Creating a More Accurate Human Development Index Using the Cobb-Douglas Production Function and Least Squares Regression Method

International Baccalaureate Mathematics: Analysis and Approaches Internal Assessment (Higher Level)

Table of Contents

Introduction

The Human Development Index (HDI) is an indicator developed by the United Nations, which measures the economic development of countries. It is represented on a scale from 0 to 1, and is calculated by taking the geometric mean of the 3 indicators it measures: life expectancy at birth, education (expected years of schooling and mean years of schooling), and power purchasing parity. In 2021, India's Human Development Index dropped from 0.645 in 2019 to 0.633. This was the first time the value had fallen since 1990, and it was caused, in part, by the repercussions of the COVID-19 pandemic. This trend was experienced by 90% of countries measured (Sharma). However, apart from this, when viewed on a global scale, India's development rankings relative to other countries have steadily reduced. While it ranked 127 in the year 2000, it now ranks 131 (Nations).

This is a surprising statistic, considering economic indicators of development and growth, such as India's infant mortality and unemployment rates have been steadily dropping since 2011 (OECD). This may be an example for a common criticism of the HDI, which is that it fails to account for other important development indicators.

Aim

The aim of this paper is to investigate and propose a more accurate Human Development Index by using additional economic indicators such as infant mortality and unemployment rates. The index will be created using the Cobb-Douglas Production function and the Least Squares Regression method. The paper then aims to provide a comparative analysis of the newly-created index and the existing one to examine whether there is a change in rankings.

Personal Engagement

Living in India, a rapidly developing economy, I've witnessed a drastic change in the urban landscape over the years, with subsidized housing, free education and healthcare being expanded. I was thus intrigued to see why India's ranking on the Human Development Index had instead decreased. Considering that the HDI only measured three indicators, I believed it was not an accurate representation of India's development and could damage the country's reputation on the international stage. I was hence compelled to explore a way to make the HDI more accurate, so that it is truly representative of India's development.

Plan

First, I plan to collect secondary data from government sources and independent credible institutions such as the World Bank. This data will include indicators of the HDI (life expectancy, power purchasing parity, and GNI per capita), but also other macroeconomic objective indicators such as infant mortality and unemployment rate. It is not plausible to collect primary data as the index will measure development from a macroeconomic perspective. The index will be created using the Least Squares Regression method and the Cobb Douglas Production Function. The results will then be compared with the original index rankings.

Background Information

Production Function

A production function is an economics equation that measures the relationship between the quantities of productive factors (oftentimes labor and capital) and the amount of product (oftentimes manufacturing goods) they can produce.

This paper focuses on the Cobb-Douglas Production Function. Developed by economists Paul Douglas and Charles Cobbs in the 1930s, the Cobb-Douglas Production Function represents the technological relationship between the amounts of two or more inputs (particularly physical capital and labor) and the amount of output that can be produced by those inputs ("Cobb-Douglas Production Function | INOMICS").

Assume the following variables:

$$G \coloneqq output\ produced$$

$$c_j \coloneqq input\ \ (such\ as\ capital\ or\ labor)$$

$$\beta \coloneqq parameter\ determining\ overall\ efficiency\ of\ production$$

$$d_j \coloneqq$$

$$parameter\ determining\ the\ responsivess\ of\ output\ produced\ (G)\ to\ changes\ in\ input\ quantities\ (c_j)$$

Hence, the general form of this function for a set of l inputs is:

$$G = f(c_1, c_2, c_3, \dots, c_l) = \beta \prod_{j=1}^{l} c_j^{d_j}$$

Apart from economic production, this function can be applied to other disciplines, such as developmental economics and econometrics. This paper intends to use the production function in the context of these two disciplines.

Theoretical Development

Theoretical Development of a Cobb Douglas Production Function in a Development Economics Context

Let us consider the following variables:

$$Y := \ Development\ Index$$

$$\gamma := constant\ parameter\ (accounts\ for\ discrepancies)$$

$$x_i := input(development\ indicator)$$

$$\alpha_i := \ parameter\ determining\ the\ responsivess\ of\ final\ value\ (Y)\ to\ changes\ in\ input\ values\ (x_i)$$

The production function to be used for a set of n inputs will thus be:

$$Y = f(x_1, x_2, x_3, \dots, x_n) = \gamma \prod_{i=1}^{n} x_i^{a_i}$$

Least Squares Regression Method Derivation and Use

The Least Squares Regression method is used to find the unknown parameters ($\gamma, a_i \dots a_n$)from the Cobb Douglas Production Function.

Assume a set of N vectors: $\{\vec{h}_1, \dots, \vec{h}_N\}$, with each being 'd' dimensional vectors and each having an associated scalar $\{k_1, \dots, k_N\}$. Assume $\vec{w}$ to be the vector of 'd' dimensions holding the unknown parameters ($\gamma, a_i \dots a_n$).

To perform least squares regression, we can assume a linear function that would allow us to estimate k from $\vec{h}$:

$$k_i \approx \vec{w} \cdot \vec{h}_i$$

≈has been used to denote the quantity of $\vec{w} \cdot \vec{h}_i$ being nearly equal to k_i but not entirely accurate. $\vec{w}$ acts as the weight of the function, similar to in the generic linear equation $y = mx + b$ where m acts as a constant and it minimizes the prediction error of this function.

We can now write a formula for finding the value of $\vec{w}$ which minimizes the squared error (SE). This is important for producing an accurate index value:

$$SE = \sum_{i=1}^{N} (k_i - \vec{h}_i \cdot \vec{w})^2$$

We can convert this into matrix form to simplify calculation. Let H be the matrix which has rows as vectors $\{\vec{h}_i\}$ and K be the vector of all stacked observations $\{k_i\}$.

$$H = \begin{bmatrix} - & \vec{h}_1 & - \\ & \vdots & \\ - & \vec{h}_N & - \end{bmatrix}$$

$$K = \begin{bmatrix} k_1 \\ \vdots \\ k_N \end{bmatrix}$$

The magnitude of the squared error of this vector equation (magnitude of error is taken to produce a numerical result) would thus be:

$$SE = |K - H\vec{w}|^2$$

In order to further simplify this expression, we can use the following explanation:

Assume b as a vector with u dimensions with the transpose of b being b^T. Thus:

$$b^T b = \begin{bmatrix} b_1 & b_2 & \cdots & b_n \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} {b_1}^2 + {b_2}^2 + \cdots + {b_n}^2 \end{bmatrix} = |b|^2$$

Thus $b^T b = |b|^2$, proving that $(K - H\vec{w})^T (K - H\vec{w}) = |K - H\vec{w}|^2$

SE is hence further simplified into $(K - H\vec{w})^T (K - H\vec{w})$

We can now derive the least squares expression using partial derivatives. Note that it is also possible to use orthogonality to derive the Least Squares Regression equation. However, it is a highly abstract method as compared to calculus. Using calculus will hence be more efficient and allow for a better explanation of the derivation.

We will now differentiate this error with respect to $\vec{w}$, eventually equating it to 0 (to have a local optimum) and then solving to obtain the value of $\vec{w}$ which minimizes the error.

$$\frac{\partial SE}{\partial \vec{w}} = \frac{\partial}{\partial \vec{w}}(K - H\vec{w})^T(K - H\vec{w})$$

Expanding the factors, we obtain:

$$= \frac{\partial}{\partial \vec{w}}(K^T - H^T\vec{w}^T)(K - H\vec{w})$$

$$= \frac{\partial}{\partial \vec{w}}(K^TK - \vec{w}HK^T - \vec{w}^TH^TK + \vec{w}^TH^TH\vec{w})$$

$\vec{w}HK^T$ is assumed to be symmetrical. If so, then $(\vec{w}HK^T)^T = (\vec{w}HK^T)$ due to matrix properties. Thus:

$$(\vec{w}HK^T) = (\vec{w}HK^T)^T = (\vec{w}H)^TK = \vec{w}^TH^TK \ (note\ (K^T)^T = K)$$

Hence:

$$\frac{\partial}{\partial \vec{w}}(K^TK - \vec{w}HK^T - \vec{w}^TH^TK + \vec{w}^TH^TH\vec{w})$$

$$= \frac{\partial}{\partial \vec{w}}(K^TK - \vec{w}^TH^TK - \vec{w}^TH^TK + \vec{w}^TH^TH\vec{w})$$

$$= \frac{\partial}{\partial \vec{w}}(K^TK - 2\vec{w}^TH^TK + \vec{w}^TH^TH\vec{w})$$

$$= -2H^TK + 2H^TH\vec{w}$$

Equating to 0:

$$-2H^TK + 2H^TH\vec{w} = 0$$

Solving for $\vec{w}$:

$$H^TH\vec{w} = H^TK \Leftrightarrow \vec{w} = (H^TH)^{-1}H^TK$$

Using the Least Squares Regression Equation to Calculate the Unknown Parameters of the Cobb Douglas Function

Recall function:

$$Y = f(x_1, x_2, x_3, \dots, x_n) = \gamma \prod_{i=1}^{n} x_i^{a_i}$$

Using a logarithmic transformation to convert the Cobb Douglas function into a linear form for simplistic calculation and representation:

$$\ln(\mathrm{Y}) = \ln(\gamma \prod_{i=1}^{n} x_i^{a_i}) \Leftrightarrow \ln(\mathrm{Y}) = \ln[\gamma(x_1^{a_1})(x_2^{a_2})(x_3^{a_3}) \dots (x_n^{a_n})]$$

$$\Leftrightarrow \ln(\mathrm{Y}) = \ln(\gamma) + a_1 \ln(x_1) + a_2 \ln(x_2) + a_3 \ln(x_3) + \dots + a_n \ln(x_n)$$

Thus, the production function has been transformed into a linear function. The Least Squares Regression equation can be used to obtain the regression coefficients $(\ln(\gamma), a_i \dots a_n)$.

Recall H as the matrix which has rows as vectors $\{\vec{h}_i\}$, each vector now containing the logarithm of the inputs to the Cobb Douglas Function. Recall K as the vector of all associated stacked scalars$\{k_i\}$, each now being the logarithm of the output of the Cobb Douglas Function. Thus, the vector $\vec{w}$, which acts as the weight in the least squares regression will be the vector holding the regression coefficients $(\ln(\gamma), a_i \dots a_n)$.

Hence:

$$\begin{bmatrix} 1 & \ln(x_{11}) & \ln(x_{12}) & \dots & \ln(x_{1n}) \\ 1 & \ln(x_{21}) & \ln(x_{22}) & \dots & \ln(x_{2n}) \\ 1 & \ln(x_{31}) & \ln(x_{32}) & \dots & \ln(x_{3n}) \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & \ln(x_{n1}) & \ln(x_{n2}) & \dots & \ln(x_{nn}) \end{bmatrix} \begin{bmatrix} \ln(\gamma) \\ a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} \ln(Y_1) \\ \ln(Y_2) \\ \ln(Y_3) \\ \vdots \\ \ln(Y_n) \end{bmatrix}$$

$$H \cdot \vec{w} = K$$

We may now solve for $\vec{w}$ (the vector of the parameters) using the closed form regression equation derived above:

$$\vec{w} = (H^T H)^{-1} H^T K$$

Finding the regression coefficients ($\vec{w}$), these values can now be plugged into the original Cobb Douglas Function to obtain a function which outputs the new index for each country (as explained underneath in data collection and calculation).

Data Collection and Calculation

Data Collection

The following essential economic indicators will be used to calculate the development index:

$$x_1 := Life\ expectancy\ at\ birth$$

$$x_2 := GNI\ Per\ Capita\ (Adjusted\ for\ Power\ Purchasing\ Parity)$$

$$x_3 := Infant\ Mortality\ Rate$$

$$x_4 := Unemployment\ Rate$$

$$x_5 := Education$$

The data being used to establish the function will be that of India's from 2011 to 2021.

Economic Indicator ↓	Years →	2021	2019	2017	2015	2013	2011
Life Expectancy at Birth (Years)		67.2	70.9	70.5	69.6	68.5	67.4
GNI Per Capita adjusted for Power Purchasing Parity (USD)		6590	6650	6116	5401	4758	4361
Infant Mortality Rate (number per 1000)		28.7	28.3	31.4	34.9	38.8	43
Unemployment Rate (%)		6	5.3	5.4	5.4	5.4	5.4
Education (arithmetic mean of Expected Years of Schooling and Mean Years of Schooling)		9.3	9.1	9.35	9.05	8.65	8.3

India's Human Development Index for the following years:

Year	HDI Value
2021	0.633
2019	0.645
2017	0.644
2015	0.629
2013	0.607
2011	0.588

Data Substitution and Creation of the Production Function:

Substituting these values into the Cobb Douglas Production Function Matrix:

$$\begin{bmatrix} 1 & \ln(67.20) & \ln(6590) & \ln(28.70) & \ln(6.00) & \ln(9.30) \\ 1 & \ln(70.90) & \ln(6650) & \ln(28.30) & \ln(5.30) & \ln(9.10) \\ 1 & \ln(70.50) & \ln(6116) & \ln(31.40) & \ln(5.40) & \ln(9.35) \\ 1 & \ln(69.60) & \ln(5401) & \ln(34.90) & \ln(5.40) & \ln(9.10) \\ 1 & \ln(68.50) & \ln(4758) & \ln(38.80) & \ln(5.40) & \ln(8.65) \\ 1 & \ln(67.40) & \ln(4361) & \ln(43.00) & \ln(5.40) & \ln(8.30) \end{bmatrix} \begin{bmatrix} \ln(\gamma) \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} \ln(0.633) \\ \ln(0.645) \\ \ln(0.644) \\ \ln(0.629) \\ \ln(0.607) \\ \ln(0.588) \end{bmatrix}$$

$$H \cdot \vec{w} = K$$

$$\begin{bmatrix} 1 & 4.21 & 8.79 & 3.36 & 1.79 & 2.23 \\ 1 & 4.26 & 8.80 & 3.34 & 1.67 & 2.21 \\ 1 & 4.26 & 8.72 & 3.45 & 1.69 & 2.24 \\ 1 & 4.24 & 8.59 & 3.55 & 1.69 & 2.21 \\ 1 & 4.23 & 8.47 & 3.66 & 1.69 & 2.16 \\ 1 & 4.21 & 8.38 & 3.76 & 1.69 & 2.12 \end{bmatrix} \begin{bmatrix} \ln(\gamma) \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} -0.45 \\ -0.44 \\ -0.44 \\ -0.46 \\ -0.50 \\ -0.53 \end{bmatrix}$$

$$H \cdot \vec{w} = K$$

Using the equation $\vec{w} = (H^T H)^{-1} H^T K$ to find $\vec{w}$:

The matrix H will be:

$$\begin{bmatrix} 1 & 4.21 & 8.79 & 3.36 & 1.79 & 2.23 \\ 1 & 4.26 & 8.80 & 3.34 & 1.67 & 2.21 \\ 1 & 4.26 & 8.72 & 3.45 & 1.69 & 2.24 \\ 1 & 4.24 & 8.59 & 3.55 & 1.69 & 2.21 \\ 1 & 4.23 & 8.47 & 3.66 & 1.69 & 2.16 \\ 1 & 4.21 & 8.38 & 3.76 & 1.69 & 2.12 \end{bmatrix}$$

The matrix H^T will hence be:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 4.21 & 4.26 & 4.26 & 4.24 & 4.23 & 4.21 \\ 8.79 & 8.80 & 8.72 & 8.59 & 8.47 & 8.38 \\ 3.36 & 3.34 & 3.45 & 3.55 & 3.66 & 3.76 \\ 1.79 & 1.67 & 1.69 & 1.69 & 1.69 & 1.69 \\ 2.23 & 2.21 & 2.24 & 2.21 & 2.16 & 2.12 \end{bmatrix}$$

Thus, $H^T H$ will be:

$$\begin{bmatrix} 6 & 25.41 & 51.75 & 21.12 & 10.22 & 13.17 \\ 25.41 & 107.614 & 219.171 & 89.4344 & 43.2787 & 55.7777 \\ 51.75 & 219.171 & 446.496 & 182.014 & 88.1605 & 113.627 \\ 21.12 & 89.4344 & 182.014 & 74.4834 & 35.962 & 46.3245 \\ 10.22 & 43.2787 & 88.1605 & 35.962 & 17.4174 & 22.4361 \\ 13.17 & 55.7777 & 113.627 & 46.3245 & 22.4361 & 28.9187 \end{bmatrix}$$

$(H^T H)^{-1}$ is:

$$\begin{bmatrix} 522241 & -77747.6 & -15270.1 & -17199.9 & -36298.1 & 27834 \\ -77747.6 & 23509.8 & -2416.76 & -1606.61 & 10977.6 & -6384.99 \\ -15270.1 & -2416.76 & 2323.3 & 2142.43 & -1147.75 & -54.1935 \\ -17199.9 & -1606.61 & 2142.43 & 2025.88 & -760.524 & -141.371 \\ -36298.1 & 10977.6 & -1147.75 & -760.524 & 5247.61 & -2985.87 \\ 27834 & -6384.99 & -54.1936 & -141.371 & -2985.87 & 2395.19 \end{bmatrix}$$

K is:

$$\begin{bmatrix} -0.45 \\ -0.44 \\ -0.44 \\ -0.46 \\ -0.50 \\ -0.53 \end{bmatrix}$$

Hence $H^T K$ will be:

$$\begin{bmatrix} -2.82 \\ -11.94 \\ -24.2921 \\ -9.9554 \\ -4.802 \\ -6.1817 \end{bmatrix}$$

Hence, $(H^T H)^{-1} H^T K$ is:

$$\begin{bmatrix} 505705.3431 & -53264.3282 & -18485.1451 & -26309.4569 & -6397.1701 & -7834.0838 \\ -53264.3269 & 10336.2434 & 152.5463 & 1021.6182 & 3254.2465 & -386.8037 \\ -18485.1456 & 152.5464 & 1382.7422 & 1629.7355 & -777.4697 & 663.4484 \\ -26309.4574 & 1021.6184 & 1629.7355 & 2021.2331 & -635.8484 & 843.8955 \\ -6397.1694 & 3254.2464 & -777.4697 & -635.8485 & 1639.7989 & -533.0039 \\ -7834.0841 & -386.8036 & 663.4484 & 843.8955 & -533.0039 & 756.5824 \end{bmatrix} \begin{bmatrix} -2.82 \\ -11.94 \\ -24.2921 \\ -9.9554 \\ -4.802 \\ -6.1817 \end{bmatrix}$$

$$= \begin{bmatrix} 1.6186632467142772 \\ -0.3897435893813963 \\ -0.08170940173465624 \\ -0.18222222224653706 \\ -0.33435897422532435 \\ 0.6731623930754722 \end{bmatrix}$$

Note that it was highly important to maintain as many decimal places as possible for each value to ensure accuracy in the final index value produced. Rounding was not considered as it had a drastic impact on the final index values.

Thus, the parameters ($\vec{w}$) are:

$$\ln(\gamma) = 1.6186632467142772$$

$$a_1 = -0.3897435893813963$$

$$a_2 = -0.08170940173465624$$

$$a_3 = -0.18222222224653706$$

$$a_4 = -0.33435897422532435$$

$$a_5 = 0.6731623930754722$$

Plugging these values into the general Cobb Douglas Function:

$$\ln(Y) = 1.6186632467142772 - 0.3897435893813963 \ln x_1 - 0.08170940173465624 \ln x_2$$
$$-0.18222222224653706 \ln x_3 - 0.33435897422532435 \ln x_4 + 0.6731623930754722 \ln x_5$$

$$\Leftrightarrow$$

$$Y = e^{(1.6186632467142772 - 0.3897435893813963 \ln x_1 - 0.08170940173465624 \ln x_2 - 0.18222222224653706 \ln x_3 - 0.33435897422532435 \ln x_4 + 0.6731623930754722 \ln x_5)}$$

Hence, we have obtained a final function that allows for the substitution of economic indicator values to obtain a new and accurate Human Development Index value for a country.

Rankings

Using the function created above, we will now calculate the new index value for a select section of countries. It proves logistically difficult to measure all countries due to a scope for human error and as well as the time required to collect data. Hence, I have chosen to compare a list of 9 countries, of which 4 are ranked below and 4 are ranked above India in the current Human Development Index rankings.

The data for 2021 for the following countries is below:

Country	Life Expectancy at Birth (Years)	GNI Per Capita adjusted for Power Purchasing Parity (USD)	Infant Mortality Rate (number per 1000)	Unemployment Rate (%)	Education (arithmetic mean of Expected Years of Schooling and Mean Years of Schooling)
Cape Verde	74.1	6230	15.284	12.64	9.45
Bangladesh	72.4	5472	21.556	5.23	9.9
Tuvalu	64.5	6351	18	24	10
Marshall Islands	65.3	4620	25	6.3	10.55
Ghana	63.8	5745	31.768	4.70	10.15

Micronesia	70.7	3696	21	16.2	9.65
Guatemala	69.2	8723	18.042	3.57	8.15
Kiribati	67.4	4063	39.322	8.6	9.9

Using the function created above, the new index values for the following countries are obtained by substituting the data for 2021. They are then ranked relative to each other.

Country	Old HDI Value	New HDI Value	Current HDI Ranking worldwide	Old Ranking (Existing HDI)	New Ranking (New HDI)
Ghana	0.632	0.744	133	6	1
Guatemala	0.627	0.730	135	8	2
Marshall Islands	0.639	0.729	131	4	3
Bangladesh	0.661	0.724	129	2	4
India	0.633	0.633	132	5	5
Kiribati	0.624	0.579	136	9	6
Cape Verde	0.662	0.545	128	1	7
Micronesia	0.628	0.511	134	7	8

Tuvalu	0.641	0.468	130	3	9

Thus, it is evident from the table that there has been a change in the index values for the 8 countries ranked alongside India. Considering the large differences in index values between India and countries above and below it, it is highly possible that countries previously not ranked near India in the original Human Development Index rankings may now appear closer due to the new indicators being included. This may alter the entire Human Development Index ranking. However, it proves logistically difficult to find which countries these are.

Note that India's index value remains the same. This is because the new index was created using economic development data from India and was correlated to its existing Human Development Index value.

Through this method, I was able to normalize the all index values from 0 to 1. Since India's index value remained the same, normalization between 0 and 1 remained the same as in the original Human Development Index while accounting for more economic indicators. Other countries, however, will display index values different than their original ones due to the different economic indicator values of infant mortality rate and unemployment rate. Nevertheless, these values will remain between 0 and 1.

The current Human Development Index is calculated by taking the geometric mean of the 3 indicators currently used. One can assume that the same method could have been used in this research process as well instead of using Least Squares Regression and the Cobb Douglas Production Function. However, compared to this method, which creates a function in which indicator values for each country can simply be substituted to obtain an index value, the geometric mean method proves inefficient and cumbersome as it would require more calculation to obtain an index value. Also, using a geometric mean would change the scale at which the index is measured and would have to be normalized. The Cobb Douglas Function and Least Squares Regression method maintains normalization since it relies on India's existing HDI value.

Conclusion

Limitations

There were a few limitations to the research conducted. A primary concern was the lack of data availability. This limited the number of indicators which could have been used to create index. Initially, metrics such as poverty rate and sector makeup were intended to be included. However, due to a lack of data, only infant mortality rate and unemployment rate were used as additional indicators. This made the new index not as accurate as it was intended to be, but still more accurate than the original index.

Another limitation of the research was the logistical difficulties in calculating the index for all different countries. It proved cumbersome to find data for countries and substitute them into the function to obtain the new index value. However, this was worked around by simply taking a sample of countries and calculating their indices rather than the entire list.

Lastly, a key limitation was the high potential for human error. It was imperative to not round the values as they had a drastic impact on the final index value. This meant that it was important to be accurate and cognizant of the weight/regression coefficient values when calculating index values.

Future Considerations

There is a wide scope for future research on this topic. With greater data availability on important indicators such as the poverty rate, the index could be made more accurate, providing governments with a reliable way to

measure their countries' development. Furthermore, it may be possible to make the potential for human error less by finding a method that allows rounding up weight values without having a large impact on the outcome.

The method that has been used to develop this new Human Development Index can now also be used to increase accuracy of other economic indexes such as the corruption perception index and the happy planet index. By including more indicators in these indices, they can become more representative of a country's actual state.

I can see my research as a foundation for future econometric analyses of economic development.

Concluding Statement

It was eye-opening to combine statistical methods with calculus to predict and measure economic development. I was able to link my interests together and was satisfied with the outcome of my research and calculations. Although there were certain limitations, I was able to develop a more accurate way of measuring human development and compare results to the original Human Development Index. I was also able to better gauge India's development relative to other countries, but with a more effective way to rank all countries, this could have been made more accurate. Nevertheless, I was able to meet the aim of my investigation. I now wish to build upon this topic in my future studies, as I believe it can be highly beneficial in the fields of policy-making and governance.

Works Cited

Cottrell, Allin. *The Cobb–Douglas Production Function*. 2019. *ECN 207 : Intermediate Macroeconomics*, http://users.wfu.edu/cottrell/ecn207/cobb-douglas.pdf. Accessed 26 October 2022.

Sharma, Neetu. "India slips down in UN Human Development Index, stands at 132 out of 191 countries." *Business Today*, 8 September 2022, https://www.businesstoday.in/latest/economy/story/india-slips-down-in-un-human-development-index-stands-at-132-out-of-191-countries-346765-2022-09-08. Accessed 05 November 2022.

Khater, E. 2012. *Hikari*, http://www.m-hikari.com/ijcms/ijcms-2012/9-12-2012/khaterIJCMS9-12-2012.pdf. Accessed 31 October 2022.

McKenzie, Tom. "Cobb-Douglas Production Function." *INOMICS*, 15 April 2020, https://inomics.com/terms/cobb-douglas-production-function-1456726. Accessed 25 October 2022.

Organisation for Economic Co-operation and Development. "Selected indicators for India." *OECD Data*, 2021, https://data.oecd.org/india.htm. Accessed 31 October 2022.

Pillow, Jonathan. *Lecture 3B notes: Least Squares Regression*. 2018. *pillow lab @ princeton*, http://pillowlab.princeton.edu/teaching/statneuro2018/slides/notes03b_LeastSquaresRegression.pdf. Accessed 26 October 2022.

White, Steve. "Cobb Douglas Production Function." *YouTube*, 2 February 2019, https://www.youtube.com/watch?v=qpaiuJA-nFU. Accessed 02 November 2022.

The World Bank. "India | Data." *World Bank Data*, 2021, https://data.worldbank.org/country/IN. Accessed 02 November 2022.

Extended Essay

Automatic Stabilizers: Bane or Boon?

To what extent were automatic stabilizers effective in cushioning income and unemployment shocks in Denmark during the COVID-19 Pandemic (2019 Q4 to 2022 Q2)?

Examination Year: May 2023

Subject: Economics

Word Count: 3958 (including footnotes)

Table of Contents

Introduction

Automatic stabilizers are an integral part of government fiscal policy in most industrialized countries(ET CONTRIBUTORS). However, less economically developed countries (LEDCs) rarely employ this concept due to limited funds. As my country, India, progresses into an industrialized country, it will have to consider whether to include these tools as part of its fiscal policy. This has motivated me to conduct research on the use of automatic stabilizers in existing fiscal policies to evaluate whether India should implement them.

Approaches to fiscal policy have been divided; countries with a higher emphasis on automatic stabilizers enact smaller discretionary fiscal stimulus(Dolls et al.). Denmark, for example, relies heavily on automatic stabilizers to address crises compared to the United States, which relies on fiscal stimulus packages. While both have similar intentions to offset negative fluctuations, their effects on households and government budgets are varied.

Considering these varying effects, it is essential to pursue this topic thoroughly. A failure to explore the effects of automatic stabilizers may lead to misinformed decisions by governments of newly industrialized countries causing them to be unprepared in the next economic crisis. As I researched further, I discovered Denmark as a prime country for analysis.

Home to nearly 6 million inhabitants, Denmark is a Scandinavian country bordering Sweden, the Netherlands, and other European countries("Denmark | History, Geography, & Culture | Britannica"). 88.2% of inhabitants live in urban areas("Urban Population (% of Total Population) - Denmark | Data"). The country also has one of the lowest Gini coefficients in the world, at 0.28, indicating low levels of inequality relative to OECD countries at 0.318(Thévenot). Denmark's GDP per Capita is estimated to be USD 68,000 (The World Bank).

This provides a clear rationale for choosing Denmark as a country of analysis. With a lower Gini-Coefficient, the effects of automatic stabilizers on cushioning income and unemployment shocks become more uniform across the population, allowing for a more accurate analysis. Denmark also has one of the most robust automatic stabilizer systems worldwide, thus providing a broader scope for analysis.

Thus, the essay will focus on the following question: *"To what extent were automatic stabilizers effective in cushioning income and unemployment shocks in Denmark during the COVID-19 Pandemic (2019 Q4 to 2022 Q2)?"*

The essay also explores the discretionary response of the United States, a country of similar economic development to Denmark, during the pandemic to evaluate the effectiveness of automatic stabilizers relative to other forms of fiscal policies. This may further explain if another fiscal policy rather than automatic stabilizers would have been effective in cushioning shocks, allowing a conclusion to be formed about whether India should consider implementing them.

Another interesting area of analysis is whether Denmark's savings rate complemented the effects of automatic stabilizers. This will be compared to saving rates in the United States for evaluation.

Hypothesis

It can be hypothesized that automatic stabilizers will be able to cushion economic shocks more effectively than discretionary policy, leading to a minimal reduction in aggregate demand. This will prevent a significant loss in GDP and eventually large spikes in unemployment.

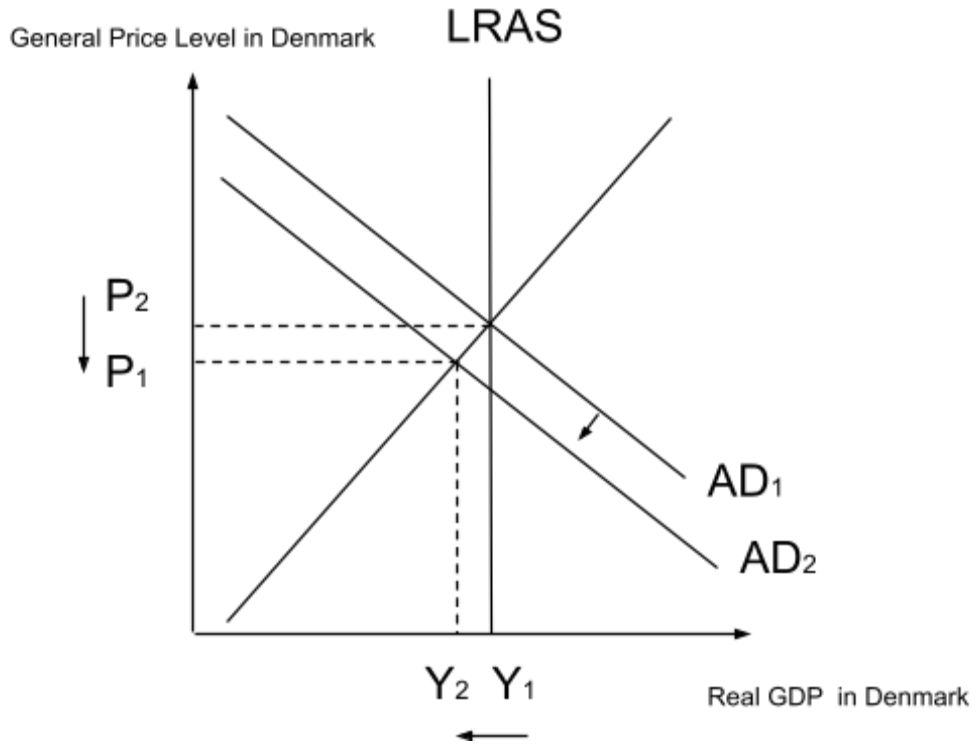


Figure 1: The effect of automatic stabilizers in reducing the shift in aggregate demand

Figure 1 shows the state of the economy at different stages and the effects of automatic stabilizers.

Before the pandemic, the economy operated at (Y1,P2) at the economy's maximum potential (LRAS). However, during the pandemic, aggregate demand decreases from AD1 to AD2, due to reductions in consumer spending. The economy now produces above the natural rate of unemployment (Y2,P1), causing the recessionary gap. However, AD does not shift to the left as much as it would without automatic stabilizers. This is because government expenditure offsets some of the shock caused by the reduction in consumer expenditure.

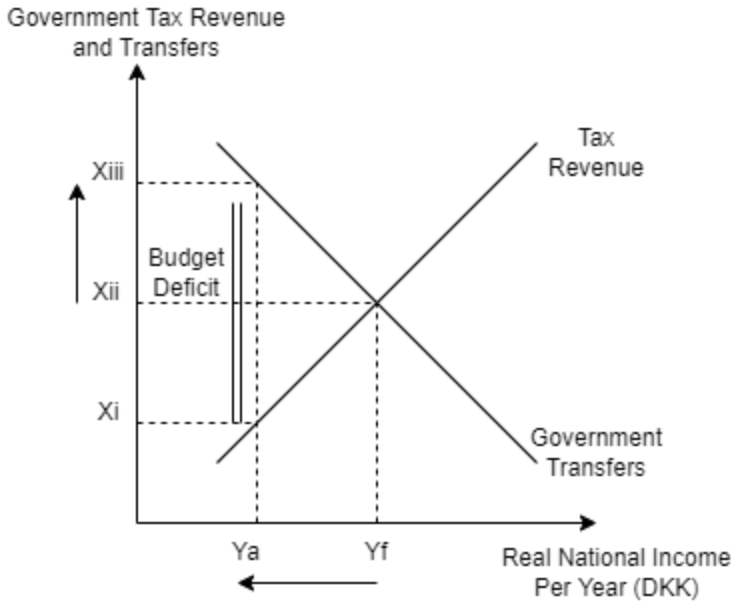


Figure 2: Creation of Budget Deficit due to Pandemic

It is also hypothesized (refer to figure 2) that using automatic stabilizers in Denmark will drastically increase the government's budget deficit (Xiii-Xi) during the pandemic, thus increasing its debt-GDP ratio and causing future debt servicing. This is because in stages of recession, government spending will automatically increase in areas such as unemployment (such as from Xii to Xiii), and national income will decrease due to more households entering lower income brackets and decreasing expenditure (Yf to Ya). This will cause expenditure to be greater than revenue.

Automatic stabilizers are also hypothesized to cushion income shocks more effectively. This is because, contrary to discretionary policy, which can take months for stimulus to arrive, automatic stabilizers arrive on a timely basis, allowing households to continue spending patterns akin to regular times. This can prevent income inequality from deepening as well as spikes in inflation in the future since funds are being allocated at regular intervals, providing a steady stream of income.

Methodology

Methodology: Primary Research

In order to assess Denmark's automatic stabilizer system and its effectiveness during the pandemic on an individual level, an asynchronous survey was conducted. The survey had thirty respondents living in Denmark's urban areas. The survey pool was intended to be diverse and consisted of varying socioeconomic backgrounds and occupations.

In order to maintain privacy in questions that concerned incomes and unemployment insurance, exact data values were not collected but rather percentages. This consideration allowed respondents to provide accurate data.

Apart from quantitative analysis, the survey also intended to focus on qualitative data. It thus collected opinions on Denmark's tax, unemployment, and business support systems. This qualitative approach proves vital when understanding how automatic stabilizers could be strengthened and consumer sentiment surrounding them. An effort was made to acquire opinion of the Danish economy and its functioning before and after the pandemic to establish whether automatic stabilizers were effective.

Methodology: Secondary Research

The essay, wherever possible to avoid source bias, uses Danish government reports and independent research papers recognized by the European Union and journal articles from reputed journals reporting on the pandemic. This approach is to obtain descriptions and data on Danish macroeconomic objective indicators such as unemployment, inflation, debt, income levels, and additional fiscal policy measures. Extensive secondary research was conducted to analyze the effectiveness of automatic stabilizers quantitatively during the pandemic.

With these statistics, it becomes possible to analyze the cushioning effect of stabilizers on income and unemployment shocks. It also becomes feasible to answer whether these stabilizers are worth implementing as the essay analyzes their effects on government budgets and on curbing post-pandemic inflation.

Furthermore, it uses statistics from the United States during the pandemic to provide a comparative analysis of the tools' effectiveness in cushioning shocks.

It was necessary, as well as most accurate, to use secondary data to collect quantitative data. Since the essay focuses on a macroeconomic concept, primary research renders comparatively ineffective due to its inability to gauge the state of the entire economy.

Body:

Background Information on Automatic Stabilizers

Automatic stabilizers are in-built mechanisms into government budgets that automatically take effect as the state of the economy changes(Lee and Sheiner). Contrary to discretionary fiscal policy, in which large-scale stimulus packages are passed through the legislature, automatic stabilizers require no additional approval from the government and can thus take effect immediately. Examples include pension schemes and unemployment insurance (transfer payments), and the progressive tax system. These tools are primarily designed to counter adverse economic shocks[1], as well as combat problems associated with expansion such as inflation("Automatic Stabilizer").

[1] Unpredictable events having a widespread effect on the economy

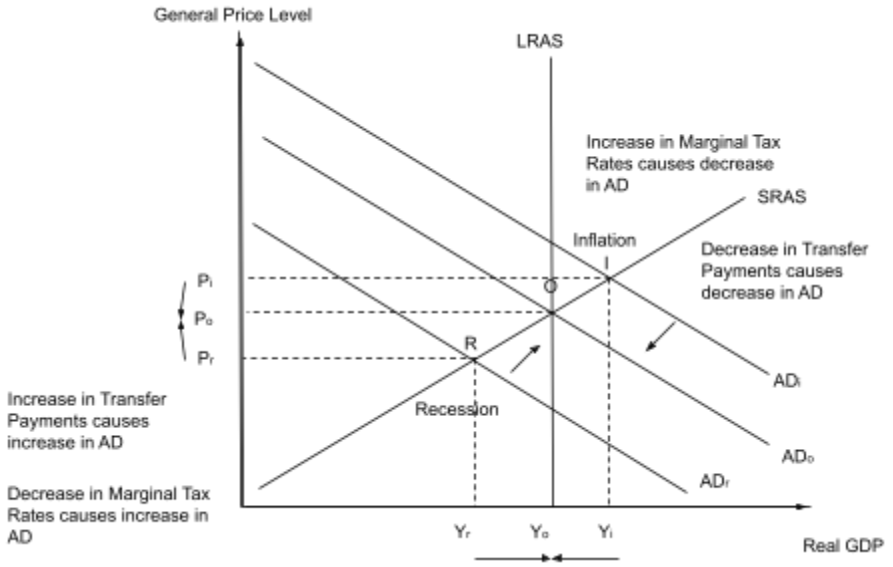


Figure 3: Effect of automatic stabilizers in shifting aggregate demand

The graph above shows the effects of automatic stabilizers in periods of inflation and recession. During a recession (R), transfer payments are increased and marginal tax rates are reduced. This encourages consumer spending (explained in next section), leading to a shift in aggregate demand to the socially optimal level (O). Real GDP increases from Yr to Yo along with price levels from Pr to Po. Similarly, during inflation (I), marginal tax rates are increased and transfer payments are decreased. This discourages consumption, leading to a shift in aggregate demand to optimal level (O). Real GDP decreases from Yi to Yo along with price level from Pi to Po.

While automatic stabilizers also include elements such as pension schemes, this essay focuses primarily on unemployment insurance and Denmark's progressive tax system, as they will have a more direct effect on cushioning income shocks[2] during a pandemic.

[2] Sudden, significant drops in income levels that can hamper an individual's ability to afford necessities

Correlations between economic development and economic sector make-up, and its relation to unemployment in the pandemic context in Denmark

Analyzing Denmark's economic structure and labor force provide insights into the specific sectors that make up the economy. With this data, it becomes feasible to analyze whether automatic stabilizers, primarily in the context of the pandemic, can depend on the type of sector an economy specializes in and how developed it is.

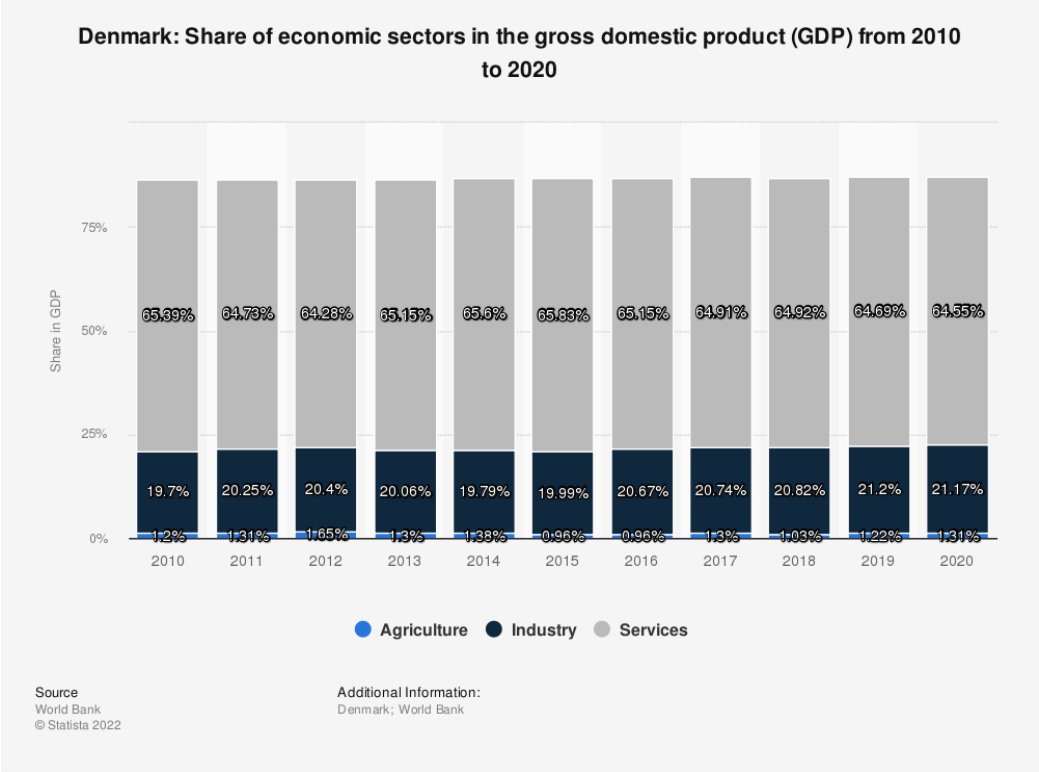


Figure 4: The share of economic sectors (of GDP) in Denmark from 2010 to 2020 (data source: Statista, The World Bank, 2022)

Figure 4 displays the sector make-up of the Danish economy before and after the onset of the COVID-19 pandemic. The Danish economy specializes in the service sector or tertiary sector. From having the largest share of GDP, the sector also employs the majority of the country's labor force; in 2017, the tertiary sector employed more than thirty times the force

employed in agriculture[3](Moody's Analytics). This finding proves that Denmark's economy strongly depends on the tertiary sector.

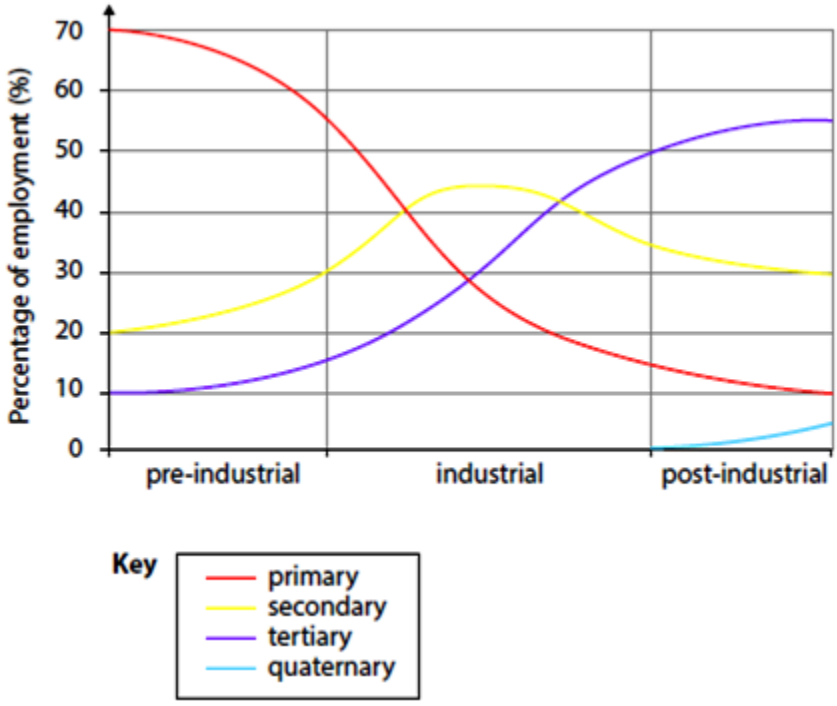


Figure 5: The Clark-Fisher Model (data source: GCSE Revision: Economic change, 2014)

The above model identifies a country's development stage based on the composition of its sectors. Considering that the tertiary sector makes up most of the Danish economy, and whose portion of the economy has remained constant over the years, it can be inferred that Denmark is in its post-industrial stage (or MEDC - More Economically Developed Country). The growth of the tertiary sector as a country becomes an MEDC can be explained. As the country emerged as an MEDC, workers' wages began to increase, allowing them more disposable income to spend on the services sector. This was due to an increased labor productivity[4], where advancements in technology lowered costs, increased wages, and also allowed more workers to switch from primary and secondary sectors to tertiary ones(Pettinger).

[3] Calculated using raw data from Moody's Analytics
[4] Real economic output per labor hour

This emphasis on the tertiary sector explains a rise in the unemployment rate- by 1.9 percentage points by June 2020 (Statista), two months after the lockdown implementation- due to the COVID-19 Pandemic. Considering the tertiary sector provides 'intangible assets'[5], often involving face-to-face interactions, the lockdowns led many service-oriented businesses to close or limit activities. This eventually resulted in reduced demand for labor, leading to layoffs. Thus, the emphasis on the tertiary sector may explain why automatic stabilizers had a drastic positive effect on the Danish economy (discussed below).

The Link Between Unemployment, Income, and GDP

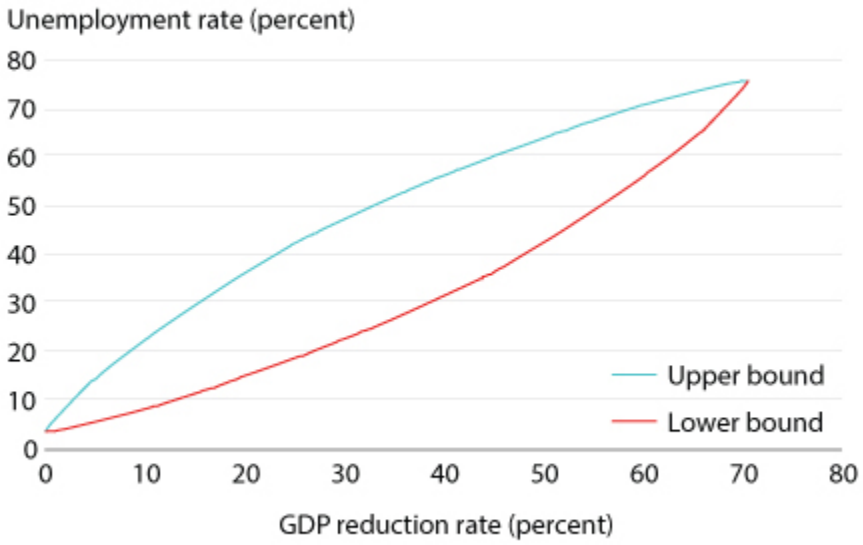


Figure 6: Relationship between unemployment and GDP (data source: St. Louis Fed, 2020)

Figure 6 shows a nearly linear correlation between a decrease in the GDP of an economy and an increase in the unemployment rate. With the pandemic causing the tertiary sector to halt services, a large share of households experienced unemployment, leading to income shocks. These shocks had repercussions across the economy, causing reductions in consumer expenditure.

[5] Services not physical in nature

This can, as per the graph and the equations below, explain why Denmark's GDP[6] contracted during the COVID-19 Pandemic by 5% in 2020(Marinov).

$$Y = C + I + G + (X - M)$$

The equation above outlines the expenditure approach[7] of measuring GDP. Considering that consumer expenditure (C) is a critical component of GDP (Y), a lack of it will lead to contraction.

However, it is essential to note that Denmark's GDP in 2020 did not contract as much as expected (Europa.eu). This is due to increased government expenditure (G) through automatic stabilizers (unemployment insurance) and extra fiscal stimulus packages. The expenditure offset some of the economic shock caused by consumer expenditure reduction.

Denmark's Unemployment Insurance System

Denmark's unemployment insurance system is known to be generous. It is uniquely known for its 'flexicurity' model, which allows employers to lay off employees without litigation and offer new roles whenever they wish (Andersen and Mailand 9-10). The system is possible due to the state-covered unemployment scheme, which is also paid by labor and trade unions and sponsored employment training programs.

Thus, Denmark has some of the highest job turnover rates at 20%. This system is beneficial for all stakeholders. It allows the unemployment rate to remain low but also keeps mobility rates high while simultaneously providing freedom to employers and maintaining the well-being of employees.

[6] Gross Domestic Product
[7] The sum of all final goods and services produced within an economy

Eligibility criteria for this scheme are relatively simple, allowing for 80% of the labor force to be covered (The World Bank). This ease of eligibility is in contrast to most OECD[8] countries, which only cover 40% of job seekers (OECD).

However, while eligibility requirements are not complex to meet[9], Denmark, second to Poland, has the strictest availability requirements in the OECD (OECD); it requires citizens to actively seek jobs on Denmark's public employment service website and be ready to take up roles within a day's notice or relocate for employment. If not, it may lead to sanctioning[10]. Thus, the system incentivizes citizens to seek employment while providing a safety net. This reduces pressure on the government to support those unemployed, reducing opportunity costs of other essential services such as education and public healthcare.

An advantage of the Danish system is its longevity. Citizens are provided benefits for up to three years. This benefit allows consumer expenditure to stay constant during the economic crisis, preventing a significant drop in GDP (as outlined above) in a particular economic quarter[11].

The cushioning of GDP contraction is in contrast to countries such as the United States, where, having to wait for large-scale stimulus payments, the GDP dropped by 8.9 percentage points (The White House) in one quarter; households potentially ran out of disposable income before the stimulus arrived, which lowered consumer expenditure drastically (20% of U.S. households lost their savings during the Pandemic (Calfas)).

[8] Organisation for Economic Co-operation and Development
[9] Being a member of a recognized unemployment insurance fund for at least one year, and a minimum income of USD 37,800 during the three preceding years
[10] Disqualification from benefits and government support
[11] 3 month period of a fiscal year

This beneficial effect of the Danish government expenditure can be quantified using the fiscal multiplier[12] explanation. It is estimated that due to the robust automatic stabilizer system and discretionary fiscal policy, the multiplier effect is 1.41 (Byrialsen and Olesen). This is in contrast to the United States, which had 0.50. This is perhaps due to the fact that the United States may not have provided as much stimulus as in Denmark's automatic stabilizer system.

The unemployment scheme also covers up to 90% of one's prior salary, but only a maximum of USD 2600 monthly; it is reasonable, as the average monthly cost of living in Denmark is approximately USD 1000 (Lončar). The extra income from the insurance can prevent aggregate demand from reducing further. One respondent in the survey commented: "I purchased multiple cheap luxury items during the lockdown." This response shows a willingness and ability to spend on non-essential goods. The same respondent also verified the effectiveness of insurance programs by stating their quick response. 63% of respondents also stated that unemployment insurance would be able to maintain their current lifestyle and allow them to obtain all necessities.

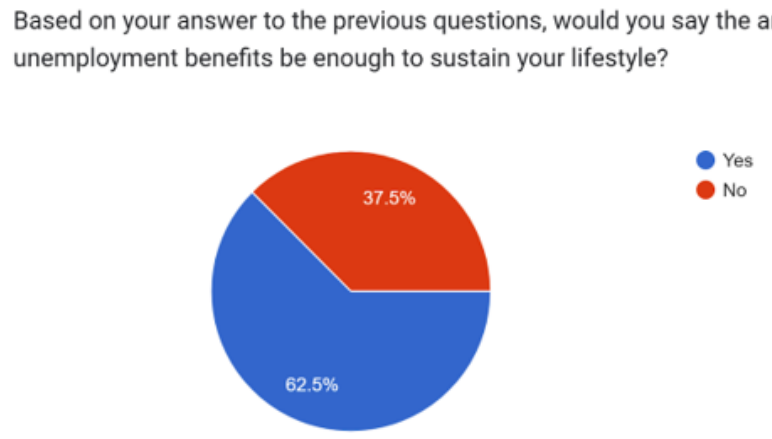


Figure 7: Primary responses to unemployment benefits (data source: primary survey)

[12] A measure of the effect of government expenditure: the higher the multiplier, the higher the effect on the economy of stimulus being spent by the government.

Thus, from a consumer and economic standpoint, unemployment insurance appears to be beneficial, providing Danish citizens with spending potential that they can use to sustain themselves and keep the economy relatively stable. Nevertheless, it is also essential to analyze the effect of unemployment insurance provision from a government expenditure standpoint.

Social expenditure by type of benefits, measure and time

	2018	2019	2020
Total social expenditures			
2.4 Unemployment benefits for unemployed persons with limited working capacities	3 676	3 694	3 954
6.1 Municipal employment measures	5 447	6 027	5 392
6.2 Active employment measures	513	511	223
6.3 Unemployment insurance benefits; unemployed people not joining training programmes	16 247	16 794	23 389
6.4 Unemployment insurance benefits and other income substitution for unemployed people in training programmes	6 192	5 843	4 611
Cash benefits gross			
2.4 Unemployment benefits for unemployed persons with limited working capacities	3 676	3 694	3 954
6.1 Municipal employment measures	0	0	0
6.2 Active employment measures	0	0	0
6.3 Unemployment insurance benefits; unemployed people not joining training programmes	14 761	15 294	21 888
6.4 Unemployment insurance benefits and other income substitution for unemployed people in training programmes	5 966	5 615	4 383

31-8-2022 Danmarks Statistik , www.statbank.dk/ESSPROS1

Figure 8: Unemployment expenditures from 2019-2020, Danish government (data source: Statistics Denmark, 2022)

It is evident that government expenditure to cushion shocks drastically increased during the Pandemic, with unemployment benefits (6.3) increasing by more than 6000 million Danish Krone. The expenditure contributed to the 26% increase in the national debt in 2020 (O'Neill).

The hike in debt may have consequences in the near future, such as a reduction in national savings that can be invested in essential services (The Danish healthcare system relies on government spending) and significant tax hikes. Considering the already high effective tax rates(discussed below), it proves controversial to increase tax burdens and may lead to a recession due to decreased disposable income.

Denmark's Progressive Tax System

The Progressive Tax system[13] is an automatic stabilizer implemented by most countries worldwide. Nevertheless, each country implements it differently. The Danish system is considered to be one of the most progressive, with high effective tax rates.

DKK salary	DKK tax 2018	DKK tax 2019	DKK tax 2020	Effective tax % 2020
100.000	22.400	22.000	22.300	22,3 %
200.000	61.600	61.000	61.700	30,9 %
300.000	100.800	99.800	100.600	33,5 %
400.000	141.200	139.800	140.800	35,2 %
500.000	182.900	181.400	182.900	36,6 %
600.000	232.400	228.800	228.200	38,0 %
700.000	287.900	284.300	284.000	40,6 %
800.000	343.300	339.700	339.900	42,4 %
900.000	398.700	395.100	395.800	43,0 %
1.000.000	454.200	450.600	451.700	45,2 %
2.000.000	1.008.400	1.004.900	1.010.700	50,5 %

Figure 9: The Danish Progressive Tax system (data source: pwc consulting)

The system has twelve brackets, with the highest effective tax rate being 50.5%. Surprisingly, the Danish government did not experience a drastic decrease in tax revenue during the Pandemic, contrary to what was hypothesized. Its revenue reduced only 0.6% of its GDP from 2019 to 2020 (The World Bank). The slight reduction could potentially be because unemployment insurance in Denmark is taxable.

A key metric to calculate when measuring tax system effectiveness is income shock cushioning. Assume a 40% proportional income tax rate. In a scenario of an income shock of DKK 100, there would be a loss of disposable income of DKK 60. The system absorbs 40% of

[13] The tax rate increases as one's income increases

the shock. The more progressive a system is, the more shock it can cushion(Dolls). Similarly, Denmark's automatic stabilization (the tax system) of households' income cushions 62% of the shock for all households and 77% for low-income households (The European Commission 2).

