

# **STUDENT DATABASE MANAGEMENT (SDM)**

A Project Report Submitted

In Partial Fulfillment of the Requirement

For the Degree in Bachelor of Science (Computer Science)

By

**K.Beihrazi**

**S1701361**

**17003427**



**Department of Computer Science St.  
Edmund's College  
Shillong - 793003**

**July, 2019**



---

## **CERTIFICATE**

This is to certify that the project work titled “STUDENT DATABASE MANAGEMENT” is bonafide work done by K.Beihrazi bearing university roll number S1701361, under my guidance during the 6th Semester in Partial Fulfillment of the Requirement for the Degree in Bachelor of Science (Computer Science)

Prof Ma'am Preeti Thapa  
Internal Guide

The project seminar was held on \_\_\_\_\_ at St. Edmund's College,  
Shillong.

Dr. S. Nagi,  
Associate Professor & H.O.D.,  
Deptt. of Computer Science,  
St. Edmund's College, Shillong.

External Examiner

# CONTENTS

1. Acknowledgement
2. Introduction
3. Synopsis
4. Feasibility Study:
  - a. Technical Feasibility
  - b. Behavioural Feasibility
  - c. Economic Feasibility
5. Analysis:
  - a. Existing Systems
  - b. Screenshots of Existing Systems
  - c. Proposed System
  - d. Comparison of Existing and Proposed System
6. Hardware and Software Requirements:
  - a. Developer Requirements
  - b. User Requirements
7. Designs:
  - a. ER Diagrams
  - b. Data Flow Diagrams
8. Future Enhancements
9. Screenshots of developed System
10. Conclusion
11. Code
12. Bibliography

## **ACKNOWLEDGEMENT**

First of all I would like to thank The Almighty for providing me a good health and able to complete this project with success on time. I would like to express my sincere gratitude and special thanks to my teachers and our Hod Dr.S.Nagi who have given me this good opportunity to do this project on 'Student Database Management' and also help me through many trials and errors during developing this project, which later on encourages me to do a lot of Research on the subject and came to aware a lot about many new interesting things related to java and its features.

Secondly I would like to thank my classmates and friends who helped me through too, last but not the least, with due regards I thank my parents for always supporting me.

# INTRODUCTION

Student database management (SDM) system is an application software that deals and maintains all kinds of details or information. This information can be of any kinds related to student details, academic records, student's attendance, course details, academy progress and other related details too. It tracks all the details of a student from the day one to the end of his course which can be used for all reporting purposes.

This is an automation system which replaces manual work that requires maintaining in the form of documents or paperwork and reduces manpower and time complexity. Since it is computerized system, the accuracy and consistent state is maintained well. Maintaining backup is easy and it is user friendly interface. The system has two types of accessing modes, Lecturer and student user. SDM is managed by admin (lecturer). Its job to maintain and update or remove and monitor the whole process. The user in return can then check his records and progress.

Overall the goal is to utilize this Student database management into data retrieval, data manipulation and provide more ease for managing and improved valuable time of Staff and maintained all the data in ordered and in a consistent state.

## SYNOPSIS

The project, 'Student Database Management' provides a simple interface for maintaining Students information and his academic progress in the institute. This software can be used by any educational institutes or colleges to maintain the records and keep tracks of it, without this management sometimes it becomes a tedious work for any institute or organization to manage and maintain the details of the students.

The Purpose of this project is to provide the admin or lecturer to be able to represent a report or view all the details regarding the students information and his progress and scores as well to be available to the user within a few seconds. This will help many stakeholders of college or institute to quickly perform the requirement operations instead of doing manually.

The objectives is to improve time flexibility, consistency and accuracy, as manual work has some cons therefore with this management system the goal is to improve it better. Backing up the data is also easy and if one lose track of it they can always check the database and resume the records where they were lost at the previous track.

The salient Features includes:

**One-click access:** One can obtain the details of the students or records by entering his/her name or the roll number in the search button.