Being able to cushion large amounts of low-income shocks is an advantage of the Danish system and explains why wealth inequality is low and stayed low during the Pandemic. Higher incomes are taxed at high rates, allowing income redistribution. Furthermore, those with lower incomes pay low effective tax rates, allowing them to have disposable incomes closer to those of the top earners. This system can be compared to the United States, which, while having a progressive system, has lower effective federal tax rates for the top percentage of earners (30.6% compared to Denmark's 52.4%) (Gravelle). This has resulted in the country having a higher Gini Coefficient than Denmark, as higher-income citizens essentially pay lower effective tax rates than lower-income households.

It can also be inferred from the ability of this system to reduce inflationary pressures, first proposed by Keynes. While a recession would cushion shocks, an expansion would prevent inflation by taking away disposable income in the form of taxes, thus preventing an inflationary spiral.

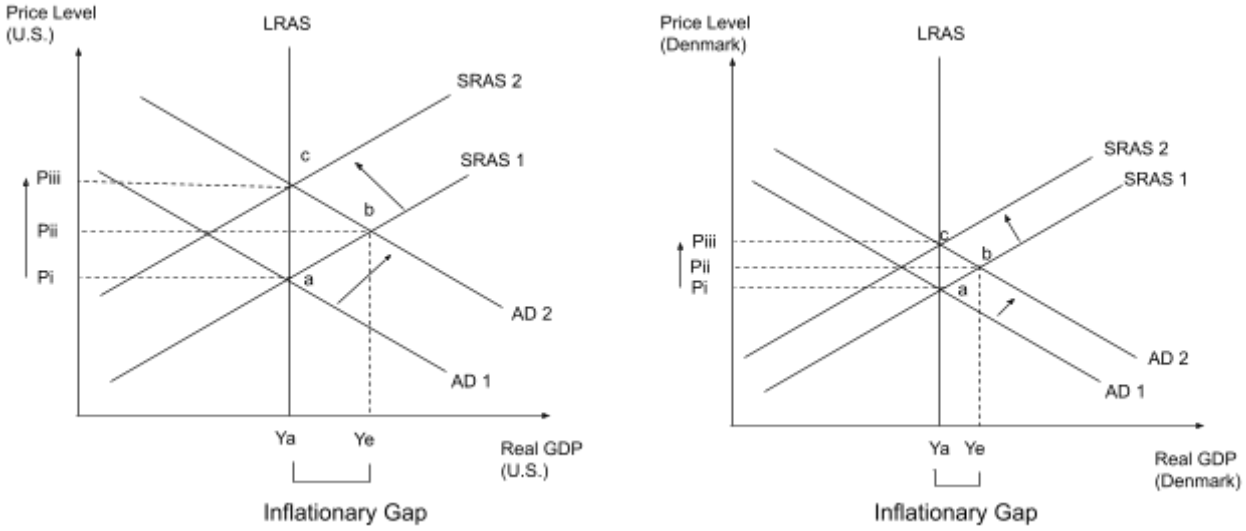


Figure 10:Theoretical difference in inflationary gap between the U.S. and Denmark due to automatic stabilizer usage

Figure 7 shows inflationary spirals in Denmark and the United States. The U.S. faces a more significant inflationary gap due to a less progressive system; as incomes increase as the Pandemic lessens, U.S residents have more disposable income than those in Denmark. Denmark's inflationary gap is smaller, leading to a minor increase in output and price than in the U.S.

Quantitatively, in June 2022, the U.S. had a rate of 9.1% (Bureau of Labor Statistics), compared to Denmark's 8.2% (Trading Economics). Furthermore, as per the Consumer Price Index[14], most OECD countries, who rely on heavier taxation rates, experienced drastically lower inflation rates than the U.S. This shows a benefit of progressive tax.

[14] Measures the average change of prices paid by consumers over a period of time for a basket of goods

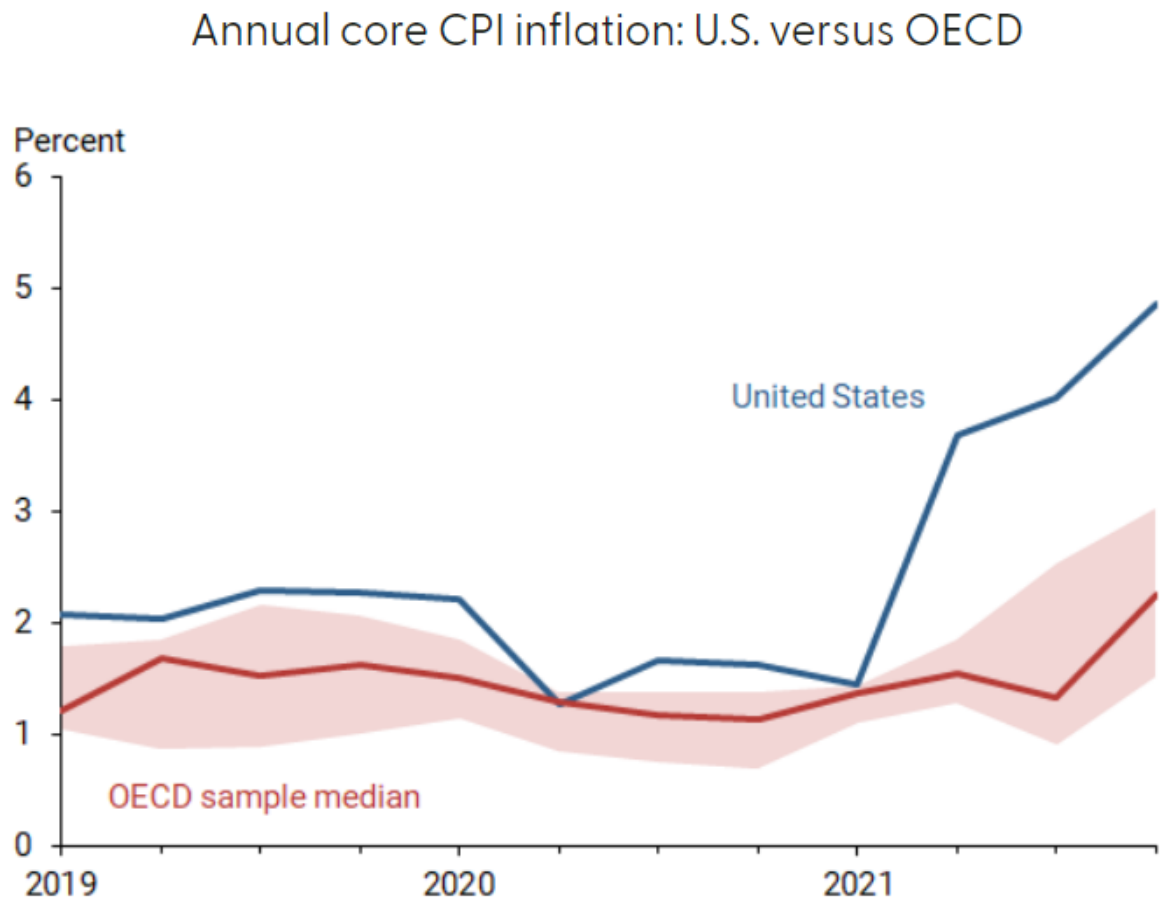


Figure 11: Inflation in the U.S. versus inflation in OECD countries (data source: OECD)

Thus, from a consumer standpoint, the system appears to be beneficial. This inference is reinforced by the primary survey results, with most respondents voicing support for such a system. While many respondents stated that the rates were exceptionally high, they also noted generous welfare benefits such as unemployment insurance, healthcare, and pensions. The government also did not experience a loss in revenue.

Saving Rates Versus Automatic Stabilizers

Savings rates can be analyzed to better understand its relationship with automatic stabilizer effectiveness.

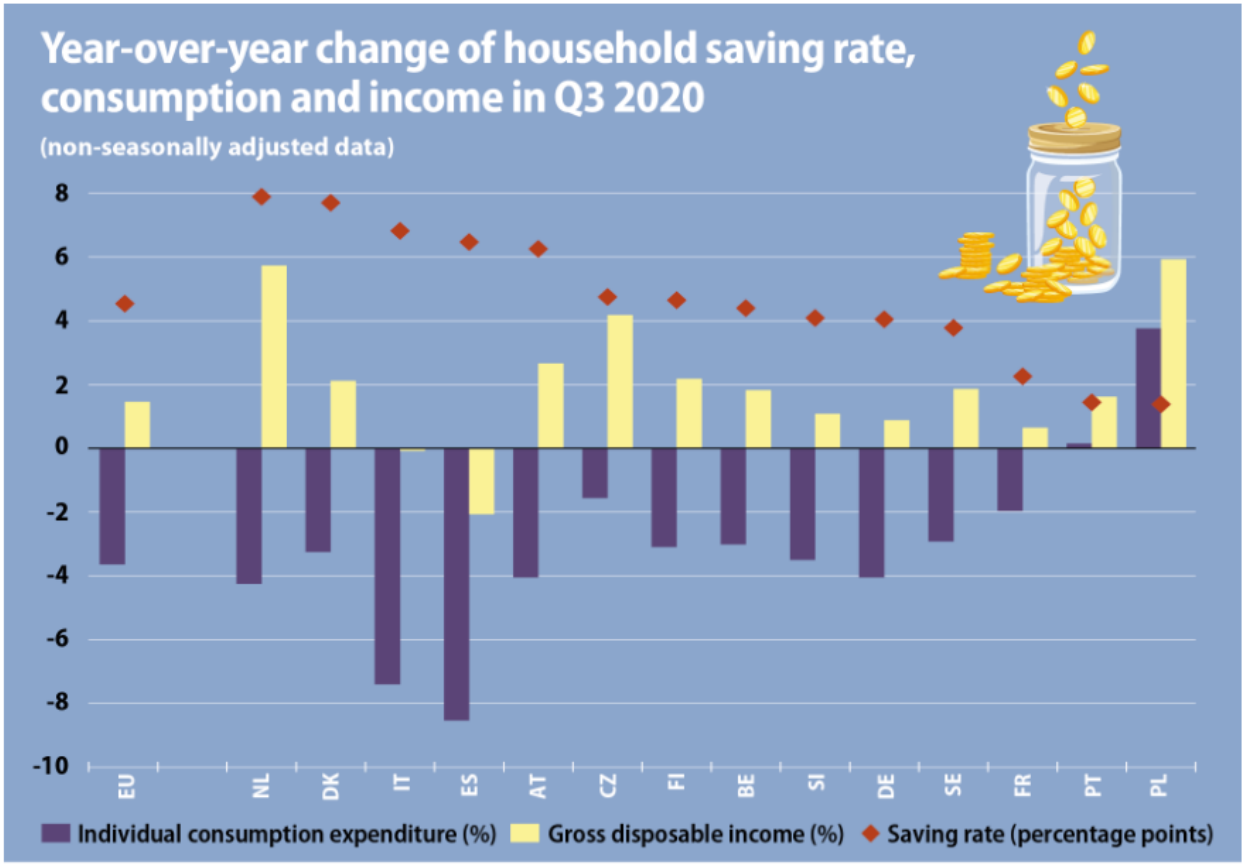


Figure 12: Change in household savings rate in the EU (data source: eurostat)

Denmark has some of the highest saving rates worldwide, and the COVID-19 Pandemic increased it by 8% (in quarter three of 2020, the gross savings rate became 30.25%). The savings rate has also resulted in Danish households having disposable income, even in economic crises. Thus, did the saving rate or automatic stabilizers cushion income shocks?

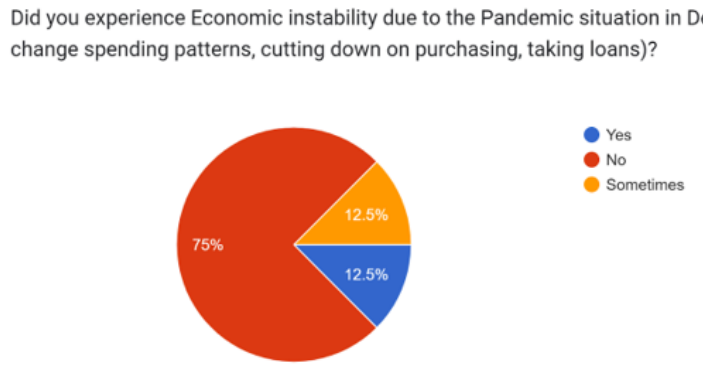


Figure 13: Primary responses to economic instability during Pandemic (data source: primary survey)

As per the survey, most respondents did not experience economic instability during the Pandemic. It can be inferred that this was because of the high saving rates and, to some extent, the welfare system. 40% of the respondents were unemployed during the Pandemic, and three were actively receiving unemployment benefits, yet most were economically stable.

Moreover, each respondent responded positively when asked about the effectiveness of automatic stabilizers such as unemployment insurance. One respondent stated it was effective, citing the stable economy during the Pandemic. Another stated that it provided a stable income for businesses. However, some respondents also did not use government support due to remaining funds or employment status.

Thus, it can be inferred that while saving rates allowed specific households to sustain themselves, automatic stabilizers were still crucial for those who lost employment, owned businesses, or were freelancers. Furthermore, with positive responses to the welfare system, Danish residents do believe in automatic stabilizers' positive effects.

Another evidence of automatic stabilizer importance despite saving rates lies in comparison. The U.S, in April 2020, had a similar savings rate (Statista) to Denmark's in July 2020. Despite the high savings rate and one-time stimulus package passed in April, personal

income dropped 4.2% in May(Statista). This indicates an apparent reduction in disposable income. In Denmark, gross disposable income, however, increased. The plausible reason is the availability of automatic stabilizers. Unlike the one-time package in the U.S, Denmark provides a steady source of income.

Conclusion

It is evident that automatic stabilizers can partially mitigate income shocks during economic crises and better prepare households for periods of recession. They can cause increases in aggregate demand while preventing inflation spikes. Thus, to answer the question posed, automatic stabilizers are highly effective from a consumer standpoint and for curbing inflation, yet they may be a source of contention for governments in debt or having strict budgets. It is essential to have adequate funding for stabilizers. Denmark, having a lower than world average debt-GDP ratio, could spend more on stabilizers without the risk of defaulting during the crisis.

Analyzing the sectoral make-up of the economy proved vital. Automatic stabilizers may be more effective in service economies rather than agricultural ones or MEDCs than LEDCs. Thus, these tools were more effective in Denmark than they would be in India at the moment.

From a consumer standpoint, there appeared to be no disadvantage when using automatic stabilizers. They were able to redistribute income, preventing inequality spikes in a Pandemic that deepened the divide. This effect shows the potential of automatic stabilizers to lift citizens from absolute and relative poverty. The unique flexicurity model protected employee and employer rights while providing an adequate social safety net, preventing a widespread drop in GDP.

It was hypothesized that saving rates may curb the effectiveness of automatic stabilizers. However, analyzing Danish saving rates and the provision of unemployment insurance showed their importance. Unemployment insurance might have contributed to the saving rates, as there was an apparent increase over the months. The insurance and low tax rates also led to a rise in disposable income, despite decreasing GDP.

There were certain limitations faced when conducting research. It was not possible to use EUROMOD, a macroeconomic simulator that models the effect of shocks on household income, due to the author of this essay not being a university student. Thus, this research had to rely solely on secondary data, which could have led to inaccuracies. Unlike for the tax system, the research did not reveal the amount of income shock unemployment insurance could cushion, which would have allowed this essay to gauge the effectiveness of insurance further.

Regardless, this essay contributes immensely to the currently opaque field of automatic stabilizers. With limited research having been done(Lee and Sheiner), this essay provides a way forward to offset shocks from future economic crises while fulfilling macroeconomic objectives and economic development levels. By analyzing Denmark, this essay provides countries with a simple framework to follow when implementing automatic stabilizers and evidence of their positive effects.

Nevertheless, the field requires further research on how discretionary fiscal policy can be integrated with automatic stabilizers to cushion shocks. With Denmark having also used discretionary fiscal policy to aid employers, it remains to be seen whether the country's economy could have withstood the effect of COVID-19 solely by relying on automatic stabilizers. Indeed, if not, then discretionary stimulus coupled with automatic stabilizers is the ideal solution.

Works Cited

Andersen, Søren Kaj, and Mikkel Mailand. "The Danish Flexicurity Model." 2005, pp. 9-10, https://faos.ku.dk/english/pdf/publications/2005/The_Danish_Flexicurity_Model_0905.pdf. Accessed 8 September 2022.

Bureau of Labor Statistics. "Consumer prices up 9.1 percent over the year ended June 2022, largest increase in 40 years : The Economics Daily: U.S." *Bureau of Labor Statistics*, 2022, https://www.bls.gov/opub/ted/2022/consumer-prices-up-9-1-percent-over-the-year-ended-june-2022-largest-increase-in-40-years.htm. Accessed 8 September 2022.

Byrialsen, Mikael Randrup, and Finn Olesen. "The macroeconomic effects of covid-19: the imperative need for a Keynesian solution." *OpenEdition*, 2021. *OpenEdition Journals*, https://journals.openedition.org/regulation/18405?lang=en. Accessed 2022.

Calfas, Jennifer. "Close to 40% of U.S. Households Say They Face Financial Difficulties as Covid-19 Pandemic Continues." *Wall Street Journal*, 14 October 2021, https://www.wsj.com/articles/close-to-40-of-u-s-households-say-they-face-financial-difficulties-as-covid-19-pandemic-continues-11634241586. Accessed 9 September 2022.

Chien, YiLi. "How Bad Can It Be? The Relationship between GDP Growth and the Unemployment Rate | St. Louis Fed." *Economic Research - St. Louis Fed*, 11 April 2020, https://research.stlouisfed.org/publications/economic-synopses/2020/04/16/how-bad-can-it-be-the-relationship-between-gdp-growth-and-the-unemployment-rate. Accessed 10 December 2022.

Dolls, Mathias. “Automatic stabilization and discretionary fiscal policy in the financial crisis - IZA Journal of Labor Policy.” *IZA Journal of Labor Policy*, 6 November 2012, https://izajolp.springeropen.com/articles/10.1186/2193-9004-1-4. Accessed 4 September 2022.

Europa.eu. *2.22. DENMARK*. 2020. *ec.europa.eu*, https://ec.europa.eu/economy_finance/forecasts/2020/summer/ecfin_forecast_summer_2020_dk_en.pdf.

The European Commission, and The European Union. “Households´ income and the cushioning effect of fiscal policy measures in the Great Lockdown.” 2020, p. 2, https://joint-research-centre.ec.europa.eu/system/files/2020-07/jrc121228_policy_brief-_the_impact_of_the_great_lockdown_on_hh_25_06_2020_1.pdf. Accessed 8 September 2022.

eurostat. “Impact of COVID-19 on household consumption and savings.” *European Commission*, 2 February 2021, https://ec.europa.eu/eurostat/web/products-eurostat-news/-/ddn-20210202-1. Accessed 10 September 2022.

Gravelle, Jane G. “Overview of the Federal Tax System in 2022.” *FAS Project on Government Secrecy*, 8 June 2022, https://sgp.fas.org/crs/misc/R45145.pdf. Accessed 8 September 2022.

Investopedia. “Automatic Stabilizer Definition - Economy.” *Investopedia*, 2021, https://www.investopedia.com/terms/a/automaticstabilizer.asp. Accessed 4 September 2022.

Kaushal, Neeraj. “View: India needs one-year programmes that can boost demand during cyclical slowdowns.” *The Economic Times*, 22 October 2019, https://economictimes.indiatimes.com/news/economy/policy/view-india-needs-one-year-programmes-that-can-boost-demand-during-cyclical-slowdowns/articleshow/71695162.cms?from=md. Accessed 4 September 2022.

Kokkinos, Patricia. “GCSE Revision: Economic change | Geography is easy.” *Geography is easy*, 19 February 2014, https://geographyiseasy.wordpress.com/2014/02/19/gcse-revision-economic-change/. Accessed 10 December 2022.

Linton, Michael IA. “Denmark | History, Geography, & Culture | Britannica.” *Encyclopedia Britannica*, 30 August 2022, https://www.britannica.com/place/Denmark. Accessed 4 September 2022.

Lončar, Zorica. “Cost of living in Denmark: Your guide - Wise, formerly TransferWise.” *Wise*, 2 October 2020, https://wise.com/gb/blog/cost-of-living-in-denmark. Accessed 10 September 2022.

Marinov, Marin A. “BSR Policy Briefing 5/2020.” *Centrum Balticum*, 20 November 2020, https://www.centrumbalticum.org/en/publications/databank/databank/bsr_policy_briefing_5_2020.6333.news. Accessed 10 September 2022.

Moody's Analytics. “Denmark Total Employment | Moody's Analytics.” *Economy.com*, https://www.economy.com/denmark/total-employment. Accessed 4 September 2022.

OECD. “How demanding are activation requirements for jobseekers?” *OECD*, 2020, https://www.oecd.org/social/strictness-benefit-eligibility.htm. Accessed 8 September 2022.

O'Neill, Aaron. "• Denmark - national debt 2017-2027." *Statista*, 2021, https://www.statista.com/statistics/531482/national-debt-of-denmark/. Accessed 10 September 2022.

Pettinger, Tejvan. "Tertiary - Service sector of the economy." *Economics Help*, 19 October 2017, https://www.economicshelp.org/tertiary-service-sector/. Accessed 4 September 2022.

Sheiner, Louise, and Michael Ng. "What are automatic stabilizers?" *Brookings Institution*, 2 July 2019, https://www.brookings.edu/blog/up-front/2019/07/02/what-are-automatic-stabilizers/. Accessed 4 September 2022.

Statista. "• Denmark: unemployment rate 2022." *Statista*, 23 May 2022, https://www.statista.com/statistics/586271/monthly-unemployment-rate-in-denmark/. Accessed 4 September 2022.

Statista. "• U.S.: personal saving rate monthly 2022." *Statista*, 5 August 2022, https://www.statista.com/statistics/246268/personal-savings-rate-in-the-united-states-by-month/. Accessed 8 September 2022.

Thévenot, Celine. "Inequality in OECD countries." *Scandinavian Journal of Public Health*, vol. 45, no. 18, 2017, pp. 9-16. *Sage Journals*, https://journals.sagepub.com/doi/full/10.1177/1403494817713108#:~:text=The%20Gini%20coefficient%2C%20a%20common,and%20Mexico%20(Figure%201). Accessed 4 September 2022.

Trading Economics. "Denmark Inflation Rate - August 2022 Data - 1981-2021 Historical - September Forecast." *Trading Economics*, 2022, https://tradingeconomics.com/denmark/inflation-cpi. Accessed 8 September 2022.

United Nations Population Division. “Urban population (% of total population) - Denmark | Data.” *World Bank Open Data*, 2021, https://data.worldbank.org/indicator/SP.URB.TOTL.IN.ZS?locations=DK. Accessed 4 September 2022.

The White House. “The U.S. Economy and the Global Pandemic.” *The White House*, 23 April 2022, https://www.whitehouse.gov/wp-content/uploads/2022/04/Chapter-3-new.pdf. Accessed 10 September 2022.

The World Bank. “GDP per capita(current US$) - Denmark.” *The World Bank*, 2021, https://data.worldbank.org/indicator/NY.GDP.PCAP.CD?locations=DK. Accessed 10 September 2022.

The World Bank. “Gross domestic savings (% of GDP) - Denmark | Data.” *World Bank Open Data*, 2021, https://data.worldbank.org/indicator/NY.GDS.TOTL.ZS?locations=DK. Accessed 8 September 2022.

The World Bank. “Labor legislation: Denmark.” *World Bank Group*, https://web.worldbank.org/archive/website01419/WEB/IMAGES/11_DENMA.PDF. Accessed 8 September 2022.

The World Bank. “Tax revenue (% of GDP) - Denmark | Data.” *World Bank Data*, 2020, https://data.worldbank.org/indicator/GC.TAX.TOTL.GD.ZS?locations=DK. Accessed 10 September 2022.

6. “To what extent is the knowledge we produce determined by the methodologies we use? Discuss with reference to history and one other area of knowledge”

As I wrote my Extended Essay in economics, I relied on a plethora of methodologies to acquire my data, mixing economic theory with real-life examples through interviewing and acquiring government data. I was intrigued to find varying information. While classical economic assumptions claimed markets are self-correcting, the 2008 financial crisis proved else wise, requiring government bailouts. Had the U.S. government relied on the theoretical model, its economy would have further plunged into recession. The varying answers from different economic methodologies encouraged me to explore *"the extent to which the knowledge produced is determined by the methodologies we use."*

A methodology is *"a body of methods, rules, postulates employed by a discipline"* (Merriam-Webster), or in this case, an area of knowledge. Knowledge is *"facts, information, and skills acquired through theoretical or practical understanding of a subject"* (Oxford Languages), or in this case, skills acquired through methodologies. I plan to focus specifically on the knowledge that methodologies produce in the areas of history and mathematics. I will compare the use of methodologies depending on the context in which they are used, as well as compare multiple methodologies from different areas of the subject and evaluate whether a similar outcome is produced.

History uses a multitude of research methodologies to understand the past; a key methodology is the study of records, or codicology. Reading writings from different contexts about a particular period can reveal contrasting perspectives on an event, prompting various historical records. This was especially true during the Pearl Harbor Bombing, a Japanese surprise attack on a United States naval base during World War II. Different accounts of the motivations behind the attack, influenced by varying contexts, led to different opinions in respective countries. Studying civilian perspectives in writings from the United States revealed Pearl Harbor as a "sneak" attack that was unjustified and immoral. Japanese perspectives, however, revealed a different outlook on the attack. They highlight their country's economic struggles as the U.S. tried to restrict Japan's oil trade. This is proven by the writings of a middle school Japanese student, who mentions, "*The Americans, British, Chinese, and Dutch; they wouldn't give us a drop of oil!"* (Texas A&M University) The Japanese hence considered the attack to be a justified response. By reading writings from two different contexts, historians have uncovered contrasting perspectives, leading to multiple conclusions about the motives behind the Pearl Harbor Attack. This shows how geographical, political, and cultural contexts can influence the knowledge produced by a particular methodology.

However, while differing historical methodologies produce different knowledge, they may also produce some common knowledge. This is true with the two different historical approaches used to explain Adolf Hitler's rise to power during World War Two: The Sonderweg and the Revisionist Schools.

The Sonderweg school arose from American historian William Shirer, who witnessed the nationalist and authoritarian views in Nazi Germany from first-hand experience. The culmination of these attitudes led him to consider the rise of Nazism (a movement focused on these same ideals) inevitable and a logical phase in Germany's national development. While the Sonderweg school relied on first-hand experience, Revisionist historians used "dispassionate objectivity," leading to a different perspective on the rise of Hitler. Historians such as John Toland viewed Hitler in a balanced view, focusing not only on his negative attributes but also on positive ones. In his book, "Hitler's War," Toland considers Hitler a product of his context. His experiences allowed him to become an adept politician, in which he could exert his influence to become a dictator, aided by the unraveling economic situation in Germany (Llewellyn).

Thus, while the different schools gave different reasons for the rise of Nazism, both concluded that Hitler's rise to power and his ideology would be inevitable. This suggests that regardless of the methodology used, the knowledge produced regarding whether Hitler would take power would have been the same.

It is important to note that codicology, however, may also produce different outcomes due to a high potential for human error. Considering the vast swaths of time that have passed since events, certain historical accounts may be lost due to issues relating to translation and revisionism. It is also possible for inaccuracies to be caused due to civilian perspectives, as historians may consider them representative of the entire community.

Yet, while history may produce varying outcomes due to different methodologies to a large extent, the same may not be the case with areas of knowledge such as mathematics. In this area of knowledge, multiple solving methodologies can be used to arrive at the same conclusion.

Consider the varying methods within algebra through which one can find the roots of a quadratic equation: the formula and factorization method. Each method has a different approach to obtaining the answer. The formula method relies on a set formula, given by $\frac{-b\pm\sqrt{b^2-4ac}}{2a}$. Here, the coefficients of a

standard quadratic equation $ax^2 + bx + c$ are substituted to obtain the roots. Similarly, the quadratic equation $x^2 + 6x + 8$, when substituted in the formula, would give roots of -4, and -2.

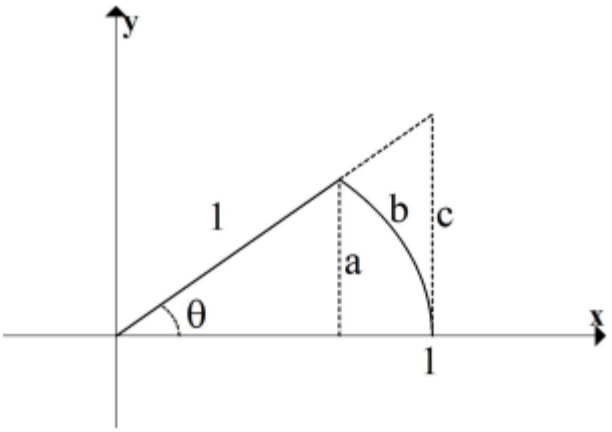
Consider now the 'factorization' method, in which the equation is represented in the form of its factors of the equation, with each being equated to 0 to find the root. Using the same example from above, $x^2 + 6x + 8$ in factored form would simply be $(x + 4)(x + 2) = 0, with\, x + 4 = 0, x + 2 = 0$. Thus, the roots would still be -4, and -2.

While these methodologies were part of algebra, methodologies from other areas of math can also be used to produce the same knowledge. Consider the different areas of calculus and algebra that have methodologies that can find the gradient of a standard line $y = mx + b$. While it is evident that 'm' is the gradient of the line, they are found through different methods. In linear algebra, it is found by dividing the difference between two vertical points by the difference between two horizontal points of the line ($\frac{y_2 - y_1}{x_2 - x_1}$). For example, the line $y = 2x + 5$ has coordinates (1, 7) and (3, 11). The gradient would hence be (11-7)/(3-1) = 2

In differential calculus this is found by using the differentiation principles (i.e. $\frac{dy}{dx} = x^n = nx^{n-1}$ with constant values becoming 0). Thus, for the line $y = mx + b$, the slope would be $\frac{dy}{dx} = mx + b = mx^0 = m$. Using the example above of $y = 2x + 5$, the slope would be $\frac{dy}{dx} = 2x + 5 = 2x^0 = 2$. Thus, despite using methodologies from different mathematical branches, the knowledge produced is the same.

A reason why mathematics may have the same outcome in most scenarios is because it is a human-created subject and relies on a set of axioms developed by humans themselves. This leaves little scope for inaccuracy, compared to fields such as history, which study unexpected actions. There are areas in math, however, that humans themselves are trying to form conclusions on, and this has led to varying outcomes.

Consider the field of limits, and the case of $\lim_{\theta \to 0} \frac{sin(\theta)}{\theta}$. A potential answer to this could be $\frac{0}{0}$, however, this would be undefined in mathematics. Thus, mathematicians use a different method, one involving the unit circle to obtain a finite value to the problem. To use this method, we draw a diagram of a partial unit circle, with two right angled triangles.



.

Figure 1: Diagram of partial unit circle

Using trigonometric ratios, we find that $sin(\theta) = \frac{a}{1} = a$. We also find that $tan(\theta) = \frac{c}{1} = c$ We can also use the arc length formula ($s = r * \theta$) to find b. $b = r * \theta = 1 * \theta = \theta$.

Through this, we can observe $a < b$ (since the sector length of a circle is greater than the radius in this case). This indicates that $sin(\theta) < \theta$.

We also observe that $b < c$, indicating that $\theta < tan(\theta)$ and eventually $cos(\theta) < \frac{sin(\theta)}{\theta}$.

Therefore: $cos(\theta) < \frac{sin(\theta)}{\theta} < 1$

Since both sides of the inequality approach to 1 as θ nears 0, it can be proven that $\lim_{\theta \to 0} \frac{sin(\theta)}{\theta} = 1$ (University of Washington).

Thus, it can be observed that while there are instances where different methodologies are used and the same outcome is obtained, there are also contrary situations.

It is evident that different methodologies can produce different knowledge, however the extent depends on the area of knowledge. Quantitative areas of knowledge such as mathematics are able to reach the same conclusion to a large extent despite different methodologies. However, there are still instances where different knowledge is produced, in areas still trying to be defined by mathematicians. Knowledge production in history, on the other hand, has a larger scope for producing varying outcomes. Unlike mathematics, in history, it is possible for different outcomes to be produced even when the same methodology is used. This is due to context, which can influence historical accounts despite the data

collection method being the same. It is interesting to note that, similar to mathematics, there were instances where the thought process was different, but the conclusion was the same, as in the explanations behind Hitler's rise to power. I believe the importance of varying outcomes due to methodology also depends on the area of knowledge being considered. For social sciences and humanities, multiple perspectives provide insights into different human behaviors, which I believe is important when learning from past mistakes that have led to conflict; multiple methodologies prove important in these subjects. Quantitative areas on the other hand, would not benefit from these approaches, as they are meant to measure a specific value that is intended to provide order to human functioning.

Word Count: 1594

Works Cited

Methods for Solving Quadratic Equations, https://www.uww.edu/documents/rock/academics/quadratic_equations_methods.pdf. Accessed 11 November 2022.

Llewellyn, Jennifer. “The historiography of Nazi Germany.” *Alpha History*, 30 July 2020, https://alphahistory.com/nazigermany/historiography-of-nazi-germany/. Accessed 11 November 2022.

Merriam-Webster. “Methodology Definition & Meaning.” *Merriam-Webster*, 1 November 2022, https://www.merriam-webster.com/dictionary/methodology. Accessed 11 November 2022.

Oxford Languages. “Our Dictionaries | Oxford Languages.” *Oxford Languages*, https://languages.oup.com/dictionaries/. Accessed 11 November 2022.

Texas A&M University. “Pearl Harbor: "Japanese vs. American Civilian Perspectives" · Narratives of World War II in the Pacific · Bell Library Exhibits.” *library.tamucc.edu*, https://library.tamucc.edu/exhibits/s/hist4350/page/pearl-harbor-japanese-vs-american-perspective. Accessed 11 November 2022.

University of Washington. “Outline of Proof of lim sin(θ) θ = 1 When we study slopes of trig functions, the limit lim sin(θ) θ often appears. My goal i.” https://sites.math.washington.edu/~aloveles/Math124Fall2017/m124%20Trig%20Limit.pdf. Accessed 11 November 2022.

Commentary 1

Title of the article	Can Milan Become Europe's Most Bike-Friendly City?
Source of the article	Bloomberg City Lab https://www.bloomberg.com/news/articles/2022-01-14/milan-plans-bike-lane-infrastructure-to-rival-paris
Date the article was published	January 14th, 2022
Date Commentary was Written	February 23rd, 2022
Word count (800 words maximum)	794 words
Key concept	Intervention
Student Code	kjt140
Section of the syllabus the article relates to	**Section 1: Microeconomics** Section 2: Macroeconomics Section 3: International Trade

Article

By

Feargus O'Sullivan

January 14, 2022 at 11:30 AM GMT+5:30

Within 15 years, Milan should have one of the most comprehensive networks of protected bicycle lanes in all of Europe.

When complete in 2035, the network — approved by the Metropolitan City of Milan in November 2021 and due to deliver its first major cycle highways by this summer — will provide Italy's most populous metro area with 750 kilometers (466 miles) of segregated lanes. Dubbed the *Cambio Biciplan* (the "Change Bike Plan") the 250 million-euro ($285 million) project's target exceeds even the 680 kilometers of tracks planned for Europe's current trailblazer for grand scale bike infrastructure, Paris and its surrounding metro area.

Reaching from Milan's core far out into the surrounding countryside, the ambition is to make cycling the first and easiest choice for getting around the Metropolitan City of Milan — a district that includes both the city proper, its suburbs and some of its immediate rural hinterland. If the plan seems grand in scale, so are the problems it seeks to tackle. The region surrounding Milan has some of Europe's worst pollution, created by a combination of dense population, large-scale industrial activity and widespread car dependency. The emissions created become especially harmful in winter, when temperature inversions commonly trap pollutants in the lower atmosphere, leaving a toxic blanket of smog cloaking the city.

Only neighboring Turin has exceeded Milan's poor national record for pollution in recent years, leading to a legal ruling against the Region of Lombardy by the European Court of Justice, in which Milan is located, in 2020. According to Milan's own research, 50% of the city's PM10 and nitrous oxide pollution comes from transportation emissions, meaning that Milan's pollution problem is substantially a car and truck problem.

The city has already started a battle to clear the toll that vehicle emissions takes on its air. It has had a congestion charge in place in the city center since 2008, and has banned diesel vehicles (in classes Euro 1-4) from most of the city since 2014. In 2020, it brought in emergency short-term driving bans enforced with fines during periods of especially acute pollution.

Then with the pandemic's onset, it converted car lanes into bike tracks and pedestrianized squares to create 36 "tactical plazas" designed to facilitate outdoor social life, repurposing 35 kilometers of road previously used for motor traffic. These have made central Milan easier to cycle but have remained only a partial, makeshift solution. Provisional bike- and pedestrian-friendly interventions in the city's street plan are now looking a little shabby and bike lanes often abruptly re-integrate with heavy traffic as they leave the city center.

The new network seeks to make sure that the kind of benefits created for downtown cyclists since the arrival of Covid become accessible to all of Milan's residents. It will seek to ensure that 80% of Milan's homes are within one kilometer's distance of a fully protected axial cycle route, making it possible for residents to conduct almost all of their daily business on two wheels. This would be a paradigm shift in a city where heavy motor traffic still makes using shared streets unsafe and unappealing for bike riders.

This attempt at full coverage is clear from the proposed network plan. Resembling a spider's web, the network will be organized around five concentric bicycle beltways emanating out from the city core — a reflection of a street plan initially laid out around concentric canals, most of which are now buried.

These beltways will be crossed by 16 spoke-like tracks connecting the city's heart with the periphery, some of which will be classified as "super-fast" — in other words, tracks for swift commuting with as many obstacles and bottlenecks as possible removed.

By intersecting spoke tracks with the beltways, Milan should avoid a classic blind spot for cycle networks, and indeed for all public transit. Too often, cities provide decent routes in and out of a city but fail to connect outer neighborhoods with each other, except through downtown routes. Moreover, the new paths will extend far beyond Milan's built-up area (and the current reach of the city's metro system) reaching out through suburban towns and farmland before reaching the network's final feature — a ring of "greenways" for leisure use that connects woodland and nature reserves.

The city says that in addition to cutting pollution, the plan will improve access to the city core for people with lower incomes, many of whom live in a donut-shaped circle of neighborhoods on Milan's fringes. Another aim is safer streets. Not only will cyclists get better protection, increasing the volume of bikes on the roads increases their visibility and thus shifts attitudes toward greater awareness and respect for their presence.

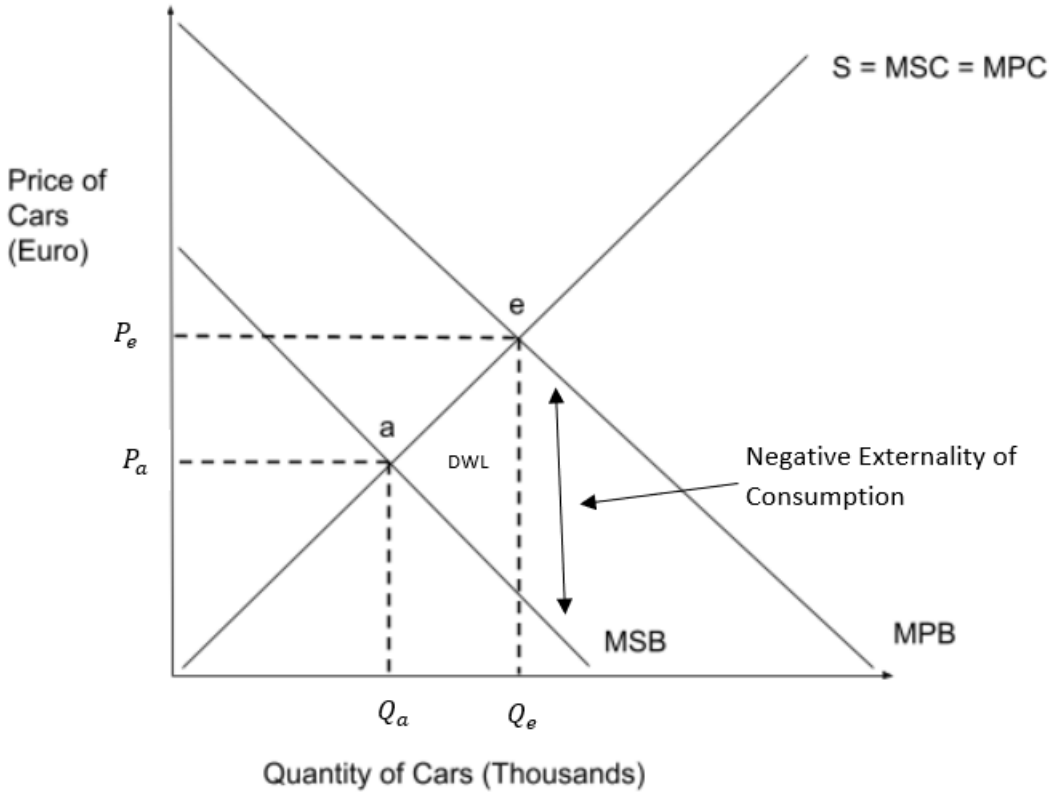
These promised benefits don't necessarily mean that the city's plans will go ahead without upset. At the heart of one of Europe's most industrially productive areas, Milan has a strong attachment to car culture in some sectors. As the home of automobile manufacturer Alfa Romeo, it's perhaps not surprising that one right-wing politician has accused the city of acting like the "Taliban" in its assault on cars. But as Europe's cities move increasingly in unison towards more comprehensive bike infrastructure — Milan cites growing networks in Paris, Berlin, Lyon and Toulouse as role models — the city's plans look like the shape of things to come.

Commentary

Since 2008, Milan's parliament has introduced legislation to restrict car consumption and encourage cycling. These **interventions** have been met with some resistance, with a right-wing politician calling a recent cycling lane legislation an assault on cars; this is expected given the high cultural influence of cars on the city.

The excess of cars has led to a market failure, causing negative externalities of consumption. Milan also experiences a positive externality of consumption, as while bicycle usage is beneficial, they are under consumed. Milan's legislations aim to combat these allocative inefficiencies.

Figure 1: Market for cars in Milan (after 2014)



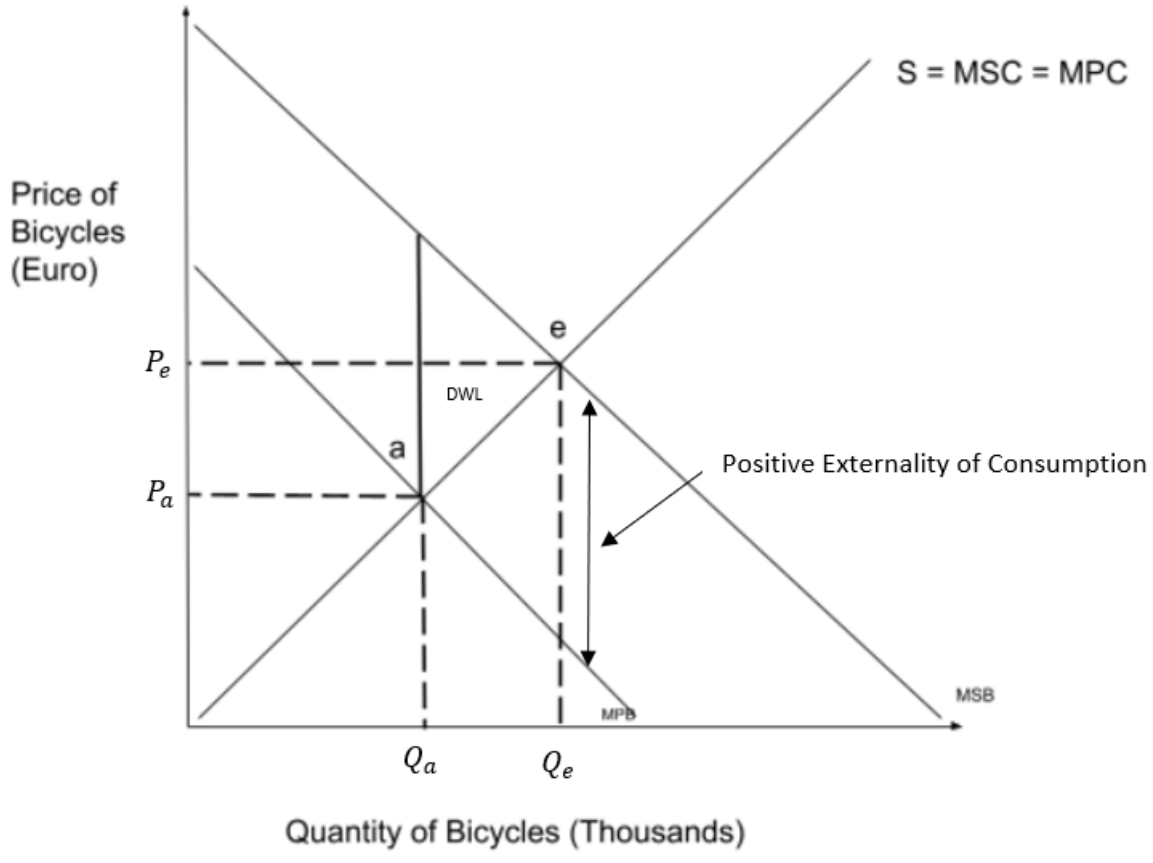
The above graph shows the market for cars after 2014 in Milan. The Marginal Private Benefit (MPB) is greater than Marginal Social Benefit (MSB), indicating allocative inefficiency. At free market

equilibrium, the market operates at point ‘e’. It creates a deadweight loss (illustrated as ‘DWL’), producing at $Q_e > Q_a$ units of cars, at price $P_e > P_a$. As mentioned in the article, Milan has a strong car culture, and is also home to a large car manufacturer, providing thousands of jobs, further enforcing dependence, and requiring **intervention** by the government.

In 2008, Milan introduced a congestion charge, and in 2014, banned certain diesel vehicles. While these **interventions** have reduced the negative externality of consumption of cars, they have still not internalized them, and the market still has a deadweight loss. Hence, even after 2014, Milan still has some of the worst pollution in Europe, with 50% coming from cars and trucks.

The government is hence now shifting its focus to bicycles, to combat the positive externality of consumption, in turn bringing about benefits to residents.

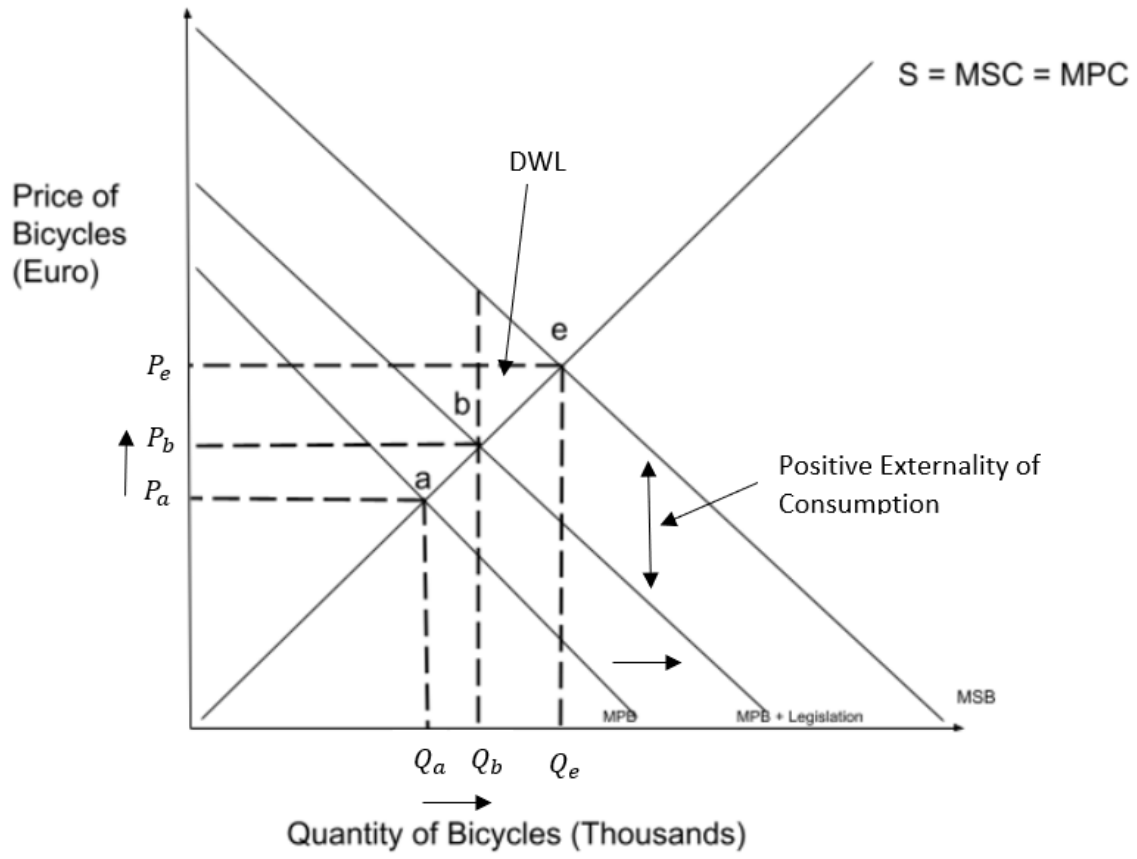
Figure 2: Market for bicycles in Milan (before 2021)



The above graph shows the market for bicycles in Milan before 2021. The market experiences a positive externality of consumption, with MSB being greater than MPB. At free market equilibrium, the market operates at point 'a', at $Q_a < Q_e$ units at price $P_a < P_e$. This has created a deadweight loss, illustrated by DWL. Despite the widespread benefits of safety, pollution reduction, and cost-efficiency, the market under-consumes, likely due to a lack of adequate infrastructure. Hence, the government is **intervening** to eliminate this market failure, by providing a complementary good: bicycle lanes.

As mentioned in the article, the **intervention** to implement bicycle lanes will make streets safer, connect residents to nature reserves, increasing well-being, and reduce pollution and noise. All of these benefits will reduce the positive externality of consumption.

Figure 3: Market for Bicycles in Milan (after 2021)



The above graph shows the market for bicycles after 2021, after the government has implemented a cycling lane legislation. We see that due to this **intervention** the MPB has shifted closer to MSB, to MPB + Legislation. The market now operates at point 'b', producing $Q_b > Q_a$ units at price $P_b > P_a$. The deadweight loss (illustrated as DWL) has thus been reduced. However, the externality has still not been internalized.

Each **intervention** has its advantages and disadvantages. On a positive note, they have reduced the negative externality of consumption of cars. The congestion charges have encouraged consumers to utilize other forms of transport, and the government gains revenue from those continuing to use vehicles. The revenue can be used to further invest in clean travel, as is being done with the bike lanes.

The **interventions** have also increased positive externality of consumption of bicycles. Milan will now be safer, less polluted, and more equitable as it provides a cheap and clean alternative to cars. The cycling lane legislation, is reducing traffic in the city centre. It is also leading to lower accidents with non-motorists, and further reducing noise pollution. The lanes are also allowing lower-income consumers to travel to the city centre with ease, spurring the economy, and are connecting neighbourhoods that were once separated by wide roads, reducing inequality.

The **interventions** also have their flaws. The congestion charging scheme and diesel ban exclude lower income residents who can't afford to pay a higher price, hindering their mobility. In the short-run, many consumers may not have adequate transport until the cycling lanes are finished being constructed. Milan is also home to one of the largest car manufacturers, Alfa Romeo. The **interventions** reduce car consumption, which can lead to mass unemployment, forcing the government to consider the opportunity cost of unemployment insurance in the short run. The construction of the cycling lanes cost $285 million, a hefty investment that could be used to invest in other merit goods, such as education and healthcare.

The concept of **intervention** is highly relevant in this situation. Without it, both markets will continue to operate at an allocatively inefficient level. Government **interventions** have brought both, the market for cars, and the market for bicycles, closer to the allocatively efficient level, making Milan safer, and cleaner.

Commentary 2

Title of the article	Fed hikes interest rates by three-quarters of a percentage point in boldest move since 1994
Source of the article	Cable News Network (CNN) Business https://edition.cnn.com/2022/06/15/economy/fed-rate-hike-decision-june/index.html
Date the article was published	June 15th, 2022
Date Commentary was Written	September 25th, 2022
Word count (800 words maximum)	798 words
Key concept	Economic Well-being
Student Code	kjt140
Section of the syllabus the article relates to	Section 1: Microeconomics **Section 2: Macroeconomics** Section 3: International Trade

Article

Washington, DC (CNN): The Federal Reserve raised interest rates by three-quarters of a percentage point on Wednesday in an aggressive move to tackle white-hot inflation that is plaguing the economy, frustrating consumers, and stifling the Biden administration.

It's the largest rate hike since 1994, and will affect millions of American businesses and households, pushing up the cost of borrowing for homes, cars and other loans in order to force a slowdown in the economy.

Until this week, economists and investors had expected the Fed to raise its benchmark interest rate by half a point, the second such move in the last 22 years. However, after a disastrous inflation report on Friday revealed that price hikes are broadening across the entire economy, expectations rose for a more dramatic rate hike.

In a post-meeting news conference, Federal Reserve Chairman Jerome Powell acknowledged that a three-quarter-point rate hike "is an unusually large one."

While Fed officials "do not expect moves of this size to be common," Powell said, he noted that the central bank would likely discuss raising rates by 75 basis points or just 50 basis points at its next meeting, on July 26-27.

The US stock market rallied after the announcement, with the Dow up more than 500 points as investors interpreted the central bank's actions as a solid commitment to bringing down inflation.

"It was what everyone was expecting," said Jeffrey Frankel, co-president at Stuart Frankel & Co. "What Powell is saying after the announcement, that he will be flexible about future rate hikes, the market seems happy at the moment."

Americans are struggling with rising costs from the grocery store to the gas pump and the Fed is mandated with the task of keeping prices stable. Surging prices on everything from food to gas -- which has hit a series of daily record highs in the past month -- have led to the lowest consumer sentiment since 1952.

According to a statement released Wednesday by the Federal Open Market Committee, the voting arm of the central bank, 10 out of 11 voting members were in favor of the hike. One lone dissenter, Kansas City Fed President Esther George, voted for a half-percentage-point hike.

The committee said in its statement it was "strongly committed to returning inflation to its 2% objective." This is likely intended as a response to criticism that the Fed has been behind the curve in tackling the nation's inflation problem. It also indicates that more aggressive hikes are not off the table. The central bank also decreased its economic projections for 2022, stoking recessionary fears: The Fed's median GDP forecast for 2022 is now 1.7%, down significantly from 2.8% in March.

Notably, the Fed does not predict a decrease in inflation this year and sees unemployment rising to 3.7% in 2022, higher than its March prediction.

When the pandemic first hit the United States, the Fed rolled out a series of emergency measures to support the economy, including slashing its interest rate to zero, making it almost free to borrow money. But while that "easy money" policy encouraged spending by households and businesses, it also fed inflation and contributed to today's overheated economy.

Now that the economy no longer needs support from the Fed, the central bank has been taking steps to "remove the punch bowl" and slow down the economy by aggressively hiking interest rates.

The Fed's actions will increase the rate that banks charge each other for overnight borrowing to 1.5%-1.75%, the highest since before the pandemic hit the United States.

The rate hike is not entirely unexpected: Some major banks, including Barclays, Jefferies, Goldman Sachs and JPMorgan, all expected the Fed to increase its rate by 75 basis points, or three-quarters of a percentage point.

The central bank announcement came at the conclusion of its two-day policymaking meeting.

Commentary

As per the article, in June of 2022, the U.S. Federal Reserve raised interest rates to 1.5-1.75. This contractionary monetary policy measure was intended to curb the inflationary spiral, or 'remove the punch bowl' to increase **economic well-being** by reducing prices and improving consumer sentiment. This move was anticipated, considering the high rates above the stable inflation rate. However, such a drastic hike was 'unusual', as the chairman of the Federal Reserve mentioned.