**Universal access with ease :** One does not need a deep study/info in any language to use this, its user friendly

**Updating & Retrieval :** Easy to add,edit, update and retrieve back information.

**Security :** The user passwords is secured and protected, the user is requested to fill the Security question incase later on if he/she forgets or tempts to re-change the password.

**Redundant Human Errors :** Improves Accuracy & Flexibility.

**Student Portal :** Students can keep tracks of their progress regarding Test Scores, courses & Attendance as well

**Builds Alumni :** An alumni network helps in myriad ways, a colleague of the same field or type can help or guide the other whos below trying to make his way up and have the Same field or goal.

On the other hand this projects tends to bring closeness among students and the teaching stuff and improves coordination between them and also acts as centralized system for latest updates.

## SYNOPSIS

### **The Product Functions include:**

There are two different users

- Admin (lecturer) who can monitor and add, edit, delete and view regarding any details/information.
- Student users, who can keep track of their progress and records but not provided with an authority to make any changes in the database.

The System will provide a login page with username and password, the details of login are secured and protected with encryption.

The lecturers job is to check and keep the records and details and update the changes so that the students in return can check the accurate latest updates.

The System will also provide with search and filter options in the database so that the accessibility becomes quick and easy.

The system provides a print option so that the information can be printed out

The inventory system keeps all the details of equipments and tools used in department, and prices tags.

build and manage the alumni network, by facilitating engagement, community- building, networking, communications and the passed out or graduate student can enroll for the alumni so that the community becomes strong and guide others in need which inturns helps the institution and makes the institution proud to have a student like him/her and the reputation of that person grows in many ways.

# FEASIBILITY STUDY

## 1. Technical Feasibility:

Judging from the project we can say that its strongly technical feasible, since all the requirements is available in the server and one can easily download it. The software equipments are not that hard to get for developing this project. This project is developed using netbeans IDE 8.0.1 with java jdk 8 and wamp server where all of our database is stored and there are some basic hardware requirements but if it satisfies that minimum hardware specification, its still strongly feasible.

## 2. Behavioural study:

This project is developed to improve the user /staff's time flexibility by reducing their manual work which usually required by going through lots of paper work and at times it's a headache when files or data are not in order or lost or its in inconsistent state. This software will inturn help him and do quick operation if required and can keep track or search anything within in a few seconds. It is user friendly and one does not need any special training or learn a course to used this software. Simple thing makes life easier and reduce stress.

## 3. Economic Feasibility:

For developing this project its highly economical feasible as from one's perspective or any organization one does not need to spent any amount to purchase the tools required for developing this, since most of the requirement equipments or softwares needed to build this project is already available in the internet.

When it comes to income or benefits since theres no any investment needed to build this project except time, its economically feasible and database management is a great demand when it comes to data world.



## ANALYSIS

### EXISTING SYSTEM:

#### ➤ College-ERP :

A college management system built using Django framework. It is designed for interactions between students and teachers. Features include attendance, marks and time table. The required equipment used for developing this software is Python and Django.

The Features of College-ERP:

- Attendance
- marks reports
- Time table

Pros :

- Its reliable, created by a group of skill-talented people.

Cons :

College-ERP is the free version of another administration software of the same name. When comparing the free version to the paid version, it becomes clear that the open-source version is lacking in a number of features, including inventory, custom reports, registration, Security and discipline.

- It doesn't come with extensive support since it is an open software
- vulnerable to malicious users and privacy security is weak.

## Screenshot of Existing System:

### Student Page

CollegeERP Samarth Logout

## Welcome Samarth

### Attendance

View the attendance status for each of your courses. The attendance of each course is also displayed as list of classes that were conducted.

[View Attendance](#)

### Marks

View the marks obtained for each of your courses. These include the marks of 3 internal assessment, 2 events and the Semester End Exam

[View Marks](#)

### TimeTable

View the timetable in a tabular form. The timetable displays all the courses of the student and the time and day at which they are conducted.