Inflationary Spiral in the U.S. Economy due to Pandemic Restrictions Easing

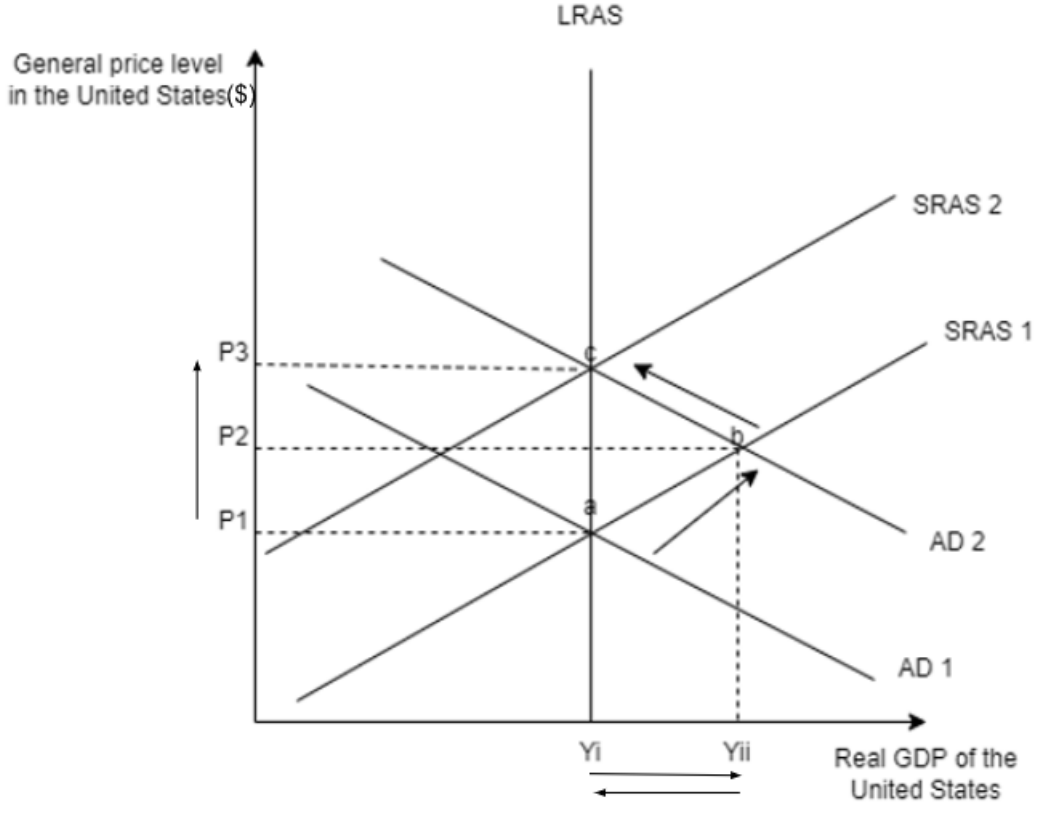
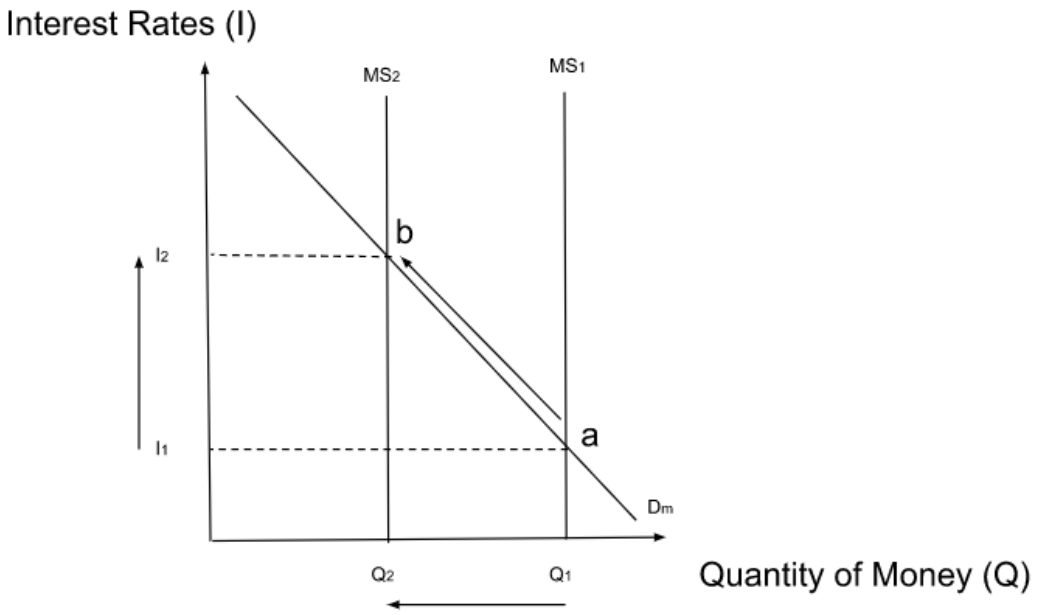


Figure 1 shows the general price level in the U.S. against its real GDP. It depicts the inflationary spiral taking place before the interest rates hike, due to the easing of Pandemic restrictions. Currently, the U.S. economy produces at point ‘a’ (Yi, P1) where AD1 equals SRAS1. Due to the “easy money” policy during the Pandemic, where the federal government made it easy to borrow, households could borrow more money. This potentially led to an increased marginal propensity to consume, leading to increased consumer expenditure. Thus, there is an increase in Aggregate Demand from AD1 to AD2. The economy now produces at point ‘b’, leading to price level P2>P1, and a real GDP value of Yii>Yi.

However, the economy is producing beyond its potential output (depicted by the LRAS curve at Yi). As prices rise, real income will decrease, leading to decreased **economic well-being**. This results in the aggregate supply shifting from SRAS1 to SRAS2, as the costs of factors of production have increased, potentially causing stagflation. The economy now operates at point ‘c’ (Yi, P3), at price level P3>P2>P1 but original real GDP value Yi<Yii. The economy now operates at a higher general price level, with the original output level. This spiral continues, creating inflationary gaps and rising prices, further reducing **economic well-being** due to an inability to purchase necessities.

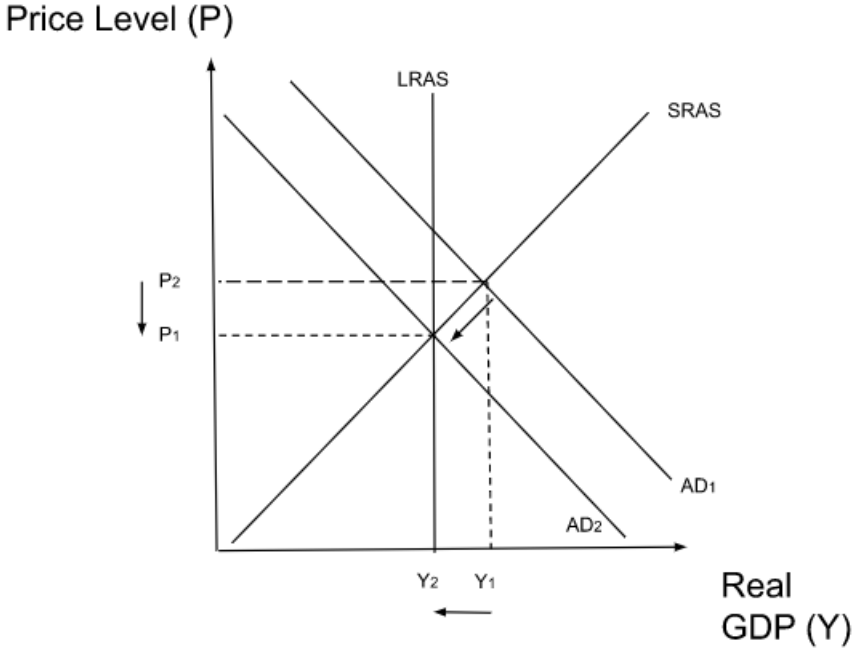
Thus, contractionary monetary policy is required. The Federal Reserve ‘raised interest rates by three-quarters of a percentage point’, in order to tackle this ‘white-hot inflation’, as the article mentions, and meet the macroeconomic objective of low and stable rate of inflation. This affects the money supply.

Interest Rates vs Quantity of Money Demanded in the U.S., June 2022



Prior to the hike, the economy was at point 'a' (Q_1, I_1). Through selling bonds, increasing the reserve ratio or the lending rate, the central bank has decreased the money supply in the economy from MS_1 to MS_2 in an attempt to reach a stable rate of inflation at 2%. Due to this decrease, the interest rates will rise from I_1 to I_2. The quantity of money will decrease to $Q_2 < Q_1$, as consumers will save more in banks due to higher returns. Hence, the marginal propensity to save will increase, leading to a reduction in aggregate demand.

Price Levels vs Real GDP in the U.S. After Hike



The above graph shows the U.S economy after the effects of the interest rate hike have taken effect, potentially in 2023 as the article mentioned. Currently, the economy produces at point 'a' ($Y_1 > Y_2\ output\ at\ price\ P_2 > P_1$), below the natural rate of unemployment. Due to the decreased money supply, interest rates have increased. This can cause reductions in consumer expenditure and investment, eventually reducing aggregate demand from $AD_1\ to\ AD_2$. The economy now meets full employment and point 'b'. The economy now produces $Y_2\ output\ at\ price\ P_1$. Thus, the inflationary gap has been addressed, increasing **economic-wellbeing** for many by increasing the purchasing power of money.

However, as the article mentions, there are consequences. By shifting the AD to the left, expenditure and investment has decreased. Thus, firms are less inclined to retain labour that they may not need anymore, leading to a rise in unemployment rates. This can lead to social problems such as poverty and crime as well as increase debt. As per the article, there would be a 3.7% increase in unemployment. Furthermore, the decrease in output will result in a decrease in GDP growth, estimated to be "1.7%, down from 2.8% in March".

Thus, this measure has its advantages and disadvantages. The bank addresses inflationary gaps independently, allowing it to increase **economic well-being** in areas of the economy most needed. However, it is important to consider "time-lags"; inflation will continue to skyrocket until the effects of the Federal Reserve occur. This will lead to a drastic decrease in **economic well-being** until prices eventually reduce. Also, the government may have to increase transfer payments due to the rising unemployment rate, potentially causing budget deficits. Lastly, the policy may lead to a deflationary gap if the hike proves too high, "stoking fears of a recession".

Economic well-being is highly important in this situation. Without adequate contractionary policy, households may experience income shocks, leading to debt and social issues such as poverty and crime. However, with contractionary policy, unemployment could potentially occur. The central bank thus

plays a vital role in making necessities more accessible, and preventing decreases in real income, increasing **economic well-being**, but should be cognizant of the disadvantages.

Commentary 3

Title of the article	India raises tax on imported cars, motorbikes, including EVs
Source of the article	Reuters.com https://www.reuters.com/business/autos-transportation/india-raises-tax-imported-cars-motorbikes-including-evs-2023-02-01/
Date the article was published	February 1st, 2023
Date Commentary was Written	February 3rd, 2023
Word count (800 words maximum)	800 words
Key concept	Choice
Student Code	kjt140
Section of the syllabus the article relates to	Section 1: Microeconomics Section 2: Macroeconomics **Section 3: International Trade**

Article

NEW DELHI, Feb 1 (Reuters) - India on Wednesday said it will raise taxes on imported cars and motorbikes, including electric vehicles (EVs), as it seeks to boost local manufacturing in line with Prime Minister Narendra Modi's "Make in India" campaign ahead of elections in 2024.

All vehicles with a landed cost of less than $40,000 will be taxed at 70%, up from 60% earlier, a move analysts say could impact demand. The landed cost includes the vehicle's price tag plus insurance and freight costs.

Import tax on all semi-knocked-down cars - where major parts are imported separately and the final vehicle is assembled in the country - will be raised to 35% from 30%.

The tax hikes, part of the government's federal budget which was presented on Wednesday, come into effect from April 1.

Taxes on automobiles in India, the world's third-largest car market, rank among the highest globally, drawing criticism from companies such as Tesla Inc (TSLA.O) which shelved plans last year to enter the market because of high tariffs.

Carmakers including South Korea's Hyundai Motor (005380.KS) and Kia Corp (000270.KS), Germany's Volkswagen AG (VOWG_p.DE), Skoda Auto and Japan's Toyota Motor Corp (7203.T) sell imported cars in the country, with some also selling electric models.

India's EV market is small - making up about 1% of total car sales in the country in 2022 - but growth has been rapid, with domestic carmakers Tata Motors (TAMO.NS) and Mahindra & Mahindra (MAHM.NS) as well as global rivals BYD (002594.SZ) and SAIC's (600104.SS) MG Motor lining up new launches.

Global carmakers who are banking on buoyant consumer demand in the premium and luxury space will be affected, Rajat Mahajan, partner at Deloitte India said, adding that some companies may absorb this cost to benefit from the recent surge in demand.

India has seen strong rebound in car sales, which rose 25% to 3.8 million in 2022. Growth in premium and luxury cars was higher, with carmakers like Mercedes-Benz (MBGn.DE) and BMW (BMWG.DE) reporting record sales in the country.

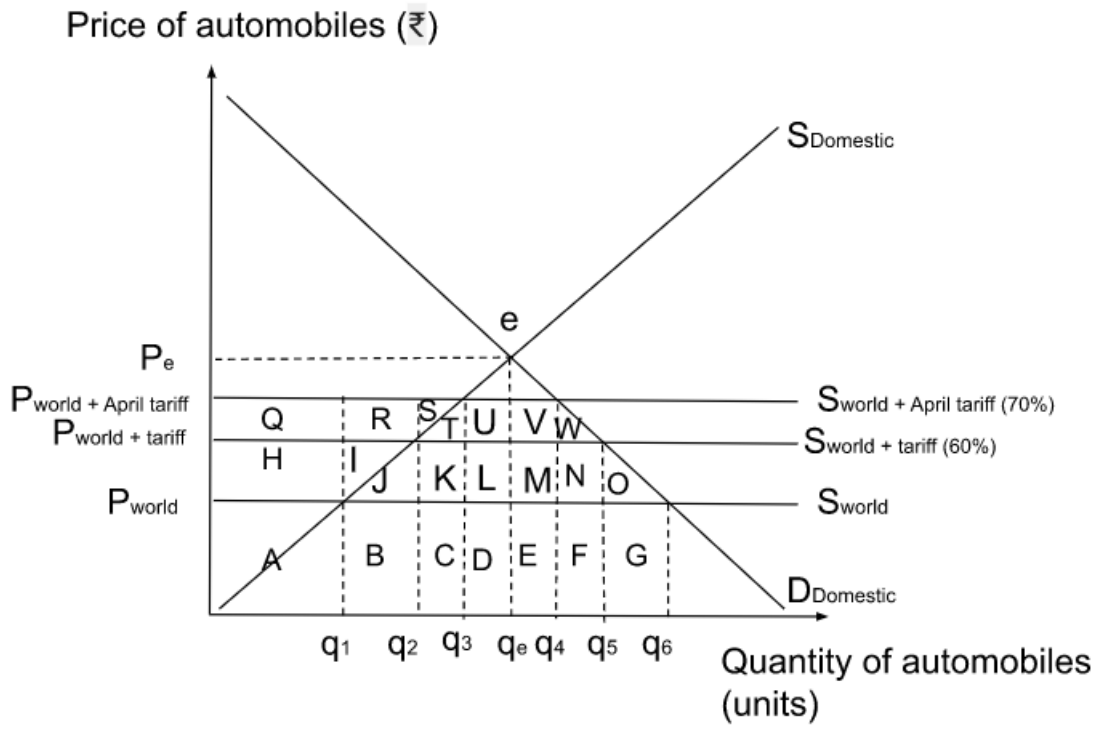
Toyota's luxury brand Lexus has warned that based on the increase in taxes, the carmaker may have to "adjust the prices" of some of its models.

"We hope to have better clarity once we study the overall impact of the same," Lexus' India president, Naveen Soni, said.

Commentary

As per the article, the Indian automobile market is the third largest in the world, composed of foreign and domestic producers alike. While the presence of imported vehicles allows greater **choice** for consumers, they may also hinder local manufacturing, which could be composed of infant industries. Thus, the government is imposing higher taxes on imported vehicles and semi-knocked down cars.

Figure 1: Market for automobiles in India after April 2023 tariff



In a free trade economy, the market operates at point 'e', supplying q_e units of automobiles at price P_e. However, since foreign producers supply vehicles at a lower price than domestic producers, the price reduces to P_{world}. This puts Indian automobile manufacturers, such as "Tata Motors" and "Mahindra & Mahindra", at a disadvantage, as they may not have developed the economies of scale to compete with foreign producers since they may be infant industries. Hence, only q_1 units of automobiles are produced by Indian manufacturers, while $q_6 - q_1$ units are imported.

The Indian government imposed a tariff of 60% on imported vehicles with a landed cost of less than $40000 and 30% on semi-knocked down cars, raising the price of automobiles to $P_{world+tariff\ (60\%)}$.

Domestic producers now supply q_2 units of automobiles. Foreign producers supply $q_5 - q_2$ units of automobiles since their factor costs have increased.

This will lead to reduced **choice** for consumers in terms of price, quantity, and quality of vehicles, as only domestic cars will be available at higher prices. The domestic producer revenue increases from area A to A+B+H+I+J, while the foreign producer revenue decreases from area B+C+D+E+F+G to C+D+E+F. The areas K+L+M+N represent government revenue. However, a welfare loss is identified (areas O+J). The area O represents a loss in consumer surplus and J represents an inefficiency of domestic producers as they produce more than what they would in a free market.

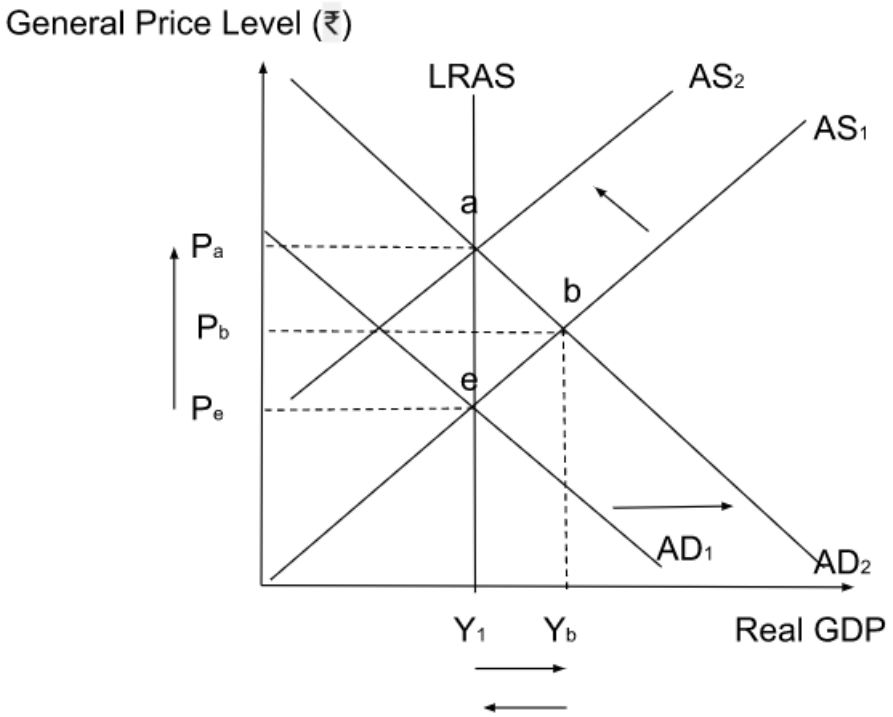
However, the government further intends to raise the tariffs of imported vehicles under $40000 by 10% and semi-knocked down cars by 5% in April to protect domestic manufacturers.

Consequently, the supply of automobiles is now reduced to q_4 units, at price level $P_{\text{world + April tariff (70\%)}}$. Domestic producers produce more automobiles, supplying q_3 units. Foreign producers supply less than previously at $q_4 - q_3$ units. Domestic revenue has increased from areas A+B+H+I+J to A+B+C+H+I+J+K+Q+R+S+T, while foreign revenue has decreased from C+D+E+F to D+E. Government revenue from the tariff has also changed from areas K+L+M+N to L+M+U+V. Since domestic producers are now producing more than in the free market, albeit inefficiently, the loss in consumer surplus increases to O+N+W with the inefficiency of producers being increased to J+K+T.

The article mentions how “taxes on automobiles in India rank among the highest globally”, causing foreign producers interested in entering the Indian market such as Lexus, Mercedes Benz, and Tesla, to face higher taxes, increasing factor cost and lowering supply. This will have an adverse effect on consumers, who will have to pay more for automobiles and will have less **choice** available to them, leading them to make irrational decisions.

Furthermore, the article mentioned car sales in India rising 25% to 3.8 million in 2022, of which the growth in premium and luxury cars such as BMW and Mercedes Benz was higher. This could increase aggregate demand as consumer expenditure is increasing. However, due to the supply shortages caused by tariffs, aggregate supply could reduce, leaving consumers with little buying **choice** as prices increase, causing inflation.

Figure 2: Market in India after imposition of April tariff



Previously, the market was operating at point 'e', at full equilibrium (AD1=AS1=LRAS). However, aggregate demand increases from AD_1 to AD_2 raising prices from P_e to P_b and increasing real GDP from Y_1 to Y_b. However, due to the tariffs, supply has reduced from AS_1 to AS_2 since factor costs for foreign producers have increased. Price has increased from P_b to P_a and Real GDP has decreased from Y_b to Y_1 leading to inflation.

The article also mentions how the tariffs have drawn criticism from foreign companies such as Tesla. Such discontent can prompt retaliatory tariffs on Indian automobile imports, which can cause unemployment in the Indian automobile industry and in complementary industries such as petroleum. This will reduce **choice** due to less income. Furthermore, the tariff will negatively affect the

consumption of electric vehicles, which accounted for 1% of total car sales in 2022 but has been drastically increasing. By preventing established companies such as Hyundai to enter the market, electric vehicles will be expensive to purchase, leaving the positive externality of consumption of electric vehicles unaddressed. This will eventually result in fewer **choices** for citizens to make healthy lifestyle choices due to the unaddressed air and noise pollution caused by petroleum and diesel substitutes.

Thus, the government must evaluate whether it should enact import substitution. It will have to make a **choice** between upholding consumer **choice**, or reducing it, and thus consumer surplus, to benefit inefficient domestic producers.

Portfolio A

Commentary 1

Criterion	Marks awarded	Marks available	Comments
A			Negative externality of consumption of cars, positive externality of consumption of bicycles. Legislation to reduce DWL.
B			Market failure, deadweight loss, free market equilibrium, externality, complementary good, MSB/MPB, allocative efficiency/inefficiency, merit good
C			Government intervenes through congestion charge to reduce negative externality of car consumption and intervenes through legislation to reduce positive externality of consumption of bicycles, to improve allocative efficiency and prevent costs to third-parties.
D			Intervention
E			.Unemployment, lower social mobility due to congestion charge taking high proportion of income of low income residents, opportunity cost, divert funding from merit goods, internalizing externality reduces third-party costs, enhancing equality and equity
Total		**14**	

Commentary 2

Criterion	Marks awarded	Marks available	Comments
A			Inflationary spiral in U.S. economy, interest rates vs quantity of money demanded, reduction in AD due to interest rate hike
B			Interest rates, inflationary spiral, marginal propensity to consume, consumer expenditure, aggregate demand, real GDP, potential output, aggregate supply, factors of production, stagflation, contractionary monetary policy, stable rate of inflation, marginal propensity to save, bonds, reserve ratio, lending rate, natural rate of unemployment, purchasing power, time lags, income shock, deficit
C			Interest rate hike due to ongoing inflationary spiral, causes decrease in supply of money, reducing aggregate demand and curbing inflation to reduce prices
D			Economic Well-Being
E			independent of political pressure, time lags, transfer payments causing budget deficit, deflationary gap, income shocks without policy causing social issue, potential for unemployment
Total		**14**	

Commentary 3

Criterion	Marks awarded	Marks available	Comments
A			Tariff diagram in India for previous tariff and new 2023 tariff, inflationary spiral (AD and SRAS shifts)
B			Tariff, domestic producer, infant industry, free trade economy, economies of scale, factor cost, revenue, welfare loss, inefficiency, irrational decisions, consumer expenditure, aggregate demand, inflation, equilibrium, retaliatory tariff, positive externality of consumption, import substitution
C			Foreign producers hurting domestic producers, prompting higher tariffs, increasing domestic revenue but creating welfare loss. Supply shortages combined with increased consumer expenditure leads to inflationary spiral, leading to higher prices and reduced choice.
D			Choice
E			Retaliatory tariffs causing unemployment in industries and complementary industries/goods. Less income means less choice. Positive externality of consumption of electric

			vehicles unaddressed, causing third-party costs.
Total		**14**	

Criterion F (when all three commentaries are marked)

Marks awarded	Marks available	Comments
	3	

Introduction:

It was quite intriguing to learn how eyeglasses correct vision and the scale at which they are used and produced; nearly 64% of the U.S. population in 2016 uses these visual aids. Innovations in vision care have made use of a well-known phenomenon known as 'refraction'. The applications of this phenomenon are manifold, from deciding what material lenses one should wear, to identifying substances in factory processes. I hence decided to investigate refraction further, particularly 'refractive index'. This experiment thus aims to investigate, to contribute to the study of optic materials, the effect of concentration in water on the solution's refractive index.

Research Question:
How does the concentration of salt in a saltwater solution affect its refractive index?

Background Information:

The discovery of refraction as an explicit phenomenon dates back as early as the 10th century, when the ability of a convex form to produce a magnified image of an object, appears to be credited to Ibn al-Haytham. He eventually published *Kitab al-Manazer (book of optics)*, which transformed the understanding of light and vision (Tbakhi and Amr). An important application of Al-Haytham's discoveries was in eyeglasses, first invented in the 13th century to correct vision problems (All About Eyes). Eyeglasses use refraction, which is now defined as the bending of a wave when it enters a medium caused by its change in speed (Britannica).

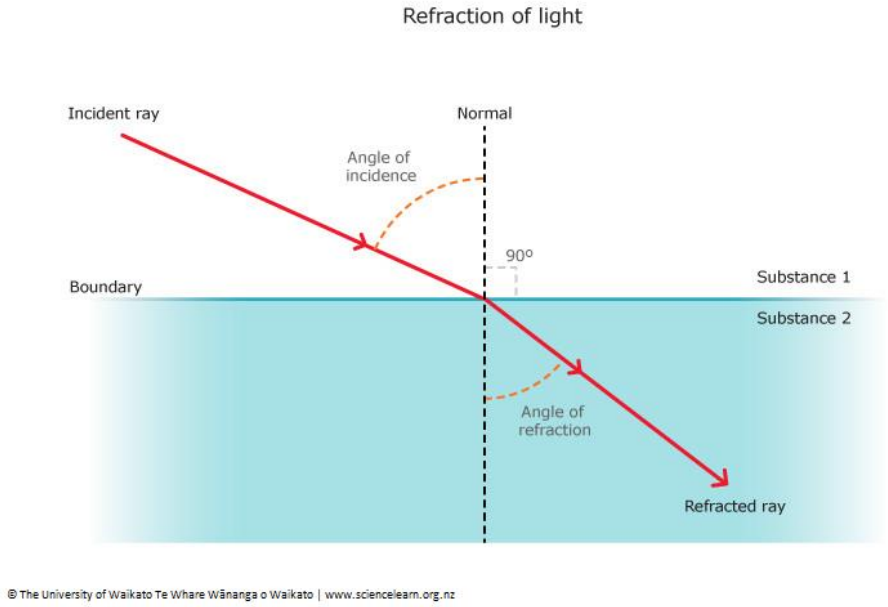


Figure 1: Diagram displaying the phenomenon of refraction (Science Learning Hub)

The extent of deviation from the normal can be quantified using an optical property known as the refractive index. The refractive index (n) of a medium is defined as the ratio of the velocity of a light ray in vacuum (c) to the velocity of light in a substance (v) ("Refractive Index"):

$$n = \frac{c}{v}$$

The refractive indices of two mediums can also be related by the bending of light, as represented in Snell's Law:

$$n_1 \sin(\theta_1) = n_2 \sin(\theta_2)$$

where n_1 represents the refractive index of the first medium, n_2 represents the refractive index of the second medium, θ_1 is the angle of incidence, and θ_2 is the angle of refraction.

There are numerous theories as to why light bends when entering a medium. A key theory is Fermat's Principle, which states that light travels between two points along the path requiring the least time compared to other nearby paths. While this theory is correct, it does not explain the scientific reason.

Since light is assumed to be made of oscillating electromagnetic fields, the reasoning behind the bending of light can be explained through Maxwell's Equations of electromagnetism. The following equations will be used:

Faraday's Law of Induction:

$$\nabla \times E = -\frac{\partial B}{\partial t}$$

Here, $\nabla \times$ is the curl operator (tendency of the vector field to rotate or circulate at a given point), E is the electric field, B is the magnetic flux density, and t is the time ("Overview of Maxwell's Equations — Electromagnetic Geophysics")

.

Gauss's Law for Static Electric Fields:

$$\varepsilon \nabla \cdot E = \rho$$

Here, $\nabla \cdot$ is the divergence operator (tendency of the vector field to converge or diverge at a given point), E is the electric field, ε is the permittivity of free space (measure of the ability of free space to store electric charge), and ρ is the charge density ("Overview of Maxwell's Equations — Electromagnetic Geophysics")

.

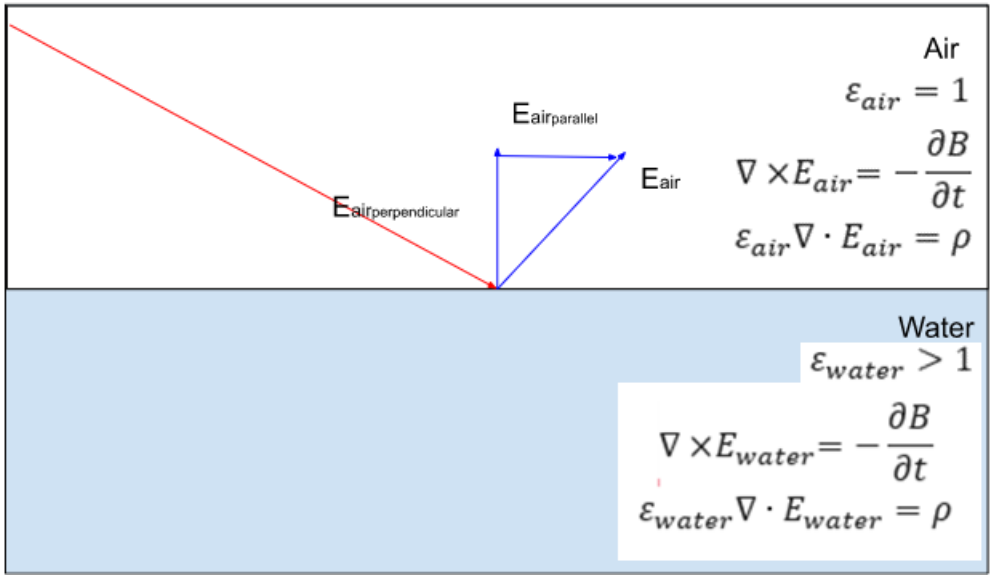


Figure 2: Diagram displaying the direction of motion light wave through air with appropriate maxwell equations (adapted from Fermilab)

We can now explain the bending of light. Assume light to be travelling from air to water at a particular incident angle. The electric field will hence be perpendicular to the direction of the light, with parallel and horizontal components. A similar electric field will be formed as the light ray traverses through water.

Since the surface belongs to both mediums, the equations for air and water can be equated as follows:

$$\nabla \times E_{air} = -\frac{\partial B}{\partial t} = \nabla \times E_{water}$$
$$\Leftrightarrow \nabla \times E_{air} = \nabla \times E_{water}$$

$$\varepsilon_{air} \nabla \cdot E_{air} = \rho = \varepsilon_{water} \nabla \cdot E_{water}$$
$$\Leftrightarrow \varepsilon_{air} \nabla \cdot E_{air} = \varepsilon_{water} \nabla \cdot E_{water}$$

Calculus can then be used to find the following restrictions:

$$E_{air_{parallel}} = E_{water_{parallel}}$$
$$\varepsilon_{air} \nabla \cdot E_{air_{perpendicular}} = \varepsilon_{water} \nabla \cdot E_{water_{perpendicular}}$$

Since $\varepsilon_{water} > \varepsilon_{air}$, $E_{water_{perpendicular}} < E_{air_{perpendicular}}$.

This causes a change in E_{water}. Light will hence travel perpendicular to the new electric field, causing a bend in light (Fermilab)

.

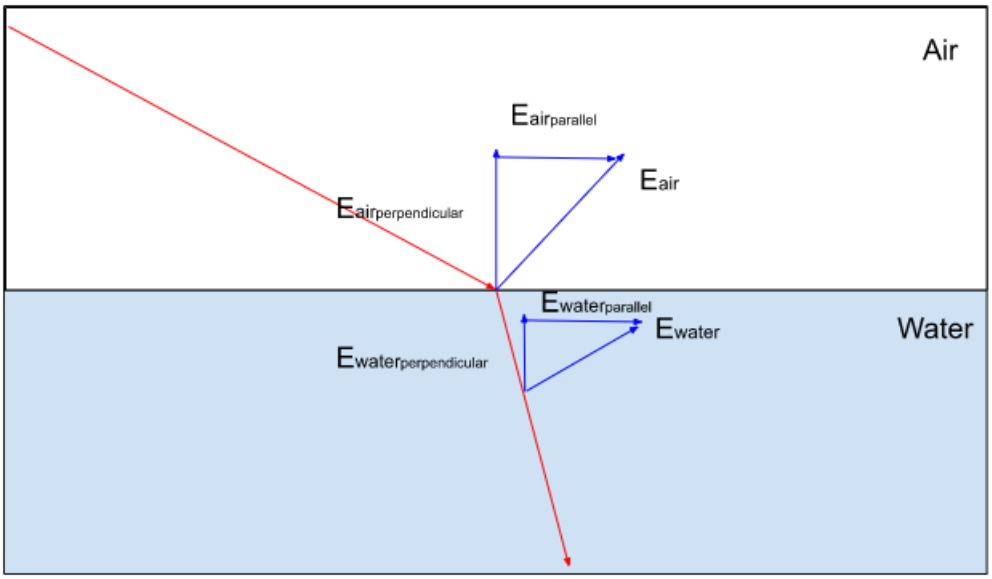


Figure 3: Diagram displaying the bending of light due to change in electric field of light wave in water (adapted from Fermilab)

Understanding how salt dissolves in water will allow us to understand how the refractive index of the solution will change. As salt dissolves in water, the forces that will attract the two molecules will be dipole-dipole. Due to the polarity, the negative element of the water (O-) will attract the positive end of salt (Na+) and vice versa. Hence, when salt (NaCl) dissolves, its components will be attracted to those of water's, leading the solution to become denser, hence affecting the refractive index. (Edubirdie)

Hypothesis:

It can be hypothesized that **an increase in the concentration of NaCl in a salt water solution will lead to an increase in the refractive index of the solution**, due to the solution becoming denser, increasing its permittivity of free space, leading to an increase in the bending of the light wave and causing the refractive angle to reduce.

Variables:

Independent Variable	Units	How it will be measured	Manipulation of variable
Concentration of salt in saltwater solution (c)	[g/mL]	Using an electronic weighing scale and an A4 sized sheet, the mass of powdered salt will be measured and dissolved into a fixed volume of water using a stirrer. The	Different masses of salt will be taken. The initial mass will be 0 g, and will increase in values of 70 g until the mass of salt added after 5 solutions will be 350 g in total. Note that the solubility constant of salt in water is 360 g per

		solution will hence become denser due to forces of attraction.	1000 mL at 25℃. Hence, the maximum amount of salt being dissolved will be 350 g.

Table 1: Independent Variable

Dependent Variable	Units	How it will be measured	Manipulation of variable
Angle of Refraction (θ_{AR_y} where AR denotes average refractive index measured across all trials and y denotes the solution number)	[°]	The entirety of the experiment will be filmed using a mobile phone kept at constant distance from the setting. Using IC Measure, a digital software to measure rotational angles, the angle of refraction will be measured, and the average (θ_{AR_y}) of the five trials of the solution number will be calculated.	As the concentration of salt increases, the angle of refraction will decrease. The variable will hence be manipulated by manipulating the independent variable.

Table 2: Dependent Variable

Controlled Variables	Units	How it will be measured	Value
Wavelength (λ)	[nm]	The wavelength will be kept constant by using a laser of green color.	550 nm
Incident Angle (θ_i)	[°]	The angle of incidence will be kept constant using a clamp stand. The laser will thus be kept at a constant angle.	$20.0 \pm 0.5°$
Temperature	[℃]	The room will be air conditioned with the windows and blinds closed.	25℃
Volume of Solution	[mL]	The volume of solution will be measured using a measuring cylinder	1000.0 ± 0.5 mL

Table 3: Controlled Variables

Apparatus:

Measuring Apparatus (quantity)(uncertainty):

1. Electronic Weighing Balance (1) (± 0.01g)
2. Protractor (1) ($\pm 0.5°$)
3. 1000 mL Measuring Cylinder (1) (± 0.5 mL)
4. A4 Sheet of paper (1)
5. Spatula (1)

Experimental Apparatus (quantity):

1. 1000 mL Beaker (1) (± 50 mL)
2. Clamp Stand (1)
3. Green Laser (1) ($\lambda \approx 550$ nm)
4. Magnetic Stirrer (1)
5. Notebook and Pen (1)
6. Blue Tack (1 packet)

Analysis Apparatus (quantity):

1. Graphic Display Calculator (1)
2. IC Measure (1)
3. LoggerPro (1)
4. Laptop (1)

Chemicals Required (quantity):

1. Distilled water (1500 mL)
2. Table salt or NaCl (500 g)

Method:

Experimental Setup:

1. Set the clamp stand on a counter
2. Using blue tack, stick the protractor to the clamp stand, ensuring that the 90° line points vertically downward
3. Attach the laser pointer to the clamp stand, and using a protractor, adjust it so that it is 20 degrees to the stand
4. Stick the laser pointer to its current position using blue tack and tighten the clamp stand to ensure minimal inaccuracy in the incident angle
5. Using a 1000 mL measuring cylinder, measure 1000 mL of distilled water, while ensuring the measurement takes place at eye level to avoid parallax error
6. Pour the distilled water from the 1000 mL measuring cylinder into a 1000 mL beaker
7. Place an A4 sheet of paper on an electronic weighing scale and reset mass to 0 g
8. Set the stirrer on the counter

Conducting Experiment:

9. Using a spatula, placed powdered salt on the electronic weighing scale until it measures 70 g
10. Empty the A4 sheet into the 1000 mL beaker containing the distilled water
11. Place the beaker on the stirrer for 5 minutes, setting the RPM to 1600 to make solution homogenous
12. Place the beaker near the clamp stand, ensuring the laser light reaches the middle of the beaker through air, and not through the glass
13. Turn the laser on
14. Using a phone camera, take 5 photos from a constant distance away from the beaker at eye-level
15. Repeat steps 9 – 15 until the concentration of the solution becomes 0.350 g/mL

Analyzing Data:

16. Organize photos into different folders by solution numbers
17. Load photos from each trial onto IC Measure on the laptop, and by drawing a vertical line upwards from the point the laser reaches the water in the beaker, measure the refraction angle for each photo in the solution number set

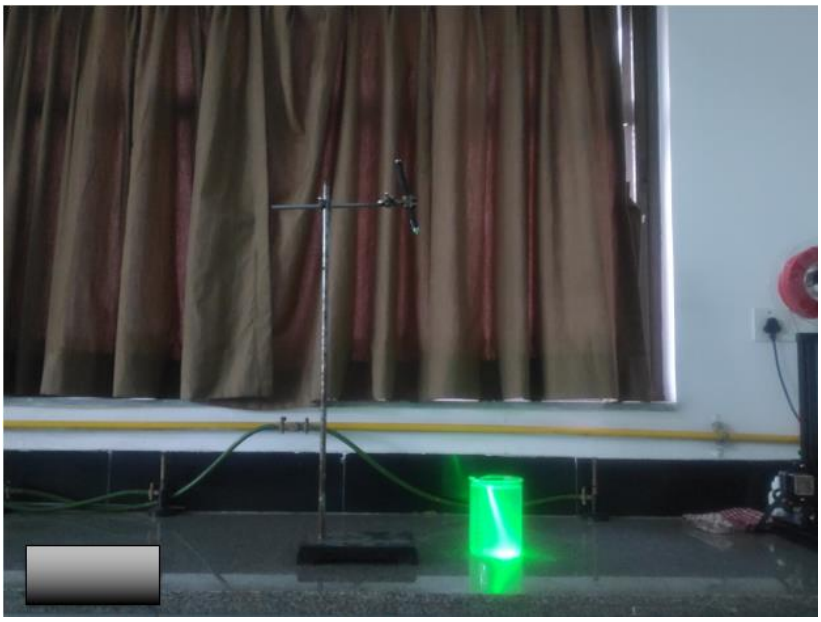


Figure 4: Experimental setup and sample photo

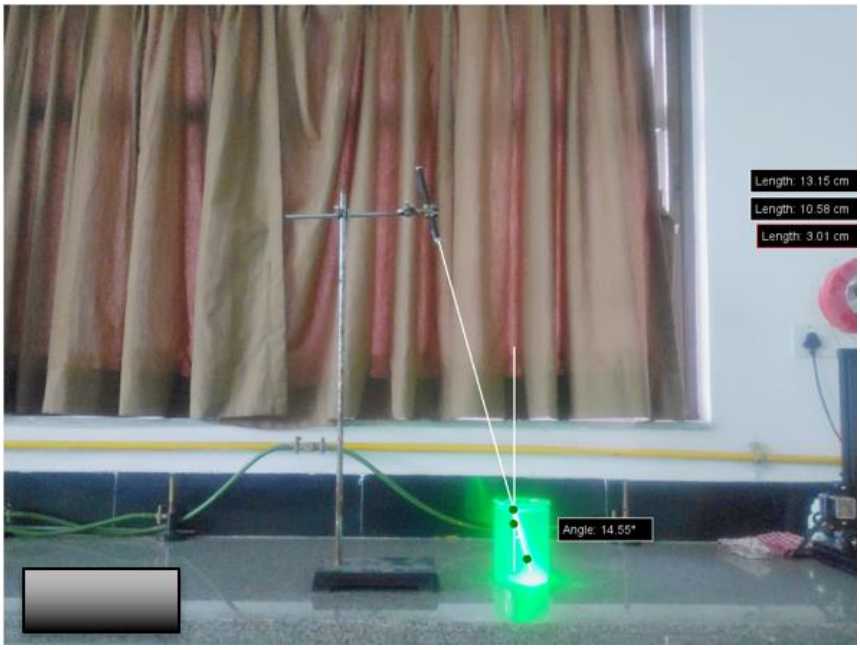


Figure 5: Sample Analysis

Safety Precautions:

Make sure to avoid eye-contact with lasers as it can cause vision impairment. Make sure to not keep the beaker on the edge of the counter to avoid spillage.

Environmental Considerations:

Attempt to reuse water by simply dissolving additional masses of salt to increase concentration. Keep laser off when not in use to avoid battery drainage and air conditioners off when experimental procedure is over.

Analysis & Data Representation:

Solution Number	**Concentration [g/mL]**	**Refraction Angle**					
	$c \pm \Delta c$	1st Trial	2nd Trial	3rd Trial	4th Trial	5th Trial	$\theta_{AR_y} \pm \Delta\theta_{AR_y}$
1	$(700.000 \pm 0.360) \times 10^{-4}$	14.3	14.5	14.9	15.0	14.4	14.600±0.350
2	$(140.000 \pm 0.107) \times 10^{-3}$	14.0	14.6	14.1	14.5	15.1	14.500±0.550

3	$(210.000 \pm 0.212) \times 10^{-3}$	13.8	14.1	14.1	14.7	14.6	14.300 $\pm$0.450
4	$(280.000 \pm 0.352) \times 10^{-3}$	14.6	13.9	13.8	14.0	14.5	14.200$\pm$0.400
5	$(350.000 \pm 0.527) \times 10^{-3}$	13.5	14.3	13.8	13.9	14.4	14.000$\pm$0.450

Table 5: Representation of Data

Since the concentration of a solution is denoted as $c_x = \frac{m_x}{V}$., where c_x is the concentration of an x solution number, m_x is the mass of solute in solution number x and V is the volume of the solution, the Gaussian uncertainty for concentration can be calculated as:

$$\frac{\Delta c_x}{c_x} = \sqrt{\left(\frac{\Delta m}{m_x}\right)^2 + \left(\frac{\Delta V}{V}\right)^2} \Leftrightarrow \Delta c_x = c_x \sqrt{\left(\frac{\Delta m}{m_x}\right)^2 + \left(\frac{\Delta V}{V}\right)^2}$$

$$\Leftrightarrow \Delta c_x = c_x \sqrt{\left(\frac{0.01}{m_x}\right)^2 + \left(\frac{0.5}{1000}\right)^2}$$

Note that Gaussian uncertainty is being used as it is more precise as compared to normal uncertainty.

Hence, the uncertainty for the first concentration (0.070 g/mL) will be:

$$\Delta c_1 = c_1 \sqrt{\left(\frac{0.01}{m_1}\right)^2 + \left(\frac{0.5}{1000}\right)^2}$$

$$\Delta c_1 = 0.070 \sqrt{\left(\left(\frac{0.01}{70}\right)^2 + \left(\frac{0.5}{1000}\right)^2\right)} = 0.000036 = 0.360 \times 10^{-4} \text{ g/mL}$$

The uncertainty of the second concentration (0.140 g/mL) was the following:

$$\Delta c_x = \Delta c_{x-1} + c_x \sqrt{\left(\frac{\Delta m}{m_x}\right)^2 + \left(\frac{\Delta V}{V}\right)^2}$$

$$\Leftrightarrow \Delta c_2 = \Delta c_1 + c_2 \sqrt{\left(\frac{0.01}{m_2}\right)^2 + \left(\frac{0.5}{1000}\right)^2}$$

$$\Leftrightarrow \Delta c_2 = 0.000036 + 0.140 \sqrt{\left(\frac{0.01}{140}\right)^2 + \left(\frac{0.5}{1000}\right)^2}$$

$$\Leftrightarrow \Delta c_2 = 0.000036 + 0.000071$$

$$\Leftrightarrow \Delta c_2 = 0.107 \times 10^{-3}$$

The average refraction angle (θ_{AR_y}) was calculated using the following method:

$$\theta_{AR_y} = \frac{\theta_1 + \theta_2 + \theta_3 + \theta_4 + \theta_5}{5}$$

For example, for the first solution, the refraction angle is

$$\theta_{AR_1} = \frac{14.3 + 14.5 + 14.9 + 15.0 + 14.4}{5} = \frac{73.1}{5} = 14.62 = 14.6°$$

The uncertainty values of the refraction angle of each solution was calculated using the following method:

$$\Delta\theta_{AR_y} = \frac{\theta_{max} - \theta_{min}}{2}$$

For example, the uncertainty of the refraction angle for the first solution is:

$$\Delta\theta_{AR_1} = \frac{15.0-14.3}{2} = 0.350°$$

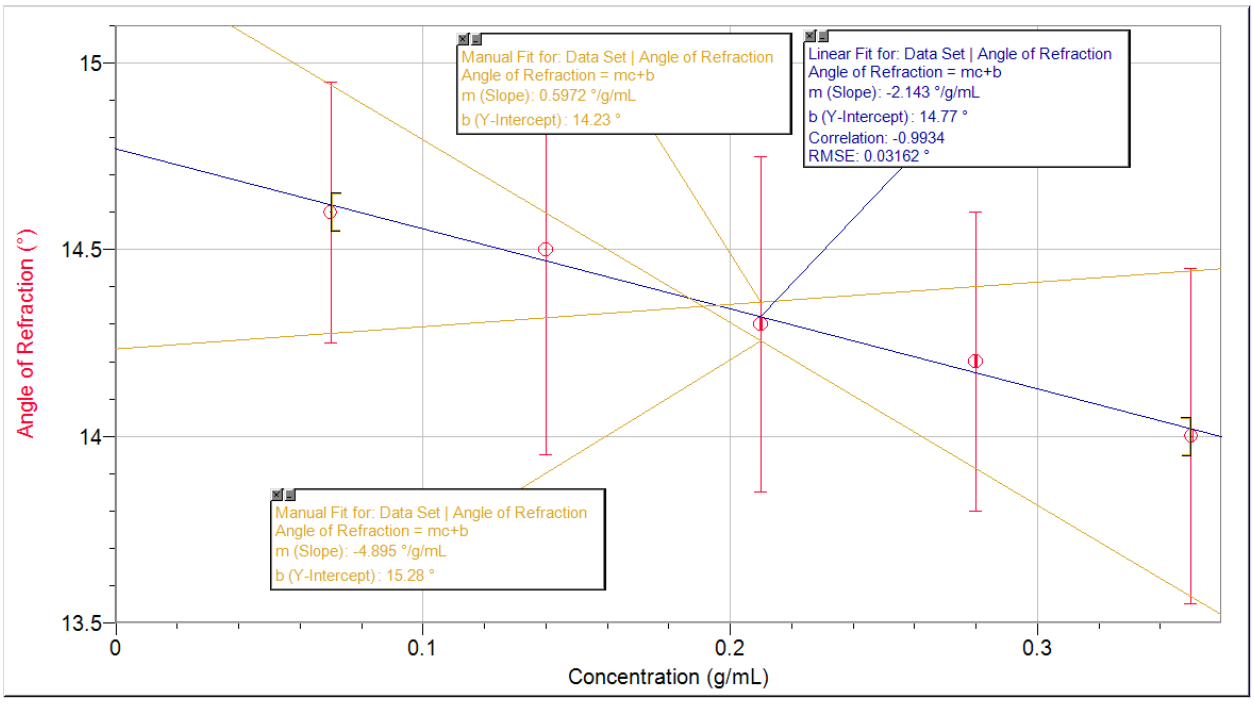


Figure 6: Graph showing relationship between concentration of solution and angle of refraction (made using Logger Pro)

It is evident that there is a negative correlation between the concentration of the solution and the angle of refraction θ_{AR_y}. This validates a part of the hypothesis which mentions a higher

concentration leading to reduced refractive angles due to greater bending of light. It is, however, important to consider the uncertainty of the values, which range from 0.350 to 0.550, indicating certain reservations regarding the values obtained, and an inability to conclude whether the relationship between the two variables is linear. This prevalence of uncertainty amongst the angle of refraction values may be the result of random errors such as visual error while analyzing through IC Measure as well as parallax error when obtaining pictures of the experiment. It is worth noting, however, that when $c = 0$, $\theta_{AR_0} = 14.77°$. A close value is noted when using Snell's Law: ($n_1 \sin(\theta_1) = n_2 \sin(\theta_2)$

$$\theta_{\mathrm{r}} = \sin^{-1}\left(\frac{(n_{air}\sin(\theta_i)}{n_{water}}\right)$$

$$\Leftrightarrow \theta_{\mathrm{r}} = \sin^{-1}\left(\frac{1 \times \sin(20°)}{1.33}\right)$$

$$\Leftrightarrow \theta_{\mathrm{r}} = 14.9°$$

Such a close difference of 0.13° may be the result of the random errors mentioned above.

Solution Number	Concentration [g/mL]	Refractive Index (n_2)	Uncertainty of Refractive Index (Δn_2)
1	0.0700	1.36	0.040
2	0.140	1.37	0.060
3	0.210	1.39	0.050
4	0.280	1.40	0.050
5	0.350	1.41	0.060

Table 6: Representation of Data

To calculate Δn_2, we will use Snell's law:

$$n_1 \sin(\theta_i) = n_2 \sin\left(\theta_{AR_y}\right) \Leftrightarrow\ n_2 = \frac{n_1 \sin(\theta_i)}{\sin\left(\theta_{AR_y}\right)}$$

$$\frac{\Delta n_2}{n_2} = \sqrt{\left(\frac{\Delta n_1}{n_1}\right)^2 + \left(\frac{\Delta \sin(\theta_i)}{\sin(\theta_i)}\right)^2 + \left(\frac{\Delta \sin\left(\theta_{AR_y}\right)}{\sin\left(\theta_{AR_y}\right)}\right)^2}$$

We can further simplify $\Delta sin(\theta_i)$ and $\Delta sin(\theta_{AR_y})$ through the following convention for a generic $\Delta sin(\theta)$:

$$\Delta sin(\theta) = \sqrt{\left(\left|\frac{dsin(\theta)}{d\theta}\right|\right)^2} = \sqrt{(|\cos(\theta)|\Delta\theta)^2} = |\cos(\theta)|\Delta\theta$$ (The University of Mississippi)

Hence:

$$\frac{\Delta n_2}{n_2} = \frac{\Delta n_1}{n_1} + \frac{\Delta \sin(\theta_i)}{\sin(\theta_i)} + \frac{\Delta \sin\left(\theta_{AR_y}\right)}{\sin\left(\theta_{AR_y}\right)}$$

$$\Leftrightarrow \frac{\Delta n_2}{n_2} = \sqrt{\left(\frac{\Delta n_1}{n_1}\right)^2 + \left(\frac{|\cos(\theta_i)|\Delta\theta_i}{\sin(\theta_i)}\right)^2 + \left(\frac{\left|\cos\left(\theta_{AR_y}\right)\right|\Delta\theta_{AR_y}}{\sin\left(\theta_{AR_y}\right)}\right)^2}$$

$\frac{\Delta n_1}{n_1}$ is negligible and can hence be assumed as 0 due to its small value.

Note that the degree values will be converted to radians as as $\frac{dy}{dx}\sin(x) = \cos(x)$ only if x is represented in radian form.

Thus the uncertainty for the first solution will be calculated as follows:

$$\frac{\Delta n_2}{1.36} = \sqrt{\left(\frac{\left(\left|\cos\left(20 \times \frac{\pi}{180}\right)\right|\right)\left(0.5 \times \frac{\pi}{180}\right)}{\sin\left(20 \times \frac{\pi}{180}\right)}\right)^2 + \left(\frac{\left(\left|\cos\left(14.6 \times \frac{\pi}{180}\right)\right|\right)\left(0.35 \times \frac{\pi}{180}\right)}{\sin\left(14.6 \times \frac{\pi}{180}\right)}\right)^2}$$

$$\Leftrightarrow \Delta n_2 = 1.36\sqrt{\left(\frac{|\cos(0.349066)| \times 0.00872665}{\sin(0.349066)}\right)^2 + \left(\frac{|\cos(0.2548181)|0.006108652}{\sin(0.2548181)}\right)^2}$$

$$\Leftrightarrow \Delta n_2 = 1.36(0.033538526745421)$$

$$\Leftrightarrow \Delta n_2 = 0.045612396373773$$

$$\Leftrightarrow \Delta n_2 \approx 0.04$$

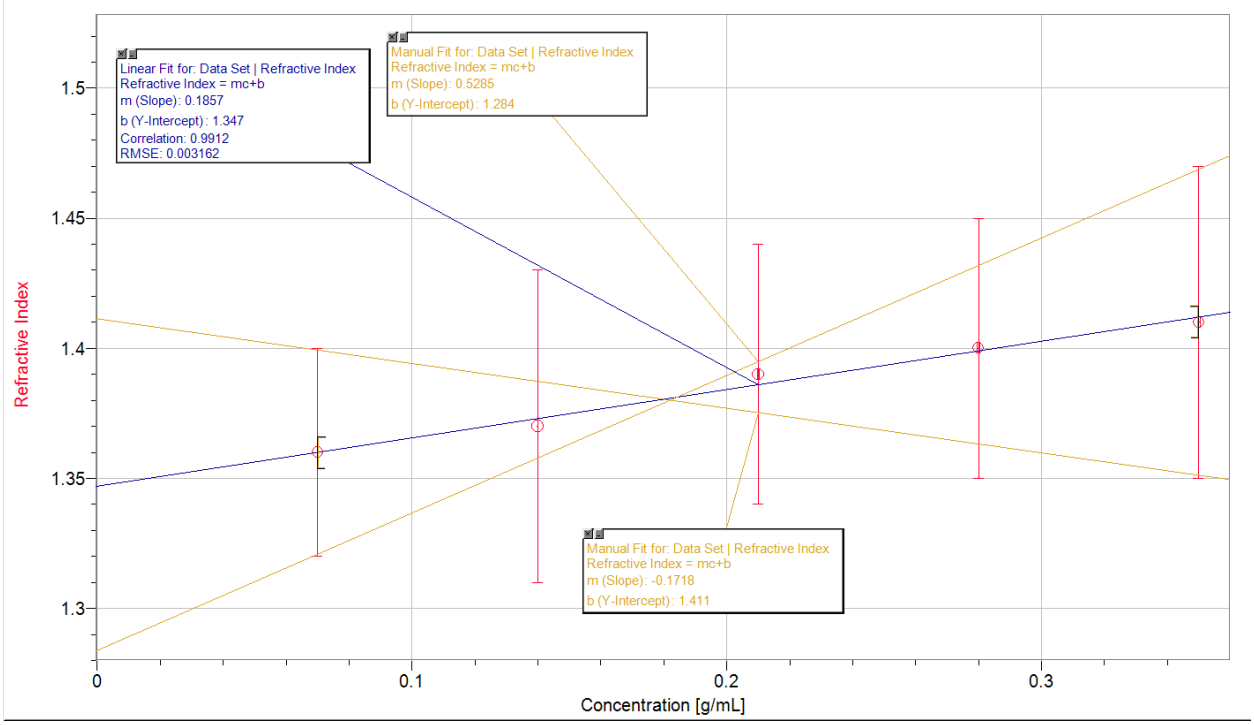


Figure 7: Graph showing relationship between concentration of solution and refractive index (made using Logger Pro)

It is evident that there exists a positive correlation between the concentration of the saltwater solution and the refractive index of the respective solution, due to the correlation coefficient being 0.9912. All data points, except the second, fall on, or above the trend line. With each increase in concentration, there is an increase, albeit minimal, in the refractive index. Note that with concentration at 0 g/mL (distilled water), the refractive index is 1.347. This is higher than the scientifically accepted value, and could be the result of random error as mentioned above, which led to a variation in the observation of the refractive angle. Note that uncertainties of concentration are plotted but not visible. As per the line of best fit, the refractive index of saltwater can be given by the following linear equation:

$$n_2 = 0.1857c + 1.347$$

Using the manual fits for data set, the slope uncertainty is:

$$\Delta m = \frac{m_{max} - m_{min}}{2}$$

$$\Leftrightarrow \Delta m = \frac{0.5285 - (-0.1718)}{2}$$

$$\Leftrightarrow \Delta m = \frac{0.5285 + 0.1718}{2}$$

$$\Leftrightarrow \Delta m = \frac{0.7003}{2}$$

$$\Leftrightarrow \Delta m = 0.35015$$

$$\Leftrightarrow \Delta m \approx 0.4$$

The slope is hence 0.19 ± 0.40

Evaluation:

In order to ensure high precision, most uncertainties were accounted for. This led to higher uncertainty of the refractive indices. Furthermore, it was noted that there was a minimal increase in the volume of the solution as a solute was added (this is a known phenomenon as per scientific consensus). Due to an inability to quantify the rise in volume due to a lack of suitable apparatus, a potential source of error was not accounted for, and could have altered the refractive index. Nevertheless, the collected data was highly precise, as proven by the correlation coefficient, which was 0.9912 for the relationship between concentration and refractive index, and -0.9934 for the relationship between the concentration and the angle of refraction. The slope for the line of best fit representing the relationship between concentration and refractive index was 0.19 ± 0.40 with an error percentage uncertainty of $\frac{0.19}{0.40} \times 100\% = 210.5\%$. This indicates that the data collected may not be entirely accurate. However, it is also important to note that the values of the existing research paper still lied within the uncertainty range of the data collected in this experiment. The difference in values obtained may be the result of considerable systematic error and random error. While efforts were made to minimize systematic error such as using distilled water and ensuring that the concentration remained under NaCl's solubility constant to ensure homogeneity, random errors such as the angle at which the photo was taken, mis clicking on the software, and parallax error contributed to a deviation in the values. Also worth noting was the highly negligible uncertainties of the concentration, enhancing accuracy of the measured solute.

Conclusion:

The results obtained validate the hypothesis, that an increase in the concentration of salt in a saltwater solution will increase the solution's refractive index. However, it is important to note that due to random errors such as parallax error, the results deviated from data found by existing scientific research rather than a systematic error. This can be compared to a graph from an existing paper (titled "Remote monitoring of water salinity by using side-polished fiber-optic U-shaped sensor") which represented the examined between salt concentration of solution and refractive index just as in this experiment. While the existing paper measured the concentration using percentage, they have been converted to concentration in g/mL.

Concentration (g/mL)	Refractive Index determined by this paper	Refractive Index determined by existing paper
0.070	1.36	1.34
0.140	1.37	1.36

0.210	1.39	1.37
0.280	1.40	1.38
0.350	1.41	1.40 (through extrapolation)

Table 7: Comparison of findings between this experiment with existing published paper (Stupar)

While deviation occurs, it is minimal, with existing research findings lying within the uncertainty range of the data found in this experiment.

The slope (m_e) of a best fit line passing through the data points of the existing paper would be:

$$m_e = \frac{1.38 - 1.34}{0.280 - 0.070} \Leftrightarrow m_e = 0.1905$$

This is in contrast to the results obtained in this experiment, where the slope was 0.1857, indicating a $\frac{0.1905-0.1857}{0.1905} \approx 0.025$ deviance. Similarly, the Y-intercept (b) of the best fit line passing through the data points of the existing paper (x_e, y_e) would be:

$$y_e = m_e x_e + b \Leftrightarrow 1.34 = 0.1905 \times 0.070 + b \Leftrightarrow b \approx 1.33$$

This is in contrast to the results obtained in this experiment where the Y-intercept was 1.347, indicating a $\frac{1.33-1.347}{1.33} \approx -0.013$ deviance.

While these findings deviate from traditionally accepted values (although minimal), they are still highly significant in their application, particularly in the field of ophthalmology. Studying protein concentrations in the human eye lens allows for an observation of the gradient of refractive index in human eye lens, thus allowing ophthalmologists to note vision errors and prescribe accurate visual aids (Pierscionek and Augousti). This experiment will allow ophthalmologists to rely on concentration more confidently when identifying vision errors, preventing prescription errors. Furthermore, refractive index and concentration is used to identify materials that can be used to correct vision error, such as polycarbonate (Lera).

Strengths and Weaknesses:

Variables	Influence of error on data	Methods to minimize error
Independent: Concentration of Salt in Saltwater Solution (c)	Considering the highly minimal uncertainty, potentially due to effective stirring and using precise equipment, measurement error is unlikely to alter the outcome of this experiment. However, the minimal rise in volume (described under), if accounted for, could have altered the uncertainty of the concentration.	Accounting for rise in volume due to addition of solute. Measuring concentration as a percentage of the total solution to match with traditional scientific reports.

Dependent: Angle of Refraction	It is possible that due to random errors such as parallax error the outcome of the experiment may have been altered, especially considering that minimal changes in refraction angles can cause large changes in refractive indices. Furthermore, scattering of light was prevalent when analyzing the photos, leading to potential error.	Capturing photos at a closer proximity to allow less visual error. Using phone holder to ensure uniformity and prevent parallax error. Test scattering of laser beforehand.
Controlled: Wavelength (λ)	The laser used remained the same throughout the experiment and the battery remained charged, allowing for a constant wavelength and monochromaticity. However, the precise value of the wavelength and its uncertainty was not accounted for due to an inability to lack of data	Using a laser pointer which has appropriate labelling or a box that accompanies it.
Incident Angle	The incident angle is considered to be a minimal source of error. The laser was not deliberately moved during the experiment, and blue tack was used to complement the clamp stand in holding the pointer in place. However, blue tack is a stretchable material and the pointer may have slightly deviated from its incident angle.	Using non-stretchable material such as glue to hold the pointer as well as using an adjustable laser pointer holder. Keeping the protractor attached to the clamp stand throughout the experiment to check the incident angle periodically.
Temperature	While the doors were shut and air conditioning was on, the temperature variated throughout the day. The presence of sun light entering the room could have altered the temperature, potentially causing the solution to become less dense, altering the refractive angle.	Conducting experiments in a dark room, preventing sunlight and external heat sources from entering. Using thermometers to constantly gauge temperature.
Volume of Solution	The volume of solution increased by a minimal amount with each addition of solute. This was not accounted for and could have altered the refractive index due to changing concentration.	Using pipette to extract additional volume and measure quantity using burette. Calculating additional solute, with graphic display calculator, to add in order to account for difference in concentration.

Table 8: Strengths and Weaknesses, analyzed by variable

Works Cited

Britannica. “Refraction | Definition, Examples, & Facts | Britannica.” *Encyclopedia Britannica*, 10 February 2023, https://www.britannica.com/science/refraction. Accessed 4 March 2023.

Edubirdie. “Sodium Chloride Solution Effect On Refractive Index - Free Essay Example.” *Edubirdie*, 2022, https://edubirdie.com/examples/sodium-chloride-solution-effect-on-refractive-index/. Accessed 4 March 2023.

Fermilab. “Why does light bend when it enters glass?” *YouTube*, 1 May 2019, https://www.youtube.com/watch?v=NLmpNM0sgYk. Accessed 4 March 2023.

Lera, Bridget. “Choosing The Right Lens — Downtown Vision - Eye Care and Vision Services in Reno and Carson City.” *Downtown Vision*, 29 November 2022, https://www.downtownvisionnv.com/blog/choosing-the-right-lens/. Accessed 5 March 2023.

“Maxwell's Equations.” *Maxwell's Equations*, https://www.maxwells-equations.com/. Accessed 5 March 2023.

“Overview of Maxwell's Equations — Electromagnetic Geophysics.” *EM GeoSci*, 2018, https://em.geosci.xyz/content/maxwell1_fundamentals/quick_guide_maxwell.html. Accessed 06 March 2023.

Pierscionek, Barbara K., and Andy T. Augousti. “The refractive index in the eye lens – implications for clinical practice and optical design.” *Eye News*, 1 August 2016, https://www.eyenews.uk.com/features/ophthalmology/post/the-refractive-index-in-the-eye-lens-implications-for-clinical-practice-and-optical-design. Accessed 5 March 2023.

Science Learning Hub. “Refraction of light — Science Learning Hub.” *Science Learning Hub*, 26 April 2012, https://www.sciencelearn.org.nz/resources/49-refraction-of-light. Accessed 1 March 2023.

Stupar, Dragan. “Fig. 5. Refractive index depending on concentration of salt in...” *ResearchGate*, 2012, https://www.researchgate.net/figure/Refractive-index-depending-on-concentration-of-salt-in-water-salt-solution-From-Fig-5_fig6_261419326. Accessed 01 March 2023.

Tbakhi, Abdelghani, and Samir S. Amr. “Ibn Al-Haytham: Father of Modern Optics.” *Annals of Saudi Medicine*, vol. 27, no. 6, 2007, pp. 464-467.

Tsokos, K. A. *Physics for the IB Diploma Coursebook*. Cambridge University Press, 2014. Accessed 05 March 2023.

The University of Mississippi. *Uncertainty calculations for index of refraction calculations*. *The University of Mississippi*, http://www.phy.olemiss.edu/~thomas/weblab/222_Miscellaneous%20folder/222_web_uncertainty_items/Snell_uncertainty_use.pdf. Accessed 05 March 2023.

Computer Science IA Cover Page

Solution title: Student Management Software for Speech & Drama Center

Candidate personal code: kjt140

Links to: Product

Directions to access product or any other additional information	To login as an admin- User Name : john@gmail.com Password: Vocal2023@john To login as a teacher/user- User Name: george@gmail.com Password: Vocal2023@george

Links to documentation are below

Planning	Planning
Design	Record of Tasks Design
Development	Development
Functionality (additional videos may be added)	Video of product
Evaluation	Evaluation
Appendices	Appendix 1(All Interactions with Client) Appendix 2 (Initial Sketches) Appendix 3(Complete Code)
WORD COUNT	**2000**

Who is the client?

My client is **Ms. Lisa D'Souza** who is the founder of VOCAL, an institution training students for Trinity College London speech and drama examinations.

Defining the Problem:

VOCAL teaches 400+ students, with 15+ teachers. Thus keeping track of student progress becomes **tedious**. Currently, VOCAL only measures progress by viewing the student's Trinity exam marks. This proves problematic as giving exams is **expensive**, and since they focus on memorization, are **not representative of the student's actual skills**. ***(Refer Appendix 1 first interaction for Interview Transcript).***

VOCAL also faces difficulty keeping track of examination requirements. Each candidate needs to have a unique piece for performing. Teachers **do not have a medium of logging** these details in. Thus, due to a **lack of communication**, multiple students **are assigned the same pieces** which leads to **disqualification from the examination**. This makes the process of meeting requirements error-prone.

Apart from that, my client does not have a **medium of storing student data**. Hence, searching for student information proves **time-consuming**. Oftentimes, students return later when applying to university, requesting an official transcript, as they do not remember which exam they

appeared for. VOCAL however does not maintain these, **preventing** the student from applying for credits in college.

I hence offered to help Ms. Lisa offset these problems with a digital solution. This would allow her to keep track of students in a coherent and organized manner, and also reduce stress when searching for information.

Rationale for Proposed Solution:

The solution will allow my client to keep track of her students in an organized and easy manner. It will also quantify student progress, allowing teachers to focus on weaknesses.

A few potential solutions brainstormed included a mobile application created using Flutter and Dart, or creating an interactive spreadsheet system using Microsoft Excel. However, these proved inadequate. A mobile application is difficult to maneuver. Flutter also does not have a large resource base. Using Microsoft Excel is not a secure option and would hinder navigability due to the presence of multiple spreadsheets without ergonomically appealing elements.

I have chosen to use Java and MySQL for the creation of the application. Apart from currently studying the language, I have chosen Java for its other benefits:

1. Platform Independent: Considering that there will be multiple devices using the same software, Java is ideal, as it is platform independent, implying that the code can be used on any operating system.
2. Object Oriented: Java allows reuse of code through inheritance, saving coding time, and allows modularity for easier troubleshooting.
3. Secure: There is rarely interaction between the software and the Operating System, which is beneficial for holding sensitive student data.
4. Economically Efficient: It does not require any high end hardware or software infrastructure, so it is economical.

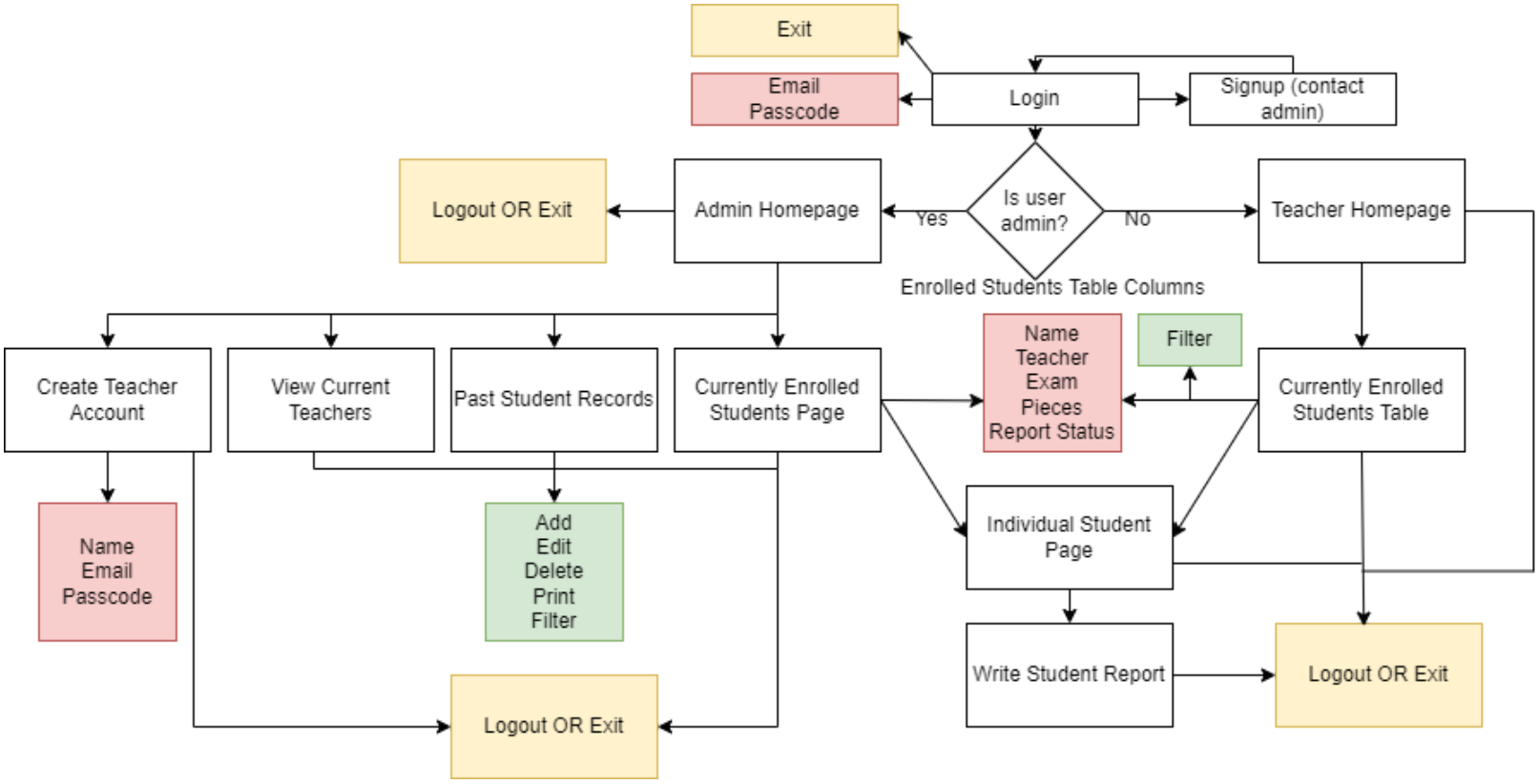
MySQL has many advantages as well. It has a client-server architecture, allowing any number of clients to communicate with the database server. It is compatible with most used Operating Systems, and can be run on different computers via the internet or local network. It is economically efficient as it is free to download from MySQL website. It is also secure, as it contains a data security layer; passwords are encrypted.

Success Criteria:

1. There must be an option to filter information on the database, through criteria such as exam, alphabetical student name, pieces, etc

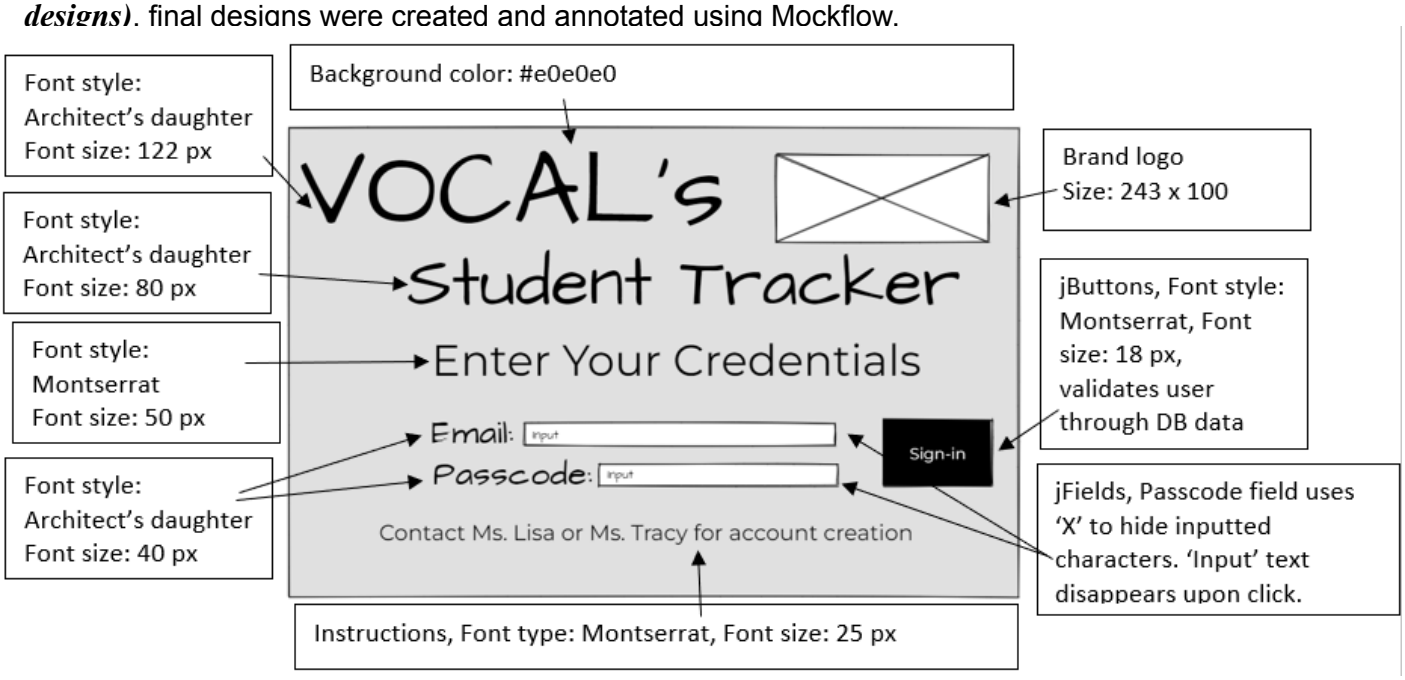
2. Teachers can view all and edit some student information, but cannot add or delete students. Administrators can change any student's information, as well as add and delete from their own page.
3. Warning and confirmation messages should be displayed when attempting to edit data
4. The application should be protected by a user login, and password
5. In case of user invalidation, the application should portray an error message
6. All lists need to be organized as a table, adding to ergonomic appeal
7. The software should allow a 'flagging' feature for students that use the same pieces, or aren't meeting exam requirements
8. The application should have two sections: administrator and user/teacher. Administrators can create, edit data, as well as add teacher accounts.
9. Only administrators can view past year information for students
10. The user can exit the application at any time, but would need to login again once opened
11. The institution's logo should be visible on each page of the application
12. The application should be simplistic and efficient to use, aiding navigability
13. All buttons need to direct user to the correct page
14. All data in application should match the data entered by the user and recorded in the database
15. Input text fields must accept information and perform validation checks
16. There should be an option to start a new term, which will erase the currently enrolled students table and label past students by the term

Overall Flowchart of Application

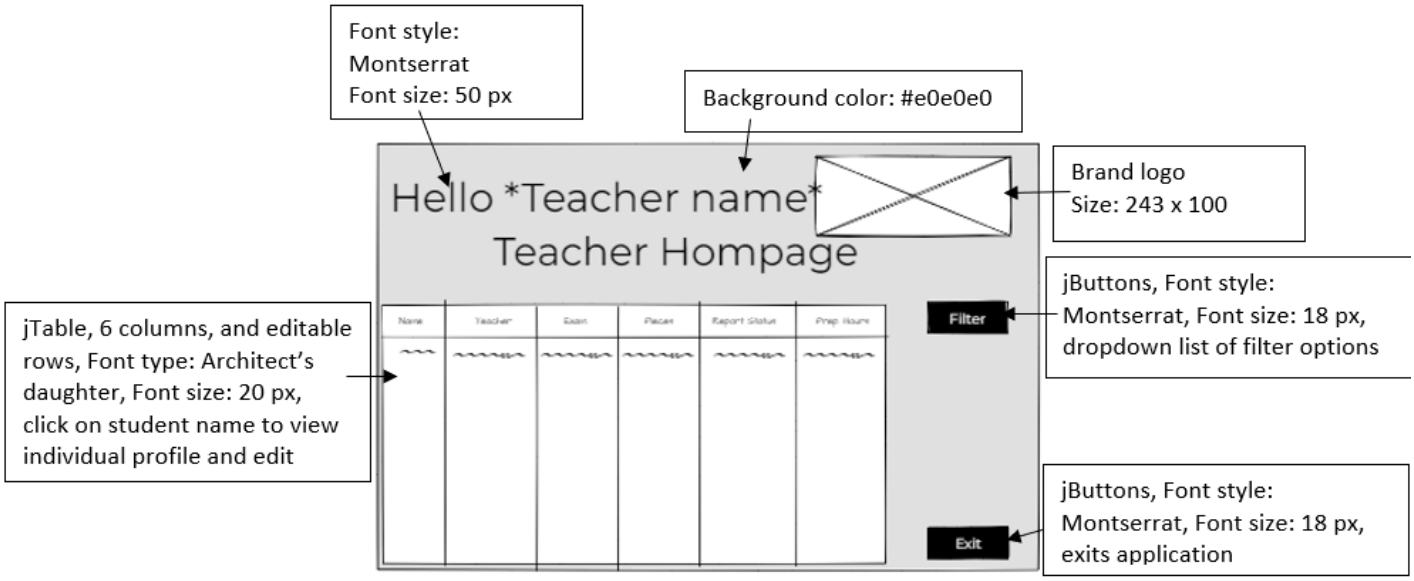


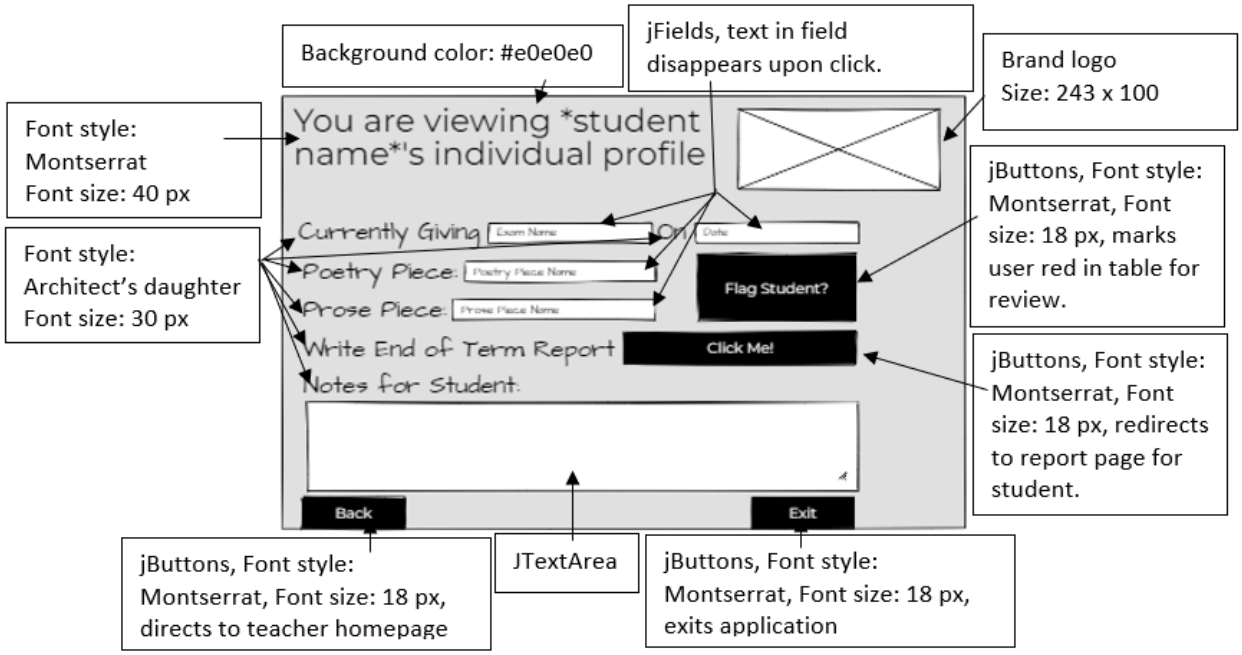
Final GUI Design

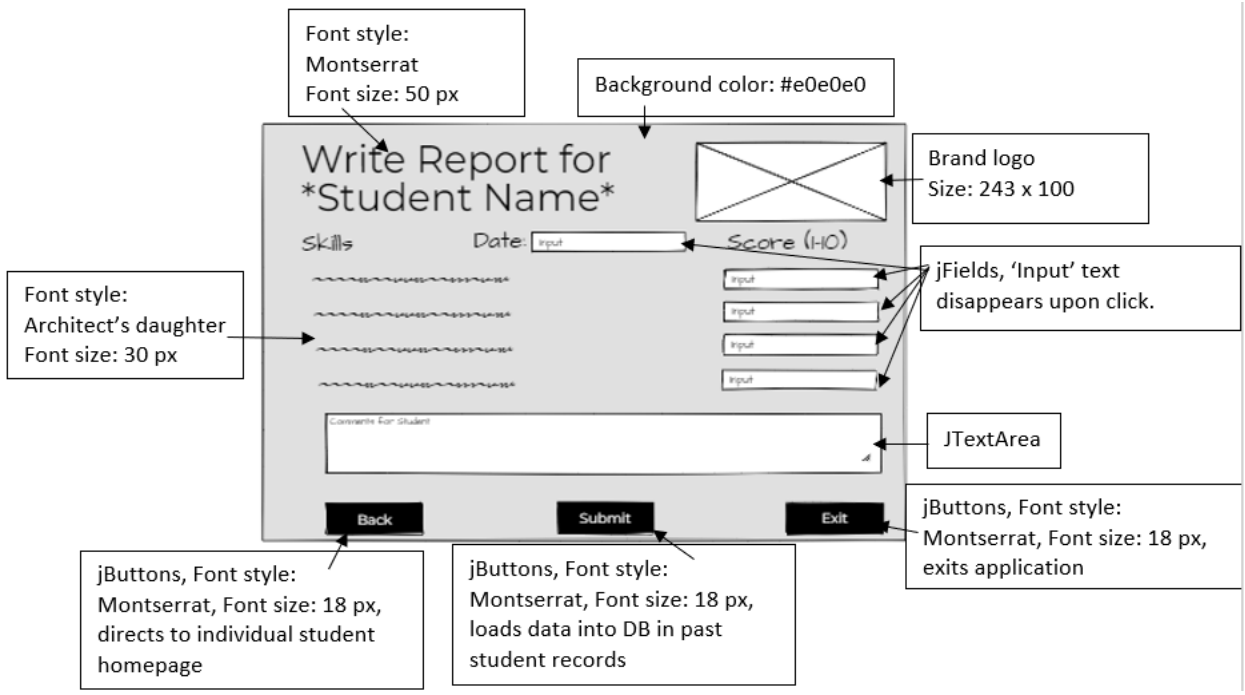
After reworking on the initial sketches based on feedback from VOCAL ***(Refer Appendix 2 for initial sketches and feedback and refer Appendix 1 second interaction for feedback on final designs)***. final designs were created and annotated using Mockflow.

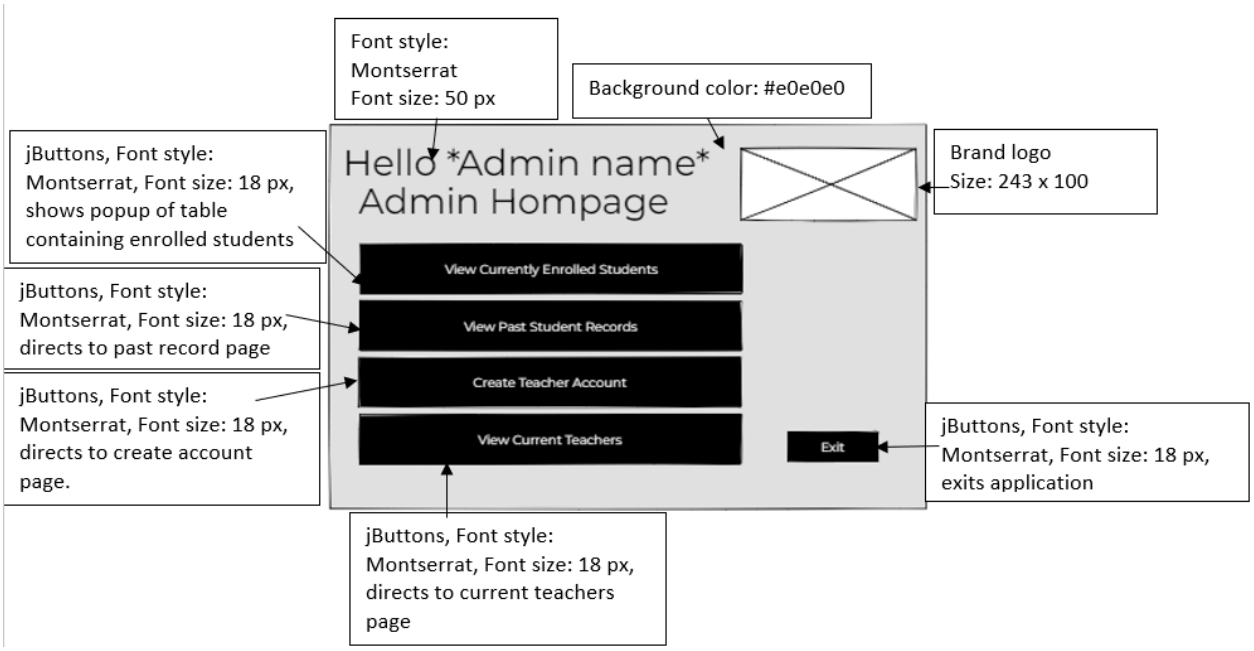


Teacher Homepage

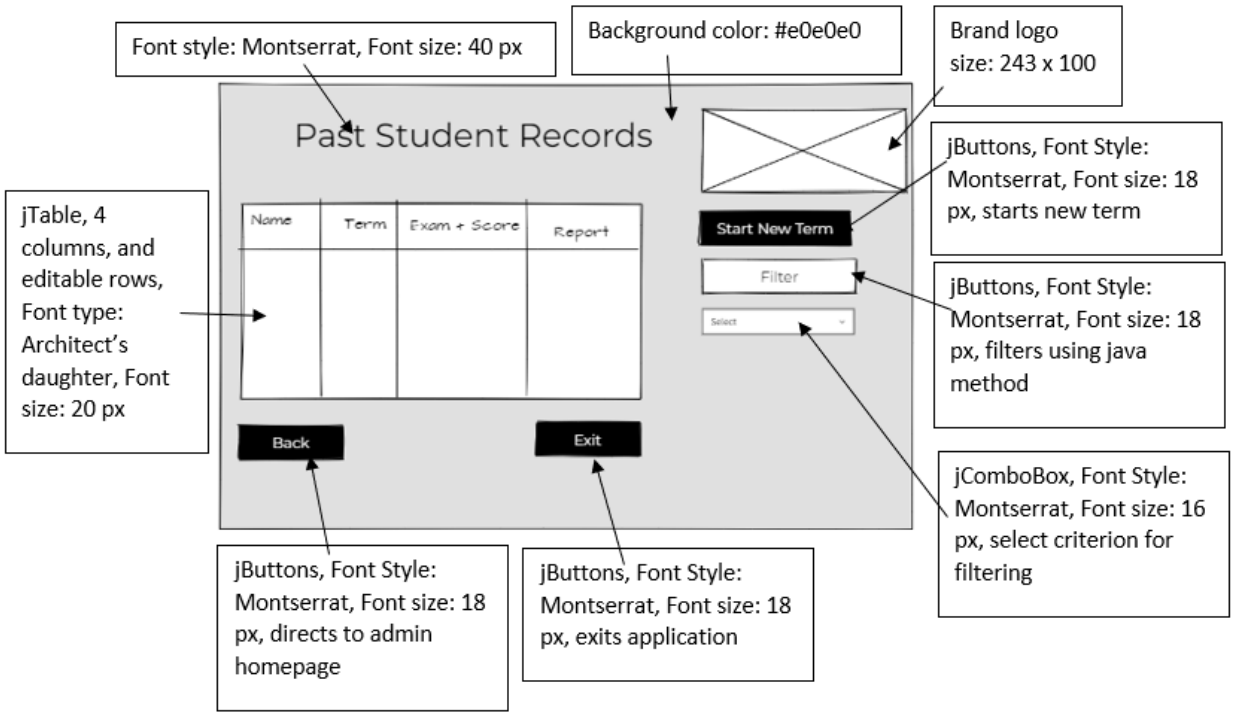


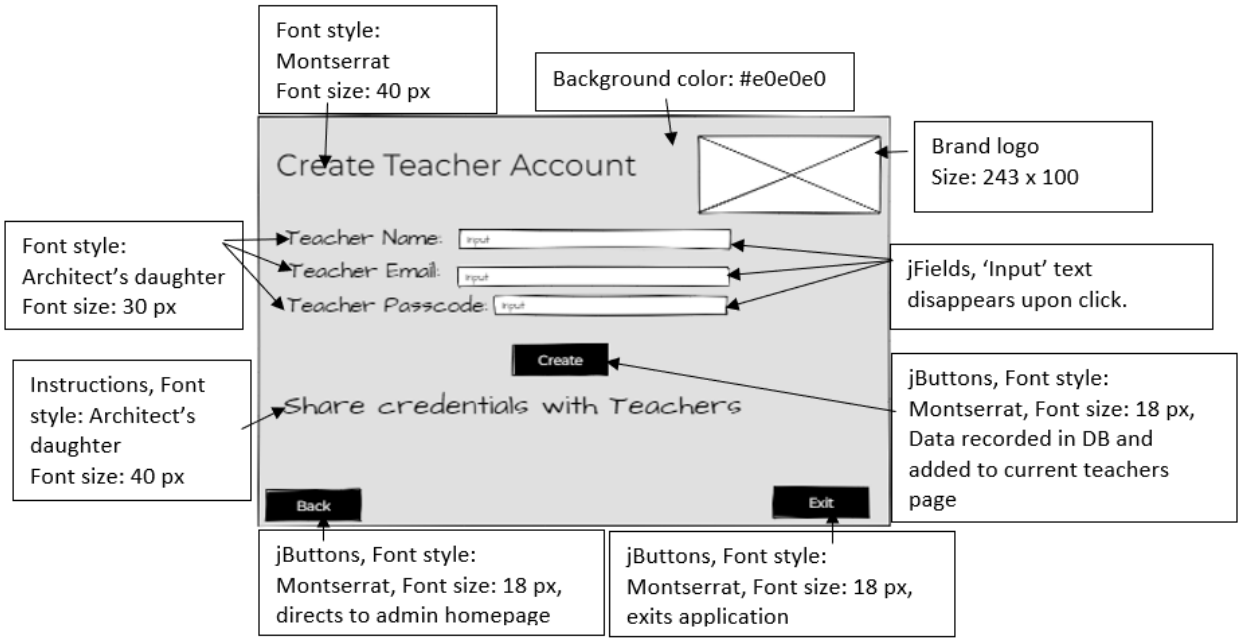

Background color: #e0e0e0
jFields, text in field disappears upon click.
Brand logo
Size: 243 x 100
Font style:
Montserrat
Font size: 40 px
You are viewing *student name*'s individual profile
jButtons, Font style: Montserrat, Font size: 18 px, marks user red in table for review.
Font style:
Architect's daughter
Font size: 30 px
Currently Giving
Exam Name
On
Date
Poetry Piece:
Poetry Piece Name
Prose Piece:
Prose Piece Name
Flag Student?
Write End of Term Report
Click Me!
jButtons, Font style: Montserrat, Font size: 18 px, redirects to report page for student.
Notes for Student:
Back
Exit
jButtons, Font style: Montserrat, Font size: 18 px, directs to teacher homepage
JTextArea
jButtons, Font style: Montserrat, Font size: 18 px, exits application


Font style:
Montserrat
Font size: 50 px
Background color: #e0e0e0
Write Report for *Student Name*
Brand logo
Size: 243 x 100
Skills
Date:
Input
Score (1-10)
Font style:
Architect's daughter
Font size: 30 px
jFields, 'Input' text disappears upon click.
Input
Input
Input
Input
Comments For Student
JTextArea
Back
Submit
Exit
jButtons, Font style: Montserrat, Font size: 18 px, exits application
jButtons, Font style: Montserrat, Font size: 18 px, directs to individual student homepage
jButtons, Font style: Montserrat, Font size: 18 px, loads data into DB in past student records



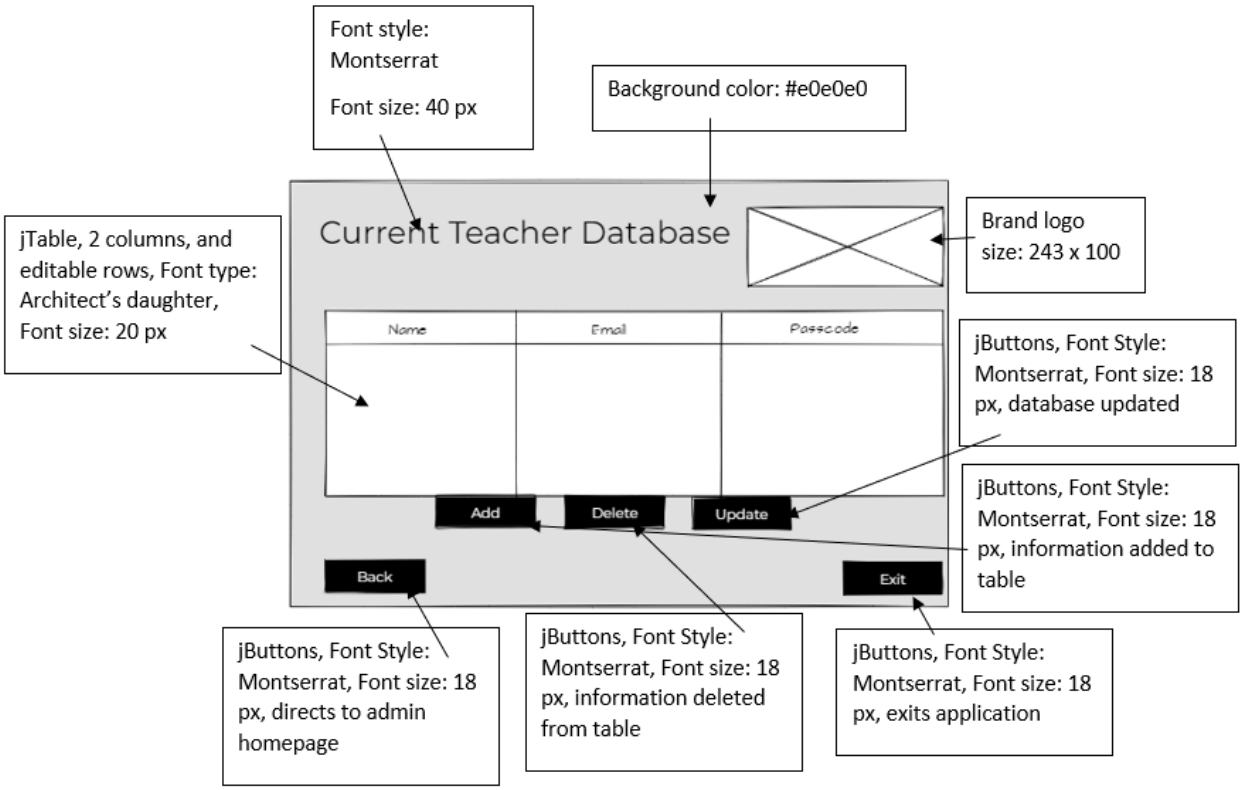
Past Student Records page



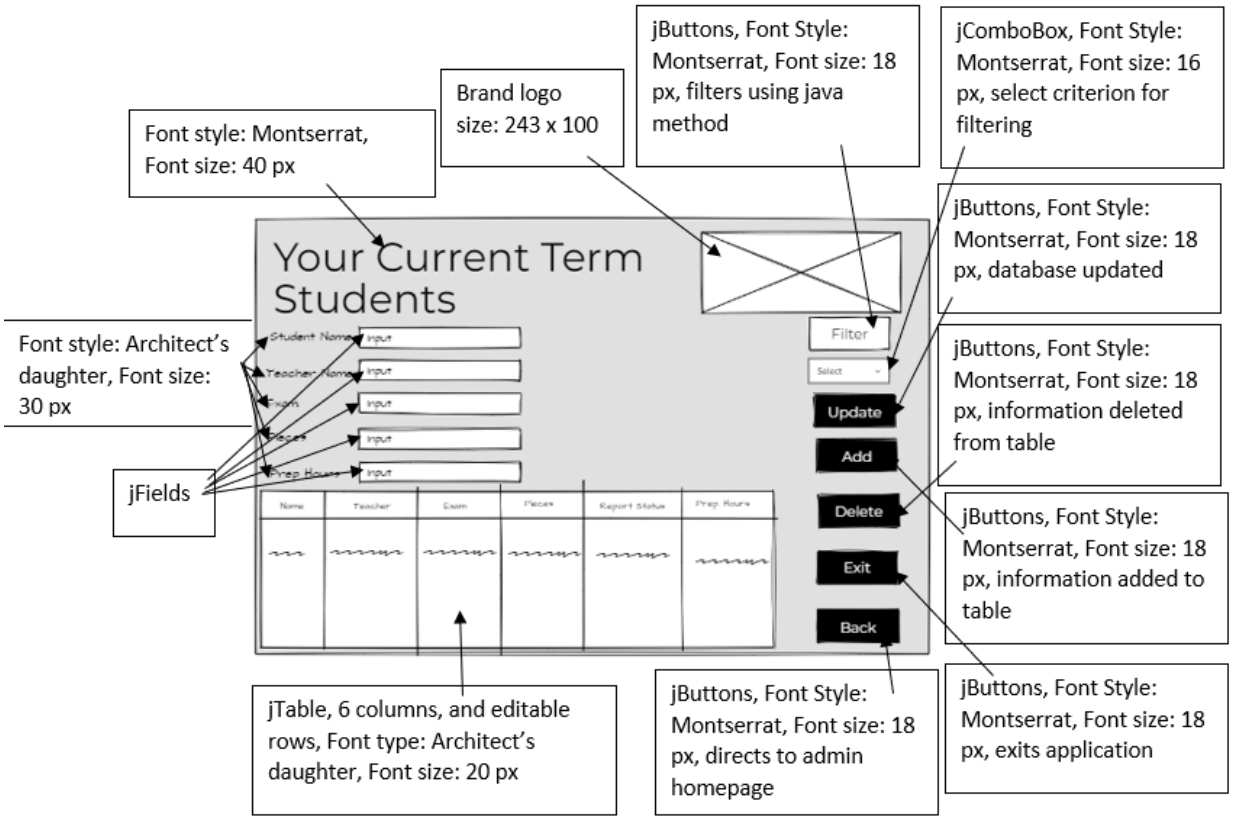


Create Teacher Account Page

Current Teacher Database Page

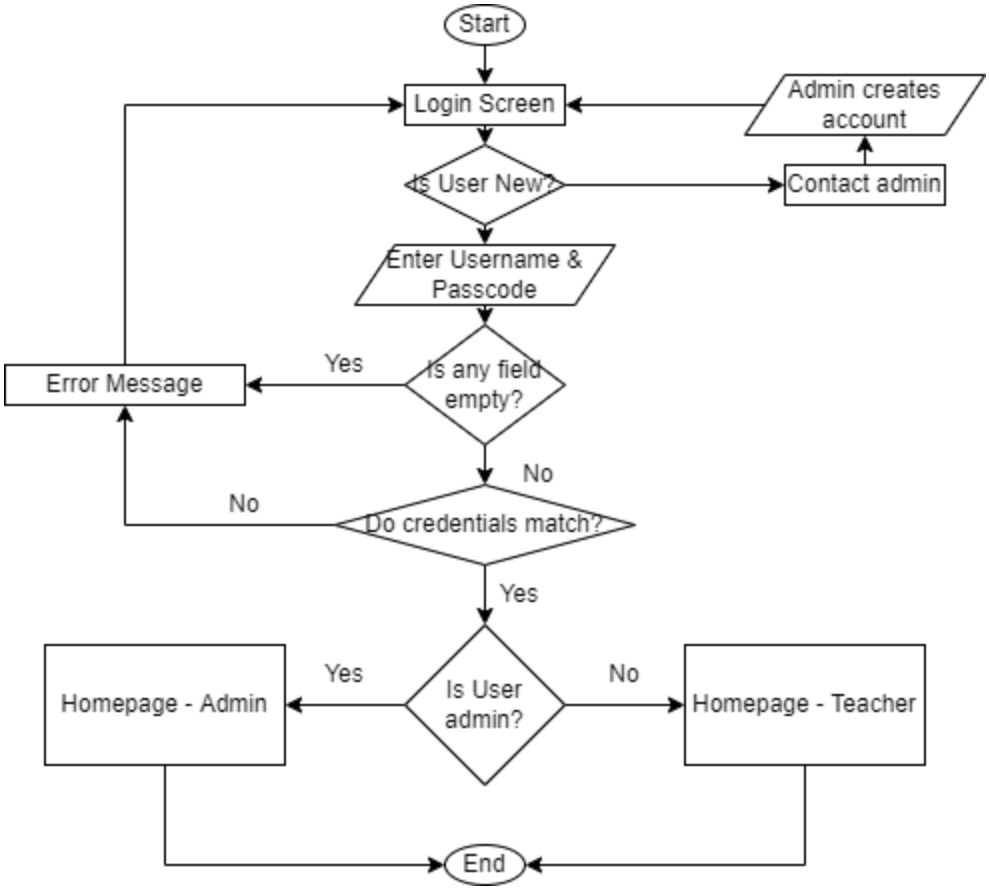


Currently Enrolled Students Page (admin version)



Flowcharts of Pages and Processes

Login Process



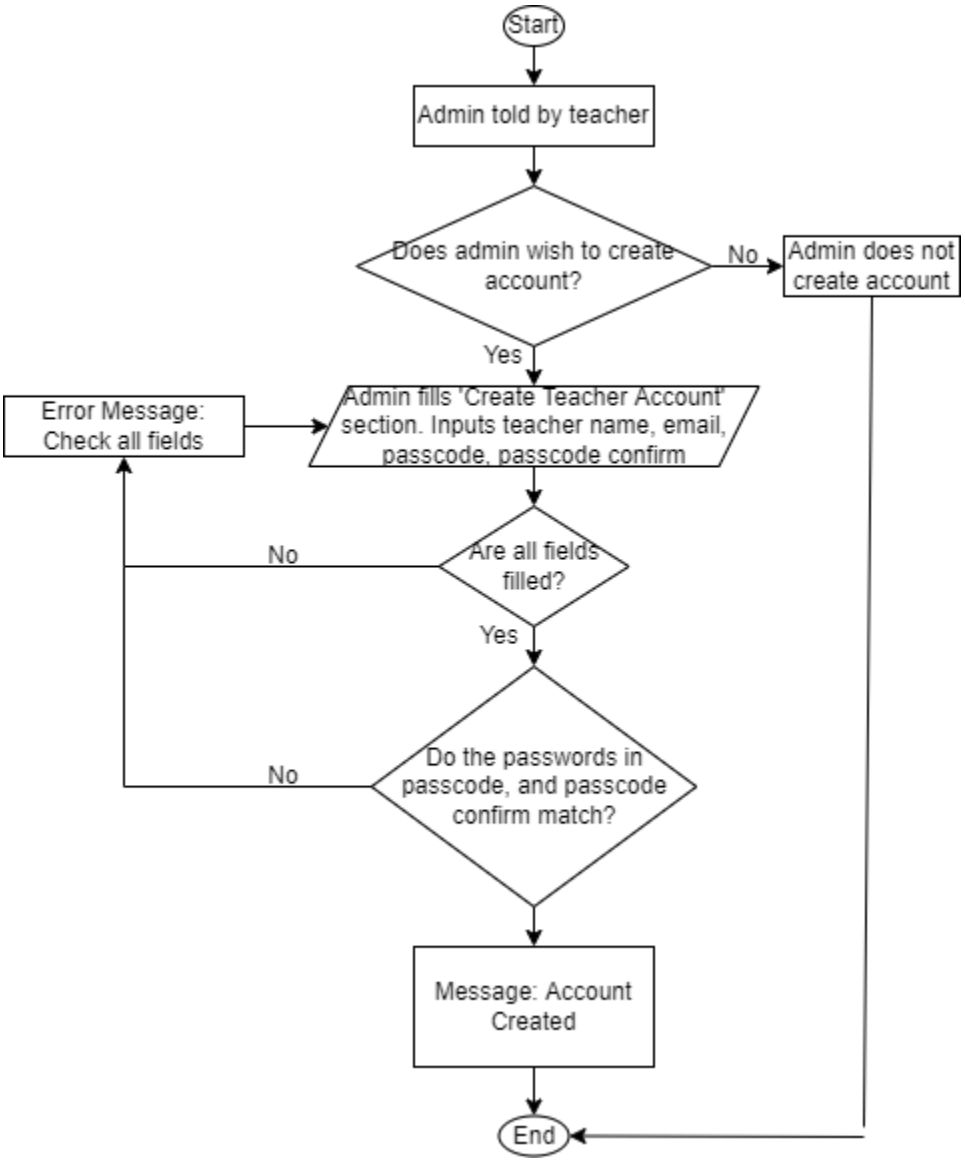
Field Name	Possible Errors	Sample Data
Email	Incorrect email (includes conventions) Missing field	sampleemail@emailaddress.com
Password	Incorrect password Missing field	Password

Algorithm for login page in pseudocode:

Start
Username = input(“EmailID”)
Password = input(“Passcode”)
Inputs are matched with database
If Username == EmailID && Password == Passcode
Output “You are now logged in successfully”

User redirected to respective homepage
Else If Username.isEmpty() == true then
Output “Username and Password do not match”
Else if Password.isEmpty() == true then
Output “Username and Password do not match”
Else
Output “Username or Password is incorrect”
End If

Account Creation Page



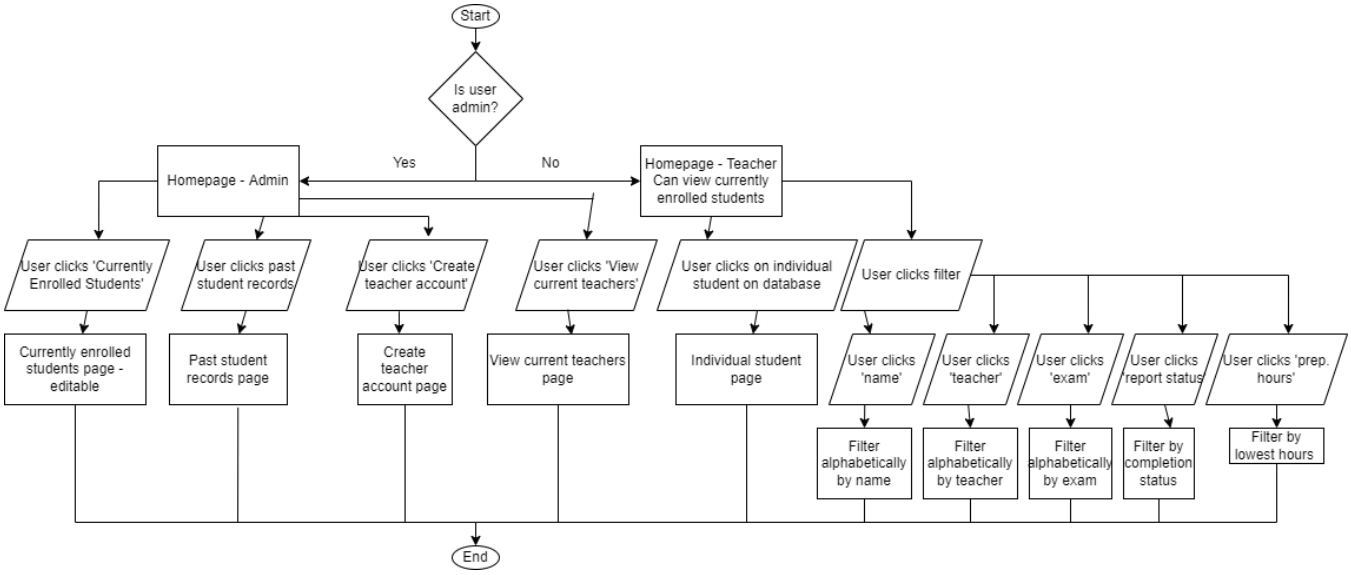
Field Name	Possible Errors	Sample Data

Teacher name	Missing field	Teacher name
Email	Incorrect convention	teachername@emailaddress.com
Passcode	Missing field	Passcode
Confirm Passcode	Missing field Does not match passcode	Passcode

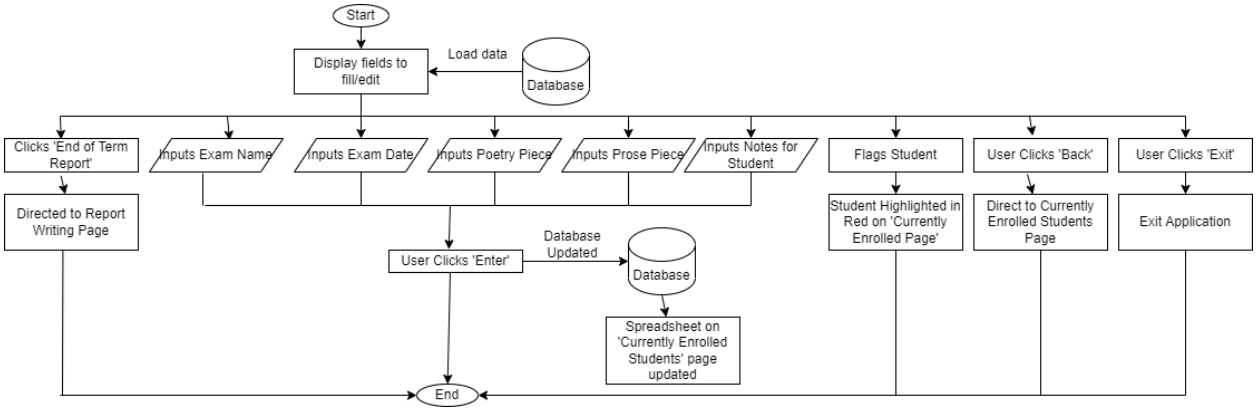
Algorithm for create teacher account page in pseudocode:

```
Start

Response = input // teacher conversing with administrator
If response = true then
TeacherName = input(“teachername”)
EmailAddress = input(“EmailID”)
Password = input(“Passcode”)
Passwordconfirm = input(“Passcode”)
If TeacherName.isEmpty() ==  true then
Output “Fields not filled correctly”
Else If EmailAddress.isEmpty() == true then
Output “Fields not filled correctly”
End If
If Passwordconfirm == Password then
Output “Account Created Successfully”
Else
Output “Recheck Password”
End If
Else If response = false then
Halt proceeding
End If
```

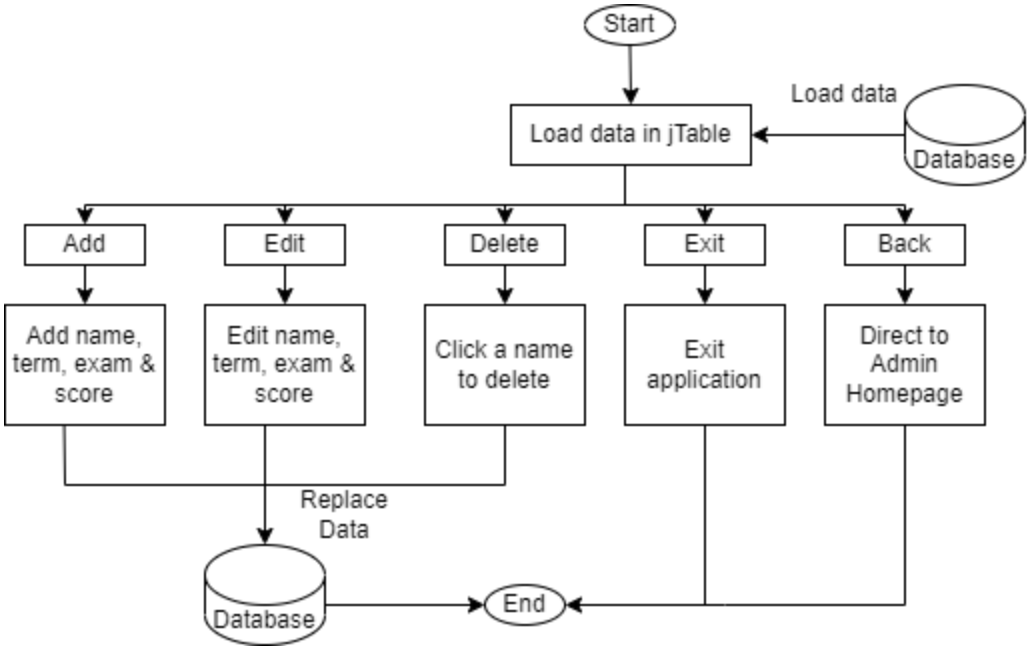
Homepage Processes Page



Individual Student Page Process

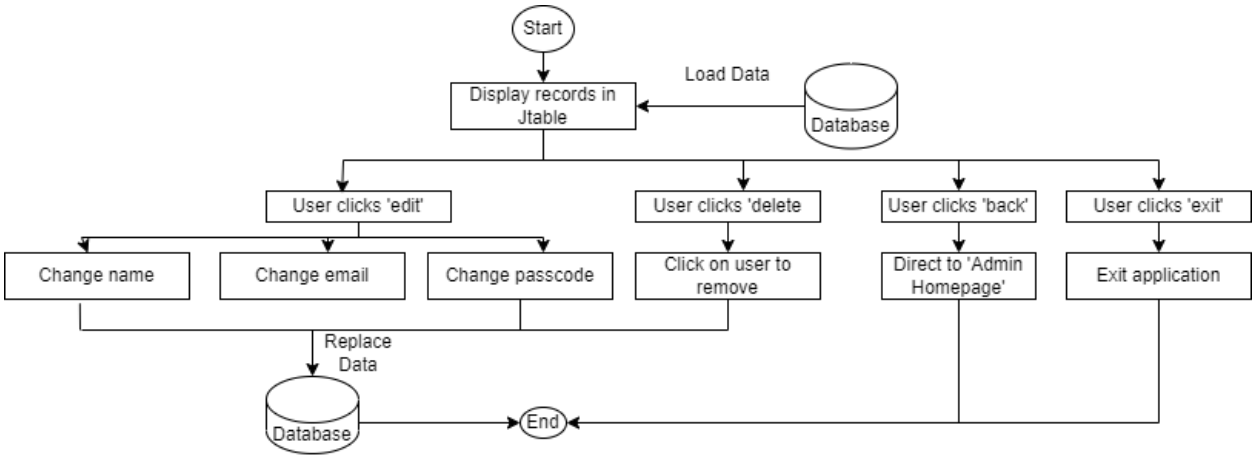


Past Student Records Process



Field Name	Possible Errors	Sample Data
Name	Missing field String input only	Student name
Exam & score	Missing field	Grade 8 acting - 98/100
Term	Missing field Integer input only	2019

Current Teachers Page Process



Field Name	Possible Errors	Sample Data
Teacher name	Missing field	Teacher name
Email	Incorrect convention	teachername@emailaddress.com
Passcode	Missing field	Passcode

Testing Method Plan

Criteria Testing Plan		
1.	There must be an option to filter information on the database, through criteria such as exam, alphabetical student name, pieces, etc	Click on the filter button, and select option on the dropdown list. Wait for the table to update.
2.	Teachers can view all and edit some student information, but cannot add or delete students. Administrators can change any student's information, as well as add and delete from their own page.	Login through teacher credentials and check if delete or add options are there. Do the same through admin credentials and attempt editing information.
3.	Warning and confirmation messages should be displayed	After editing, wait for a popup message to appear asking to confirm.

	when attempting to edit data for all users, on all tables	
4.	The application should be protected by a user login, and password	Ensure invalid credentials are inputted and tested. Ensure password field is used.
5.	In case of user invalidation, the application should portray an error message	Input incorrect credentials, and wait for a popup message to show.
6.	All lists need to be organized as a table, adding to ergonomic appeal	Compare layout of student, past students, current teachers table against created designs. Ensure all table-like conventions are adhered to.
7.	The software should allow a ‘flagging’ feature for students that use the same pieces, or aren’t meeting exam requirements	Go to the student's individual page and click the ‘flag student’ button. Go back to the main table and check if the student's row has been highlighted in red.
8.	The application should have two sections: administrator and user/teacher. Administrators can	Login using an administrator credential and verify homepage and ensure all editing features are present. Repeat likewise with teacher credentials.

	create, edit data, and add teacher accounts.	
9.	Only administrators can view past year information for students	Login using admin credentials and go to ‘past student records’ to see if records are editable and available. Similarly, login using teacher credentials, ensuring page is not present.
10.	The user can exit the application at any time, but would need to login again once opened	Login into software, and then click exit. Open software again and check if login is required.
11.	The institution’s logo should be visible on each page of the application	Check the top right hand corner of each page if the logo is present.
12.	The application should be simplistic and efficient to use, aiding navigability	Wait for feedback from user.
13.	All buttons need to direct user to the correct page	Test all buttons to test whether they reach the right page
14.	All data in application should match the data entered by the user and recorded in database	Crosscheck between SQL table and application table after editing to ensure uniformity.

15.	Input text fields must accept information and perform validation checks	Test validation by inputting incorrect data type or format and clicking continue
16.	There should be an option to start a new term, which will erase the currently enrolled students table and label past students by the term	Click “Start New Term”, and validate whether the currently enrolled students table is clear and term is visible on past student records page.

Table Schema

Table: multiuserlogin

Field Name	Description	Data Type	Default Value
IDTEACHER	Primary Key for teacher (unique identifier)	UIUD(36)	Randomly Generated
TeacherName	Name of teacher	VARCHAR(20)	Name added by admin
EmailAddress	Username for teacher	VARCHAR(20)	Email added by admin
Passcode	Access code for teacher to software	VARCHAR(20)	Password added by admin
UserType	Differentiates between admin and user	VARCHAR(20)	Usertype set by admin

Table: currentlyenrolledstudentsadmin

Field Name	Description	Data Type	Default Value

ID	Unique identifier	UIUD(36)	Randomly Generated
studentname	Currently enrolled student name	VARCHAR(20)	Name added by admin
teachername	Teacher name	VARCHAR(20)	NULL
examname	Exam student is appearing for	VARCHAR(40)	NULL
piecenames	Pieces student is using	VARCHAR(60)	NULL
reportstatus	Checks whether report is written or not	VARCHAR(15)	“Not Done”
prephours	Number of hours student has spent in class	TINYINT(255)	0
flag	Boolean operator	Boolean	0

Table: current_teachers

Field Name	Description	Data Type	Default Value
teacherName	Teacher name	VARCHAR(20)	Name added by admin
emailName	Username for teacher	VARCHAR(20)	Email added by admin
password	Access code for teacher to software	VARCHAR(20)	Password added by admin

Table: past_student_records

Field Name	Description	Data Type	Default Value
sID	Foreign Key	UIUD (36)	Randomly generated in currentlyenrolledstudentsadmin table

Sname	Name of student	VARCHAR(20)	Name added by admin
ExamScore	Name of exam given and score	VARCHAR(255), TINYTEXT	NULL
Term	Term exam was given	TINYINT(4)	NULL
Report	Checks report status	VARCHAR(15)	“Not Done”

Requirements

Serial Number	Requirement	Purpose
1.	SQLite Studio (version (3.35.4)) https://sqlitestudio.pl/	This database management system will be used to store the backend information of the application and will handle user files and data.
2.	Apache NetBeans IDE 13 for Windows	This IDE will be used for designing and coding the application using Java.
3.	Chrome	This is a web browser that will allow access to tutorials on sites such as YouTube, as well as downloadable software that may be needed when coding.
4.	Laptop: Minimum Requirements - 1. 2GB Minimum Storage 2. 8GB RAM 3. Processor - Intel Core i5 or higher	In order to create the software as well as download the software, a laptop is required. In order to deal with multiple students’ data, a larger storage space is required.

Method Definitions

Class	Method name	Description
All classes	*connectDB()*	Connection to database
All classes	*buttonexit*	Exits system
Create account	*buttoncreate*	Performs convention checks, saves information to database, and portrays on ‘current_teachers’
Create account	*buttonback1*	Redirects to admin homepage
Login	*buttonlogin*	Performs validation check, portrays error message if error, checks whether user is admin or teacher, and redirects to respective homepage
Teacher Home	*buttonfilter*	Provides dropdown list to pick criteria for filtering
Individual Student Page	*buttonflag*	Saves input into database, highlights row in red on ‘current_students’ table
Individual Student Page	*buttonreport*	Directs to student report page
Individual Student Page	*buttonback2*	Redirects to teacher homepage
Admin Homepage	*buttonpaststudents*	Redirects to ’past students page’
Admin Homepage	*buttonteacheraccount*	Redirects to create teacher account page
Admin Homepage	*buttoncurrentteachers*	Redirects to current teachers page
Admin Homepage	*buttoncurrentstudents*	Redirects to current student page for admin
Student report page	*buttonback3*	Redirects to individual student page
Student report page	*buttonsubmit*	Checks if all fields are filled, updates report status on ‘current_students’, saves to

		database, becomes viewable on ‘past_students’ table
Past student records	*buttonfilter2*	Provides dropdown list with filter options for ‘past_students’ table
Current Teacher Page	*buttonaddCTP*	Adds row that user can fill information in
Current Teacher Page	*buttondeleteCTP*	Deletes information in row user clicks on
Current Teacher Page	*buttonupdateCTP*	Updates database after table is edited

Validation

Scenario	Character Type	Name of check	Description	Popup error message
Username	String	Verification, Format	Check if the username is present in the database. Also check if conventions have been met	“Username or \ password incorrect” “Username not meeting conventions”
Password	Textfield	Verification	Check if the password is present in the database, and matches the username.	“Username or password incorrect”
Confirming password (create account page)	Textfield	Verification	Check if this field is the same as the password entered above.	“Password does not match with above field entry”
Email convention (create account page)	String	Format	Check if the username meets valid email conventions.	“Username does not meet email conventions”
All Tables	Integer String	Length Character type	Perform checks on length of inputs, as well as type	“Length limit exceeded”

	Character		of characters being inputted.	“Incorrect character type used”
All Forms/Reports/ Tables	Any	Missing fields	Checks if any field is left unfilled before submitting.	“All fields not filled”

(Refer Appendix 3 for complete code)

Sr. No.	Methods Used	Implementation
1	Field Validations	Sign-in, Currently Enrolled Students Admin, Past Student Records, Create Teacher Account, Report, Individual Student, Current Teachers pages
2	Personalized UI Components	Teacher and Admin Homepages
3	Authentication	Sign-in Page
4	Dynamic and Responsive UI	Currently Enrolled Students Admin, Individual Student, Report pages
6	UI and SQL Connection	All Pages
7	Primary	Currently Enrolled Students Admin Page, MySQL
8	User Defined Methods	All Pages
9	Database handling and querying in MySQL	All Pages
10	Table Customization for Color Coding Rows	Currently Enrolled Students Admin, Teacher Homepage, Individual Student pages
11	Flag Button - Color Coding Rows	Currently Enrolled Students Admin, Teacher Homepage, Individual Student pages
12	Filtering	Currently Enrolled Students Admin, Teacher Homepage, Past Student Records pages
13	Exception Handling	All pages
14	Use of Additional Libraries	All pages
15	Basic operations: add, update, delete	Current Teachers Page, Currently Enrolled Students Page

Field Validations

```
private void buttonloginActionPerformed(java.awt.event.ActionEvent evt) {
   try{
      String regex = "^[a-zA-Z0-9+_.-]+@[a-zA-Z0-9.-]+$";
      Pattern pattern = Pattern.compile(regex);
      Matcher matcher = pattern.matcher(edEmail.getText());
      if(matcher.matches()) {
         String query = "SELECT * FROM `multiuserlogin` WHERE EmailAddress=? and Passcode=? and UserType=?";
      // load and register JDBC driver for MySQL
      con = DriverManager.getConnection("jdbc:mysql://localhost/multiuserlogin", "root", "");
      pst = con.prepareStatement(query);
      pst.setString(1, edEmail.getText());
      pst.setString(2, jPasswordField1.getText());
      pst.setString(3, String.valueOf(usertype.getSelectedItem()));
      ResultSet rs=pst.executeQuery();
      if(rs.next()) {
      JOptionPane.showMessageDialog(this, "username and password matched and you are signed in as " + rs.getString("usertype") );
      if(usertype.getSelectedIndex() == 0){
         adminhomepage adminhp = new adminhomepage();
         adminhp.setName(rs.getString("TeacherName"));
         adminhp.setVisible(true);
         this.setVisible(false);
      }else {
        teacherhomepage t = new teacherhomepage();
        t.setName(rs.getString("TeacherName"));
        t.setVisible(true);
        this.setVisible(false);
      }}
      else{
            JOptionPane.showMessageDialog(this, "username and password do not match");}

      }
```

Personalized UI Components

The user defined method ‘setName’ uses this stored value to display the name in the next page after signing in.

```
String adminname;
public void setName(String name){
    adminname = name;

}
   public void labelName() {
       namelabel.setText(adminname);

   };
```

Authentication

The code connects to the ‘multiuserlogin’ page using the ‘getConnection’ method. If there is a match between SQL table and inputted information, the user is directed to the correct homepage.

```
private void btnSignInActionPerformed(java.awt.event.ActionEvent evt) {
   try{
       String query = "SELECT * FROM `multiuserlogin` WHERE EmailAddress=? and Passcode=? and UserType=?";
       // load and register JDBC driver for MySQL
       con = DriverManager.getConnection("jdbc:mysql://localhost/multiuserlogin", "root", "");
       pst = con.prepareStatement(query);
       pst.setString(1, edEmail.getText());
       pst.setString(2, edPassword.getText());
       pst.setString(3, String.valueOf(usertype.getSelectedItem()));
       ResultSet rs=pst.executeQuery();
       if(rs.next()) {
       JOptionPane.showMessageDialog(this, "username and password matched and you are logined as " + rs.getString("usertype") );
       if(usertype.getSelectedIndex() == 0){
           adminhomepage adminhp = new adminhomepage();
           adminhp.setName(rs.getString("TeacherName"));
           adminhp.setVisible(true);
           this.setVisible(false);
       }else {
         teacherhomepage t = new teacherhomepage();
         t.setName(rs.getString("TeacherName"));
         t.setVisible(true);
         this.setVisible(false);
       }}
       else{
               JOptionPane.showMessageDialog(this, "username and password do not match");}
   }catch(Exception ex){
   JOptionPane.showMessageDialog(this, ex.getMessage());
}
```

Dynamic and Responsive UI

This code uses DefaultTableModel to form a table model, from which information can be taken for display in the text fields above the table for readability.

```
private void catMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    if(cat.getSelectedColumn() == 5){
        DefaultTableModel RecordTable = (DefaultTableModel)cat.getModel();

    int SelectedRows = cat.getSelectedRow();
            String sid = RecordTable.getValueAt(SelectedRows, 0).toString();
            String sname = RecordTable.getValueAt(SelectedRows, 1).toString();
            individualstudentpage isp = new individualstudentpage();
            isp.setStudent(sid, sname);

           isp.setVisible(true);
           this.setVisible(false);


    }
    else{
    DefaultTableModel RecordTable = (DefaultTableModel)cat.getModel();
    int SelectedRows = cat.getSelectedRow();
    tfsname.setText(RecordTable.getValueAt(SelectedRows, 1).toString());
    tftname.setText(RecordTable.getValueAt(SelectedRows, 2).toString());
    tfexam.setText(RecordTable.getValueAt(SelectedRows, 3).toString());
    tfpieces.setText(RecordTable.getValueAt(SelectedRows, 4).toString());

    tfpreph.setText(RecordTable.getValueAt(SelectedRows, 6).toString());
    }
}
```

UI and SQL Connection

JDBC driver is used to connect with SQL databases. The getConnection user defined method establishes connection between the user-interface and the corresponding databases.

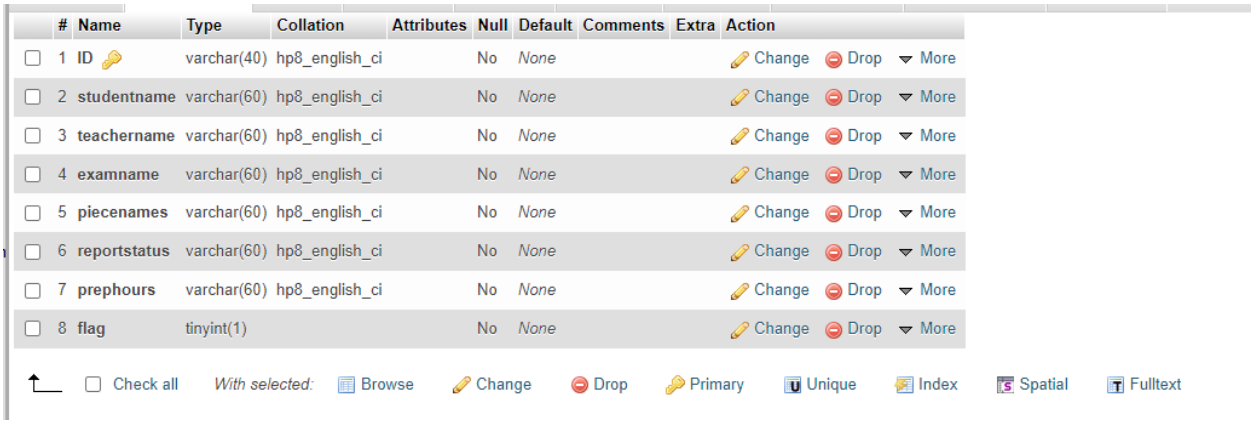
```
 private static final String username = "root";
 private static final String password = "";
 private static final String dataConn = "jdbc:mysql://localhost:3306/currentlyenrolledstudentsadmin";
 private static final String dataConn2 = "jdbc:mysql://localhost:3306/individualstudentreport";

Connection sqlConn = null;
PreparedStatement pst = null;
ResultSet rs = null;
int q, i, id, deleteItem;
```

```
public Connection getConnection() {
    try{
        Class.forName("com.mysql.jdbc.Driver");

    }catch(ClassNotFoundException ex) {
        System.out.println(ex.getMessage());
    }
    Connection con = null;
    try{
        con = DriverManager.getConnection("jdbc:mysql://localhost/individualstudentreport", "root", "");
    } catch(SQLException ex) {
        System.out.println(ex.getMessage());

    }
    return con;

}
```

Primary Keys

A primary key has been used to uniquely identify each student, by importing the UIUD package and generating a random 36 character value for each student.

```
try {
    Class.forName("com.mysql.jdbc.Driver");
    sqlConn = DriverManager.getConnection(dataConn, username, password);
    pst = sqlConn.prepareStatement("INSERT INTO `currentlyenrolledstudentsadmin`(`ID`,`studentname`, `teachername`, `examname`, `piecenames`, `reportstatus`, "
            + "`prephours`) VALUES (?,?,?,?,?,?,?)");
  String studentid = UUID.randomUUID().toString().replaceAll("_", "");
    System.out.println(studentid);
    pst.setString(1, studentid);

    pst.setString(2, tfsname.getText());
    pst.setString(3, tftname.getText());
    pst.setString(4, tfexam.getText());
    pst.setString(5, tfpieces.getText());
    pst.setString(6, "Not Done");
    pst.setString(7, tfpreph.getText());
```



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	varchar(40)	hp8_english_ci		No	None			Change Drop More
2	studentname	varchar(60)	hp8_english_ci		No	None			Change Drop More
3	teachername	varchar(60)	hp8_english_ci		No	None			Change Drop More
4	examname	varchar(60)	hp8_english_ci		No	None			Change Drop More
5	piecenames	varchar(60)	hp8_english_ci		No	None			Change Drop More
6	reportstatus	varchar(60)	hp8_english_ci		No	None			Change Drop More
7	prephours	varchar(60)	hp8_english_ci		No	None			Change Drop More
8	flag	tinyint(1)			No	None			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index Spatial Fulltext

User Defined Methods

Methods have been created for ease of code. This method takes information from the ‘individualstudentreport’ database to fill the report page for a particular student that has been selected.

```
String sid1;
String sname1;
public void setStudent(String sid, String sname) {
    sid1 = sid;
    sname1 = sname;
    fillfields();

}
 JFrame prevPage;
public void setPreviousPage(JFrame prevPage1) {
    prevPage = prevPage1;
}
```

```
public void fillfields(){
    try {
        sqlConn = DriverManager.getConnection(dataConn2, username, password);
        pst = sqlConn.prepareStatement("SELECT * FROM `individualstudentreport` WHERE IDSTUDENT = '"+sid1+"';");

        rs = pst.executeQuery();
        rs.next();
        btDate.setText(rs.getString("Date"));
        txtS1.setText(rs.getString("Skill1"));
        txtS2.setText(rs.getString("Skill2"));
        txtS3.setText(rs.getString("Skill3"));
        txtS4.setText(rs.getString("Skill4"));
        txtNote.setText(rs.getString("Notes"));

    } catch (SQLException ex) {
        Logger.getLogger(reportpage.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

Database handling and querying in MySQL

SQL commands are used in the ‘add’, ‘delete’, and ‘update’ buttons to update tables, as well as when the window is opened to automatically fill the table.

```
private void btnaddActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        Class.forName("com.mysql.jdbc.Driver");
        sqlConn = DriverManager.getConnection(dataConn, username, password);
        pst = sqlConn.prepareStatement("INSERT INTO `currentlyenrolledstudentsadmin`(`ID`,`studentname`, `teachername`, `examname`, `piecenames`, `reportstatus`, "
                + "`prephours`) VALUES (?,?,?,?,?,?,?)");
      String studentid = UUID.randomUUID().toString().replaceAll("_", "");
        System.out.println(studentid);
        pst.setString(1, studentid);

        pst.setString(2, tfsname.getText());
        pst.setString(3, tftname.getText());
        pst.setString(4, tfexam.getText());
        pst.setString(5, tfpieces.getText());
        pst.setString(6, "Not Done");
        pst.setString(7, tfpreph.getText());

        pst.executeUpdate();
        JOptionPane.showMessageDialog(this, "Record Added");
        upDateDB();
```

Update code:

```
private void BtnSubmitActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Connection con = getConnection();
    Statement st;
    Connection con2 = getConnection2();
    Statement st2;
    Connection con3 = getConnection3();
    Statement st3;

    try {
        st = con.createStatement();
        String date = btDate.getText();
        String skill1 = txtS1.getText();
        String skill2 = txtS2.getText();
        String skill3 = txtS3.getText();
        String skill4 = txtS4.getText();
        String notes = txtNote.getText();


            String updateQuery = "UPDATE `individualstudentreport` SET `Date`='"+date+"',`Skill1`='"+skill1+"',`Skill2`='"+skill2+"',`Skill3`='"+skill3+"',"
                    + "`Skill4`='"+skill4+"', `Notes`='"+notes+"' WHERE `IDSTUDENT` = '"+sid1+"'";
            st.addBatch(updateQuery);



        int[] updatedRow =  st.executeBatch();
```

Deletion code:

```
private void btndeleteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        Class.forName("com.mysql.jdbc.Driver");
        sqlConn = DriverManager.getConnection(dataConn, username, password);
        pst = sqlConn.prepareStatement("DELETE FROM `currentlyenrolledstudentsadmin` WHERE ID = ?");
            DefaultTableModel model = (DefaultTableModel)cat.getModel();

                String id = model.getValueAt(cat.getSelectedRow(),0).toString();
                pst.setString(1,id);


        pst.executeUpdate();
        JOptionPane.showMessageDialog(this, "Record Deleted");
        //upDateDB();




    }
    catch(ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (SQLException ex){
        java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
```

Fill information in table code:

```
public void fillfields(){
    try {
        sqlConn = DriverManager.getConnection(dataConn2, username, password);
        pst = sqlConn.prepareStatement("SELECT * FROM `individualstudentreport` WHERE IDSTUDENT = '"+sid1+"';");

        rs = pst.executeQuery();
        rs.next();
        btDate.setText(rs.getString("Date"));
        txtS1.setText(rs.getString("Skill1"));
        txtS2.setText(rs.getString("Skill2"));
        txtS3.setText(rs.getString("Skill3"));
        txtS4.setText(rs.getString("Skill4"));
        txtNote.setText(rs.getString("Notes"));

    } catch (SQLException ex) {
        Logger.getLogger(reportpage.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

Using Window Listener to automatically fill table from MySQL

The WindowListener automatically implements methods called when the window opens. Here, the fillTable user defined method has been called to fill the jTable when the window opens.

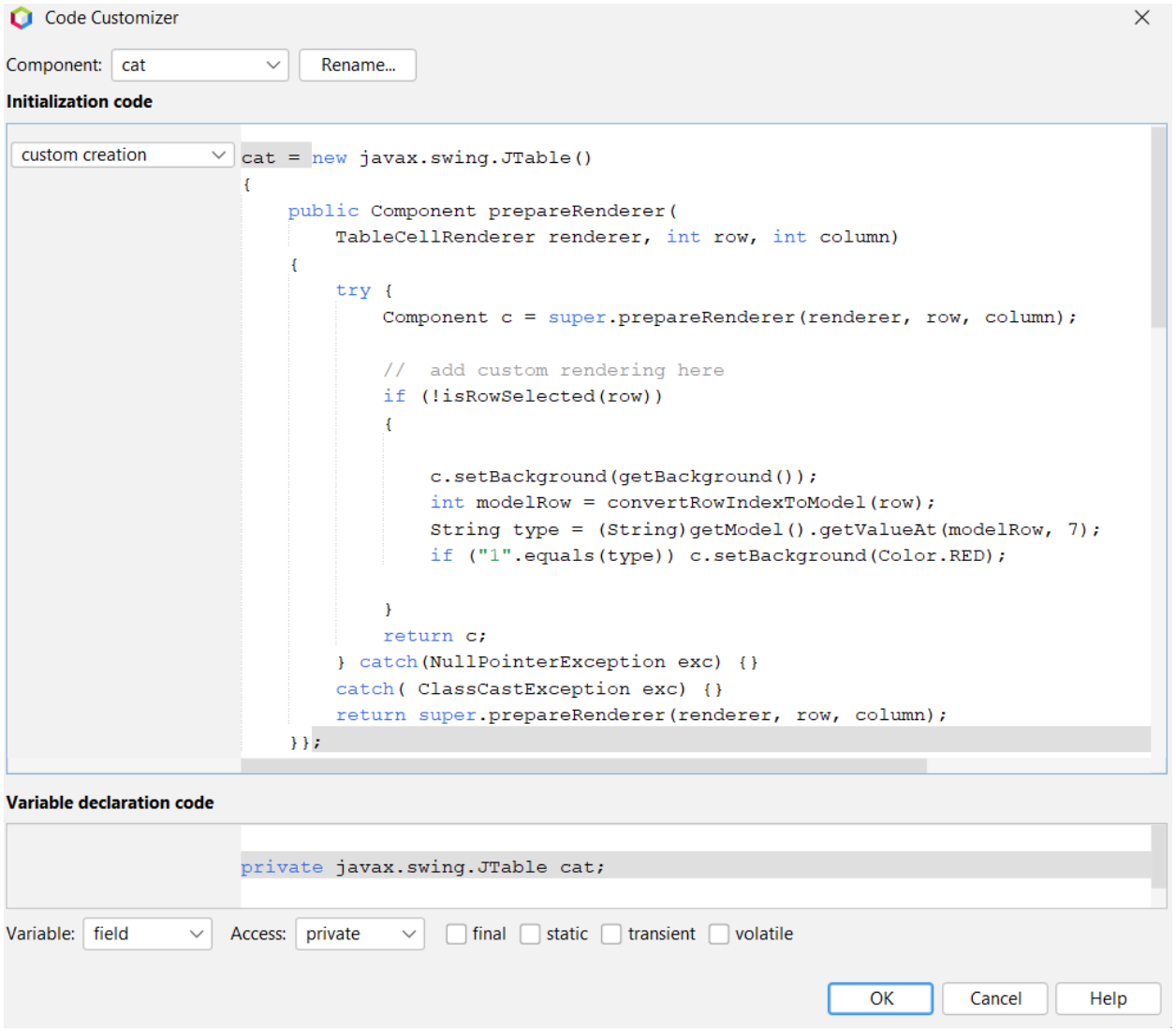
```
private void initSelfListeners() {
 WindowListener taskStarterWindowListener = new WindowListener() {
     @Override
     public void windowOpened(WindowEvent e) {
         fillTable();
         System.out.println("Performing task...");
     }
```

The fillTable method also fills jTables by connecting to SQL databases:

```
public void fillTable() {
    Connection con = getConnection();
    Statement ps;
    ResultSet rs;
    DefaultTableModel model = (DefaultTableModel) psrtab.getModel();
    try {
        ps = con.createStatement();
        rs = ps.executeQuery("SELECT * FROM `paststudentrecords`");

        while(rs.next()) {
            Object[] row = new Object[psrtab.getColumnCount()];
            row[0] = rs.getString("sID");
            row[1] = rs.getString("Sname");
            row[2] = rs.getInt("Term");
            row[3] = rs.getString("ExamScore");
            row[4] = rs.getString("Report");

            model.addRow(row);
        }
    } catch(SQLException ex) {
        System.out.println(ex.getMessage());

    }

}
```

Table Customization for Color Coding Rows

Certain jTables were customized using the ‘Customize Code’ option, to implement a color coding feature when the flag button is clicked. TableCellRenderer, setBackground, and other methods have been imported.



Flag Button Code to Update Table (Ternary Operator)

Using a ternary operator, a particular row in a table can change color to red, or plain, depending on whether the student has been flagged or not. This is achieved through a flag variable in the SQL table that uses values ‘0’ and ‘1’.

```
    try {
        // TODO add your handling code here:
        sqlConn = DriverManager.getConnection(dataConn, username, password);
        pst = sqlConn.prepareStatement("SELECT * FROM `currentlyenrolledstudentsadmin` WHERE ID = '"+sid1+"';");

        rs = pst.executeQuery();
        rs.next();
        boolean flag = rs.getBoolean("flag");
        Connection con = getConnection();
        Statement st;

        st = con.createStatement();
        String updateQuery = "UPDATE `currentlyenrolledstudentsadmin` SET `flag`='"+(flag ? 0 : 1)+"' WHERE `ID` = '"+sid1+"'";
        st.addBatch(updateQuery);

        int[] updatedRow =  st.executeBatch();
        System.out.println(updatedRow.length);

    } catch (SQLException ex) {
        Logger.getLogger(individualstudentpage.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

Filtering Rows

The TableRowSorter method has been imported and used to filter tables. The head columns of the table are also clickable for filtering.

```
    private void btnfilter1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
psrtab.setRowSorter(null);
TableRowSorter<TableModel> sorter = new TableRowSorter<TableModel>(psrtab.getModel());
psrtab.setRowSorter(sorter);

ArrayList<RowSorter.SortKey> sortKeys = new ArrayList<>(25);
sortKeys.add(new RowSorter.SortKey(DropDownFilter.getSelectedIndex()+1, SortOrder.ASCENDING));
sortKeys.add(new RowSorter.SortKey(0, SortOrder.ASCENDING));
sorter.setSortKeys(sortKeys);
    }
```

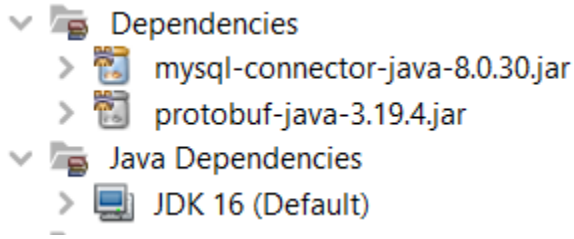
Exception Handling

Exception handling has been used when using SQL commands to prevent java errors.

```
try {
    Class.forName("com.mysql.jdbc.Driver");
    sqlConn = DriverManager.getConnection(dataConn, username, password);
    pst = sqlConn.prepareStatement("DELETE FROM `multiuserlogin` WHERE IDTEACHER = ?");
    DefaultTableModel model = (DefaultTableModel)ctdtab.getModel();
    String id = model.getValueAt(ctdtab.getSelectedRow(),0).toString();

    pst.setString(1,id);
    pst.executeUpdate();
    JOptionPane.showMessageDialog(this, "Record Deleted");
}
catch(ClassNotFoundException ex) {
    java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
} catch (SQLException ex){
    java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
}
```

Use of Additional Libraries

. Mysql-connector has been used to establish database connectivity. Protobuf-java has been used to serialize data.



Basic Operations: Add, Update, Delete

Add Operation:

```
private void buttonaddCTSActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        Class.forName("com.mysql.jdbc.Driver");
        sqlConn = DriverManager.getConnection(dataConn, username, password);
        pst = sqlConn.prepareStatement("INSERT INTO `currentlyenrolledstudentsadmin`(`ID`,`studentname`, `teachername`, `examname`, `piecenames`, `reportstatus`, "
                + "`prephours`, flag) VALUES (?,?,?,?,?,?,?,0)");
      String studentid = UUID.randomUUID().toString().replaceAll("_", "");
        System.out.println(studentid);
        pst.setString(1, studentid);

        pst.setString(2, tfsname.getText());
        pst.setString(3, tftname.getText());
        pst.setString(4, tfexam.getText());
        pst.setString(5, tfpieces.getText());
        pst.setString(6, "Not Done");
        pst.setString(7, tfpreph.getText());

        pst.executeUpdate();
        JOptionPane.showMessageDialog(this, "Record Added");
        upDateDB();

        sqlConn = DriverManager.getConnection(dataConn2, username, password);
        pst = sqlConn.prepareStatement("INSERT INTO `individualstudentreport`(`IDSTUDENT`, `Studentname`, `Date`, `Skill1`, `Skill2`, `Skill3`, "
                + "`Skill4`, `Notes`) VALUES (?,?,null,null,null,null,null,null)");
       System.out.println(studentid);
        pst.setString(1, studentid);

        pst.setString(2, tfsname.getText());
                    pst.executeUpdate();

    Connection con = getConnection();
    Statement ps;
    ResultSet rs;
    try {
        ps = con.createStatement();
        rs = ps.executeQuery("SELECT term FROM `term`");
        rs.next();
        int term = rs.getInt("term");

        sqlConn = DriverManager.getConnection(dataConn3, username, password);
        pst = sqlConn.prepareStatement("INSERT INTO `paststudentrecords`(`sID`, `Sname`, `Term`, `ExamScore`, `Report`) VALUES (?,?,"+String.valueOf(term)+",?,?)");
       System.out.println(studentid);
        pst.setString(1, studentid);

        pst.setString(2, tfsname.getText());
        pst.setString(3, tfexam.getText());
        pst.setString(4, "Not Done");
                    pst.executeUpdate();



    } catch(SQLException ex) {
        System.out.println(ex.getMessage());

    }


    }
    catch(ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (SQLException ex){
        java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);



    }


}
```

Update Operation:

```
    private void buttonupdateCTSActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        Connection con = getConnection();
        Connection con2 = getConnection2();
        Connection con3 = getConnection3();
        Statement st;
        Statement st2;
        Statement st3;
        DefaultTableModel model = (DefaultTableModel)cat.getModel();
        try {
            st = con.createStatement();
            st2 = con2.createStatement();
            st3 = con3.createStatement();
            for(int i = 0; i<model.getRowCount();i++) {
                String id = model.getValueAt(i,0).toString();
                String fn = model.getValueAt(i, 1).toString();
                String tn = model.getValueAt(i, 2).toString();
                String en = model.getValueAt(i, 3).toString();
                String pn = model.getValueAt(i, 4).toString();
                String rst = model.getValueAt(i, 5).toString();
                String ph = model.getValueAt(i, 6).toString();
                String updateQuery = "UPDATE `currentlyenrolledstudentsadmin` SET `studentname`='"+fn+"',`teachername`='"+tn+"',`examname`='"+en+"',"
                        + "`piecenames`='"+pn+"',`reportstatus`='"+rst+"',`prephours`='"+ph+"' WHERE `ID` = '"+id+"'";
                st.addBatch(updateQuery);
                String updateQuery2 = "UPDATE `individualstudentreport` SET `Studentname`='"+fn+"' WHERE `IDSTUDENT` = '"+id+"'";
                st2.addBatch(updateQuery2);
```

```
                String updateQuery3 = "UPDATE `paststudentrecords` SET `Sname`='"+fn+"',`ExamScore`='"+en+"',`Report`='"+rst+"' WHERE `sID` = '"+id+"'";
                st3.addBatch(updateQuery3);


            }
            int[] updatedRow =  st.executeBatch();
            int[] updatedRow2 =  st2.executeBatch();
            int[] updatedRow3 =  st3.executeBatch();
System.out.println(updatedRow.length);
```

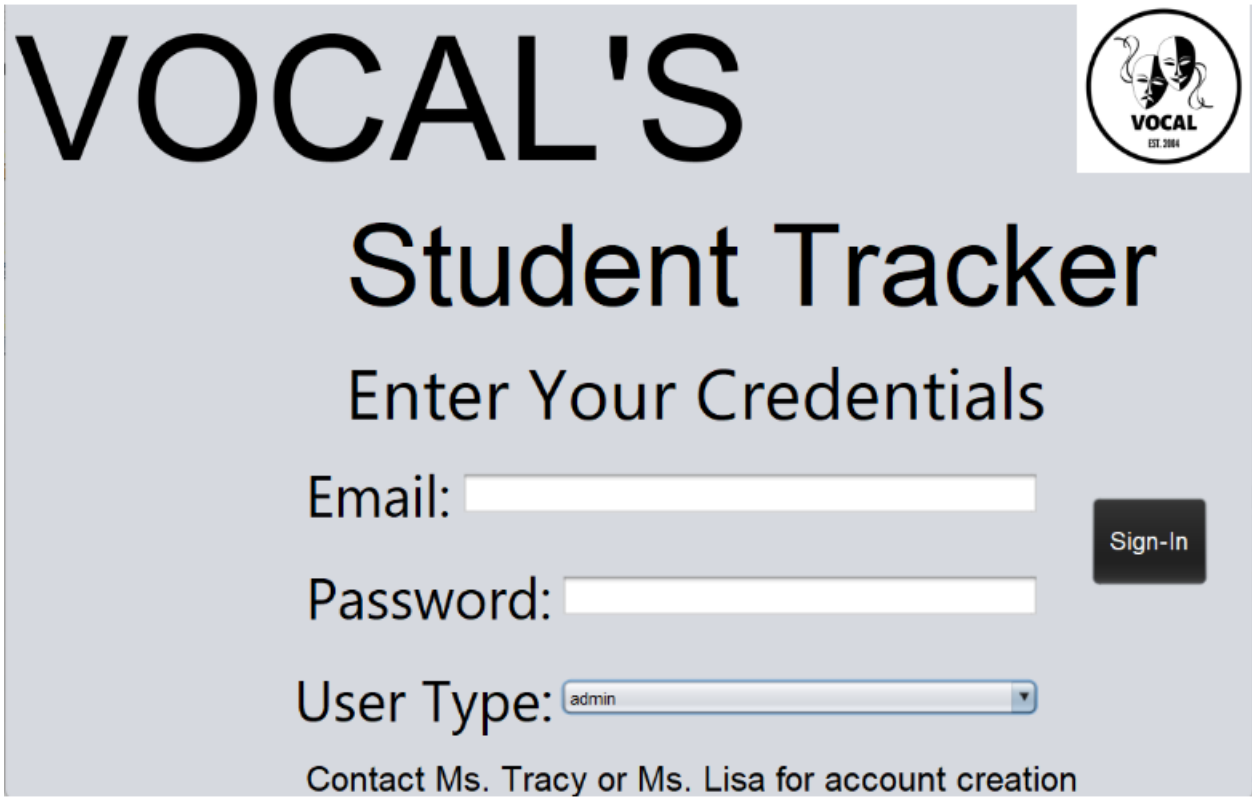
```
            }*/
        } catch (SQLException ex) {
            Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(Level.SEVERE, null, ex);
        }


    }
```

Delete Operation:

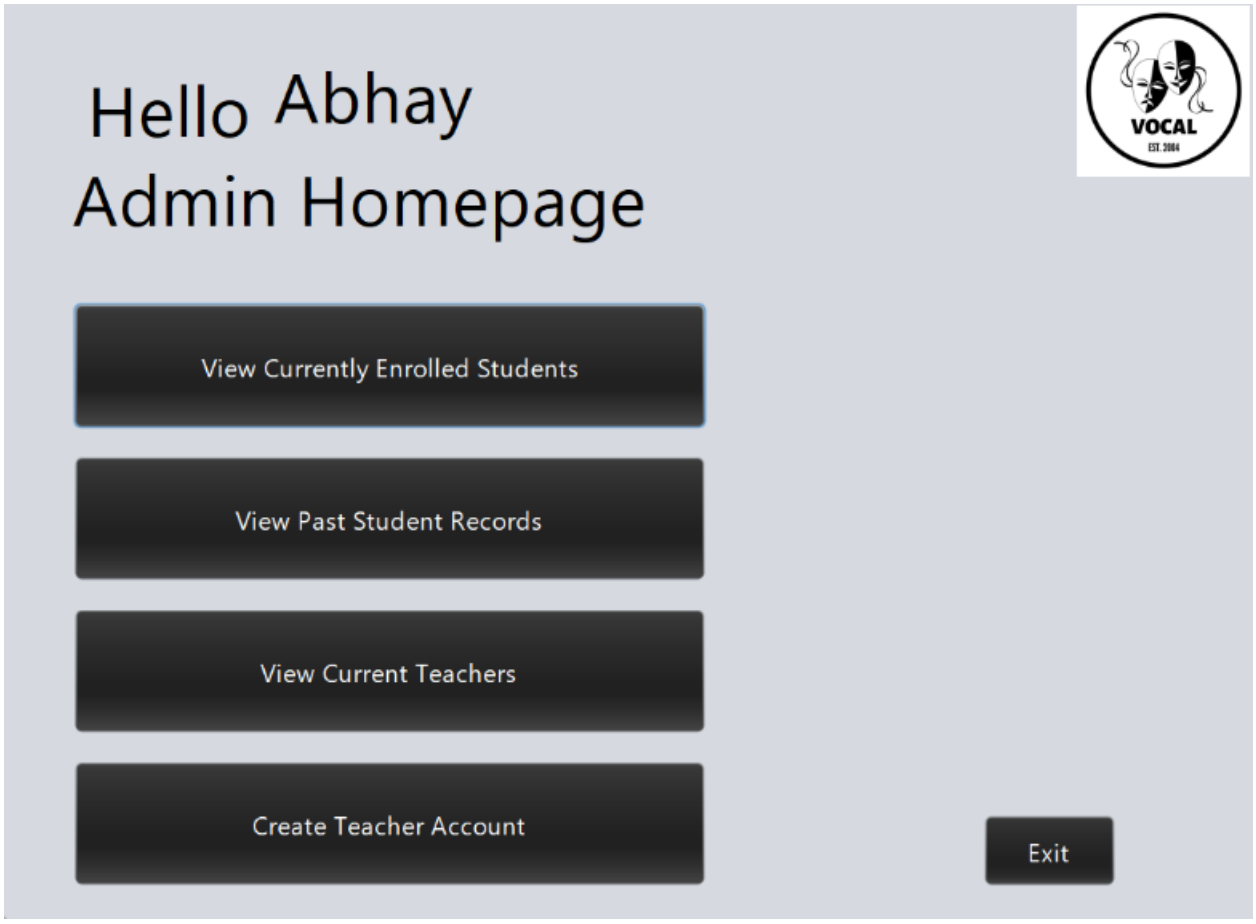
```
    private void buttondeleteCTSActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        try {
            Class.forName("com.mysql.jdbc.Driver");
            sqlConn = DriverManager.getConnection(dataConn, username, password);
            pst = sqlConn.prepareStatement("DELETE FROM `currentlyenrolledstudentsadmin` WHERE ID = ?");
                    DefaultTableModel model = (DefaultTableModel)cat.getModel();

                        String id = model.getValueAt(cat.getSelectedRow(),0).toString();
                        pst.setString(1,id);


            pst.executeUpdate();
            JOptionPane.showMessageDialog(this, "Record Deleted");
            //upDateDB();




        }
        catch(ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (SQLException ex){
            java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);


    }
    ((DefaultTableModel)cat.getModel()).removeRow(cat.getSelectedRow());

}
```

Coded Program Screenshots

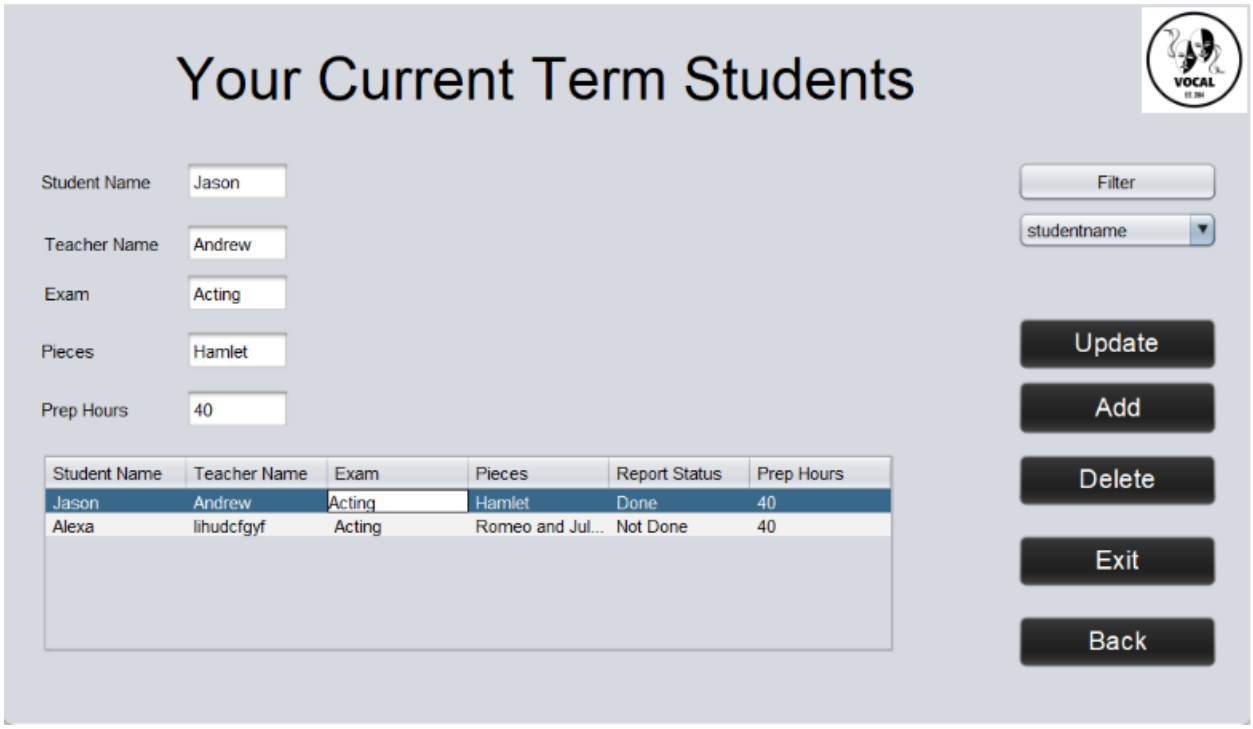
Sign-In Page:



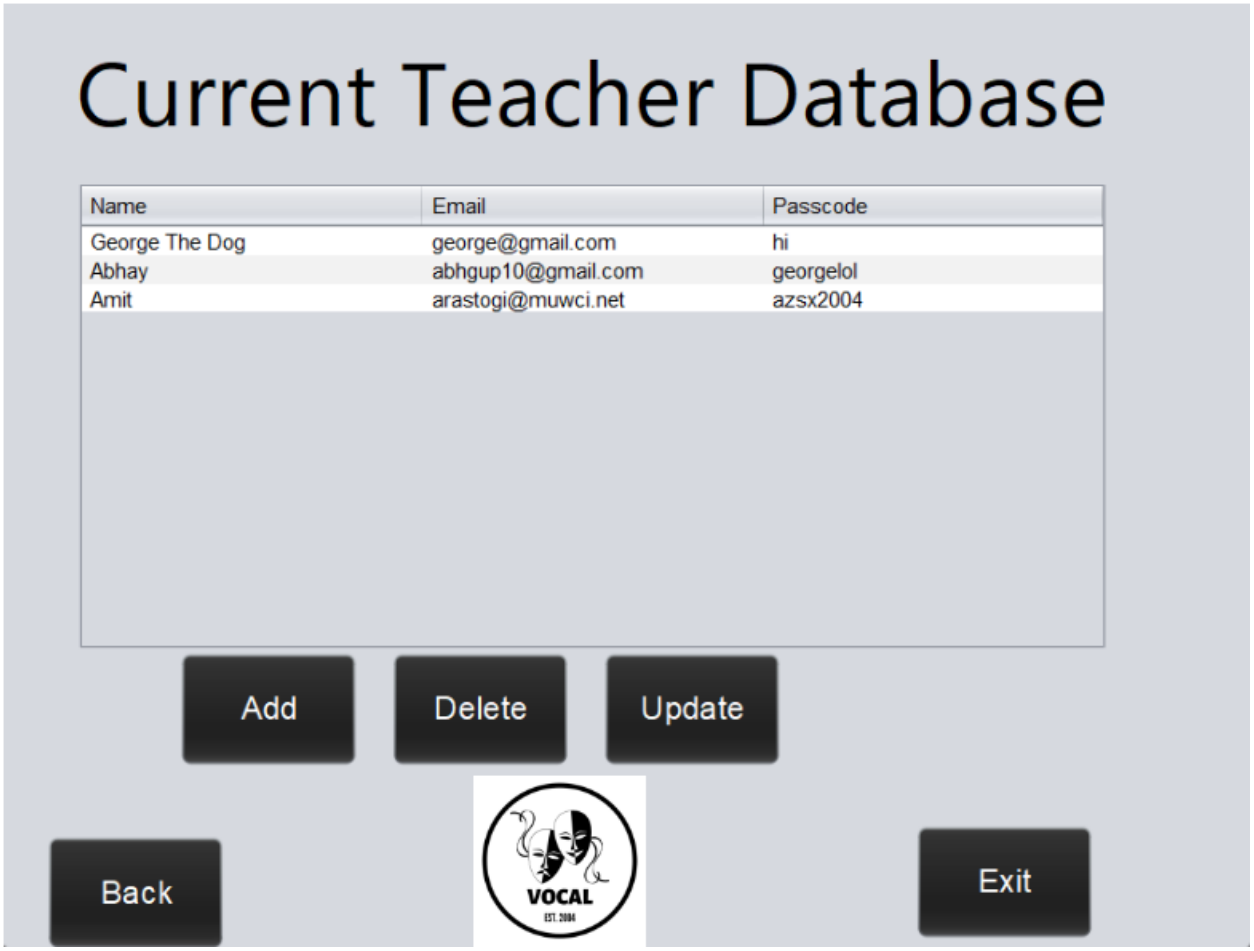
Admin HomePage



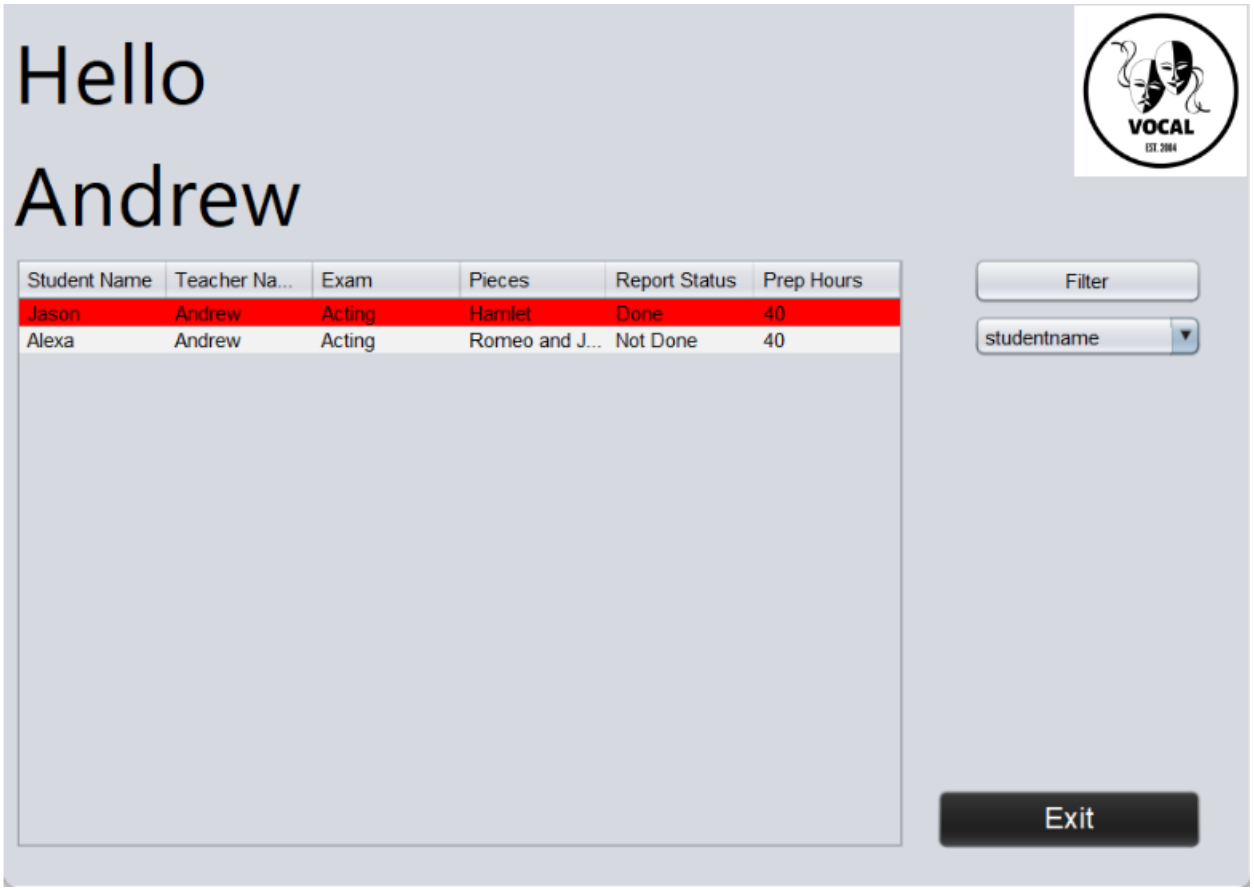
Currently Enrolled Students Admin Page:



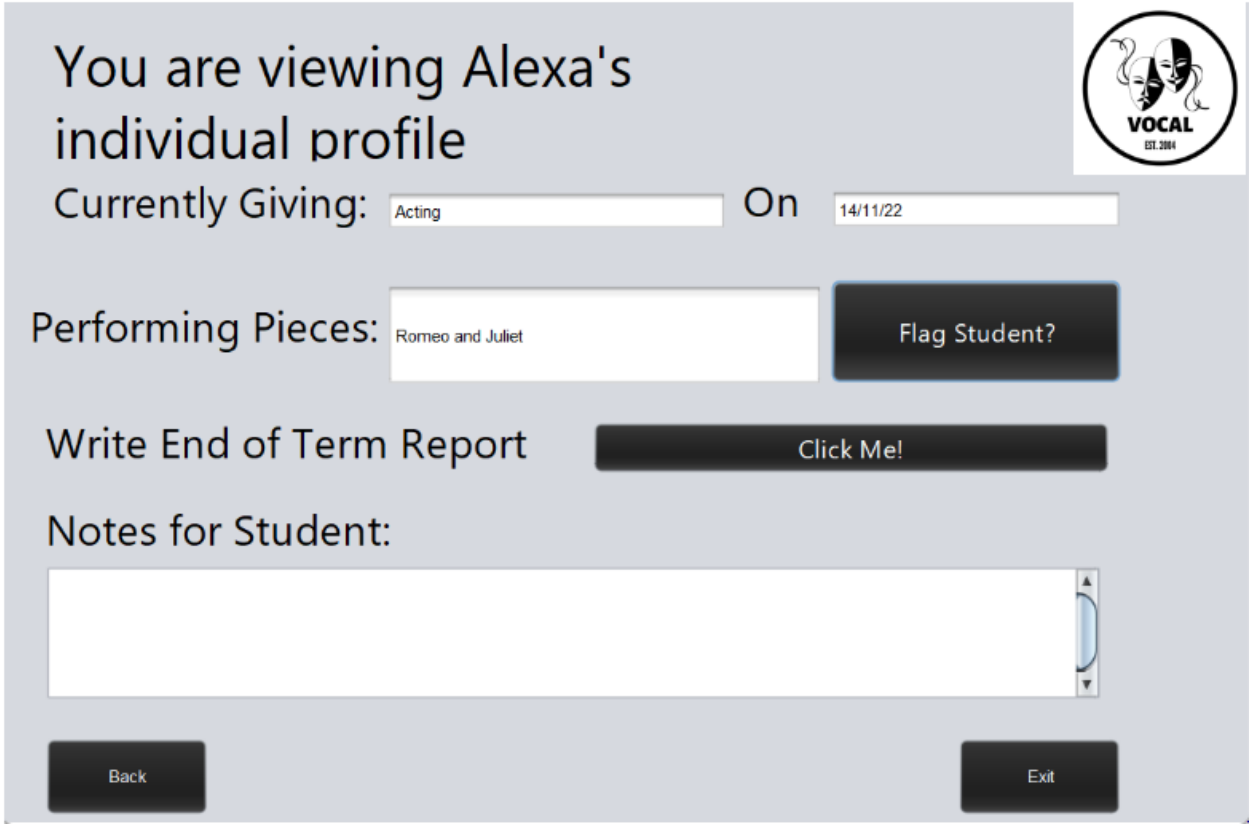
Current Teachers Admin Page:



Teacher HomePage:

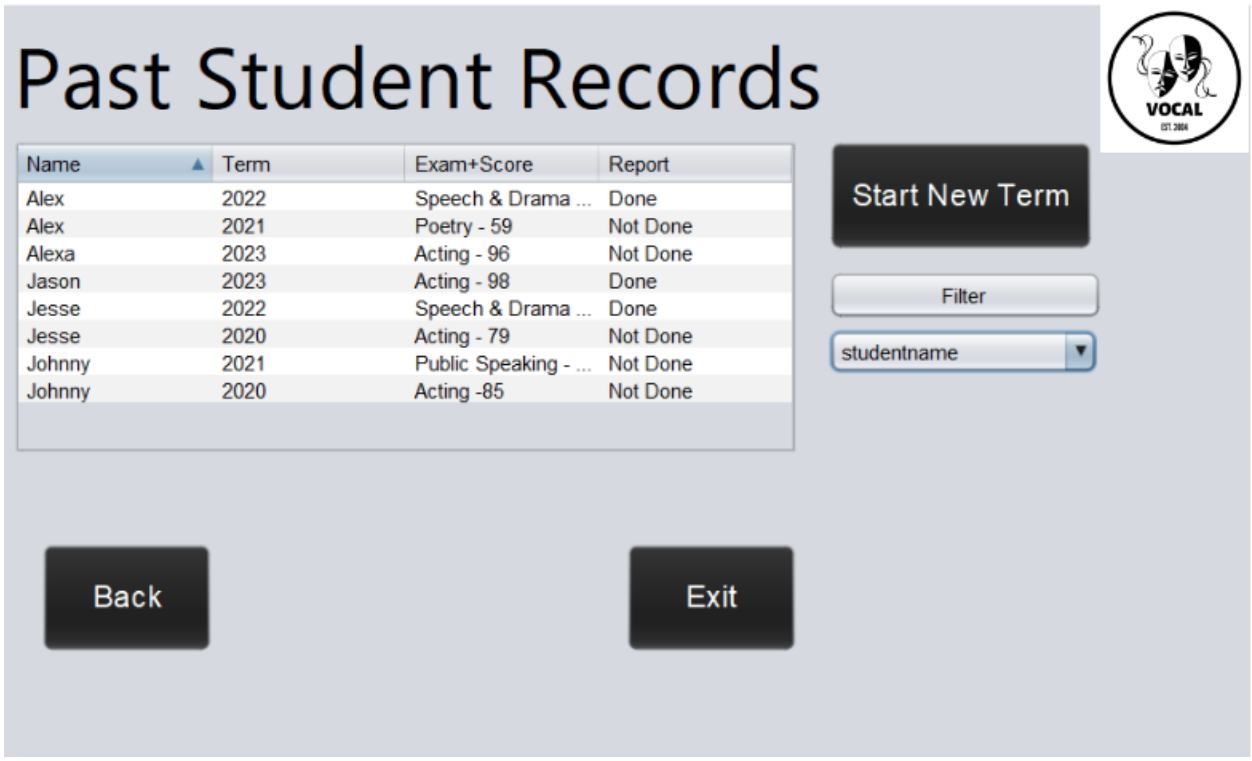


Individual Student HomePage:



Student Report Page:

Past Student Records:



:

Works Cited

BTech Days. “How to transfer data between One JFrame to Another JFrame Java Swing With Netbeans.” *YouTube*, 13 February 2020, https://www.youtube.com/watch?v=r4asbD08ZKI. Accessed 14 August 2022.

Code With Arjun. “Connect Java with Mysql Database | Java JDBC | Java Database connectivity | JDBC MySql | ArjunCodes.” *YouTube*, 11 July 2021, https://www.youtube.com/watch?v=AHFBPxWebFQ. Accessed 14 August 2022.

“How To Create A MySQL Database In phpMyAdmin.” *YouTube*, 15 December 2019, https://www.youtube.com/watch?v=DwHxDlaOsyw. Accessed 21 August 2022.

javatpoint. *Javatpoint: Tutorials List*, https://www.javatpoint.com/. Accessed 12 July 2022.

“LOGIN Form with MYSQL Database JDBC | JAVA SWING GUI.” *YouTube*, 28 March 2020, https://www.youtube.com/watch?v=6K9OpqDM_RM. Accessed 12 July 2022.

“Newest 'java' Questions.” *Stack Overflow*, https://stackoverflow.com/questions/tagged/java. Accessed 15 July 2022.

1BestCsharp blog. “How To Highlight Checked Table Row Using Radio Button In Javascript [with Source code].” *YouTube*, 6 April 2017, https://www.youtube.com/watch?v=2UMpdFtx3Ew. Accessed 25 July 2022.

ProgrammingWizardsTV. “Java Swing S1E2 : JList - Filter/Search using JTextField.” *YouTube*, 2 May 2018, https://www.youtube.com/watch?v=qPMesvqZsmA. Accessed 11 August 2022.

w3 schools. “Java Tutorial.” *W3Schools*, https://www.w3schools.com/java/. Accessed 11 July 2022.

(Refer Appendix 1 third and fourth interactions for interview transcript)

Success Criteria	Feedback from Client
There must be an option to filter information on the database, through criteria such as exam, alphabetical student name, pieces, etc	Mentioned how filter will be beneficial since the center handles many students (lines 111-112)
Teachers can view all and edit some student information, but cannot add or delete students. Administrators can change any student's information, as well as add and delete from their own page.	No feedback. Requirements met.
Warning and confirmation messages should be displayed when attempting to edit data	No feedback. Requirements met.
The application should be protected by a user login, and password	No feedback. Requirements met.
In case of user invalidation, the application should portray an error message	No feedback. Requirements met.
All lists need to be organized as a table, adding to ergonomic appeal	Mentions how the tables are slightly difficult to read but also mentions how they give her a consolidated list of students which is helpful. (lines 112, 139-142)

The software should allow a ‘flagging’ feature for students that use the same pieces, or aren’t meeting exam requirements	Could be made easier to use by incorporating the feature in the table page itself. (lines 116)
The application should have two sections: administrator and user/teacher. Administrators can create, edit, add teacher accounts	No feedback. Requirements met.
Only administrators can view past year information for students	Client acknowledges this is “great” (line 122)
The user can exit the application at any time, but would need to login again once opened	No feedback. Requirement met.
The institution’s logo should be visible on each page of the application	No feedback. Requirement met.
The application should be simplistic and efficient to use, aiding navigability	The client rated this an 8/10. Text could be replaced by icons in some places. (lines 146-149)
All buttons need to direct user to the correct page	No feedback. Requirement met.
All data in application should match the data entered by the user and recorded in database	No feedback. Requirement met.

Input text fields must accept information and perform validation checks	No feedback. Requirement met.
There should be an option to start a new term, which will erase the currently enrolled students table and label past students by the term	No feedback. Requirement met.

Evaluation of Success Criteria:

The product had an effective filter feature that allowed the client to analyze information easily. It also demonstrated proper validation techniques, aiding security and usability. It maintained an easily accessible database of past students as well as their reports. The system also included an effective flagging feature, allowing teachers to identify struggling students. The aesthetics were also appealing to the user, who mentioned “simplicity” and “navigability” as key advantage.The tables made it easier to consolidate student information, as well as add and delete records while portraying appropriate warning messages. Through the Past Students Records page, the client could also start new terms, automatically deleting all records on the current enrolled students table, saving time. The administrator also had access to more private records and more functionality.

Future Development Recommendations:

Table Readability: The table appears difficult to read. One possible way to overcome this is to convert text in buttons to icons, or increase the vertical size of the table.

Shifting Flag option to currently enrolled students page: The client mentioned how having a flag button on the currently enrolled students page rather than having to go to an individual student’s profile page would make her tasks less cumbersome.

Test Plan:

Screen	Test Strategy	Expected Result	Success Criteria	Actual Outcome
Login Page	Typing incorrect email:	Error message showing	4, 5	Error message displays

	Typing incorrect password:	password and email does not match and administrator should be contacted. User required to try again.	4, 5	Error message displays
	Trying a valid email address and password	Message showing login successful. Redirect to respective page (either Teacher Homepage or Admin Homepage).	4	Login message occurs. Redirection is successful.
	Trying a newly created email address and password		4	Login message occurs. Redirection is successful.
Admin Homepage	Clicking on “View Currently Enrolled Students”	Redirect to “Currently Enrolled Students” page	12, 13	Redirection successful
	Clicking on “View Past Student Records”	Redirect to “Past Student Records” page	12, 13	Redirection successful
	Clicking on “Create Teacher Account”	Redirect to “Create Teacher Account” page	12, 13	Redirection successful
	Clicking on “View Current Teachers”	Redirect to “Current Teachers” page	12, 13	Redirection successful
Currently Enrolled Students Page	Clicking on “Filter” button and selecting following options: **ADD**	Dropdown list of options to filter. If option is picked, table should rearrange to represent filter option.	1	Drop-down list shows all options. Table rearranges as per criteria selected.
	Verifying whether all data	Data matches across SQL tables and the	14	Data matched

	in table matches the data entered by the user and is recorded in SQL database	application tables		
	Clicking on “Add” button.	A row should be added to the end of the table that can be filled by the user OR it should add the data the user has inputed in the text fields to the table. Window opens for confirmation before.	2, 6	Row is added that can be filled. Information is added to row if it has been entered in the text fields above.
	Clicking on “Delete” button.	The row selected should be deleted from the table and the database. Window opens for confirmation before.	2, 6	Row is deleted and record is removed from database
	Clicking on “Exit” button.	The application should logout and close. Window opens for confirmation before.	10, 12, 13	Application logs out and closes successfully
	Clicking on student name in table	Redirection to the individual student's profile page.	13	Student name does not redirect but clicking on the report status field does

Past Student Records Page	Clicking on “Filter” button	Dropdown list of options to filter. If option is picked, table should rearrange to represent filter option	1	Drop-down list shows all options. Table rearranges as per criteria selected.
	Verifying that administrators can see page but teachers cannot	Teacher account page should not have option to be directed to this page while administrator homepage should.	9	Teachers cannot see this page, while administrators can.
	Verifying that past student records contains student information from previous terms	The data in past student record table should match that of the corresponding SQL database.	14	Data matches
	Clicking on “back” button..	User redirected to admin homepage. Window opens for confirmation before.	12, 13	User is redirected to correct page
	Clicking on “exit” button	Application logs out and closes. Window opens for confirmation before.	10, 12, 13	Application logs out and closes successfully
	Clicking on “Start New Term” button	Currently enrolled students table should be cleared and new term should be started. Will show new term once entries are added to past	16	New term is started. Currently enrolled students table is cleared for new entries. The new entries load into the past students record

		student records		page but with the new term.
	Clicking report fields to view student report button	User redirected to student report for that particular term	13	User redirected to correct page successfully
Create Teacher Account Page	Clicking on “back” button	User redirected to admin homepage. Window opens for confirmation before.	12, 13	User redirected to correct page successfully
	Clicking on “exit” button	Application logs out and closes. Window opens for confirmation before.	10, 12, 13	Application logs out and closes successfully
	Entering inputs in text fields	Textfields allow information to be entered. Validation check for email field.	15	Text fields accept information. Error message is shown in validation mistake
	Clicking on “Create” button	New teacher account created that can be used for login. Updated in table and database.	4, 6, 14	Account is created and loaded into table and database. Login works successfully.
Current Teacher Database Page	Clicking on “Delete” button	Deletes row selected and removes account details from database, halting login. Window opens for confirmation before.	2, 3, 6, 14	Row is successfully deleted and record removed from database. Login for these credentials is unsuccessful

	Verifying whether records in table match those of database	Records in application table should match with those of the corresponding SQL table (current teachers)	14	Match successful
	Clicking on “back” button	User redirected to admin homepage. Window opens for confirmation before.	12, 13	User redirected to correct page successfully
	Clicking on “exit” button	Application logs out and closes. Window opens for confirmation before.	10, 12, 13	Application logs out successfully and closes
Individual Student Page	Entering inputs in text fields	Textfields should accept information and perform validation checks	15	Textfields accept information and show error message if validation error exists
	Clicking on “flag” button	Row on currently enrolled students page table for particular student is highlighted in red. It is de-highlighted if unflag is pressed.	7	Row for particular student is highlighted I. Red successfully. Highlight is removed when student flag is removed
	Clicking on “click me” button for end of term report.	User redirected to writing report for particular student	12, 13	User redirected to Correct page successfully
	Clicking on “back” button	User redirected to currently	12, 13	User redirected to correct page

		enrolled students page. Window opens for confirmation before.		successfully
	Clicking on “exit” button	Application logs out and closes. Window opens for confirmation before.	10, 12, 13	Application logs out successfully and closes
	Entering notes in text field	Text field should accept information	15	Text field accepts information and shows error message if validation error exists
Report Page	Entering inputs in text fields	Text fields should accept information while performing validation checks	15	Textfields accepts information and shows error of validation error exists
	Clicking on “submit” button	Report recorded in database. Status on currently enrolled students page table updated to "done". Report becomes accessible in past student records page. Window opens for confirmation before.	2, 3, 14	Report is stored in database for particular student. Becomes accessible in past student records page.
	Clicking on “back” button	User redirected to individual student's profile page. Window opens for confirmation	12, 13	User redirected to correct page successfully

		before.		
	Clicking on “exit” button	Application logs out and closes successfully. Window opens for confirmation before.	10, 12,13	application logs out Successful

Client: Ms. Lisa
Founder of VOCAL, a Speech and Drama institute training 400+ students (K-12) in multiple centers. Currently trains students for Trinity College London and examinations.

Initial Interaction: 30th May, 2022

Me: Goodmorning Ms. Lisa! Thank you so much for agreeing to meet me. I'm looking forward to working with you over the next couple of months, and am excited to start developing software for your esteemed institution.

Ms. Lisa: Hi! Thank you for picking me as your client. Let's get started.

Me: I just had a few questions regarding your institution and the problems you're facing, so that we can work together to form the perfect solution!

Ms. Lisa: Alright let's get into it!

Me: How many students does VOCAL train across how many centers, and how many teachers?

Ms. Lisa: So during COVID, on average, we train around 400 students each term. We employ 15 teachers, and have 1 main branch, another branch in ***REDACTED*** and an online branch.

Me: Wow that's a lot! What is your age range, and how many different exams do you train for?

Ms. Lisa: So we have a very wide age range. We start training from 2.5 years itself, with Trinity College London's 'Young Performer's Certificate'. 40% of our batch is pre-primary, 20-30% middle school to high school, and the remaining are college students. We train in 7-8 different subjects, ranging from acting to public speaking. In order to graduate, students must pick either acting or public speaking, or they can choose to pursue certifications in both.

Me: So how do you normally keep track of your students and their progress?

Ms. Lisa: We don't normally have a way to keep track of students except for viewing their scores after they give the exam, however, we wish we had a way to do that, because the exam results sometimes aren't reflective of the student's actual potential. Normally we divide our students between teachers, and they're supposed to keep tabs on their progress, but we don't have a quantifiable way to actually do that, and whenever the teacher changes, it's

extremely difficult for them to gauge the student's skills because there is no documentation of the same. It also becomes difficult for us to decide which exam the student should now go for, and whether they should go for a particular subject, as we have to wait for the exam score to come out.

Me: Okay thanks for letting me know that. Would you prefer to have a way to track student progress apart from the exam in a sharable, quantifiable manner?

Ms. Lisa: Indeed. That'll be great for us because then we can grade them on personal skills rather than exam skills and exam scores. We could have a set of personal skills that can be graded on a skill each session, and by the end of the term, a progress report could be created. This could show what students need to work on, such as character, or speech. When they get transferred to another teacher, it'll be easier to place them and also determine what exam they can go for. Each teacher could do this for their students, and then share it with us and the child's parents. Because now, parents don't have a way to see if their child has improved, because we have stopped conducting Trinity exams due to COVID-19 as well. The progress reports would also allow us to hold teachers accountable.

Me: Okay I'll keep that in mind. I was just thinking of the number of students you have, and was wondering if you maintain a record of the students with you and the exams they are doing?

Ms. Lisa: Unfortunately, we have no way of actually knowing what students are with us unless we ask the teacher, or look back at fee payment receipts in the bank account.

Me: Have you ever tried to maintain a uniform record?
Ms. Lisa: No not really. We don't have an easy way of doing so.

Me: Okay so the current process of asking the teacher, or searching for receipts must be quite tedious and time-consuming right?

Ms. Lisa: Indeed it is. It's really difficult to keep track of the students with us, and the exams they're doing. Oftentimes, students come back to us three years later requesting a complete transcript of exactly which exams they did, as they don't remember, and we don't have those records. Once the student receives the certificate, we don't log it, but we just hand it over to the student. This proves tedious to find this kind of information as I need to go back to multiple exam schedules from Trinity. We also tend to forget the exam the student last did, and thus have difficulty suggesting their next exam. It would be great for the software to allow us to store student records year by year, the exams they did and when they did it.

Me: Alright I'll keep that in mind as well! Are there any other specific features you'd like added to the software, from helping teachers, to helping students, etc.

Ms. Lisa: Ah yes! There is one more issue we're facing that I think the software can solve. For each exam our students do, they must have their own unique prose and poetry piece. There seems to not be any communication between teachers, and they end up assigning the same prose and poetry to different students, which poses a problem. They also sometimes forget which student has been assigned which piece. Maybe a feature that can alleviate this problem would be quite beneficial.

Me: Alright that'll be great. Would you also like to have a section where teachers can fill in exam scores?

Ms. Lisa: Yes please, as that way we can keep a transcript of how many distinctions and merits the student has.

I think that's it for now, but I'll be keeping in touch with you regularly about the solution. Thank you so much for these specifications. It'll help me create the perfect solution for you!

Ms. Lisa: No worries. I'm excited for this solution!

Second Interaction: Feedback on designs: 20th July, 2022

Me: Hi Ms. Lisa! I hope you are doing well! I wanted to seek your opinion on a potential layout and design for your application.

Ms. Lisa: Hi ***Name Redacted***, I'm alright. Sure, show me what you've done.

Ms. Lisa: I like how there's a minimal focus on colors. It makes the application easy to read and less jarring to the eyes. The teacher homepage screen looks perfect to me. However, the page where you view each individual student seems a bit cluttered. Maybe you can combine the "prose" and "poem" fields into one.The "view all reports" button also isn't needed since I can view the reports from past student records, and teachers anyways don't need to see past reports of students.

Me: Noted

Ms. Lisa: The past students record page also seems a bit cluttered. Is it possible to remove the edit button and just be able to click the table to edit a record?

Me: Yes that is possible.

Ms. Lisa: That would be great then. I don't think the "add" and "delete" buttons have to be there. It will add confusion and can lead to error. The names will automatically get added from the current students page so there doesn't need to be addition and deletion on this page.

Me: Okay. Yes. The past students page is linked to the current students page, so the add and delete button is a little redundant.
Ms. Lisa: Also try putting the “filter” button on the side rather than on top. Everything else seems great! I’m excited to be able to use the application.
.
Me: Thank you for your feedback Ms. Lisa! I shall keep in touch.

Third Interaction: Feedback on actual application: 14th October, 2022

Me: Hi Ms. Lisa. It’s been a long time since we last spoke but I have finally completed the application.

Ms. Lisa: Great! I’m excited. Show me what you’ve done.

Me: Okay. As mentioned in the success criteria, the application needed to have two different sections: administrator and teacher. I have been able to create this. The administrator can create new accounts and new passwords. Currently we are on the sign-in page. I have created sample administrator credentials. If I were to enter the wrong password, I would receive an error message, as you can see. I will now enter the credentials to login.

Ms. Lisa: The layout of the sign in page is great and it hasn’t changed since you showed me the initial designs in July.

Me: We’re now on the admin homepage, where you have a range of options. Would you like to go in order?

Ms. Lisa: Yes.

Me: We first have the currently enrolled students page, where you can view all of the students in the current term, as well as filter, add, delete, and click on them, which leads you to the individual student page.

Ms. Lisa: I like that there’s a lot of functionality that I can use for the table. The filtering will be really helpful since we have so many students .Although a table format seems slightly difficult to read and can be altered later.

Me: Clicking on an individual student page allows you to view the information in detail, and also flag the student which highlights them in red in the table. You can also add some notes. All of this is viewable by different teachers.

Ms. Lisa: I like how the prose and poem fields have been combined as I suggested. The flagging feature is great! However, maybe we can have the option to flag students on the table itself rather than having to go to their individual profile.

Me: We're now on the past student records page, where you can view all the past students and their reports from the previous year. If you wish to start a new year/term, you can simply click the button. This will start a new year and will clear the current students page so you can start adding the students registered for the new term. Their information will be stored in past student records.

Ms. Lisa: It's great to see that I have access to previous records, as I had requested.
Me: Noted. We now have the Create Teacher Account page, where you can create new accounts for teachers and share the passwords with them. Next, we have the View Current Teachers Page. Here, you can view all existing accounts, change passwords or delete accounts.

Ms. Lisa: The administrators have control of the teacher accounts, which I had initially requested. This is great! Give me some time to try it out and I'll let you know what I think.

Me: Alright!

Fourth Interaction: Feedback on actual application after using: 18th October, 2022

Me: Hi Ms. Lisa. I hope you've had the chance to try out the application.

Ms. Lisa: Yes I have. It's been working well so far, and it's certainly made my job of managing students easier.

Me: Out of a scale of 10, how much easier would you say it's made your job?

Ms. Lisa: I would give it a solid 8. It lets me see which students are overlapping, keep track of fees since I have a consolidated list of students, as well as log their pieces in so I don't get confused. As I mentioned last week, the table can sometimes be a little difficult to read since it's so small. Maybe that could be fixed.

Me: Alright. Do you think it meets all success criteria adequately?

Ms. Lisa: I think it goes beyond expectations for most success criteria.

Me: What do you think could be some additions to this application?

Ms. Lisa: Of course making the table a little more readable. We could replace the text buttons with icons. For example, the add icon can simply be replaced with a + sign.

Me: Okay. In terms of simplicity of use and aesthetics, how much would you rate this product out of 10?

Ms. Lisa: Certainly an 8. I wanted the application to be simplistic in its color scheme and also be organized. The homepage gives me options which is great. It can be made more simplistic by incorporating the suggestions I mentioned above.

Me: Do you think a future feature could be an email sender that allows you to contact a student or parent with a note from the application itself?

Ms. Lisa: Yes. I certainly think that will be helpful! It will streamline my process and make it much easier.

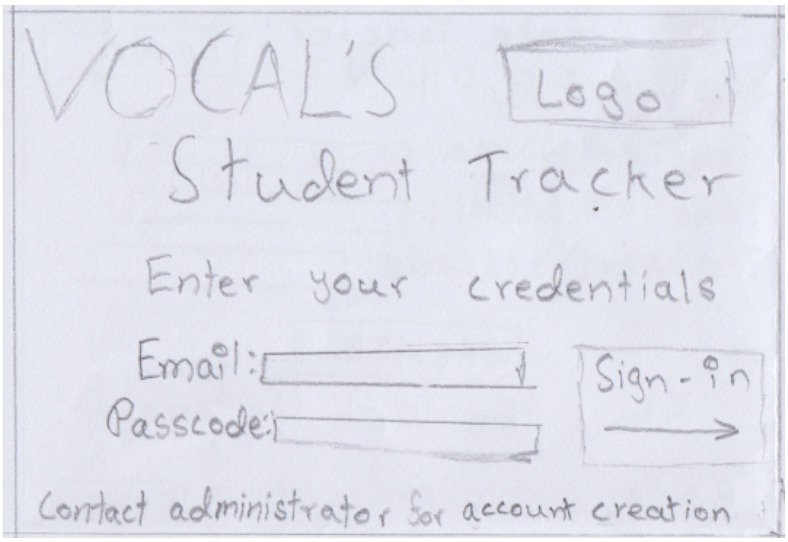
Me: Alright. I've noted all this down and hope to implement it in the near future. Thank you for your time and I hope this application serves you well.

Ms. Lisa: It was great getting to interact with and get help from you. I think it'll help VOCAL a lot. Thank you for all this!

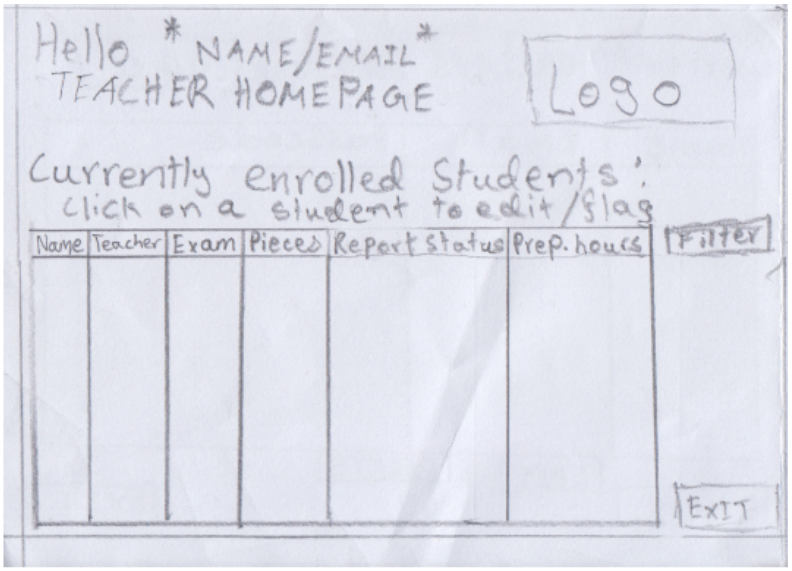
Initial sketches:

Prior to creating annotated designs, rough sketches were created to plan the initial design and flow of the application. The purpose was to obtain feedback from the client before creating in-depth designs.

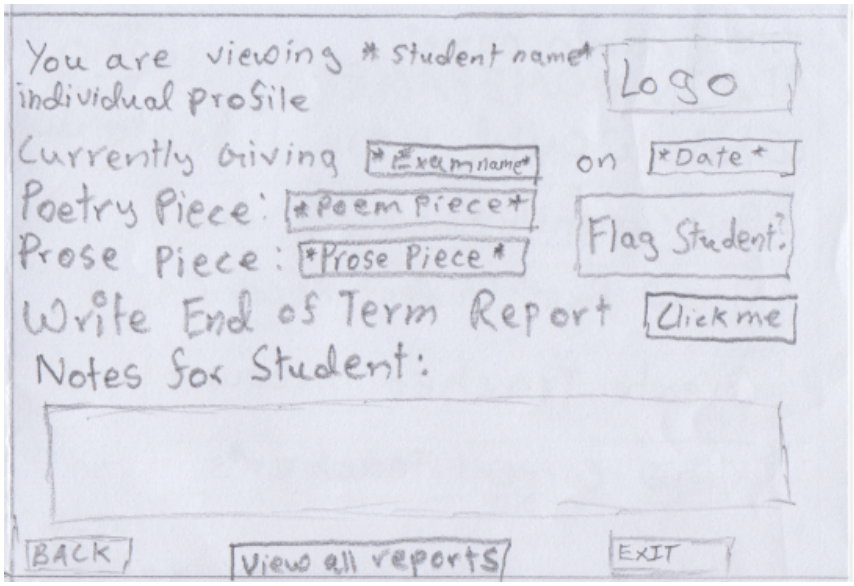
Login Screen (Same for all users):



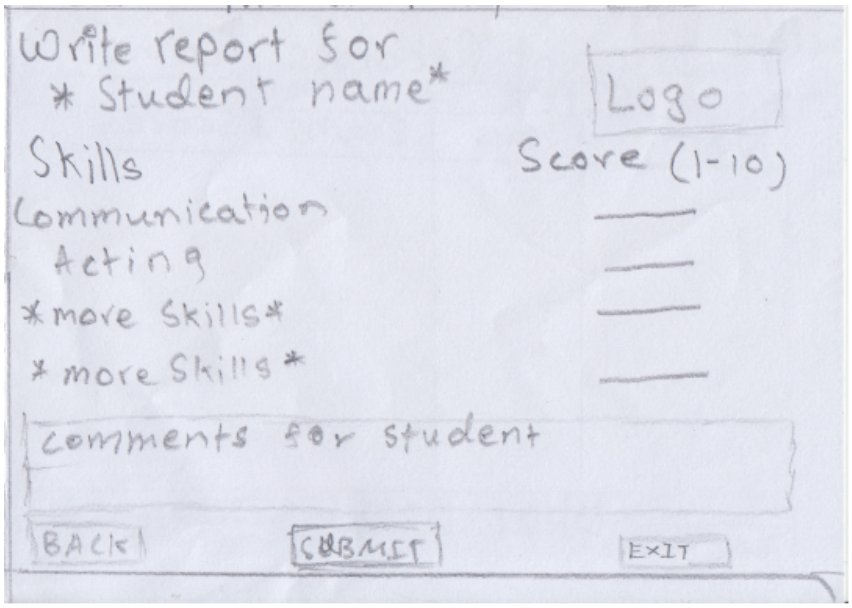
Teacher Homepage:



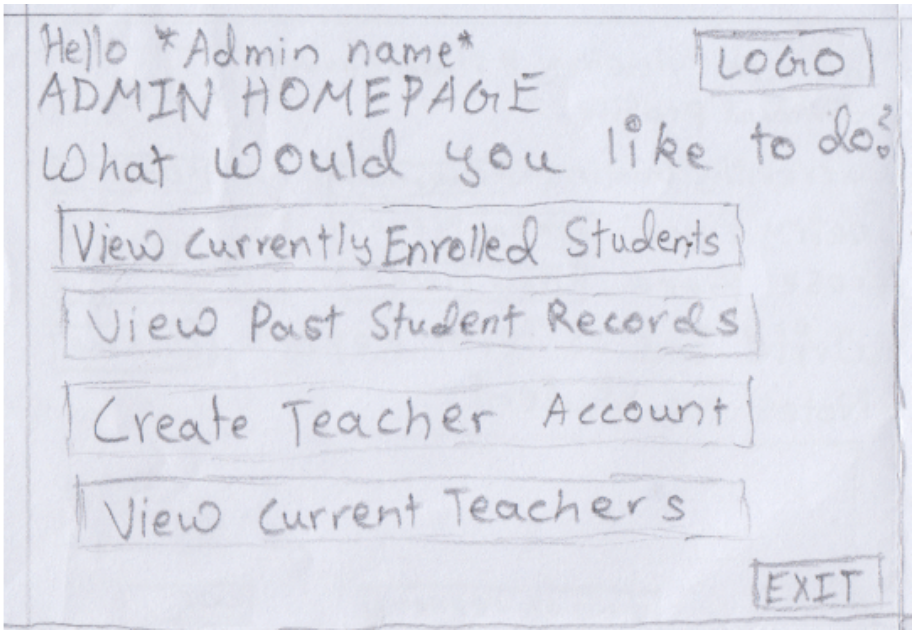
Individual Student Page (Viewable by both admin and teacher users by clicking on students from currently enrolled database):



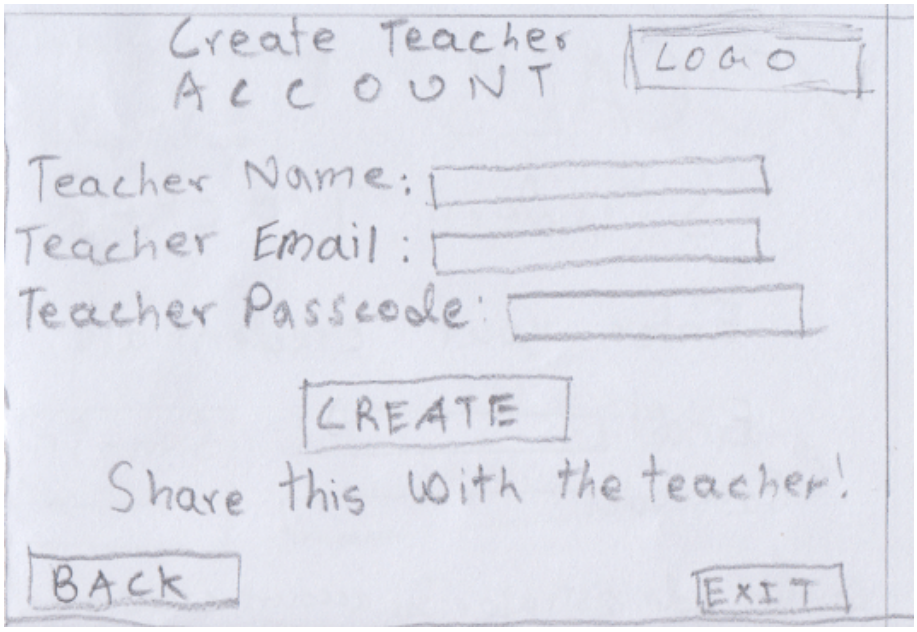
Write Report Page (only available for teachers to write reports for students. Admin can view all reports):



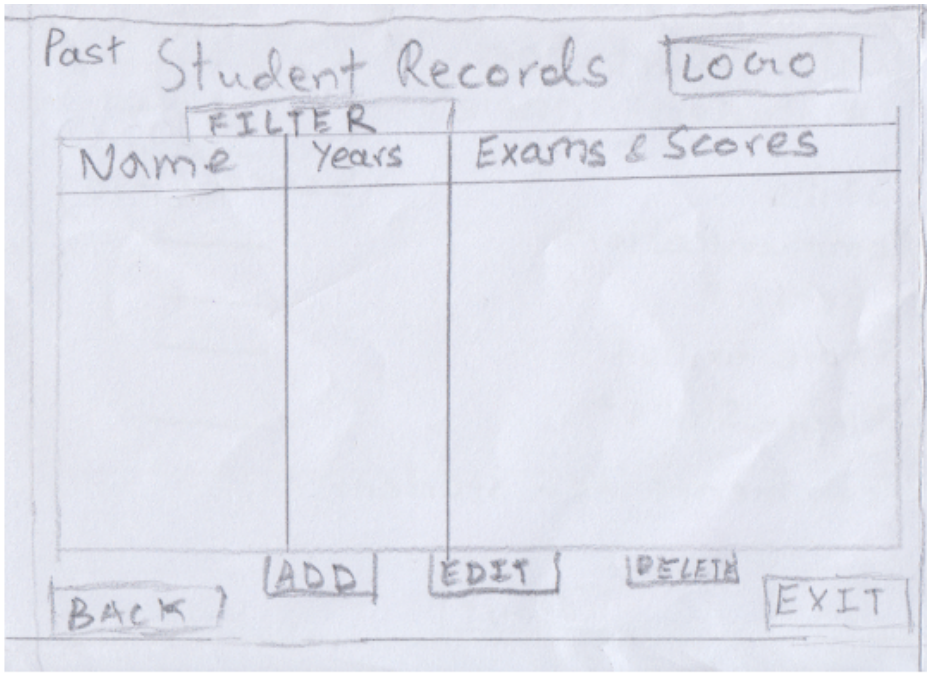
Administrator Homepage:



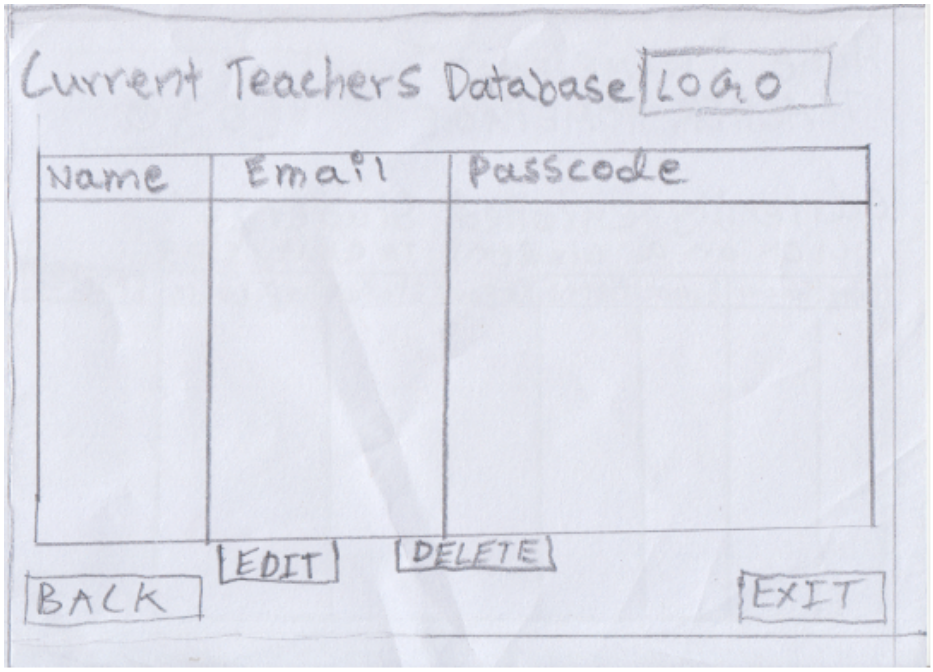
Create teacher account page:



Past student records page (only viewable by admin from admin homepage)



Current Teachers Database:



Conversation with Client (IN-PERSON, Dated 13th June, 2022)

Key takeaways/feedback and suggestions:

1. Overall aesthetic is perfect. It's easy to read, comprehend, and follow

2. Love the security features (eg. only admin gets to see past student records and information, admin can revoke and authorize access, only admin sees progress reports rather than all teachers)
3. It would be great to have a “start new year” button on the currently enrolled page where the table gets cleared and we can input new student names for the new term
4. Also helpful would be if the past student records can be updated automatically from the currently enrolled student page. When I click ‘create new year’, the table can clear and transfer the student names and years to the past student records, where all I need to do is fill their examination score
5. It would be helpful to have a past records column in the past student records page

Sign In Page:

```
 Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to
edit this template
 */
package com.mycompany.csiaf;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import javax.naming.spi.DirStateFactory.Result;
import javax.swing.JOptionPane;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

/**
 *
 *
 */
public class signinf extends javax.swing.JFrame {
    Connection con = null;
    PreparedStatement pst = null;
    Result rs = null;

    /**
     * Creates new form signinf
     */
    public signinf() {
        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the
form.
     * WARNING: Do NOT modify this code. The content of this method is
always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
```

```
    private void initComponents() {

        jLabel6 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jLabel1 = new javax.swing.JLabel();
        edEmail = new javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        buttonlogin = new javax.swing.JButton();
        jLabel3 = new javax.swing.JLabel();
        jLabel7 = new javax.swing.JLabel();
        usertype = new javax.swing.JComboBox<>();
        jPasswordField1 = new javax.swing.JPasswordField();
        jLabel8 = new javax.swing.JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jLabel6.setFont(new java.awt.Font("Arial", 0, 122)); // NOI18N
        jLabel6.setText("VOCAL'S");

        jLabel5.setFont(new java.awt.Font("Arial", 0, 80)); // NOI18N
        jLabel5.setText("Student Tracker");

        jLabel4.setFont(new java.awt.Font("Segoe UI", 0, 50)); // NOI18N
        jLabel4.setText("Enter Your Credentials");

        jLabel1.setFont(new java.awt.Font("Segoe UI", 0, 40)); // NOI18N
        jLabel1.setText("Email:");

        edEmail.setText("Input");
        edEmail.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusGained(java.awt.event.FocusEvent evt) {
                edEmailFocusGained(evt);
            }
        });
        edEmail.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                edEmailMouseClicked(evt);
            }
        });
        edEmail.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
                edEmailActionPerformed(evt);
            }
        });

        jLabel2.setFont(new java.awt.Font("Segoe UI", 0, 40)); // NOI18N
        jLabel2.setText("Password:");

        buttonlogin.setBackground(new java.awt.Color(0, 0, 0));
        buttonlogin.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        buttonlogin.setForeground(new java.awt.Color(255, 255, 255));
        buttonlogin.setText("Sign-In");
        buttonlogin.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonloginActionPerformed(evt);
            }
        });

        jLabel3.setFont(new java.awt.Font("Arial", 0, 25)); // NOI18N
        jLabel3.setText("Contact Ms. Tracy or Ms. Lisa for account
creation");

        jLabel7.setFont(new java.awt.Font("Segoe UI", 0, 40)); // NOI18N
        jLabel7.setText("User Type:");

        usertype.setModel(new javax.swing.DefaultComboBoxModel<>(new
String[] { "admin", "user" }));

        jPasswordField1.setText("jPasswordField1");

        jLabel8.setIcon(new
javax.swing.ImageIcon("C:/Users/Abhay/OneDrive/Documents/NetBeansProjects/C
SIAF/src/main/java/image/vocallogo.png")
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addComponent(jLabel6,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE, 540,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jLabel8,
javax.swing.GroupLayout.PREFERRED_SIZE, 141,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(6, 6, 6)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                            .addComponent(jLabel4)
                            .addGroup(layout.createSequentialGroup()
                                .addGap(6, 6, 6)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)

.addGroup(layout.createSequentialGroup()
                                        .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 104,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                                        .addComponent(edEmail,
javax.swing.GroupLayout.PREFERRED_SIZE, 407,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(layout.createSequentialGroup()
                                        .addComponent(jLabel7)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(usertype,
javax.swing.GroupLayout.PREFERRED_SIZE, 337,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                                        .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(jPasswordField1,
javax.swing.GroupLayout.PREFERRED_SIZE, 325,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                        .addComponent(buttonlogin)))))
                        .addGap(25, 25, 25))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel5)

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(161, Short.MAX_VALUE)
                .addComponent(jLabel3)
                .addGap(146, 146, 146))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jLabel6,
javax.swing.GroupLayout.PREFERRED_SIZE, 103,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jLabel8,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(63, 63, 63)
                .addComponent(jLabel5)
                .addGap(6, 6, 6)
```

```
                .addComponent(jLabel4)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jLabel1))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(18, 18, 18)
                        .addComponent(edEmail,
javax.swing.GroupLayout.PREFERRED_SIZE, 31,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(jLabel2))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(7, 7, 7)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                            .addComponent(jPasswordField1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(buttonlogin,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE))))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jLabel7))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(25, 25, 25)
                        .addComponent(usertype,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(18, 18, 18)
                .addComponent(jLabel3)
                .addContainerGap(44, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void edEmailActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void buttonloginActionPerformed(java.awt.event.ActionEvent evt)
{
       try{
           String regex = "^[a-zA-Z0-9+_.-]+@[a-zA-Z0-9.-]+$";
           Pattern pattern = Pattern.compile(regex);
           Matcher matcher = pattern.matcher(edEmail.getText());
           if(matcher.matches()) {
               String query = "SELECT * FROM `multiuserlogin` WHERE
EmailAddress=? and Passcode=? and UserType=?";
           // load and register JDBC driver for MySQL
           con =
DriverManager.getConnection("jdbc:mysql://localhost/multiuserlogin",
"root", "");
           pst = con.prepareStatement(query);
           pst.setString(1, edEmail.getText());
           pst.setString(2, jPasswordField1.getText());
           pst.setString(3, String.valueOf(usertype.getSelectedItem()));
           ResultSet rs=pst.executeQuery();
           if(rs.next()) {
           JOptionPane.showMessageDialog(this, "username and password
matched and you are signed in as " + rs.getString("usertype") );
           if(usertype.getSelectedIndex() == 0){
               adminhomepage adminhp = new adminhomepage();
               adminhp.setName(rs.getString("TeacherName"));
               adminhp.setVisible(true);
               this.setVisible(false);
           }else {
             teacherhomepage t = new teacherhomepage();
             t.setName(rs.getString("TeacherName"));
```

```
            t.setVisible(true);
            this.setVisible(false);
          }}
          else{
                  JOptionPane.showMessageDialog(this, "username and
password do not match");}

          }
          else {
              JOptionPane.showMessageDialog(this, "Please enter a valid
email address");

          }


       }catch(Exception ex){
       JOptionPane.showMessageDialog(this, ex.getMessage());    }
    }

    private void edEmailMouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:

    }

    private void edEmailFocusGained(java.awt.event.FocusEvent evt) {
        // TODO add your handling code here:
        if("Input".equals(edEmail.getText())) {
        edEmail.setText("");
        }
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel
setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with
the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
```

```
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(signinf.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(signinf.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(signinf.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(signinf.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new signinf().setVisible(true);
            }
        });
        }

    // Variables declaration - do not modify
    private javax.swing.JButton buttonlogin;
    private javax.swing.JTextField edEmail;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
```

```
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JPasswordField jPasswordField1;
    private javax.swing.JComboBox<String> usertype;
    // End of variables declaration
        }
```

Admin Home Page:

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to
edit this template
 */
package com.mycompany.csiaf;

import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import javax.swing.JFrame;
import javax.swing.JOptionPane;

/**
 *
 */
public class adminhomepage extends javax.swing.JFrame {
    String adminname;
    public void setName(String name){
        adminname = name;

    }
       public void labelName() {
           namelabel.setText(adminname);

       };

    private void initSelfListeners() {
    WindowListener taskStarterWindowListener = new WindowListener() {
        @Override
        public void windowOpened(WindowEvent e) {
```

```
            labelName();
            System.out.println("Performing task..."); //Perform task here.
In this case, we are simulating a startup (only once) time-consuming task
that would use a worker.
        }

        @Override
        public void windowClosing(WindowEvent e) {
            //Do nothing...Or something...You decide!
        }

        @Override
        public void windowClosed(WindowEvent e) {
            //Do nothing...Or drink coffee...NVM; always drink coffee!
        }

        @Override
        public void windowIconified(WindowEvent e) {
            //Do nothing...Or do EVERYTHING!
        }

        @Override
        public void windowDeiconified(WindowEvent e) {
            //Do nothing...Or break the law...
        }

        @Override
        public void windowActivated(WindowEvent e) {


            //Do nothing...Procrastinate like me!
        }

        @Override
        public void windowDeactivated(WindowEvent e) {
            //Do nothing...And please don't notice I have way too much free
time today...
        }
    };

    //Here is where the magic happens! We make (a listener within) the
frame start listening to the frame's own events!
    this.addWindowListener(taskStarterWindowListener);
```

```
}

    /**
     * Creates new form adminhomepage
     */
    public adminhomepage() {
        initComponents();
        initSelfListeners();
    }

    /**
     * This method is called from within the constructor to initialize the
form.
     * WARNING: Do NOT modify this code. The content of this method is
always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        namelabel = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        buttonpaststudents = new javax.swing.JButton();
        buttonteacheraccount = new javax.swing.JButton();
        buttoncurrentteachers = new javax.swing.JButton();
        buttonexit = new javax.swing.JButton();
        buttoncurrentstudents = new javax.swing.JButton();
        jLabel3 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        namelabel.setFont(new java.awt.Font("Segoe UI", 0, 50)); // NOI18N
        namelabel.setText("Hello");

        jLabel2.setFont(new java.awt.Font("Segoe UI", 0, 50)); // NOI18N
        jLabel2.setText("Admin Homepage");

        buttonpaststudents.setBackground(new java.awt.Color(0, 0, 0));
        buttonpaststudents.setFont(new java.awt.Font("Segoe UI", 0, 18));
// NOI18N
```

```
        buttonpaststudents.setForeground(new java.awt.Color(255, 255,
255));
        buttonpaststudents.setText("View Past Student Records");
        buttonpaststudents.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonpaststudentsActionPerformed(evt);
            }
        });

        buttonteacheraccount.setBackground(new java.awt.Color(0, 0, 0));
        buttonteacheraccount.setFont(new java.awt.Font("Segoe UI", 0, 18));
// NOI18N
        buttonteacheraccount.setForeground(new java.awt.Color(255, 255,
255));
        buttonteacheraccount.setText("Create Teacher Account");
        buttonteacheraccount.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonteacheraccountActionPerformed(evt);
            }
        });

        buttoncurrentteachers.setBackground(new java.awt.Color(0, 0, 0));
        buttoncurrentteachers.setFont(new java.awt.Font("Segoe UI", 0,
18)); // NOI18N
        buttoncurrentteachers.setForeground(new java.awt.Color(255, 255,
255));
        buttoncurrentteachers.setText("View Current Teachers");
        buttoncurrentteachers.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttoncurrentteachersActionPerformed(evt);
            }
        });

        buttonexit.setBackground(new java.awt.Color(0, 0, 0));
        buttonexit.setFont(new java.awt.Font("Segoe UI", 0, 18)); // NOI18N
        buttonexit.setForeground(new java.awt.Color(255, 255, 255));
        buttonexit.setText("Exit");
        buttonexit.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonexitActionPerformed(evt);
```

```
            }
        });

        buttoncurrentstudents.setBackground(new java.awt.Color(0, 0, 0));
        buttoncurrentstudents.setFont(new java.awt.Font("Segoe UI", 0,
18)); // NOI18N
        buttoncurrentstudents.setForeground(new java.awt.Color(255, 255,
255));
        buttoncurrentstudents.setText("View Currently Enrolled Students");
        buttoncurrentstudents.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttoncurrentstudentsActionPerformed(evt);
            }
        });

        jLabel3.setFont(new java.awt.Font("Segoe UI", 0, 50)); // NOI18N
        jLabel3.setText("Hello");

        jLabel8.setIcon(new
javax.swing.ImageIcon("C:/Users/Abhay/OneDrive/Documents/NetBeansProjects/C
SIAF/src/main/java/image/vocallogo.png")
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(49, 49, 49)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING, false)
                        .addComponent(buttonteacheraccount,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                        .addComponent(buttoncurrentteachers,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
```

```
Short.MAX_VALUE)
                        .addComponent(buttonpaststudents,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                        .addComponent(buttoncurrentstudents,
javax.swing.GroupLayout.DEFAULT_SIZE, 449, Short.MAX_VALUE))
                    .addComponent(jLabel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 489,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 208,
Short.MAX_VALUE)
                .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 95,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(99, 99, 99))
            .addGroup(layout.createSequentialGroup()
                .addGap(201, 201, 201)
                .addComponent(namelabel,
javax.swing.GroupLayout.PREFERRED_SIZE, 276,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jLabel8,
javax.swing.GroupLayout.PREFERRED_SIZE, 141,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(59, 59, 59)
                    .addComponent(jLabel3,
javax.swing.GroupLayout.PREFERRED_SIZE, 136,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addContainerGap(745, Short.MAX_VALUE)))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
```

```
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(41, 41, 41)
                        .addComponent(namelabel))
                    .addComponent(jLabel8,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING)
                    .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 52,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel2)
                        .addGap(41, 41, 41)
                        .addComponent(buttoncurrentstudents,
javax.swing.GroupLayout.PREFERRED_SIZE, 90,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(buttonpaststudents,
javax.swing.GroupLayout.PREFERRED_SIZE, 90,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(buttoncurrentteachers,
javax.swing.GroupLayout.PREFERRED_SIZE, 90,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(buttonteacheraccount,
javax.swing.GroupLayout.PREFERRED_SIZE, 90,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(41, 41, 41)
                    .addComponent(jLabel3)
                    .addContainerGap(658, Short.MAX_VALUE)))
        );
```

```
        pack();
    }// </editor-fold>

    private void
buttonteacheraccountActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        createteacheraccount cta = new createteacheraccount();
        cta.setName(adminname);
               cta.setVisible(true);
               this.setVisible(false);
    }
private JFrame frame;
    private void buttonexitActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        frame = new JFrame("Exit");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
exit", "MYSQL Connector",
                JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)

        {
            System.exit(0);
        }
    }

    private void
buttoncurrentteachersActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        currentteachers ct = new currentteachers();
        ct.setName(adminname);
               ct.setVisible(true);
               this.setVisible(false);
    }

    private void
buttonpaststudentsActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        paststudentrecords psr = new paststudentrecords();
        psr.setName(adminname);
               psr.setVisible(true);
               this.setVisible(false);
    }
```

```
    private void
buttoncurrentstudentsActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        currentlyenrolledstudentsadmin cesa = new
currentlyenrolledstudentsadmin();
        cesa.setName(adminname);
               cesa.setVisible(true);
               this.setVisible(false);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel
setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with
the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(adminhomepage.class.getName()).log(java.
util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(adminhomepage.class.getName()).log(java.
util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(adminhomepage.class.getName()).log(java.
```

```
util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(adminhomepage.class.getName()).log(java.
util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new adminhomepage().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton buttoncurrentstudents;
    private javax.swing.JButton buttoncurrentteachers;
    private javax.swing.JButton buttonexit;
    private javax.swing.JButton buttonpaststudents;
    private javax.swing.JButton buttonteacheraccount;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel namelabel;
    // End of variables declaration
}
```

Individual Student Page:

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to
edit this template
 */
package com.mycompany.csiaf;

import java.sql.Connection;
import java.sql.DriverManager;
```

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;



/**
 *
 */
public class individualstudentpage extends javax.swing.JFrame {

    private static final String username = "root";
    private static final String password = "";
    private static final String dataConn =
"jdbc:mysql://localhost:3306/currentlyenrolledstudentsadmin";
    private static final String dataConn2 =
"jdbc:mysql://localhost:3306/individualstudentreport";

    String adminname;
    public void setName(String name){
        adminname = name;

    }

   Connection sqlConn = null;
   PreparedStatement pst = null;
   ResultSet rs = null;
   int q, i, id, deleteItem;

    /**
     * Creates new form individualstudentpage
     */
    public individualstudentpage() {
        initComponents();
    }
    String sid1;
    String sname1;
```

```
    String examname1;
    String date1;
    String pieces1;
    boolean flag1;
    String notes1;
    public void setStudent(String sid, String sname, String examname,
String date, String pieces, boolean flag, String notes) {
        sid1 = sid;
        sname1 = sname;
        examname1 = examname;
        date1 = date;
        pieces1 = pieces;
        flag1 = flag;
        notes1 = notes;

       jLabel1.setText("You are currently viewing " + sname1 + "'s");
       txtExamName.setText(examname1);
       txtDate.setText(date1);
       txtPPieces.setText(pieces1);
       jTextArea2.setText(notes1);
       buttonflag.setText(flag1 ? "Unflag student?":"Flag student?");
    }
    String backpage;
    public void backfunctionsetpage(String backpage1) {
        backpage = backpage1;


    }
    public Connection getConnection() {
        try{
            Class.forName("com.mysql.jdbc.Driver");

        }catch(ClassNotFoundException ex) {
            System.out.println(ex.getMessage());
        }
        Connection con = null;
        try{
            con =
DriverManager.getConnection("jdbc:mysql://localhost/currentlyenrolledstuden
tsadmin", "root", "");
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());
```

```
        }
        return con;

    }

    public Connection getConnection3() {
        try{
            Class.forName("com.mysql.jdbc.Driver");

        }catch(ClassNotFoundException ex) {
            System.out.println(ex.getMessage());
        }
        Connection con = null;
        try{
            con =
DriverManager.getConnection("jdbc:mysql://localhost/paststudentrecord",
"root", "");
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }
        return con;

    }

    /**
     * This method is called from within the constructor to initialize the
form.
     * WARNING: Do NOT modify this code. The content of this method is
always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jScrollPane1 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        buttonreport = new javax.swing.JButton();
        buttonflag = new javax.swing.JButton();
        jScrollPane2 = new javax.swing.JScrollPane();
```

```
        jTextArea2 = new javax.swing.JTextArea();
        jLabel7 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        jLabel9 = new javax.swing.JLabel();
        jLabel10 = new javax.swing.JLabel();
        jLabel12 = new javax.swing.JLabel();
        txtExamName = new javax.swing.JTextField();
        txtDate = new javax.swing.JTextField();
        txtPPieces = new javax.swing.JTextField();
        buttonback2 = new javax.swing.JButton();
        buttonexit = new javax.swing.JButton();
        buttonsave = new javax.swing.JButton();
        jLabel11 = new javax.swing.JLabel();

        jTextArea1.setColumns(20);
        jTextArea1.setRows(5);
        jScrollPane1.setViewportView(jTextArea1);


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setSize(new java.awt.Dimension(876, 581));

        jLabel1.setFont(new java.awt.Font("Segoe UI", 0, 40)); // NOI18N
        jLabel1.setText("You are viewing Alexa's");

        jLabel2.setFont(new java.awt.Font("Segoe UI", 0, 30)); // NOI18N
        jLabel2.setText("Performing Pieces:");

        buttonreport.setBackground(new java.awt.Color(0, 0, 0));
        buttonreport.setFont(new java.awt.Font("Segoe UI", 0, 18)); //
NOI18N
        buttonreport.setForeground(new java.awt.Color(255, 255, 255));
        buttonreport.setText("Click Me!");
        buttonreport.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonreportActionPerformed(evt);
            }
        });

        buttonflag.setBackground(new java.awt.Color(0, 0, 0));
        buttonflag.setFont(new java.awt.Font("Segoe UI", 0, 18)); // NOI18N
        buttonflag.setForeground(new java.awt.Color(255, 255, 255));
```

```
        buttonflag.setText("Flag Student?");
        buttonflag.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonflagActionPerformed(evt);
            }
        });

        jTextArea2.setColumns(20);
        jTextArea2.setRows(5);
        jScrollPane2.setViewportView(jTextArea2);

        jLabel7.setFont(new java.awt.Font("Segoe UI", 0, 40)); // NOI18N
        jLabel7.setText("individual profile");

        jLabel8.setFont(new java.awt.Font("Segoe UI", 0, 30)); // NOI18N
        jLabel8.setText("Currently Giving:");

        jLabel9.setFont(new java.awt.Font("Segoe UI", 0, 30)); // NOI18N
        jLabel9.setText("Write End of Term Report");

        jLabel10.setFont(new java.awt.Font("Segoe UI", 0, 30)); // NOI18N
        jLabel10.setText("Notes for Student:");

        jLabel12.setFont(new java.awt.Font("Segoe UI", 0, 30)); // NOI18N
        jLabel12.setText("On");

        txtExamName.setText("Acting");

        txtDate.setText("14/11/22");
        txtDate.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtDateActionPerformed(evt);
            }
        });

        txtPPieces.setText("Romeo and Juliet");

        buttonback2.setBackground(new java.awt.Color(0, 0, 0));
        buttonback2.setForeground(new java.awt.Color(255, 255, 255));
        buttonback2.setText("Back");
        buttonback2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonback2ActionPerformed(evt);
```

```
            }
        });

        buttonexit.setBackground(new java.awt.Color(0, 0, 0));
        buttonexit.setForeground(new java.awt.Color(255, 255, 255));
        buttonexit.setText("Exit");
        buttonexit.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonexitActionPerformed(evt);
            }
        });

        buttonsave.setBackground(new java.awt.Color(0, 0, 0));
        buttonsave.setForeground(new java.awt.Color(255, 255, 255));
        buttonsave.setText("Save");
        buttonsave.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonsaveActionPerformed(evt);
            }
        });

        jLabel11.setIcon(new
javax.swing.ImageIcon("C:/Users/Abhay/OneDrive/Documents/NetBeansProjects/C
SIAF/src/main/java/image/vocallogo.png")
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(37, 37, 37)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel7,
javax.swing.GroupLayout.PREFERRED_SIZE, 472,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addContainerGap(376, Short.MAX_VALUE))
                    .addGroup(layout.createSequentialGroup()
```

```
                        .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 582,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jLabel11,
javax.swing.GroupLayout.PREFERRED_SIZE, 141,
javax.swing.GroupLayout.PREFERRED_SIZE))))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(31, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addComponent(jLabel10,
javax.swing.GroupLayout.PREFERRED_SIZE, 261,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel9,
javax.swing.GroupLayout.PREFERRED_SIZE, 376,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(buttonreport,
javax.swing.GroupLayout.PREFERRED_SIZE, 367,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 749,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(buttonback2,
javax.swing.GroupLayout.PREFERRED_SIZE, 116,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 210,
Short.MAX_VALUE)
                        .addComponent(buttonsave,
javax.swing.GroupLayout.PREFERRED_SIZE, 116,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(205, 205, 205)
                        .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 116,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(91, 91, 91))
            .addGroup(layout.createSequentialGroup()
                .addGap(20, 20, 20)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING)
                    .addComponent(jLabel8,
javax.swing.GroupLayout.PREFERRED_SIZE, 230,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel2))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(txtExamName,
javax.swing.GroupLayout.PREFERRED_SIZE, 241,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(jLabel12))
                    .addComponent(txtPPieces))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING, false)
                    .addComponent(buttonflag,
javax.swing.GroupLayout.DEFAULT_SIZE, 206, Short.MAX_VALUE)
                    .addComponent(txtDate))
                .addContainerGap())
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(17, 17, 17)
```

```
                        .addComponent(jLabel1))
                    .addComponent(jLabel11,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(7, 7, 7)
                .addComponent(jLabel7,
javax.swing.GroupLayout.PREFERRED_SIZE, 37,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(9, 9, 9)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(1, 1, 1)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                            .addComponent(jLabel8)
                            .addComponent(txtExamName,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                        .addComponent(jLabel12)
                        .addComponent(txtDate,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(35, 35, 35)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                    .addComponent(txtPPieces,
javax.swing.GroupLayout.PREFERRED_SIZE, 72,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel2)
                    .addComponent(buttonflag,
javax.swing.GroupLayout.PREFERRED_SIZE, 73,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                    .addComponent(buttonreport)
                    .addComponent(jLabel9))
                .addGap(18, 18, 18)
                .addComponent(jLabel10)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 96,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(26, 26, 26)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                    .addComponent(buttonback2,
javax.swing.GroupLayout.PREFERRED_SIZE, 55,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 55,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(buttonsave,
javax.swing.GroupLayout.PREFERRED_SIZE, 55,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );

        pack();
    }// </editor-fold>

    private void buttonreportActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        reportpage rp = new reportpage();
        rp.setStudent(sid1, sname1);
        rp.setPreviousPage(this);
               rp.setVisible(true);
               this.setVisible(false);
    }

    private void buttonflagActionPerformed(java.awt.event.ActionEvent evt)
{
        try {
```

```
            // TODO add your handling code here:
            int confirmation = JOptionPane.showConfirmDialog(this, "Are you
sure you want to "+(flag1 ? "unflag":"flag")+" this student?", (flag1 ?
"Unflag":"Flag")+" Student", JOptionPane.YES_NO_OPTION,
JOptionPane.QUESTION_MESSAGE);
            if(confirmation == JOptionPane.YES_OPTION){
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            pst = sqlConn.prepareStatement("SELECT * FROM
`currentlyenrolledstudentsadmin` WHERE ID = '"+sid1+"';");

            rs = pst.executeQuery();
            rs.next();
            boolean flag = rs.getBoolean("flag");
            Connection con = getConnection();
            Statement st;

            st = con.createStatement();
            String updateQuery = "UPDATE `currentlyenrolledstudentsadmin`
SET `flag`='"+(flag ? 0 : 1)+"' WHERE `ID` = '"+sid1+"'";
            st.addBatch(updateQuery);

            int[] updatedRow =  st.executeBatch();
            System.out.println(updatedRow.length);
            flag1 = !flag;
            buttonflag.setText(flag1 ? "Unflag student?":"Flag student?");

        }} catch (SQLException ex) {

Logger.getLogger(individualstudentpage.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }

    private void buttonback2ActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        if(backpage == "admin") {
           currentlyenrolledstudentsadmin cesa = new
currentlyenrolledstudentsadmin();
        cesa.setName(adminname);
               cesa.setVisible(true);
               this.setVisible(false);
```

```
        }
        else {
            teacherhomepage thp = new teacherhomepage();
            thp.setName(adminname);
            thp.setVisible(true);
            this.setVisible(false);

        }
    }

    private void buttonexitActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
    }

    private void txtDateActionPerformed(java.awt.event.ActionEvent evt) {
    }

    private void buttonsaveActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:


                JFrame frame = new JFrame("Save");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
save", "MYSQL Connector",
            JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)
    {


        Connection con = getConnection();

        Connection con3 = getConnection3();
        Statement st;

        Statement st3;

        try {

            st = con.createStatement();

            st3 = con3.createStatement();
```

```
                String updateQuery = "UPDATE
`currentlyenrolledstudentsadmin` SET
`examname`='"+txtExamName.getText()+"'," +
"`notes`='"+jTextArea2.getText()+"'," +  "`date`='"+txtDate.getText()+"',"
                        + "`piecenames`='"+txtPPieces.getText()+"' WHERE
`ID` = '"+sid1+"'";
                st.addBatch(updateQuery);

                String updateQuery3 = "UPDATE `paststudentrecords` SET
`ExamScore`='"+txtExamName.getText()+"' WHERE `sID` = '"+sid1+"'";
                st3.addBatch(updateQuery3);



            int[] updatedRow =  st.executeBatch();

            int[] updatedRow3 =  st3.executeBatch();
System.out.println(updatedRow.length);


            /*try{
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            String value1 = tfsname.getText();
            String value2 = tftname.getText();
            String value3 = tfexam.getText();
            String value4 = tfpieces.getText();
            String value5 = tfreportstat.getText();
            String value6 = tfpreph.getText();
            String sql = "UPDATE `currentlyenrolledstudentsadmin` SET
`studentname`='"+value1+"',`teachername`='"+value2+"',`examname`='"+value3+
"',`piecenames`='"+value4+"',`reportstatus`='"+value5+"',`prephours`='"+val
ue6+"' WHERE `studentname`='"+value1+"'";
            pst = sqlConn.prepareStatement(sql);
            pst.execute();
            JOptionPane.showMessageDialog(null, "Updated");

            }catch(Exception e) {
            JOptionPane.showMessageDialog(null, e);

            }*/
            /*try {
            Class.forName("com.mysql.jdbc.Driver");
```

```
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            pst = sqlConn.prepareStatement("UPDATE
`currentlyenrolledstudentsadmin` SET `studentname`=
?,`teachername`=?,`examname`=?,`piecenames`=?,`reportstatus`=?,`prephours`=
? WHERE `studentname`= ?");
            pst.setString(1, tfsname.getText());
            pst.setString(2, tftname.getText());
            pst.setString(3, tfexam.getText());
            pst.setString(4, tfpieces.getText());
            pst.setString(5, tfreportstat.getText());
            pst.setString(6, tfpreph.getText());

            pst.executeUpdate();
            JOptionPane.showMessageDialog(this, "Record Updated");
            upDateDB();



            }
            catch(ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
            } catch (SQLException ex){

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);




            }*/
        } catch (SQLException ex) {

Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(Level.
SEVERE, null, ex);
        }}

    }

    /**
     * @param args the command line arguments
```

```
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel
setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with
the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(individualstudentpage.class.getName()).l
og(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(individualstudentpage.class.getName()).l
og(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(individualstudentpage.class.getName()).l
og(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(individualstudentpage.class.getName()).l
og(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new individualstudentpage().setVisible(true);
            }
```

```
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton buttonback2;
    private javax.swing.JButton buttonexit;
    private javax.swing.JButton buttonflag;
    private javax.swing.JButton buttonreport;
    private javax.swing.JButton buttonsave;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JTextArea jTextArea1;
    private javax.swing.JTextArea jTextArea2;
    private javax.swing.JTextField txtDate;
    private javax.swing.JTextField txtExamName;
    private javax.swing.JTextField txtPPieces;
    // End of variables declaration
}
```

Student Report Page:

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to
edit this template
 */
package com.mycompany.csiaf;

import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.Connection;
import java.sql.DriverManager;
```

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/**
 *
 */

public class reportpage extends javax.swing.JFrame {
    private static final String username = "root";
    private static final String password = "";
    private static final String dataConn =
"jdbc:mysql://localhost:3306/currentlyenrolledstudentsadmin";
    private static final String dataConn2 =
"jdbc:mysql://localhost:3306/individualstudentreport";

   Connection sqlConn = null;
   PreparedStatement pst = null;
   ResultSet rs = null;
   int q, i, id, deleteItem;
   private void initSelfListeners() {
    WindowListener taskStarterWindowListener = new WindowListener() {
        @Override
        public void windowOpened(WindowEvent e) {
            /*fillTable();*/
            System.out.println("Performing task..."); //Perform task here.
In this case, we are simulating a startup (only once) time-consuming task
that would use a worker.
        }

        @Override
        public void windowClosing(WindowEvent e) {
            //Do nothing...Or something...You decide!
        }

        @Override
        public void windowClosed(WindowEvent e) {
```

```
            //Do nothing...Or drink coffee...NVM; always drink coffee!
        }

        @Override
        public void windowIconified(WindowEvent e) {
            //Do nothing...Or do EVERYTHING!
        }

        @Override
        public void windowDeiconified(WindowEvent e) {
            //Do nothing...Or break the law...
        }

        @Override
        public void windowActivated(WindowEvent e) {


            //Do nothing...Procrastinate like me!
        }

        @Override
        public void windowDeactivated(WindowEvent e) {
            //Do nothing...And please don't notice I have way too much free
time today...
        }
    };

    //Here is where the magic happens! We make (a listener within) the
frame start listening to the frame's own events!
    this.addWindowListener(taskStarterWindowListener);
}

    /**
     * Creates new form reportpage
     */
    String sid1;
    String sname1;
    public void setStudent(String sid, String sname) {
        sid1 = sid;
        sname1 = sname;
        fillfields();

    }
```

```
     JFrame prevPage;
    public void setPreviousPage(JFrame prevPage1) {
        prevPage = prevPage1;
    }
    public reportpage() {
        initComponents();
        initSelfListeners();
    }
    public Connection getConnection() {
        try{
            Class.forName("com.mysql.jdbc.Driver");

        }catch(ClassNotFoundException ex) {
            System.out.println(ex.getMessage());
        }
        Connection con = null;
        try{
            con =
DriverManager.getConnection("jdbc:mysql://localhost/individualstudentreport
", "root", "");
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }
        return con;

    }
    public void fillfields(){
        try {
            sqlConn = DriverManager.getConnection(dataConn2, username,
password);
            pst = sqlConn.prepareStatement("SELECT * FROM
`individualstudentreport` WHERE IDSTUDENT = '"+sid1+"';");

            rs = pst.executeQuery();
            rs.next();
            btDate.setText(rs.getString("Date"));
            txtS1.setText(rs.getString("Skill1"));
            txtS2.setText(rs.getString("Skill2"));
            txtS3.setText(rs.getString("Skill3"));
            txtS4.setText(rs.getString("Skill4"));
            txtNote.setText(rs.getString("Notes"));
```

```
        } catch (SQLException ex) {
            Logger.getLogger(reportpage.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }

    public Connection getConnection2() {
        try{
            Class.forName("com.mysql.jdbc.Driver");

        }catch(ClassNotFoundException ex) {
            System.out.println(ex.getMessage());
        }
        Connection con = null;
        try{
            con =
DriverManager.getConnection("jdbc:mysql://localhost/currentlyenrolledstuden
tsadmin", "root", "");
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }
        return con;

    }

    public Connection getConnection3() {
        try{
            Class.forName("com.mysql.jdbc.Driver");

        }catch(ClassNotFoundException ex) {
            System.out.println(ex.getMessage());
        }
        Connection con = null;
        try{
            con =
DriverManager.getConnection("jdbc:mysql://localhost/paststudentrecord",
"root", "");
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }
        return con;
```

```
    }

    /**
     * This method is called from within the constructor to initialize the
form.
     * WARNING: Do NOT modify this code. The content of this method is
always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        jLabel9 = new javax.swing.JLabel();
        jLabel10 = new javax.swing.JLabel();
        jLabel11 = new javax.swing.JLabel();
        jLabel12 = new javax.swing.JLabel();
        jLabel13 = new javax.swing.JLabel();
        jLabel14 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        txtNote = new javax.swing.JTextArea();
        buttonexit = new javax.swing.JButton();
        buttonback3 = new javax.swing.JButton();
        buttonsubmit = new javax.swing.JButton();
        txtS1 = new javax.swing.JTextField();
        txtS2 = new javax.swing.JTextField();
        txtS3 = new javax.swing.JTextField();
        txtS4 = new javax.swing.JTextField();
        btDate = new javax.swing.JTextField();
        jLabel15 = new javax.swing.JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jLabel1.setFont(new java.awt.Font("Segoe UI", 0, 40)); // NOI18N
        jLabel1.setText("Write Report ");

        jLabel8.setFont(new java.awt.Font("Segoe UI", 0, 24)); // NOI18N
        jLabel8.setText("Communication");
```

```
        jLabel9.setFont(new java.awt.Font("Segoe UI", 0, 30)); // NOI18N
        jLabel9.setText("Score (1-10)");

        jLabel10.setFont(new java.awt.Font("Segoe UI", 0, 30)); // NOI18N
        jLabel10.setText("Skills");

        jLabel11.setFont(new java.awt.Font("Segoe UI", 0, 30)); // NOI18N
        jLabel11.setText("Date:");

        jLabel12.setFont(new java.awt.Font("Segoe UI", 0, 24)); // NOI18N
        jLabel12.setText("Creative Thinking");

        jLabel13.setFont(new java.awt.Font("Segoe UI", 0, 24)); // NOI18N
        jLabel13.setText("Speaking");

        jLabel14.setFont(new java.awt.Font("Segoe UI", 0, 24)); // NOI18N
        jLabel14.setText("Acting");

        txtNote.setColumns(20);
        txtNote.setRows(5);
        jScrollPane1.setViewportView(txtNote);

        buttonexit.setBackground(new java.awt.Color(0, 0, 0));
        buttonexit.setFont(new java.awt.Font("Segoe UI", 0, 18)); // NOI18N
        buttonexit.setForeground(new java.awt.Color(255, 255, 255));
        buttonexit.setText("Exit");
        buttonexit.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonexitActionPerformed(evt);
            }
        });

        buttonback3.setBackground(new java.awt.Color(0, 0, 0));
        buttonback3.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        buttonback3.setForeground(new java.awt.Color(255, 255, 255));
        buttonback3.setText("Back");
        buttonback3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonback3ActionPerformed(evt);
            }
        });

        buttonsubmit.setBackground(new java.awt.Color(0, 0, 0));
```

```
        buttonsubmit.setFont(new java.awt.Font("Segoe UI", 0, 18)); //
NOI18N
        buttonsubmit.setForeground(new java.awt.Color(255, 255, 255));
        buttonsubmit.setText("Submit");
        buttonsubmit.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonsubmitActionPerformed(evt);
            }
        });

        jLabel15.setIcon(new
javax.swing.ImageIcon("C:/Users/Abhay/OneDrive/Documents/NetBeansProjects/C
SIAF/src/main/java/image/vocallogo.png")
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(16, 16, 16)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(jLabel10,
javax.swing.GroupLayout.PREFERRED_SIZE, 78,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(116, 116, 116))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                                .addComponent(jLabel8)
                                .addGap(25, 25, 25)))
                        .addGap(249, 249, 249)
                        .addComponent(txtS1))
```

```
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                            .addComponent(jLabel14)
                            .addComponent(buttonback3,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jLabel13)
                            .addComponent(jLabel12))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING)
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(jLabel9,
javax.swing.GroupLayout.PREFERRED_SIZE, 172,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(42, 42, 42))
                            .addComponent(txtS2,
javax.swing.GroupLayout.PREFERRED_SIZE, 281,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtS3,
javax.swing.GroupLayout.PREFERRED_SIZE, 281,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtS4,
javax.swing.GroupLayout.PREFERRED_SIZE, 281,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addGap(106, 106, 106))
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(342, 342, 342)
                        .addComponent(buttonsubmit,
javax.swing.GroupLayout.PREFERRED_SIZE, 95,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(221, 221, 221)
                        .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 95,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(30, 30, 30)
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 597,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap(93, Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 598,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(208, 208, 208)
                        .addComponent(btDate,
javax.swing.GroupLayout.PREFERRED_SIZE, 258,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 141,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(136, 136, 136)
                    .addComponent(jLabel11,
javax.swing.GroupLayout.PREFERRED_SIZE, 78,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addContainerGap(632, Short.MAX_VALUE)))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING, false)
                    .addGroup(layout.createSequentialGroup()
```

```
                        .addGap(20, 20, 20)
                        .addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(btDate,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING)
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING, false)
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(jLabel9)
                                .addGap(31, 31, 31)
                                .addComponent(txtS1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                                .addComponent(txtS2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                                .addComponent(txtS3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(18, 18, 18)
                                .addComponent(txtS4,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(48, 48, 48))
                            .addGroup(layout.createSequentialGroup()
                                .addGap(10, 10, 10)
                                .addComponent(jLabel10)
                                .addGap(18, 18, 18)
                                .addComponent(jLabel8)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING)

.addGroup(layout.createSequentialGroup()
                                        .addComponent(jLabel12)
                                        .addGap(38, 38, 38))
                                    .addComponent(jLabel13))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(jLabel14)
                                .addGap(35, 35, 35)))
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 62,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                            .addComponent(buttonback3,
javax.swing.GroupLayout.PREFERRED_SIZE, 52,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(buttonsubmit,
javax.swing.GroupLayout.PREFERRED_SIZE, 52,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 52,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(16, 16, 16))
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(102, 102, 102)
                    .addComponent(jLabel11)
                    .addContainerGap(394, Short.MAX_VALUE)))
        );

        pack();
    }// </editor-fold>

    private void buttonexitActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
    }

    private void buttonback3ActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
//        adminhomepage ahp = new adminhomepage();
//        ahp.setVisible(true);
//        this.setVisible(false);
             prevPage.setVisible(true);
             this.setVisible(false);
    }

    private void buttonsubmitActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        Connection con = getConnection();
        Statement st;
        Connection con2 = getConnection2();
        Statement st2;
        Connection con3 = getConnection3();
        Statement st3;
        JFrame frame = new JFrame("Submit");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
submit", "MYSQL Connector",
            JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)
    {
        try {
            st = con.createStatement();
            String date = btDate.getText();
```

```
            String skill1 = txtS1.getText();
            String skill2 = txtS2.getText();
            String skill3 = txtS3.getText();
            String skill4 = txtS4.getText();
            String notes = txtNote.getText();


                String updateQuery = "UPDATE `individualstudentreport` SET
`Date`='"+date+"',`Skill1`='"+skill1+"',`Skill2`='"+skill2+"',`Skill3`='"+s
kill3+"',"
                        + "`Skill4`='"+skill4+"', `Notes`='"+notes+"' WHERE
`IDSTUDENT` = '"+sid1+"'";
                st.addBatch(updateQuery);



            int[] updatedRow =  st.executeBatch();
System.out.println(updatedRow.length);

        st2 = con2.createStatement();
        boolean reportDone = false;
            if (date != "" && skill1 != "" && skill2 != "" && skill3 != ""
&& skill4 != "" && notes != "" && date != null && skill1 != null && skill2
!= null && skill3 != null && skill4 != null && notes != null) {
                reportDone = true;
            }


                String updateQuery2 = "UPDATE
`currentlyenrolledstudentsadmin` SET `reportstatus`='"+ (reportDone ?
"Done" : "Not Done") +"' WHERE `ID` = '"+sid1+"'";
                st2.addBatch(updateQuery2);



            int[] updatedRow2 =  st2.executeBatch();
System.out.println(updatedRow2.length);


        st3 = con3.createStatement();


                String updateQuery3 = "UPDATE `paststudentrecords` SET
```

```
`Report`='"+ (reportDone ? "Done" : "Not Done") +"' WHERE `sID` =
'"+sid1+"'";
                st3.addBatch(updateQuery3);



            int[] updatedRow3 =  st3.executeBatch();
System.out.println(updatedRow3.length);


            /*try{
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            String value1 = tfsname.getText();
            String value2 = tftname.getText();
            String value3 = tfexam.getText();
            String value4 = tfpieces.getText();
            String value5 = tfreportstat.getText();
            String value6 = tfpreph.getText();
            String sql = "UPDATE `currentlyenrolledstudentsadmin` SET
`studentname`='"+value1+"',`teachername`='"+value2+"',`examname`='"+value3+
"',`piecenames`='"+value4+"',`reportstatus`='"+value5+"',`prephours`='"+val
ue6+"' WHERE `studentname`='"+value1+"'";
            pst = sqlConn.prepareStatement(sql);
            pst.execute();
            JOptionPane.showMessageDialog(null, "Updated");

            }catch(Exception e) {
            JOptionPane.showMessageDialog(null, e);

            }*/
            /*try {
            Class.forName("com.mysql.jdbc.Driver");
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            pst = sqlConn.prepareStatement("UPDATE
`currentlyenrolledstudentsadmin` SET `studentname`=
?,`teachername`=?,`examname`=?,`piecenames`=?,`reportstatus`=?,`prephours`=
? WHERE `studentname`= ?");
            pst.setString(1, tfsname.getText());
            pst.setString(2, tftname.getText());
            pst.setString(3, tfexam.getText());
            pst.setString(4, tfpieces.getText());
```

```
            pst.setString(5, tfreportstat.getText());
            pst.setString(6, tfpreph.getText());

            pst.executeUpdate();
            JOptionPane.showMessageDialog(this, "Record Updated");
            upDateDB();



            }
            catch(ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
            } catch (SQLException ex){

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);




            }*/
        } catch (SQLException ex) {

Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(Level.
SEVERE, null, ex);
        }}
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel
setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with
the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
```

```
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(reportpage.class.getName()).log(java.uti
l.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(reportpage.class.getName()).log(java.uti
l.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(reportpage.class.getName()).log(java.uti
l.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(reportpage.class.getName()).log(java.uti
l.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new reportpage().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JTextField btDate;
    private javax.swing.JButton buttonback3;
    private javax.swing.JButton buttonexit;
    private javax.swing.JButton buttonsubmit;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
```

```
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel14;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextArea txtNote;
    private javax.swing.JTextField txtS1;
    private javax.swing.JTextField txtS2;
    private javax.swing.JTextField txtS3;
    private javax.swing.JTextField txtS4;
    // End of variables declaration
}
```

Create Teacher Account Page:

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to
edit this template
 */
package com.mycompany.csiaf;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.UUID;
import javax.swing.JFrame;
import javax.swing.JOptionPane;

/**
 *
 */
public class createteacheraccount extends javax.swing.JFrame {
    private static final String username = "root";
    private static final String password = "";
```

```
    private static final String dataConn =
"jdbc:mysql://localhost:3306/multiuserlogin";
    String adminname;
    public void setName(String name){
        adminname = name;

    }

   Connection sqlConn = null;
   PreparedStatement pst = null;
   ResultSet rs = null;
   int q, i, id, deleteItem;

    /**
     * Creates new form createteacheraccount
     */
    public createteacheraccount() {
        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the
form.
     * WARNING: Do NOT modify this code. The content of this method is
always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        btnSignIn = new javax.swing.JButton();
        jLabel6 = new javax.swing.JLabel();
        jLabel7 = new javax.swing.JLabel();
        edtpasscode = new javax.swing.JTextField();
        jLabel8 = new javax.swing.JLabel();
        edtname = new javax.swing.JTextField();
        jLabel9 = new javax.swing.JLabel();
        edtemail = new javax.swing.JTextField();
        jLabel10 = new javax.swing.JLabel();
        buttoncreate = new javax.swing.JButton();
        buttonback1 = new javax.swing.JButton();
        buttonexit = new javax.swing.JButton();
```

```
        jLabel15 = new javax.swing.JLabel();

        btnSignIn.setBackground(new java.awt.Color(0, 0, 0));
        btnSignIn.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        btnSignIn.setForeground(new java.awt.Color(255, 255, 255));
        btnSignIn.setText("Sign-In");
        btnSignIn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnSignInActionPerformed(evt);
            }
        });


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jLabel6.setFont(new java.awt.Font("Arial", 0, 30)); // NOI18N
        jLabel6.setText("Teacher Passcode:");

        jLabel7.setFont(new java.awt.Font("Arial", 0, 40)); // NOI18N
        jLabel7.setText("Create Teacher Account");

        edtpasscode.setText("Input");
        edtpasscode.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusGained(java.awt.event.FocusEvent evt) {
                edtpasscodeFocusGained(evt);
            }
        });
        edtpasscode.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                edtpasscodeActionPerformed(evt);
            }
        });

        jLabel8.setFont(new java.awt.Font("Arial", 0, 30)); // NOI18N
        jLabel8.setText("Teacher Name:");

        edtname.setText("Input");
        edtname.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusGained(java.awt.event.FocusEvent evt) {
                edtnameFocusGained(evt);
            }
        });
        edtname.addActionListener(new java.awt.event.ActionListener() {
```

```
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                edtnameActionPerformed(evt);
            }
        });

        jLabel9.setFont(new java.awt.Font("Arial", 0, 30)); // NOI18N
        jLabel9.setText("Teacher Email:");

        edtemail.setText("Input");
        edtemail.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusGained(java.awt.event.FocusEvent evt) {
                edtemailFocusGained(evt);
            }
        });
        edtemail.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                edtemailActionPerformed(evt);
            }
        });

        jLabel10.setFont(new java.awt.Font("Arial", 0, 30)); // NOI18N
        jLabel10.setText("Share credentials with teacher");

        buttoncreate.setBackground(new java.awt.Color(0, 0, 0));
        buttoncreate.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        buttoncreate.setForeground(new java.awt.Color(255, 255, 255));
        buttoncreate.setText("Create Account");
        buttoncreate.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttoncreateActionPerformed(evt);
            }
        });

        buttonback1.setBackground(new java.awt.Color(0, 0, 0));
        buttonback1.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        buttonback1.setForeground(new java.awt.Color(255, 255, 255));
        buttonback1.setText("Back");
        buttonback1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonback1ActionPerformed(evt);
            }
        });
```

```
        buttonexit.setBackground(new java.awt.Color(0, 0, 0));
        buttonexit.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        buttonexit.setForeground(new java.awt.Color(255, 255, 255));
        buttonexit.setText("Exit");
        buttonexit.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonexitActionPerformed(evt);
            }
        });

        jLabel15.setIcon(new
javax.swing.ImageIcon("C:/Users/Abhay/OneDrive/Documents/NetBeansProjects/C
SIAF/src/main/java/image/vocallogo.png")
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jLabel9,
javax.swing.GroupLayout.PREFERRED_SIZE, 228,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(edtemail,
javax.swing.GroupLayout.PREFERRED_SIZE, 407,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(145, 145, 145)
                        .addComponent(jLabel10,
javax.swing.GroupLayout.PREFERRED_SIZE, 421,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addContainerGap()
```

```
                        .addComponent(jLabel8,
javax.swing.GroupLayout.PREFERRED_SIZE, 228,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(edtname,
javax.swing.GroupLayout.PREFERRED_SIZE, 407,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING, false)
                        .addGroup(layout.createSequentialGroup()
                            .addGap(31, 31, 31)
                            .addComponent(buttonback1,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createSequentialGroup()
                            .addComponent(jLabel6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(edtpasscode,
javax.swing.GroupLayout.PREFERRED_SIZE, 407,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addGap(0, 19, Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()
                .addGap(326, 326, 326)
                .addComponent(buttoncreate)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 141,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(10, 10, 10)
                    .addComponent(jLabel7,
javax.swing.GroupLayout.PREFERRED_SIZE, 462,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addContainerGap(219, Short.MAX_VALUE)))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(62, 62, 62)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                            .addComponent(jLabel8,
javax.swing.GroupLayout.PREFERRED_SIZE, 58,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(edtname,
javax.swing.GroupLayout.PREFERRED_SIZE, 31,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(jLabel9,
javax.swing.GroupLayout.PREFERRED_SIZE, 58,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(142, 142, 142)
                        .addComponent(edtemail,
javax.swing.GroupLayout.PREFERRED_SIZE, 31,
javax.swing.GroupLayout.PREFERRED_SIZE)))
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jLabel6,
javax.swing.GroupLayout.PREFERRED_SIZE, 58,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(17, 17, 17)
                        .addComponent(edtpasscode,
javax.swing.GroupLayout.PREFERRED_SIZE, 31,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(buttoncreate,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel10,
javax.swing.GroupLayout.PREFERRED_SIZE, 58,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                    .addComponent(buttonback1,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(38, 38, 38)
                    .addComponent(jLabel7,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE, 103,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addContainerGap(460, Short.MAX_VALUE)))
        );

        pack();
    }// </editor-fold>

    private void edtpasscodeActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
    }

    private void edtnameActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void edtemailActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnSignInActionPerformed(java.awt.event.ActionEvent evt) {


    }

    private void buttoncreateActionPerformed(java.awt.event.ActionEvent
evt) {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            pst = sqlConn.prepareStatement("INSERT INTO
`multiuserlogin`(`IDTEACHER`,`TeacherName`, `EmailAddress`, `Passcode`,
`UserType`) VALUES (?,?,?,?,?)");
           System.out.println(UUID.randomUUID().toString().length());
            pst.setString(1, UUID.randomUUID().toString().replaceAll("_",
""));

            pst.setString(2, edtname.getText());
            pst.setString(3, edtemail.getText());
            pst.setString(4, edtpasscode.getText());
            pst.setString(5, "User");
```

```
            pst.executeUpdate();
            JOptionPane.showMessageDialog(this, "Record Added");




        }
        catch(ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (SQLException ex){

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);




        }

    }

    private void buttonback1ActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        adminhomepage ahp = new adminhomepage();
        ahp.setName(adminname);
               ahp.setVisible(true);
               this.setVisible(false);
    }
private JFrame frame;
    private void buttonexitActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        frame = new JFrame("Exit");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
exit", "MYSQL Connector",
                JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)
        {
```

```
            System.exit(0);
        }
    }

    private void edtnameFocusGained(java.awt.event.FocusEvent evt) {
        // TODO add your handling code here:
        if("Input".equals(edtname.getText())) {
        edtname.setText("");
        }
    }

    private void edtemailFocusGained(java.awt.event.FocusEvent evt) {
        // TODO add your handling code here:
        if("Input".equals(edtemail.getText())) {
        edtemail.setText("");
        }

    }

    private void edtpasscodeFocusGained(java.awt.event.FocusEvent evt) {
        // TODO add your handling code here:
        if("Input".equals(edtpasscode.getText())) {
        edtpasscode.setText("");
        }
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel
setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with
the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
```

```
javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(createteacheraccount.class.getName()).lo
g(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(createteacheraccount.class.getName()).lo
g(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(createteacheraccount.class.getName()).lo
g(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(createteacheraccount.class.getName()).lo
g(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new createteacheraccount().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnSignIn;
    private javax.swing.JButton buttonback1;
    private javax.swing.JButton buttoncreate;
    private javax.swing.JButton buttonexit;
    private javax.swing.JTextField edtemail;
    private javax.swing.JTextField edtname;
    private javax.swing.JTextField edtpasscode;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
```

```
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    // End of variables declaration
}
```

Teacher Homepage:

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to
edit this template
 */
package com.mycompany.csiaf;

import java.awt.Color;
import java.awt.Component;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.RowSorter;
import javax.swing.SortOrder;
import javax.swing.table.DefaultTableCellRenderer;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableCellRenderer;
import javax.swing.table.TableModel;
import javax.swing.table.TableRowSorter;

/**
 *
 */
public class teacherhomepage extends javax.swing.JFrame {
```

```
    private static final String username = "root";
    private static final String password = "";
    private static final String dataConn =
"jdbc:mysql://localhost:3306/currentlyenrolledstudentsadmin";

   Connection sqlConn = null;
   PreparedStatement pst = null;
   ResultSet rs = null;
   int q, i, id, deleteItem;
   String teachername;
    public void setName(String name){
        teachername = name;

    }
       public void labelName() {
           namelabel.setText(teachername);

       };
    private void initSelfListeners() {
    WindowListener taskStarterWindowListener = new WindowListener() {
        @Override
        public void windowOpened(WindowEvent e) {
            fillTable();
            System.out.println("Performing task...");
            labelName();
            cth.setDefaultRenderer(Object.class, new TableCellRenderer(){
    private DefaultTableCellRenderer DEFAULT_RENDERER =  new
DefaultTableCellRenderer();

            @Override
            public Component getTableCellRendererComponent(JTable table,
Object value, boolean isSelected, boolean hasFocus, int row, int column) {
                Component c =
DEFAULT_RENDERER.getTableCellRendererComponent(table, value, isSelected,
hasFocus, row, column);
        if(cth.getModel().getValueAt(row, 7).toString().equals("true")){


        c.setBackground(Color.RED);


}
        System.out.println(value);
```

```
       //Add below code here
                return c;
            }

        });
            //Perform task here. In this case, we are simulating a startup
(only once) time-consuming task that would use a worker.
        }

        @Override
        public void windowClosing(WindowEvent e) {
            //Do nothing...Or something...You decide!
        }

        @Override
        public void windowClosed(WindowEvent e) {
            //Do nothing...Or drink coffee...NVM; always drink coffee!
        }

        @Override
        public void windowIconified(WindowEvent e) {
            //Do nothing...Or do EVERYTHING!
        }

        @Override
        public void windowDeiconified(WindowEvent e) {
            //Do nothing...Or break the law...
        }

        @Override
        public void windowActivated(WindowEvent e) {


            //Do nothing...Procrastinate like me!
        }

        @Override
        public void windowDeactivated(WindowEvent e) {
            //Do nothing...And please don't notice I have way too much free
time today...
        }
    };
```

```
    //Here is where the magic happens! We make (a listener within) the
frame start listening to the frame's own events!
    this.addWindowListener(taskStarterWindowListener);
}

    /**
     * Creates new form teacherhomepage
     */
    public teacherhomepage() {
        initComponents();

                initSelfListeners();


    }
    public Connection getConnection() {
        try{
            Class.forName("com.mysql.jdbc.Driver");

        }catch(ClassNotFoundException ex) {
            System.out.println(ex.getMessage());
        }
        Connection con = null;
        try{
            con =
DriverManager.getConnection("jdbc:mysql://localhost/currentlyenrolledstuden
tsadmin", "root", "");
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }
        return con;

    }
    public void fillTable() {
        DefaultTableModel StudentTable = (DefaultTableModel)cth.getModel();
        StudentTable.setRowCount(0);
        Connection con = getConnection();
        Statement ps;
        ResultSet rs;
        DefaultTableModel model = (DefaultTableModel) cth.getModel();
        try {
            ps = con.createStatement();
```

```
            rs = ps.executeQuery("SELECT * FROM
`currentlyenrolledstudentsadmin`");

            while(rs.next()) {
                Object[] row = new Object[cth.getColumnCount()];
                row[0] = rs.getString("ID");
                row[1] = rs.getString("studentname");
                row[2] = rs.getString("teachername");
                row[3] = rs.getString("examname");
                row[4] = rs.getString("piecenames");
                row[5] = rs.getString("reportstatus");
                row[6] = rs.getString("prephours");
                row[7] = rs.getBoolean("flag");
                row[8] = rs.getString("date");
                row[9] = rs.getString("notes");

                model.addRow(row);


            }

        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }

    }


    /**
     * This method is called from within the constructor to initialize the
form.
     * WARNING: Do NOT modify this code. The content of this method is
always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jScrollPane1 = new javax.swing.JScrollPane();
        cth = new javax.swing.JTable()
        {
```

```
            public Component prepareRenderer(
                TableCellRenderer renderer, int row, int column)
            {
                try {
                    Component c = super.prepareRenderer(renderer, row,
column);

                    //  add custom rendering here
                    if (!isRowSelected(row))
                    {

                        c.setBackground(getBackground());
                        int modelRow = convertRowIndexToModel(row);
                        String type =
(String)getModel().getValueAt(modelRow, 7);
                        if ("1".equals(type)) c.setBackground(Color.RED);

                    }
                    return c;
                } catch(NullPointerException exc) {}
                catch( ClassCastException exc) {}
                return super.prepareRenderer(renderer, row, column);
            }}
            ;
            jLabel3 = new javax.swing.JLabel();
            namelabel = new javax.swing.JLabel();
            buttonexit = new javax.swing.JButton();
            buttonfilter = new javax.swing.JButton();
            jComboBoxfilter = new javax.swing.JComboBox<>();
            jLabel15 = new javax.swing.JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

            cth.setModel(new javax.swing.table.DefaultTableModel(
                new Object [][] {
                    {null, null, null, null, null, null, null, null, null,
null},
                    {null, null, null, null, null, null, null, null, null,
null},
                    {null, null, null, null, null, null, null, null, null,
null},
                    {null, null, null, null, null, null, null, null, null,
```

```
null}
                },
                new String [] {
                    "ID", "Student Name", "Teacher Name", "Exam", "Pieces",
"Report Status", "Prep Hours", "flag", "Date", "Notes"
                }
            ) {
                Class[] types = new Class [] {
                    java.lang.Object.class, java.lang.Object.class,
java.lang.Object.class, java.lang.Object.class, java.lang.Object.class,
java.lang.Object.class, java.lang.Object.class, java.lang.String.class,
java.lang.String.class, java.lang.String.class
                };
                boolean[] canEdit = new boolean [] {
                    false, true, true, true, true, true, true, false, true,
true
                };

                public Class getColumnClass(int columnIndex) {
                    return types [columnIndex];
                }

                public boolean isCellEditable(int rowIndex, int
columnIndex) {
                    return canEdit [columnIndex];
                }
            });
            cth.addMouseListener(new java.awt.event.MouseAdapter() {
                public void mouseClicked(java.awt.event.MouseEvent evt) {
                    cthMouseClicked(evt);
                }
            });
            jScrollPane1.setViewportView(cth);
            if (cth.getColumnModel().getColumnCount() > 0) {
                cth.getColumnModel().getColumn(0).setMinWidth(0);
                cth.getColumnModel().getColumn(0).setMaxWidth(0);
                cth.getColumnModel().getColumn(7).setMinWidth(0);
                cth.getColumnModel().getColumn(7).setPreferredWidth(0);
                cth.getColumnModel().getColumn(7).setMaxWidth(0);
                cth.getColumnModel().getColumn(8).setMinWidth(0);
                cth.getColumnModel().getColumn(8).setPreferredWidth(0);
                cth.getColumnModel().getColumn(8).setMaxWidth(0);
                cth.getColumnModel().getColumn(9).setMinWidth(0);
```

```
                cth.getColumnModel().getColumn(9).setPreferredWidth(0);
                cth.getColumnModel().getColumn(9).setMaxWidth(0);
            }

            jLabel3.setFont(new java.awt.Font("Segoe UI", 0, 50)); //
NOI18N
            jLabel3.setText("Hello");

            namelabel.setFont(new java.awt.Font("Segoe UI", 0, 50)); //
NOI18N

            buttonexit.setBackground(new java.awt.Color(0, 0, 0));
            buttonexit.setFont(new java.awt.Font("Arial", 0, 18)); //
NOI18N
            buttonexit.setForeground(new java.awt.Color(255, 255, 255));
            buttonexit.setText("Exit");
            buttonexit.addActionListener(new
java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt)
{
                    buttonexitActionPerformed(evt);
                }
            });

            buttonfilter.setText("Filter");
            buttonfilter.addActionListener(new
java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt)
{
                    buttonfilterActionPerformed(evt);
                }
            });

            jComboBoxfilter.setModel(new
javax.swing.DefaultComboBoxModel<>(new String[] { "Student Name", "Teacher
Name", "Exam", "Report Status", "Prep Hours" }));
            jComboBoxfilter.addItemListener(new
java.awt.event.ItemListener() {
                public void itemStateChanged(java.awt.event.ItemEvent evt)
{
                    jComboBoxfilterItemStateChanged(evt);
                }
            });
```

```
            jComboBoxfilter.addActionListener(new
java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt)
{
                    jComboBoxfilterActionPerformed(evt);
                }
            });

            jLabel15.setIcon(new
javax.swing.ImageIcon("C:/Users/Abhay/OneDrive/Documents/NetBeansProjects/C
SIAF/src/main/java/image/vocallogo.png")
            );

            javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
            getContentPane().setLayout(layout);
            layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 531,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                                .addGroup(layout.createSequentialGroup()
                                    .addGap(0, 29, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                                        .addComponent(jComboBoxfilter,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                        .addComponent(buttonfilter,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)))
```

```
                                .addComponent(buttonexit,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)))
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(jLabel3,
javax.swing.GroupLayout.PREFERRED_SIZE, 136,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(namelabel,
javax.swing.GroupLayout.PREFERRED_SIZE, 276,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 119,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addContainerGap())
            );
            layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                    .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                            .addComponent(jLabel3)
                            .addComponent(namelabel))
                        .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 111,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING, false)
```

```
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(buttonfilter)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(jComboBoxfilter,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 37,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 354,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGap(23, 23, 23))
            );

            pack();
        }// </editor-fold>

    private void cthMouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
        DefaultTableModel RecordTable = (DefaultTableModel)cth.getModel();

        int SelectedRows = cth.getSelectedRow();
                String sid = RecordTable.getValueAt(SelectedRows,
0).toString();
                String sname = RecordTable.getValueAt(SelectedRows,
1).toString();
                String examname = (String)
RecordTable.getValueAt(SelectedRows, 3);
                String pieces = (String)
RecordTable.getValueAt(SelectedRows,4);
                boolean flag = (boolean)
RecordTable.getValueAt(SelectedRows,7);
                String date = (String)
RecordTable.getValueAt(SelectedRows,8);
                String notes = (String)
RecordTable.getValueAt(SelectedRows,9);
```

```
                individualstudentpage isp = new individualstudentpage();
                isp.setName(teachername);

                isp.setStudent(sid, sname, examname, date, pieces, flag,
notes);

               isp.setVisible(true);
               this.setVisible(false);




    }

    private void buttonexitActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        JFrame frame = new JFrame("Exit");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
exit", "MYSQL Connector",
            JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)
    {
        System.exit(0);
        }
    }

    private void buttonfilterActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:

        cth.setRowSorter(null);
        TableRowSorter<TableModel> sorter = new
TableRowSorter<TableModel>(cth.getModel());
        cth.setRowSorter(sorter);

        ArrayList<RowSorter.SortKey> sortKeys = new ArrayList<>(25);
        sortKeys.add(new
RowSorter.SortKey(jComboBoxfilter.getSelectedIndex()+1,
SortOrder.ASCENDING));
```

```
        sortKeys.add(new RowSorter.SortKey(0, SortOrder.ASCENDING));
        sorter.setSortKeys(sortKeys);

    }

    private void jComboBoxfilterItemStateChanged(java.awt.event.ItemEvent
evt) {
        // TODO add your handling code here:
    }

    private void jComboBoxfilterActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel
setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with
the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(teacherhomepage.class.getName()).log(jav
a.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(teacherhomepage.class.getName()).log(jav
```

```
a.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(teacherhomepage.class.getName()).log(jav
a.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(teacherhomepage.class.getName()).log(jav
a.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new teacherhomepage().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton buttonexit;
    private javax.swing.JButton buttonfilter;
    private javax.swing.JTable cth;
    private javax.swing.JComboBox<String> jComboBoxfilter;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel namelabel;
    // End of variables declaration
}
```

Past Student Records Page:

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to
edit this template
 */
package com.mycompany.csiaf;
```

```
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.UUID;
import java.util.Vector;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.RowSorter;
import javax.swing.SortOrder;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableModel;
import javax.swing.table.TableRowSorter;

/**
 *
 */
public class paststudentrecords extends javax.swing.JFrame {
    private static final String username = "root";
    private static final String password = "";
    private static final String dataConn =
"jdbc:mysql://localhost:3306/currentlyenrolledstudentsadmin";
    String adminname;
    public void setName(String name){
        adminname = name;

    }

   Connection sqlConn = null;
   PreparedStatement pst = null;
   ResultSet rs = null;
   int q, i, id, deleteItem;
   private void initSelfListeners() {
    WindowListener taskStarterWindowListener = new WindowListener() {
        @Override
```

```
        public void windowOpened(WindowEvent e) {
            fillTable();
            System.out.println("Performing task..."); //Perform task here.
In this case, we are simulating a startup (only once) time-consuming task
that would use a worker.
        }

        @Override
        public void windowClosing(WindowEvent e) {
            //Do nothing...Or something...You decide!
        }

        @Override
        public void windowClosed(WindowEvent e) {
            //Do nothing...Or drink coffee...NVM; always drink coffee!
        }

        @Override
        public void windowIconified(WindowEvent e) {
            //Do nothing...Or do EVERYTHING!
        }

        @Override
        public void windowDeiconified(WindowEvent e) {
            //Do nothing...Or break the law...
        }

        @Override
        public void windowActivated(WindowEvent e) {


            //Do nothing...Procrastinate like me!
        }

        @Override
        public void windowDeactivated(WindowEvent e) {
            //Do nothing...And please don't notice I have way too much free
time today...
        }
    };

    //Here is where the magic happens! We make (a listener within) the
frame start listening to the frame's own events!
```

```
    this.addWindowListener(taskStarterWindowListener);
}

    /**
     * Creates new form paststudentrecords
     */
    public paststudentrecords() {
        initComponents();
        initSelfListeners();
    }
    public Connection getConnection() {
        try{
            Class.forName("com.mysql.jdbc.Driver");

        }catch(ClassNotFoundException ex) {
            System.out.println(ex.getMessage());
        }
        Connection con = null;
        try{
            con =
DriverManager.getConnection("jdbc:mysql://localhost/paststudentrecord",
"root", "");
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }
        return con;

    }
    public void fillTable() {
        Connection con = getConnection();
        Statement ps;
        ResultSet rs;
        DefaultTableModel model = (DefaultTableModel) psrtab.getModel();
        try {
            ps = con.createStatement();
            rs = ps.executeQuery("SELECT * FROM `paststudentrecords`");

            while(rs.next()) {
                Object[] row = new Object[psrtab.getColumnCount()];
                row[0] = rs.getString("sID");
                row[1] = rs.getString("Sname");
                row[2] = rs.getInt("Term");
```

```
                row[3] = rs.getString("ExamScore");
                row[4] = rs.getString("Report");

                model.addRow(row);
            }
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }

    }




    public void upDateDB(){
        try{
        Class.forName("com.mysql.jdbc.Driver");
        sqlConn = DriverManager.getConnection(dataConn, username,
password);
        pst = sqlConn.prepareStatement("SELECT * FROM
`currentlyenrolledstudentsadmin`");

        rs = pst.executeQuery();
        ResultSetMetaData stData = rs.getMetaData();
        q = stData.getColumnCount();

        DefaultTableModel StudentTable =
(DefaultTableModel)psrtab.getModel();
        StudentTable.setRowCount(0);

        while(rs.next()) {
            Vector columnData = new Vector();
            for(i = 1;i<=q;i++) {
                columnData.add(rs.getString("ID"));
                columnData.add(rs.getString("studentname"));
                columnData.add(rs.getString("teachername"));
                columnData.add(rs.getString("examname"));
                columnData.add(rs.getString("piecenames"));
                columnData.add(rs.getString("reportstatus"));
                columnData.add(rs.getString("prephours"));
```

```
            }
            StudentTable.addRow(columnData);

        }
        }
        catch(Exception ex) {
            JOptionPane.showMessageDialog(null, ex);

        }

    }

    /**
     * This method is called from within the constructor to initialize the
form.
     * WARNING: Do NOT modify this code. The content of this method is
always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jLabel2 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        psrtab = new javax.swing.JTable();
        buttonexit = new javax.swing.JButton();
        bback = new javax.swing.JButton();
        btnNewTerm = new javax.swing.JButton();
        buttonfilter2 = new javax.swing.JButton();
        DropDownFilter = new javax.swing.JComboBox<>();
        jLabel15 = new javax.swing.JLabel();

        jLabel2.setText("Teacher Name");


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jLabel4.setFont(new java.awt.Font("Segoe UI", 0, 50)); // NOI18N
        jLabel4.setText("Past Student Records");
```

```
        psrtab.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "ID", "Name", "Term", "Exam+Score", "Report"
            }
        ) {
            boolean[] canEdit = new boolean [] {
                false, true, true, true, true
            };

            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
        });
        psrtab.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                psrtabMouseClicked(evt);
            }
        });
        jScrollPane1.setViewportView(psrtab);
        if (psrtab.getColumnModel().getColumnCount() > 0) {
            psrtab.getColumnModel().getColumn(0).setMinWidth(0);
            psrtab.getColumnModel().getColumn(0).setMaxWidth(0);
        }

        buttonexit.setBackground(new java.awt.Color(0, 0, 0));
        buttonexit.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        buttonexit.setForeground(new java.awt.Color(255, 255, 255));
        buttonexit.setText("Exit");
        buttonexit.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonexitActionPerformed(evt);
            }
        });

        bback.setBackground(new java.awt.Color(0, 0, 0));
        bback.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        bback.setForeground(new java.awt.Color(255, 255, 255));
        bback.setText("Back");
        bback.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
                bbackActionPerformed(evt);
            }
        });

        btnNewTerm.setBackground(new java.awt.Color(0, 0, 0));
        btnNewTerm.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        btnNewTerm.setForeground(new java.awt.Color(255, 255, 255));
        btnNewTerm.setText("Start New Term");
        btnNewTerm.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnNewTermActionPerformed(evt);
            }
        });

        buttonfilter2.setText("Filter");
        buttonfilter2.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonfilter2ActionPerformed(evt);
            }
        });

        DropDownFilter.setModel(new javax.swing.DefaultComboBoxModel<>(new
String[] { "Name", "Term", "Exam+Score", "Report" }));
        DropDownFilter.addItemListener(new java.awt.event.ItemListener() {
            public void itemStateChanged(java.awt.event.ItemEvent evt) {
                DropDownFilterItemStateChanged(evt);
            }
        });
        DropDownFilter.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                DropDownFilterActionPerformed(evt);
            }
        });

        jLabel15.setIcon(new
javax.swing.ImageIcon("C:/Users/Abhay/OneDrive/Documents/NetBeansProjects/C
SIAF/src/main/java/image/vocallogo.png")
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
```

```
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING, false)
                            .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGroup(layout.createSequentialGroup()
                                .addGap(16, 16, 16)
                                .addComponent(bback,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                        .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                            .addComponent(buttonfilter2,
javax.swing.GroupLayout.PREFERRED_SIZE, 159,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(DropDownFilter,
javax.swing.GroupLayout.PREFERRED_SIZE, 157,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(btnNewTerm))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel4)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE))))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jLabel4))
                    .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 117,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 39,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(btnNewTerm,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(buttonfilter2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(DropDownFilter,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 183,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(51, 51, 51)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                    .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(bback,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>
private JFrame frame;
    private void buttonexitActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        frame = new JFrame("Exit");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
exit", "MYSQL Connector",
            JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)
    {
        System.exit(0);
        }
    }

    private void bbackActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        adminhomepage ahp = new adminhomepage();
        ahp.setName(adminname);
               ahp.setVisible(true);
               this.setVisible(false);
    }

    private void btnNewTermActionPerformed(java.awt.event.ActionEvent evt)
{
          JFrame frame = new JFrame("Save");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
start a new term", "MYSQL Connector",
            JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)
    {
```

```
        JFrame newframe = new JFrame();
    Object result = JOptionPane.showInputDialog(newframe, "Term Name:");
    System.out.println(result);
        try {
            // TODO add your handling code here:
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            pst = sqlConn.prepareStatement("DELETE FROM `term`");
            pst.execute();
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            pst = sqlConn.prepareStatement("DELETE FROM
`currentlyenrolledstudentsadmin`");
            pst.execute();

            pst = sqlConn.prepareStatement("INSERT INTO `term`(`term`)
VALUES (?)");
            pst.setInt(1, Integer.parseInt(String.valueOf(result)));



            pst.executeUpdate();
        } catch (SQLException ex) {

Logger.getLogger(paststudentrecords.class.getName()).log(Level.SEVERE,
null, ex);
        }}

    }

    private void psrtabMouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
        DefaultTableModel RecordTable =
(DefaultTableModel)psrtab.getModel();

        int SelectedRows = psrtab.getSelectedRow();
                String sid = RecordTable.getValueAt(SelectedRows,
0).toString();
                String sname = RecordTable.getValueAt(SelectedRows,
1).toString();
        reportpage rp = new reportpage();
        rp.setStudent(sid, sname);
        rp.setPreviousPage(this);
```

```
                rp.setVisible(true);
                this.setVisible(false);
    }

    private void DropDownFilterItemStateChanged(java.awt.event.ItemEvent
evt) {
        // TODO add your handling code here:
    }

    private void DropDownFilterActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
    }

    private void buttonfilter2ActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
psrtab.setRowSorter(null);
TableRowSorter<TableModel> sorter = new
TableRowSorter<TableModel>(psrtab.getModel());
psrtab.setRowSorter(sorter);

ArrayList<RowSorter.SortKey> sortKeys = new ArrayList<>(25);
sortKeys.add(new RowSorter.SortKey(DropDownFilter.getSelectedIndex()+1,
SortOrder.ASCENDING));
sortKeys.add(new RowSorter.SortKey(0, SortOrder.ASCENDING));
sorter.setSortKeys(sortKeys);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new paststudentrecords().setVisible(true);
            }
        });
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel
setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with
the default look and feel.
```

```
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(paststudentrecords.class.getName()).log(
java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(paststudentrecords.class.getName()).log(
java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(paststudentrecords.class.getName()).log(
java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(paststudentrecords.class.getName()).log(
java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new paststudentrecords().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JComboBox<String> DropDownFilter;
    private javax.swing.JButton bback;
    private javax.swing.JButton btnNewTerm;
```

```
    private javax.swing.JButton buttonexit;
    private javax.swing.JButton buttonfilter2;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTable psrtab;
    // End of variables declaration
}
```

Current Teachers Page:

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to
edit this template
 */
package com.mycompany.csiaf;

import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/**
 *
 */
public class currentteachers extends javax.swing.JFrame {
     private static final String username = "root";
    private static final String password = "";
    private static final String dataConn =
```

```
"jdbc:mysql://localhost:3306/multiuserlogin";
    String adminname;
    public void setName(String name){
        adminname = name;

    }

   Connection sqlConn = null;
   PreparedStatement pst = null;
   ResultSet rs = null;
   int q, i, id, deleteItem;
   private void initSelfListeners() {
    WindowListener taskStarterWindowListener = new WindowListener() {
        @Override
        public void windowOpened(WindowEvent e) {
            fillTable();
            System.out.println("Performing task..."); //Perform task here.
In this case, we are simulating a startup (only once) time-consuming task
that would use a worker.
        }

        @Override
        public void windowClosing(WindowEvent e) {
            //Do nothing...Or something...You decide!
        }

        @Override
        public void windowClosed(WindowEvent e) {
            //Do nothing...Or drink coffee...NVM; always drink coffee!
        }

        @Override
        public void windowIconified(WindowEvent e) {
            //Do nothing...Or do EVERYTHING!
        }

        @Override
        public void windowDeiconified(WindowEvent e) {
            //Do nothing...Or break the law...
        }

        @Override
        public void windowActivated(WindowEvent e) {
```

```
            //Do nothing...Procrastinate like me!
        }

        @Override
        public void windowDeactivated(WindowEvent e) {
            //Do nothing...And please don't notice I have way too much free
time today...
        }
    };

    //Here is where the magic happens! We make (a listener within) the
frame start listening to the frame's own events!
    this.addWindowListener(taskStarterWindowListener);
}

    /**
     * Creates new form currentteachers
     */
    public currentteachers() {
        initComponents();
        initSelfListeners();
    }
    public Connection getConnection() {
        try{
            Class.forName("com.mysql.jdbc.Driver");

        }catch(ClassNotFoundException ex) {
            System.out.println(ex.getMessage());
        }
        Connection con = null;
        try{
            con =
DriverManager.getConnection("jdbc:mysql://localhost/multiuserlogin",
"root", "");
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }
        return con;

    }
```

```
    public void fillTable() {
        Connection con = getConnection();
        Statement ps;
        ResultSet rs;
        DefaultTableModel model = (DefaultTableModel) ctdtab.getModel();
        try {
            ps = con.createStatement();
            rs = ps.executeQuery("SELECT * FROM `multiuserlogin`");
            /*need to add the where UserType is user*/
            while(rs.next()) {
                Object[] row = new Object[ctdtab.getColumnCount()];
                row[0] = rs.getString("IDTEACHER");
                row[1] = rs.getString("TeacherName");
                row[2] = rs.getString("EmailAddress");
                row[3] = rs.getString("Passcode");
                row[4] = rs.getString("UserType");


                model.addRow(row);


            }

        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }

    }

    /**
     * This method is called from within the constructor to initialize the
form.
     * WARNING: Do NOT modify this code. The content of this method is
always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jLabel5 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
```

```
        jScrollPane1 = new javax.swing.JScrollPane();
        ctdtab = new javax.swing.JTable();
        buttonaddCTP = new javax.swing.JButton();
        buttondeleteCTP = new javax.swing.JButton();
        buttonexit = new javax.swing.JButton();
        buttonback4 = new javax.swing.JButton();
        buttonupdateCTP = new javax.swing.JButton();
        jLabel15 = new javax.swing.JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jLabel5.setFont(new java.awt.Font("Arial", 0, 80)); // NOI18N

        jLabel4.setFont(new java.awt.Font("Segoe UI", 0, 50)); // NOI18N
        jLabel4.setText("Current Teacher Database");

        ctdtab.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "TeacherID", "Name", "Email", "Passcode", "Usertype"
            }
        ) {
            Class[] types = new Class [] {
                java.lang.String.class, java.lang.String.class,
java.lang.String.class, java.lang.String.class, java.lang.String.class
            };
            boolean[] canEdit = new boolean [] {
                false, true, true, true, false
            };

            public Class getColumnClass(int columnIndex) {
                return types [columnIndex];
            }

            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
        });
        ctdtab.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
```

```
                ctdtabMouseClicked(evt);
            }
        });
        jScrollPane1.setViewportView(ctdtab);
        if (ctdtab.getColumnModel().getColumnCount() > 0) {
            ctdtab.getColumnModel().getColumn(0).setMinWidth(0);
            ctdtab.getColumnModel().getColumn(0).setMaxWidth(0);
            ctdtab.getColumnModel().getColumn(4).setMinWidth(0);
            ctdtab.getColumnModel().getColumn(4).setMaxWidth(0);
        }

        buttonaddCTP.setBackground(new java.awt.Color(0, 0, 0));
        buttonaddCTP.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        buttonaddCTP.setForeground(new java.awt.Color(255, 255, 255));
        buttonaddCTP.setText("Add");
        buttonaddCTP.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonaddCTPActionPerformed(evt);
            }
        });

        buttondeleteCTP.setBackground(new java.awt.Color(0, 0, 0));
        buttondeleteCTP.setFont(new java.awt.Font("Arial", 0, 18)); //
NOI18N
        buttondeleteCTP.setForeground(new java.awt.Color(255, 255, 255));
        buttondeleteCTP.setText("Delete");
        buttondeleteCTP.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttondeleteCTPActionPerformed(evt);
            }
        });

        buttonexit.setBackground(new java.awt.Color(0, 0, 0));
        buttonexit.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        buttonexit.setForeground(new java.awt.Color(255, 255, 255));
        buttonexit.setText("Exit");
        buttonexit.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonexitActionPerformed(evt);
            }
        });
```

```
        buttonback4.setBackground(new java.awt.Color(0, 0, 0));
        buttonback4.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N
        buttonback4.setForeground(new java.awt.Color(255, 255, 255));
        buttonback4.setText("Back");
        buttonback4.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonback4ActionPerformed(evt);
            }
        });

        buttonupdateCTP.setBackground(new java.awt.Color(0, 0, 0));
        buttonupdateCTP.setFont(new java.awt.Font("Arial", 0, 18)); //
NOI18N
        buttonupdateCTP.setForeground(new java.awt.Color(255, 255, 255));
        buttonupdateCTP.setText("Update");
        buttonupdateCTP.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                buttonupdateCTPActionPerformed(evt);
            }
        });

        jLabel15.setIcon(new
javax.swing.ImageIcon("C:/Users/Abhay/OneDrive/Documents/NetBeansProjects/C
SIAF/src/main/java/image/vocallogo.png")
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(99, 99, 99)
                        .addComponent(buttonaddCTP,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                        .addGap(18, 18, 18)
                        .addComponent(buttondeleteCTP,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(buttonupdateCTP,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(25, 25, 25)
                        .addComponent(buttonback4,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(384, 384, 384)
                        .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap(215, Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()
                .addGap(42, 42, 42)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 575,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel4))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 128,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(153, 153, 153)
                    .addComponent(jLabel5)
                    .addContainerGap(671, Short.MAX_VALUE)))
        );
        layout.setVerticalGroup(
```

```
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(65, 65, 65)
                        .addComponent(jLabel4))
                    .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 124,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(147, 147, 147)
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 261,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                            .addComponent(buttonaddCTP,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(buttondeleteCTP,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(buttonupdateCTP,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(buttonback4,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                        .addGap(0, 0, Short.MAX_VALUE)
                        .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                        .addContainerGap())))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(212, 212, 212)
                    .addComponent(jLabel5)
                    .addContainerGap(462, Short.MAX_VALUE)))
        );

        pack();
    }// </editor-fold>

    private void buttonaddCTPActionPerformed(java.awt.event.ActionEvent
evt) {
createteacheraccount cta = new createteacheraccount();
        cta.setName(adminname);
               cta.setVisible(true);
               this.setVisible(false);

    }

    private void buttondeleteCTPActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        try {
            Class.forName("com.mysql.jdbc.Driver");
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            pst = sqlConn.prepareStatement("DELETE FROM `multiuserlogin`
WHERE IDTEACHER = ?");
            DefaultTableModel model = (DefaultTableModel)ctdtab.getModel();
            String id =
model.getValueAt(ctdtab.getSelectedRow(),0).toString();

            pst.setString(1,id);
            pst.executeUpdate();
            JOptionPane.showMessageDialog(this, "Record Deleted");
        }
        catch(ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
        } catch (SQLException ex){

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
        }

((DefaultTableModel)ctdtab.getModel()).removeRow(ctdtab.getSelectedRow());
    }
private JFrame frame;
    private void buttonexitActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        frame = new JFrame("Exit");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
exit", "MYSQL Connector",
                JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)
        {
           System.exit(0);
        }
    }

    private void buttonback4ActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        adminhomepage ahp = new adminhomepage();
        ahp.setName(adminname);
               ahp.setVisible(true);
               this.setVisible(false);
    }

    private void buttonupdateCTPActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        Connection con = getConnection();
        Statement st;
        DefaultTableModel model = (DefaultTableModel)ctdtab.getModel();
        try {
            st = con.createStatement();
            for(int i = 0; i<model.getRowCount();i++) {
                String id = model.getValueAt(i,0).toString();
                String fn = model.getValueAt(i, 1).toString();
                String tn = model.getValueAt(i, 2).toString();
                String en = model.getValueAt(i, 3).toString();
```

```
                String updateQuery = "UPDATE `multiuserlogin` SET
`TeacherName`='"+fn+"',`EmailAddress`='"+tn+"',`Passcode`='"+en+"',`UserTyp
e`='User' WHERE `IDTEACHER` = '"+id+"'";
                st.addBatch(updateQuery);


            }
            int[] updatedRow =  st.executeBatch();
System.out.println(updatedRow.length);
            JOptionPane.showMessageDialog(this, "Record Updated");









        } catch (SQLException ex) {

Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(Level.
SEVERE, null, ex);
        }
    }

    private void ctdtabMouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new currentteachers().setVisible(true);
            }
        });
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel
setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with
the default look and feel.
         * For details see
```

```
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(currentteachers.class.getName()).log(jav
a.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(currentteachers.class.getName()).log(jav
a.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(currentteachers.class.getName()).log(jav
a.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(currentteachers.class.getName()).log(jav
a.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new currentteachers().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton buttonaddCTP;
    private javax.swing.JButton buttonback4;
    private javax.swing.JButton buttondeleteCTP;
    private javax.swing.JButton buttonexit;
```

```
    private javax.swing.JButton buttonupdateCTP;
    private javax.swing.JTable ctdtab;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JScrollPane jScrollPane1;
    // End of variables declaration
}
```

Currently Enrolled Students Admin Page:

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to
edit this template
 */
package com.mycompany.csiaf;

import com.mysql.cj.xdevapi.Result;
import java.awt.Color;
import java.awt.Component;
import java.awt.List;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.Vector;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
```

```
import java.util.UUID;
import javax.swing.JTable;
import javax.swing.RowSorter;
import javax.swing.SortOrder;
import javax.swing.table.DefaultTableCellRenderer;
import javax.swing.table.TableCellRenderer;
import javax.swing.table.TableModel;
import javax.swing.table.TableRowSorter;


/**
 *
 */

public class currentlyenrolledstudentsadmin extends javax.swing.JFrame {
    private static final String username = "root";
    private static final String password = "";
    private static final String dataConn =
"jdbc:mysql://localhost:3306/currentlyenrolledstudentsadmin";
    private static final String dataConn2 =
"jdbc:mysql://localhost:3306/individualstudentreport";
    private static final String dataConn3 =
"jdbc:mysql://localhost:3306/paststudentrecord";

    String adminname;
    public void setName(String name){
        adminname = name;

    }

   Connection sqlConn = null;
   PreparedStatement pst = null;
   ResultSet rs = null;
   int q, i, id, deleteItem;


    /**
     * Creates new form currentlyenrolledstudentsadmin
     */
   private void initSelfListeners() {
    WindowListener taskStarterWindowListener = new WindowListener() {
        @Override
        public void windowOpened(WindowEvent e) {
```

```
            fillTable();
            System.out.println("Performing task...");
        }

        @Override
        public void windowClosing(WindowEvent e) {
            //Do nothing...Or something...You decide!
        }

        @Override
        public void windowClosed(WindowEvent e) {
            //Do nothing...Or drink coffee...NVM; always drink coffee!
        }

        @Override
        public void windowIconified(WindowEvent e) {
            //Do nothing...Or do EVERYTHING!
        }

        @Override
        public void windowDeiconified(WindowEvent e) {
            //Do nothing...Or break the law...
        }

        @Override
        public void windowActivated(WindowEvent e) {


            //Do nothing...Procrastinate like me!
        }

        @Override
        public void windowDeactivated(WindowEvent e) {
            //Do nothing...And please don't notice I have way too much free
time today...
        }
    };

    //Here is where the magic happens! We make (a listener within) the
frame start listening to the frame's own events!
    this.addWindowListener(taskStarterWindowListener);
}
    public currentlyenrolledstudentsadmin() {
```

```
        initComponents();
        cat.setDefaultRenderer(Object.class, new TableCellRenderer(){
    private DefaultTableCellRenderer DEFAULT_RENDERER =  new
DefaultTableCellRenderer();

            @Override
            public Component getTableCellRendererComponent(JTable table,
Object value, boolean isSelected, boolean hasFocus, int row, int column) {
                Component c =
DEFAULT_RENDERER.getTableCellRendererComponent(table, value, isSelected,
hasFocus, row, column);
        if(cat.getModel().getValueAt(row, 7).toString().equals("true")){


        c.setBackground(Color.RED);


}
        System.out.println(value);
       //Add below code here
                return c;
            }

        });
        initSelfListeners();


    }


    /**
     * This method is called from within the constructor to initialize the
form.
     * WARNING: Do NOT modify this code. The content of this method is
always
     * regenerated by the Form Editor.
     */
    //function to get connection
    public Connection getConnection() {
        try{
            Class.forName("com.mysql.jdbc.Driver");
```

```
        }catch(ClassNotFoundException ex) {
            System.out.println(ex.getMessage());
        }
        Connection con = null;
        try{
            con =
DriverManager.getConnection("jdbc:mysql://localhost/currentlyenrolledstuden
tsadmin", "root", "");
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }
        return con;

    }
    public Connection getConnection2() {
        try{
            Class.forName("com.mysql.jdbc.Driver");

        }catch(ClassNotFoundException ex) {
            System.out.println(ex.getMessage());
        }
        Connection con = null;
        try{
            con =
DriverManager.getConnection("jdbc:mysql://localhost/individualstudentreport
", "root", "");
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }
        return con;

    }
    public Connection getConnection3() {
        try{
            Class.forName("com.mysql.jdbc.Driver");

        }catch(ClassNotFoundException ex) {
            System.out.println(ex.getMessage());
        }
        Connection con = null;
        try{
```

```
            con =
DriverManager.getConnection("jdbc:mysql://localhost/paststudentrecord",
"root", "");
        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }
        return con;

    }


    public void fillTable() {
        DefaultTableModel StudentTable = (DefaultTableModel)cat.getModel();
        StudentTable.setRowCount(0);
        Connection con = getConnection();
        Statement ps;
        ResultSet rs;
        DefaultTableModel model = (DefaultTableModel) cat.getModel();
        try {
            ps = con.createStatement();
            rs = ps.executeQuery("SELECT * FROM
`currentlyenrolledstudentsadmin`");

            while(rs.next()) {
                Object[] row = new Object[cat.getColumnCount()];
                row[0] = rs.getString("ID");
                row[1] = rs.getString("studentname");
                row[2] = rs.getString("teachername");
                row[3] = rs.getString("examname");
                row[4] = rs.getString("piecenames");
                row[5] = rs.getString("reportstatus");
                row[6] = rs.getString("prephours");
                row[7] = rs.getBoolean("flag");
                row[8] = rs.getString("date");
                row[9] = rs.getString("notes");

                model.addRow(row);


            }

        } catch(SQLException ex) {
```

```
            System.out.println(ex.getMessage());

        }


    }

    public void upDateDB(){
        try{
        Class.forName("com.mysql.jdbc.Driver");
        sqlConn = DriverManager.getConnection(dataConn, username,
password);
        pst = sqlConn.prepareStatement("SELECT * FROM
`currentlyenrolledstudentsadmin`");

        rs = pst.executeQuery();
        ResultSetMetaData stData = rs.getMetaData();
        q = stData.getColumnCount();

        DefaultTableModel StudentTable = (DefaultTableModel)cat.getModel();
        StudentTable.setRowCount(0);

        while(rs.next()) {
            Vector columnData = new Vector();
            for(i = 1;i<=q;i++) {
                columnData.add(rs.getString("ID"));
                columnData.add(rs.getString("studentname"));
                columnData.add(rs.getString("teachername"));
                columnData.add(rs.getString("examname"));
                columnData.add(rs.getString("piecenames"));
                columnData.add(rs.getString("reportstatus"));
                columnData.add(rs.getString("prephours"));


            }
            StudentTable.addRow(columnData);

        }
        }
        catch(Exception ex) {
            JOptionPane.showMessageDialog(null, ex);

        }
```

```
    }
    public void upDateDBview(){
        try{
        Class.forName("com.mysql.jdbc.Driver");
        sqlConn = DriverManager.getConnection(dataConn, username,
password);
        pst = sqlConn.prepareStatement("SELECT * FROM
`currentlyenrolledstudentsadmin`");

        rs = pst.executeQuery();
        ResultSetMetaData stData = rs.getMetaData();
        q = stData.getColumnCount();

        DefaultTableModel StudentTable = (DefaultTableModel)cat.getModel();
        StudentTable.setRowCount(0);

        while(rs.next()) {
            Vector columnData = new Vector();
            for(i = 1;i<=q;i++) {
                                columnData.add(rs.getString("ID"));

                columnData.add(rs.getString("studentname"));
                columnData.add(rs.getString("teachername"));
                columnData.add(rs.getString("examname"));
                columnData.add(rs.getString("piecenames"));
                columnData.add(rs.getString("reportstatus"));
                columnData.add(rs.getString("prephours"));


            }


        }
        }
        catch(Exception ex) {
            JOptionPane.showMessageDialog(null, ex);

        }

    }
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
```

```
    private void initComponents() {

        jLabel7 = new javax.swing.JLabel();
        buttondeleteCTS = new javax.swing.JButton();
        buttonaddCTS = new javax.swing.JButton();
        buttonupdateCTS = new javax.swing.JButton();
        buttonexit = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        cat = new javax.swing.JTable()
        {
            public Component prepareRenderer(
                TableCellRenderer renderer, int row, int column)
            {
                try {
                    Component c = super.prepareRenderer(renderer, row,
column);

                    //  add custom rendering here
                    if (!isRowSelected(row))
                    {

                        c.setBackground(getBackground());
                        int modelRow = convertRowIndexToModel(row);
                        String type =
(String)getModel().getValueAt(modelRow, 7);
                        if ("1".equals(type)) c.setBackground(Color.RED);
                        System.out.println(type);

                    }
                    return c;
                } catch(NullPointerException exc) {}
                catch( ClassCastException exc) {}
                return super.prepareRenderer(renderer, row, column);
            }};
            jLabel1 = new javax.swing.JLabel();
            tfsname = new javax.swing.JTextField();
            jLabel2 = new javax.swing.JLabel();
            tftname = new javax.swing.JTextField();
            jLabel3 = new javax.swing.JLabel();
            tfexam = new javax.swing.JTextField();
            jLabel4 = new javax.swing.JLabel();
            tfpieces = new javax.swing.JTextField();
            jLabel6 = new javax.swing.JLabel();
```

```
            tfpreph = new javax.swing.JTextField();
            jComboBoxfilter = new javax.swing.JComboBox<>();
            btnfilter = new javax.swing.JButton();
            buttonback = new javax.swing.JButton();
            jLabel15 = new javax.swing.JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

            jLabel7.setFont(new java.awt.Font("Arial", 0, 40)); // NOI18N
            jLabel7.setText("Your Current Term Students");

            buttondeleteCTS.setBackground(new java.awt.Color(0, 0, 0));
            buttondeleteCTS.setFont(new java.awt.Font("Arial", 0, 18)); //
NOI18N
            buttondeleteCTS.setForeground(new java.awt.Color(255, 255,
255));
            buttondeleteCTS.setText("Delete");
            buttondeleteCTS.addActionListener(new
java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt)
{
                    buttondeleteCTSActionPerformed(evt);
                }
            });

            buttonaddCTS.setBackground(new java.awt.Color(0, 0, 0));
            buttonaddCTS.setFont(new java.awt.Font("Arial", 0, 18)); //
NOI18N
            buttonaddCTS.setForeground(new java.awt.Color(255, 255, 255));
            buttonaddCTS.setText("Add");
            buttonaddCTS.addActionListener(new
java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt)
{
                    buttonaddCTSActionPerformed(evt);
                }
            });

            buttonupdateCTS.setBackground(new java.awt.Color(0, 0, 0));
            buttonupdateCTS.setFont(new java.awt.Font("Arial", 0, 18)); //
NOI18N
            buttonupdateCTS.setForeground(new java.awt.Color(255, 255,
```

```
255));
            buttonupdateCTS.setText("Update");
            buttonupdateCTS.addActionListener(new
java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt)
{
                    buttonupdateCTSActionPerformed(evt);
                }
            });

            buttonexit.setBackground(new java.awt.Color(0, 0, 0));
            buttonexit.setFont(new java.awt.Font("Arial", 0, 18)); //
NOI18N
            buttonexit.setForeground(new java.awt.Color(255, 255, 255));
            buttonexit.setText("Exit");
            buttonexit.addActionListener(new
java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt)
{
                    buttonexitActionPerformed(evt);
                }
            });

            cat.setModel(new javax.swing.table.DefaultTableModel(
                new Object [][] {

                },
                new String [] {
                    "ID", "Student Name", "Teacher Name", "Exam", "Pieces",
"Report Status", "Prep Hours", "flag", "date", "notes"
                }
            ) {
                Class[] types = new Class [] {
                    java.lang.String.class, java.lang.String.class,
java.lang.String.class, java.lang.String.class, java.lang.String.class,
java.lang.String.class, java.lang.String.class, java.lang.String.class,
java.lang.Object.class, java.lang.Object.class
                };
                boolean[] canEdit = new boolean [] {
                    true, true, true, true, true, false, true, false, true,
true
                };
```

```
                public Class getColumnClass(int columnIndex) {
                    return types [columnIndex];
                }

                public boolean isCellEditable(int rowIndex, int
columnIndex) {
                    return canEdit [columnIndex];
                }
            });
            cat.addMouseListener(new java.awt.event.MouseAdapter() {
                public void mouseClicked(java.awt.event.MouseEvent evt) {
                    catMouseClicked(evt);
                }
            });
            jScrollPane1.setViewportView(cat);
            if (cat.getColumnModel().getColumnCount() > 0) {
                cat.getColumnModel().getColumn(0).setMinWidth(0);
                cat.getColumnModel().getColumn(0).setPreferredWidth(0);
                cat.getColumnModel().getColumn(0).setMaxWidth(0);
                cat.getColumnModel().getColumn(7).setMinWidth(0);
                cat.getColumnModel().getColumn(7).setPreferredWidth(0);
                cat.getColumnModel().getColumn(7).setMaxWidth(0);
                cat.getColumnModel().getColumn(8).setMinWidth(0);
                cat.getColumnModel().getColumn(8).setPreferredWidth(0);
                cat.getColumnModel().getColumn(8).setMaxWidth(0);
                cat.getColumnModel().getColumn(9).setMinWidth(0);
                cat.getColumnModel().getColumn(9).setPreferredWidth(0);
                cat.getColumnModel().getColumn(9).setMaxWidth(0);
            }

            jLabel1.setText("Student Name");

            tfsname.addActionListener(new java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt)
{
                    tfsnameActionPerformed(evt);
                }
            });

            jLabel2.setText("Teacher Name");

            jLabel3.setText("Exam");
```

```
            jLabel4.setText("Pieces");

            jLabel6.setText("Prep Hours");

            jComboBoxfilter.setModel(new
javax.swing.DefaultComboBoxModel<>(new String[] { "Student Name", "Teacher
Name", "Exam", "Pieces", "Report Status", "Prep Hours" }));
            jComboBoxfilter.addItemListener(new
java.awt.event.ItemListener() {
                public void itemStateChanged(java.awt.event.ItemEvent evt)
{
                    jComboBoxfilterItemStateChanged(evt);
                }
            });
            jComboBoxfilter.addActionListener(new
java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt)
{
                    jComboBoxfilterActionPerformed(evt);
                }
            });

            btnfilter.setText("Filter");
            btnfilter.addActionListener(new java.awt.event.ActionListener()
{
                public void actionPerformed(java.awt.event.ActionEvent evt)
{
                    btnfilterActionPerformed(evt);
                }
            });

            buttonback.setBackground(new java.awt.Color(0, 0, 0));
            buttonback.setFont(new java.awt.Font("Arial", 0, 18)); //
NOI18N
            buttonback.setForeground(new java.awt.Color(255, 255, 255));
            buttonback.setText("Back");
            buttonback.addActionListener(new
java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt)
{
                    buttonbackActionPerformed(evt);
                }
            });
```

```
            jLabel15.setIcon(new
javax.swing.ImageIcon("C:/Users/Abhay/OneDrive/Documents/NetBeansProjects/C
SIAF/src/main/java/image/vocallogo.png")
            );

            javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
            getContentPane().setLayout(layout);
            layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                    .addGap(26, 26, 26)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING, false)
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(jLabel4)
                                .addGap(62, 62, 62)
                                .addComponent(tfpieces,
javax.swing.GroupLayout.PREFERRED_SIZE, 71,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                                    .addComponent(jLabel2)
                                    .addComponent(jLabel3))
                                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING)
                                    .addComponent(tftname,
javax.swing.GroupLayout.PREFERRED_SIZE, 71,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(tfexam,
javax.swing.GroupLayout.PREFERRED_SIZE, 71,
javax.swing.GroupLayout.PREFERRED_SIZE)))
```

```
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(jLabel6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(tfpreph,
javax.swing.GroupLayout.PREFERRED_SIZE, 71,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 582,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(jLabel1)
                            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                                .addGroup(layout.createSequentialGroup()
                                    .addGap(6, 6, 6)
                                    .addComponent(tfsname,
javax.swing.GroupLayout.PREFERRED_SIZE, 71,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addComponent(jLabel7,
javax.swing.GroupLayout.PREFERRED_SIZE, 516,
javax.swing.GroupLayout.PREFERRED_SIZE))))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 146,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                                .addComponent(buttonaddCTS,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(buttonexit,
javax.swing.GroupLayout.Alignment.TRAILING,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(buttondeleteCTS,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(buttonupdateCTS,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(jComboBoxfilter,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(btnfilter,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(buttonback,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGap(23, 23, 23))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                            .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 133,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addContainerGap())))
            );
            layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                        .addComponent(jLabel7,
javax.swing.GroupLayout.PREFERRED_SIZE, 103,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 114,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGap(62, 62, 62)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                        .addComponent(btnfilter,
javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                            .addComponent(jLabel1)
                            .addComponent(tfsname,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                        .addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                                .addComponent(jLabel2)
                                .addComponent(tftname,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                                .addComponent(jLabel3)
                                .addComponent(tfexam,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGap(11, 11, 11)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
```

```
LINE)
                                .addComponent(jLabel4)
                                .addComponent(tfpieces,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                            .addGap(6, 6, 6)
                            .addComponent(jComboBoxfilter,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(buttonupdateCTS,
javax.swing.GroupLayout.PREFERRED_SIZE, 37,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASE
LINE)
                        .addComponent(buttonaddCTS,
javax.swing.GroupLayout.PREFERRED_SIZE, 38,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(tfpreph,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(jLabel6))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(buttondeleteCTS,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE, 37,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGap(18, 18, 18)
                            .addComponent(buttonexit,
javax.swing.GroupLayout.PREFERRED_SIZE, 37,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGap(18, 18, 18)
                            .addComponent(buttonback,
javax.swing.GroupLayout.PREFERRED_SIZE, 37,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addGap(39, 39, 39))
            );

            pack();
        }// </editor-fold>

    private void buttondeleteCTSActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
           JFrame frame = new JFrame("Save");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
delete", "MYSQL Connector",
            JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)
    {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            pst = sqlConn.prepareStatement("DELETE FROM
`currentlyenrolledstudentsadmin` WHERE ID = ?");
                    DefaultTableModel model =
(DefaultTableModel)cat.getModel();

                        String id =
model.getValueAt(cat.getSelectedRow(),0).toString();
                        pst.setString(1,id);


            pst.executeUpdate();
            JOptionPane.showMessageDialog(this, "Record Deleted");
            //upDateDB();
```

```
        }
        catch(ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (SQLException ex){

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);




        }

((DefaultTableModel)cat.getModel()).removeRow(cat.getSelectedRow());}

    }

    private void buttonaddCTSActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
           JFrame frame = new JFrame("Save");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
add", "MYSQL Connector",
            JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)
    {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            pst = sqlConn.prepareStatement("INSERT INTO
`currentlyenrolledstudentsadmin`(`ID`,`studentname`, `teachername`,
`examname`, `piecenames`, `reportstatus`, "
                    + "`prephours`, flag) VALUES (?,?,?,?,?,?,?,0)");
          String studentid = UUID.randomUUID().toString().replaceAll("_",
"");
            System.out.println(studentid);
            pst.setString(1, studentid);

            pst.setString(2, tfsname.getText());
```

```
            pst.setString(3, tftname.getText());
            pst.setString(4, tfexam.getText());
            pst.setString(5, tfpieces.getText());
            pst.setString(6, "Not Done");
            pst.setString(7, tfpreph.getText());

            pst.executeUpdate();
            JOptionPane.showMessageDialog(this, "Record Added");
            upDateDB();

            sqlConn = DriverManager.getConnection(dataConn2, username,
password);
            pst = sqlConn.prepareStatement("INSERT INTO
`individualstudentreport`(`IDSTUDENT`, `Studentname`, `Date`, `Skill1`,
`Skill2`, `Skill3`, "
                    + "`Skill4`, `Notes`) VALUES
(?,?,null,null,null,null,null,null)");
           System.out.println(studentid);
            pst.setString(1, studentid);

            pst.setString(2, tfsname.getText());
                        pst.executeUpdate();

        Connection con = getConnection();
        Statement ps;
        ResultSet rs;
        try {
            ps = con.createStatement();
            rs = ps.executeQuery("SELECT term FROM `term`");
            rs.next();
            int term = rs.getInt("term");

            sqlConn = DriverManager.getConnection(dataConn3, username,
password);
            pst = sqlConn.prepareStatement("INSERT INTO
`paststudentrecords`(`sID`, `Sname`, `Term`, `ExamScore`, `Report`) VALUES
(?,?,"+String.valueOf(term)+",?,?)");
           System.out.println(studentid);
            pst.setString(1, studentid);

            pst.setString(2, tfsname.getText());
            pst.setString(3, tfexam.getText());
            pst.setString(4, "Not Done");
```

```
                        pst.executeUpdate();



        } catch(SQLException ex) {
            System.out.println(ex.getMessage());

        }


        }
        catch(ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (SQLException ex){

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);




        }}






    }

    private void buttonupdateCTSActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:

        Connection con = getConnection();
        Connection con2 = getConnection2();
        Connection con3 = getConnection3();
        Statement st;
        Statement st2;
        Statement st3;
        DefaultTableModel model = (DefaultTableModel)cat.getModel();
           JFrame frame = new JFrame("Save");
```

```
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
update", "MYSQL Connector",
            JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)
    {
        try {
            st = con.createStatement();
            st2 = con2.createStatement();
            st3 = con3.createStatement();
            for(int i = 0; i<model.getRowCount();i++) {
                String id = model.getValueAt(i,0).toString();
                String fn = model.getValueAt(i, 1).toString();
                String tn = model.getValueAt(i, 2).toString();
                String en = model.getValueAt(i, 3).toString();
                String pn = model.getValueAt(i, 4).toString();
                String rst = model.getValueAt(i, 5).toString();
                String ph = model.getValueAt(i, 6).toString();
                String updateQuery = "UPDATE
`currentlyenrolledstudentsadmin` SET
`studentname`='"+fn+"',`teachername`='"+tn+"',`examname`='"+en+"',"
                        +
"`piecenames`='"+pn+"',`reportstatus`='"+rst+"',`prephours`='"+ph+"' WHERE
`ID` = '"+id+"'";
                st.addBatch(updateQuery);
                String updateQuery2 = "UPDATE `individualstudentreport` SET
`Studentname`='"+fn+"' WHERE `IDSTUDENT` = '"+id+"'";
                st2.addBatch(updateQuery2);
                String updateQuery3 = "UPDATE `paststudentrecords` SET
`Sname`='"+fn+"',`ExamScore`='"+en+"',`Report`='"+rst+"' WHERE `sID` =
'"+id+"'";
                st3.addBatch(updateQuery3);


            }
            int[] updatedRow =  st.executeBatch();
            int[] updatedRow2 =  st2.executeBatch();
            int[] updatedRow3 =  st3.executeBatch();
System.out.println(updatedRow.length);


            /*try{
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            String value1 = tfsname.getText();
```

```
            String value2 = tftname.getText();
            String value3 = tfexam.getText();
            String value4 = tfpieces.getText();
            String value5 = tfreportstat.getText();
            String value6 = tfpreph.getText();
            String sql = "UPDATE `currentlyenrolledstudentsadmin` SET
`studentname`='"+value1+"',`teachername`='"+value2+"',`examname`='"+value3+
"',`piecenames`='"+value4+"',`reportstatus`='"+value5+"',`prephours`='"+val
ue6+"' WHERE `studentname`='"+value1+"'";
            pst = sqlConn.prepareStatement(sql);
            pst.execute();
            JOptionPane.showMessageDialog(null, "Updated");

            }catch(Exception e) {
            JOptionPane.showMessageDialog(null, e);

            }*/
            /*try {
            Class.forName("com.mysql.jdbc.Driver");
            sqlConn = DriverManager.getConnection(dataConn, username,
password);
            pst = sqlConn.prepareStatement("UPDATE
`currentlyenrolledstudentsadmin` SET `studentname`=
?,`teachername`=?,`examname`=?,`piecenames`=?,`reportstatus`=?,`prephours`=
? WHERE `studentname`= ?");
            pst.setString(1, tfsname.getText());
            pst.setString(2, tftname.getText());
            pst.setString(3, tfexam.getText());
            pst.setString(4, tfpieces.getText());
            pst.setString(5, tfreportstat.getText());
            pst.setString(6, tfpreph.getText());

            pst.executeUpdate();
            JOptionPane.showMessageDialog(this, "Record Updated");
            upDateDB();



            }
            catch(ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
            } catch (SQLException ex){

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);




            }*/
        } catch (SQLException ex) {

Logger.getLogger(currentlyenrolledstudentsadmin.class.getName()).log(Level.
SEVERE, null, ex);
        }}




    }
private JFrame frame;
    private void buttonexitActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        frame = new JFrame("Exit");
        if(JOptionPane.showConfirmDialog(frame, "Confirm if you want to
exit", "MYSQL Connector",
                JOptionPane.YES_NO_OPTION) == JOptionPane.YES_OPTION)
        {
           System.exit(0);
        }
    }

    private void catMouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
        if(cat.getSelectedColumn() == 5){
            DefaultTableModel RecordTable =
(DefaultTableModel)cat.getModel();

        int SelectedRows = cat.getSelectedRow();
                String sid = RecordTable.getValueAt(SelectedRows,
0).toString();
                String sname = RecordTable.getValueAt(SelectedRows,
1).toString();
```

```
                String examname = (String)
RecordTable.getValueAt(SelectedRows, 3);
                String pieces = (String)
RecordTable.getValueAt(SelectedRows,4);
                boolean flag = (boolean)
RecordTable.getValueAt(SelectedRows,7);
                String date = (String)
RecordTable.getValueAt(SelectedRows,8);
                String notes = (String)
RecordTable.getValueAt(SelectedRows,9);
                individualstudentpage isp = new individualstudentpage();
                isp.backfunctionsetpage("admin");
                isp.setName(adminname);
                isp.setStudent(sid, sname, examname, date, pieces, flag,
notes);

               isp.setVisible(true);
               this.setVisible(false);


        }
        else{
        DefaultTableModel RecordTable = (DefaultTableModel)cat.getModel();
        int SelectedRows = cat.getSelectedRow();
        tfsname.setText(RecordTable.getValueAt(SelectedRows,
1).toString());
        tftname.setText(RecordTable.getValueAt(SelectedRows,
2).toString());
        tfexam.setText(RecordTable.getValueAt(SelectedRows, 3).toString());
        tfpieces.setText(RecordTable.getValueAt(SelectedRows,
4).toString());

        tfpreph.setText(RecordTable.getValueAt(SelectedRows,
6).toString());
        }
    }

    private void tfsnameActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void jComboBoxfilterActionPerformed(java.awt.event.ActionEvent
evt) {
```

```
        // TODO add your handling code here:
    }

    private void jComboBoxfilterItemStateChanged(java.awt.event.ItemEvent
evt) {
        // TODO add your handling code here:
    }

    private void btnfilterActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:

        cat.setRowSorter(null);
TableRowSorter<TableModel> sorter = new
TableRowSorter<TableModel>(cat.getModel());
cat.setRowSorter(sorter);

ArrayList<RowSorter.SortKey> sortKeys = new ArrayList<>(25);
sortKeys.add(new RowSorter.SortKey(jComboBoxfilter.getSelectedIndex()+1,
SortOrder.ASCENDING));
sortKeys.add(new RowSorter.SortKey(0, SortOrder.ASCENDING));
sorter.setSortKeys(sortKeys);



    }

    private void buttonbackActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        adminhomepage ahp = new adminhomepage();
        ahp.setName(adminname);
               ahp.setVisible(true);
               this.setVisible(false);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new currentlyenrolledstudentsadmin().setVisible(true);
            }
        });
```

```
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel
setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with
the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(currentlyenrolledstudentsadmin.class.get
Name()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new currentlyenrolledstudentsadmin().setVisible(true);
            }
        });
    }
```

```
    // Variables declaration - do not modify
    private javax.swing.JButton btnfilter;
    private javax.swing.JButton buttonaddCTS;
    private javax.swing.JButton buttonback;
    private javax.swing.JButton buttondeleteCTS;
    private javax.swing.JButton buttonexit;
    private javax.swing.JButton buttonupdateCTS;
    private javax.swing.JTable cat;
    private javax.swing.JComboBox<String> jComboBoxfilter;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextField tfexam;
    private javax.swing.JTextField tfpieces;
    private javax.swing.JTextField tfpreph;
    private javax.swing.JTextField tfsname;
    private javax.swing.JTextField tftname;
    // End of variables declaration

    /*private Connection getConnection() {
        throw new UnsupportedOperationException("Not supported yet."); //
Generated from
nbfs://nbhost/SystemFileSystem/Templates/Classes/Code/GeneratedMethodBody
    }*/
}
```

Global Issue: Futility of War and Destruction Caused by it

Literary Text: An Irish Airman Foresees his Death by William Butler Yeats

Literary Body of Work: William Butler Yeats Poetry Collection

Non-Literary Text: The Siege of Aleppo 005 by Javier Manzano

Non-Literary Body of Work: Syrian Civil War Photo Collection by Javier Manzano

- Introduction: War-Middle East/Russia,GI,compare,contrast - Manzano non-religion, non-nationalistic contrary to Yeats.

- Irish Airman: emotional/psychological destruction - line 1 “I know”, numbness, context and Britain, “waste of breath” freedom, “lonely impulse”
- Irish Airman: futility, life is worthless, line 7 Kiltartan poverty reference, line 3, 4 “hate” and “love”

- Yeats BOW: Common “futility”, “religion”, “Irish” elements - Easter 1916 - religion as comfort, rebels, symbolism of unmoving stone, questioning unnecessary sacrifice, mourning
- Yeats BOW: Common “destruction” elements - Rose of Battle - symbolism - Rose, women, ireland, return for love of Ireland, futility, methodology

- Aleppo 005: Personal effect, blood and rubble, soldier, crying, sad, restrained, questioning death purpose, war glorious, cremation
- Aleppo 005: Urban destruction, infrastructure, gray hue, isolated atmosphere, livelihoods destroyed, anguish vs glory

- Manzano BOW: Common ‘death’ element, blood, saving, civilian shot, emotional devastation
- Manzano BOW: Urban destruction(Karmel Jabl), rebel looking, children, Turkey, emotional appeal

- Conclusion: Summary, importance of analysis - appeal to broad audience, makes audience reflect on futility and destruction.

"A Doll's House" by Henrik Ibsen- An Essay

"A Doll's House" is a play written by Norwegian playwright Henrik Ibsen in 1879. It evoked many reactions from the audience and was considered controversial for its times. On closer examination of the text, Ibsen wished for a deliberate reaction from the public, as he focused on challenging social norms to spread awareness. Through the play's contrasting characters and interactions, Ibsen highlights and sometimes questions social norms focusing on constrictive gender roles, money, and parental obligations.

Focusing on Nora and her interactions, Ibsen is able to highlight the constrictive nature of gender roles. A prime example is a conversation between Nora and Ms Linde. Here, we see Nora outlining how revealing the loan she took to Torvald would "completely wreck" her marriage. This was because her action was contrary to typical gender roles, as Nora explains when she says, "it'd be so painful and humiliating for him to know that he owed anything to me". Nora's reasons for not telling Torvald about the loan emphasize the strict division of roles, in which women are not meant to support their husbands financially. Any challenge of this would bring "humiliation" to Torvald and disrupt the relationship. Nora also exhibits strict gender role when recounting her experience working to repay the loan. "It was almost like being a man", she mentions, implying "work" to be an area of a man's realm rather than an independent woman's, further highlighting the segregation of gender roles. Thus, Ibsen is able to show the constrictive nature of gender roles through conversations in the play, eventually fulfilling his purpose of spreading awareness.

Ibsen, through his work, also highlights the effect of money in aiding power dynamics. This is seen right at the beginning of the play, with Torvald dictating how much Nora can spend

on Christmas presents. In doing so, Torvald belittles Nora, calling her a "spendthrift" and "little person" when she requests money from him. This name-calling shows how Nora's financial dependence allows Torvald to exert power over her and be condescending. The intersection between money and power dynamics also plays into Nora and Ms Linde's conversation. When asked why she married someone she did not love, Ms Linde outlines how she couldn't "refuse his offer". Her mother was "bedridden and helpless", and she had "two younger brothers to look after". This need for financial support led to her not being able to marry her true love, Krogstad. The interaction highlights the power that money brought in Norwegian society, in which women like Ms Linde were forced to sacrifice their careers and personal life to be financially stable. Thus, Ibsen successfully highlights the effects power dynamics in Norwegian society by centring around money.

Another critical concept Ibsen focuses on, and challenges is parental obligations. Throughout his play, he makes references to traditional parental roles through Torvald and Nora. Paternal roles are reflected in Nora's dialogues where, at the beginning of the play, she states, "he'll have a big salary and lots of commission". This statement implies a paternal role of financially supporting the family. Torvald, on the other hand, outlines maternal roles. When speaking of Krogstad's immoral actions with Nora, he states, "almost anyone who has gone bad in life has had a deceitful mother". This shows a maternal obligation to raise and influence children to be morally sound. However, as the play progresses, these roles are frequently questioned. With Nora taking a loan, she undertakes a paternal obligation of providing for the family. Torvald, at the end of the play, is left to raise the children as Nora embarks on a journey of self-discovery, leading to a possible exchange of parental roles. Thus, Ibsen successfully brings out parental norms in the plot and challenges them.

Henrik Ibsen was undoubtedly one of the most influential writers of his time. By including interactions between characters with different financial levels and life experiences throughout

the play, he successfully shed light on the strict division of gender roles, the relationship between money and power dynamics, and parental obligations. Through his work, he evoked discussion on social issues, spreading awareness and fulfilling his goal of improving women and, ultimately, human rights.

Aishwary, this is a very good essay. Keep it up.

9 + 9 + 9

27/30

Chinua Achebe was a 20th-century author born in Nigeria. Often known as 'The Father of African Literature,' Achebe aimed to use his experiences and Nigerian identity to retell the story of Africa, producing anti-colonial work, elevating African literature, and combatting Western biases. This response examines how Achebe's motivations are reflected in his book 'Things Fall Apart.' Set in pre-colonial Nigeria, the novel revolves around Okonkwo, a well-respected member in his village, Umuofia. Unable to cope with the change in environment due to colonialism, he eventually commits suicide.

One primary purpose Chinua Achebe had when writing was to capture the fear, loss, and violence faced by many Africans due to colonialism. In Things Fall Apart, evidence lies in the destruction of the village of 'Abame.' This instills fear in the people of Umuofia, with Obierika stating, "I am greatly afraid." He also speaks of the slave trade ("and took slaves away across the seas"). The village's destruction and practices, such as the slave trade, show the threat and fear other villages, such as Umuofia, face. Achebe also shows loss through Christianity and its impact on Ibo culture and the community. For example, when messengers came to preach Christianity, they used lines such as "leave your wicked ways and false Gods," creating a sense of distrust in the Ibo religion. This attracted more villagers to Christianity, eventually dividing the community. The division led to conflict after a convert unmasked an 'egwugwu,' and the Church was burned. Thus, Achebe shows how colonialism divided communities, caused fear and violence, and eventually led to a loss in culture.

Achebe also aimed to retell the story of Africa from an authentic lens. This is evident in Achebe's writing style, where he makes common references to Ibo linguistic features to showcase Africa's culture. For example, instead of referring to common words such as 'house' and 'outcast,' Achebe uses Ibo vocabulary such as 'obi' and 'osu.' He also uses proverbs, with one of them being "a man who pays respect to the great paves the way for his own greatness." These features allow Achebe to retell a story that shows the complexity and uniqueness of the culture, contrary to the common colonialist perspective. Achebe also contrasts African and Western perspectives in the novel to combat readers' biases and change Africa's image. The end of the novel describes the commissioner writing a book named 'The Pacification of the Primitive Tribes of the Lower Niger.' The title suggests how the British viewed Africa as a continent without culture and knowledge. The inclusion of this title at the end forces readers to reflect more deeply on the Ibo culture described throughout the novel. The British perspective hence sounds ignorant to the reader after reading about the deep cultural practices of the Ibo people. This allows Achebe to challenge perspectives, in turn changing the perception of Africa. Hence, by portraying Ibo cultural elements and contrasting their complexity and uniqueness with the Western perspective of colonization, Achebe is able to take control of the narrative for the African continent, elevating African literature and shaping its story authentically.

Another purpose of Chinua Achebe was to show the sophistication of the Ibo people, to combat Western biases. In the novel, Achebe shows this sophistication through the Ibo judicial system. In one chapter regarding a domestic violence case, he describes the 'egwugwu' or the nine masked spirits who deliberate over the proceedings, and the fear and authority they wield ("there was room for running away if any of the egwugwu should go towards them"). He also captures similarities to the Western court system when he describes how the two parties stand on opposite sides. Through this, Achebe shows the Ibo people with already established systems, contradicting the Western perspective of them being 'savages,' as

shown in books such as the Heart of Darkness. He is able to effectively counter colonialist rhetoric and show the complexity and sophistication of the Ibo people.

Achebe also used his one experiences and family history to write his books, to show authenticity and emphasize the effect of colonization on family relations and villages. Okonkwo, for example, resembles Achebe's parents and their experience with colonialism. Living in an Ibo village, Achebe's parents experienced the rise of colonialism and saw the changes in culture and the alienation between tribesmen due to it. When his parents moved to Ogidi in 1935, they further saw the division in the community, with their own family being divided between traditional values and the new religion. Okonkwo experiences the same. His family becomes divided due to Nwoye's conversion to Christianity, and when he returns to Umuofia, the village has transformed. This is proven by the narrator saying, "The clan had undergone such profound change during his exile that it was barely recognizable." This allows Achebe to show an accurate representation of the Ibo people's plight through characters in the book. By capturing the real-life instances he and many other Africans experienced, Achebe is able to show the extent to which colonization changed lives.

Achebe was one of the most influential African authors. By using religion and examples of colonialist violence, Achebe captured the loss and threat faced by many Africans. Through Ibo cultural elements, he was able to retell Africa's story as one ripe with a unique culture. By describing Ibo society and its systems, he captured the sophistication of the African people, again combatting Western biases. His real-life experiences allowed him to paint Africa's image authentically, one representing many Africans' experiences.

Sources (Apart from Things Fall Apart novel) in MLA 9:

1. Hinlicky, Rachael. “Theme and Biographical Analysis of Things Fall Apart by Rachael Hinlicky.” *Theme and Biographical Analysis of Things Fall Apart by Rachael Hinlicky | M.A.R. Habib | Rutgers University*, 2014, https://habib.camden.rutgers.edu/2014/05/20/theme-and-biographical-analysis-of-things-fall-apart-by-rachael-hinlicky/.

It is a good essay. Some work still needed in the Evaluation of the evidence. Introduction is not good. Work on it!

8+8+8= 24

W.B. Yeats was a 20th-century poet born in Ireland. An Irish nationalist, and a staunch advocate for Irish Independence, Yeats was arguably one of the most influential Irish poets. His poems often had common traits and symbols, courtesy of his upbringing and nationalistic views.

One commonality about Yeats' body of work is his frequent reference to Irish independence. Yeats' work focuses a lot on Irish independence and his personal views on it. It also contains his nationalistic views. For example, his poem 'Easter 1916' celebrates the martyrs of the Easter uprising against the British Government. However, he also presents his view. For example, in the line "A terrible beauty is born," he shows reluctance in commemorating the uprising. In another poem, 'Sixteen Dead Men', Yeats lambasts those who call for peace in Ireland by writing about the sixteen dead men from the Easter uprising; his portrayal of the British as cruel and brutal further shows his positive views on Irish independence.

Yeats' body of work also contains frequent reference to folklore and mythological symbols. Yeats often used Celtic and Irish folklore references to convey his messages and portray his nationalism. One poem, 'The Song of Wandering Aengus,' is entirely dedicated to the Irish God of Love. His use of an Irish God, rather than the conventional Greek and Roman, portrays his nationalistic attitude and desire to elevate Irish folk and literature. In another poem, 'To the Rose upon the Rood of Time,' Yeats refers to folktales such as Cuchulain's fighting of the sea ("Cuchulain battling with the bitter tide") and figures such as Fergus ("Who cast round Fergus dreams") while speaking to the Rose, showing his admiration for Ireland and his nationalistic attitudes.

A central common theme in his body of work is his infatuation with and reference to Maud Gonne, a woman whom he loved but was rejected multiple times by. He refers to her in most of his poems. In one poem, "When you are old," published in 1893, Yeats directly addresses Maud Gonne and his knowledge of her sorrows ("And loved the sorrows of your changing face") and exhibits a desire to protect her. In "A Prayer for My Daughter," published in 1921, Yeats includes the line "Have I not seen the loveliest woman born," a reference to Maud Gonne, to give an example of why his daughter should not give in to wild feelings and should be temperate, as people who love deeply, hate deeply too, like Maud Gonne.

Hence, we see that Maud Gonne is referenced multiple times, but as time passes, Yeats' body of work becomes more and more hostile towards her, potentially because of her marriage with John MacBride and the rejections Yeats faced from her.

Yeats' body of work was perhaps one of the most influential in Ireland and Europe. By referencing Irish folktales and mythological symbols, he attempted to fight against the British control of the school system, and re-educate the Irish about their heritage to promote nationalism. Along with spiritualism and nationalism, Yeats' constant reference to Maud Gonne also gave his body of work a romantic touch, making his poems unique and personal.

Sources:

Bell, Matthew. "Yeats, Nationalism, and Myth." *Matthew Bell - "Yeats, Nationalism, and Myth"*, https://writing.colostate.edu/gallery/phantasmagoria/bell.htm.

"A Prayer for My Daughter Summary & Analysis by William Butler Yeats." *LitCharts*, https://www.litcharts.com/poetry/william-butler-yeats/a-prayer-for-my-daughter.

"WB Yeats Poems Inspired by Maud Gonne." *Orna Ross*, 6 Nov. 2013, https://www.ornaross.com/wb-yeats-poems-inspired-by-maud-gonne/.

This is good work Abhay! Work on my suggestions.

A - 8
B - 8
D - 9

25