[View TimeTable](#)

CollegeERP Samarth Logout

### Attendance

Course ID	Course name	Attended classes	Total classes	Attendance %	Classes to attend
CS510	Database Management System	4	5	86.67	2
CS520	UNIX	0	0	0	0
CS530	Software Engineering	8	10	80.0	0
CS540	Computer Networks	0	0	0	0
CS550	Language Processor	0	0	0	0
MA510	Linear Algebra	0	0	0	0

CollegeERP

[Home](#)
[Attendance](#)
[Marks](#)
[Time Table](#)
[Reports](#)

Student name

Dakshath

Farhan

Nihal

Nikhil

Rajat

Renu

Anudeep

Samarth

Shahid

Shravan

Boss

Tejus

Present

Absent

Present

Absent

Present

Absent

Present

Absent

Present

Absent

Present

Absent

Present

Absent

Type here to search

7:35 PM

11/15/2018

CollegeERP

[Home](#)
[Attendance](#)
[Marks](#)
[Time Table](#)
[Reports](#)

Student Name

Dakshath

Farhan

Nihal

Nikhil

Rajat

Renu

Anudeep

Samarth

Shahid

Shravan

Boss

Tejus

Vijeth

Total Marks

20

20

20

20

20

20

20

20

20

20

20

20

20

Enter Marks

20

15

12

19

14

12

15

9

8

7

15

14

12

Type here to search

7:36 PM

11/15/2018

## **PROPOSED SYSTEM**

### **Whats the goal and motive of SDM?**

- 1) To help the users -Maximum Organization with Minimum paperwork
- 2) It manages all the informations regarding the Student bio-data, Academic Score, exams, Stuff inventory, alumni, Courses and Attendance.
- 3) It keeps tracks of all Students in detail from day one to the end of his course.
- 4) The main purpose is to reduce the manual Work and decrease the efforts & improves time flexibility and working facilities .

### **How will this SDM help to a student?**

This Application will allow the Students to access & view their latest update reports anytime when they wish to see.

For new Students they can login and view their department -Their Teachers name and which teacher will be teaching which Subjects and so on.

Better Communication means SDM system allows for easier communication between students- teachers and even allows for the school alumni to stay connected with each other

Their info can be retrieve back by reporting to the admin or the staff.

### **How will this SDM help to a Teacher?**

Ease of editing and inserting new records and delete too eg, Grades

Searching the reports or details of a student becoms easy and maintaining reports becomes more flexible compared to spreadsheets.

The attendance management system allows to maintain a quick and accurate record of the student And check individual attendance percentage quickly to see if he/she is eligible for the exam or not.

Consistency becomes easy (its all in order and grouped)

Increased Data Security - One of the best features of using this is to manage the student database that it is very secure and ensures that all the secure institute or college data is kept safe. The system grants access to only those with prior authorization, thus ensuring that all the confidential student-related data is kept secure.

## **Comparison of Existing and Proposed System:**

Existing System	Proposed System
1.Less features and not so strong functionalities 2. Security is not so strong, if database is breached password can be stolen 3.Printing can't be done if required 4. Its an automation system.	1. Features are many compared to existing system and good functionalities with quick search and filter options 2. Security is strong as the user's passwords are encrypted end to end. 3.Printing access is provided 4. Its an automation system.

## Hardware and Software Requirements

### 1. Developer Requirements:

Minimum Hardware Configuration	Recommend Hardware Configuration
Processor: Intel(R) Pentium(R) Dual CPU E2200 @2.2.GHz 2.20 GHz  Memenory(RAM): 4 GB  Disk Space: 1.5 GB of free disk space	Processor: intel Core i5 or equivalent  Memenory(RAM): 6GB or above  Disk Space: 2 GB of free disk space

### 2. Developer Software Requirements:

- jdk-8u162-windows-x64 (JAVA SE 8.01 version)
- Netbeans IDE 8.0.1
- Wamp Server2.2e
- Microsoft visual c++ 2015
- Mysql-8.0.12-winx64

# Hardware and Software Requirements

## 1. User Requiements:

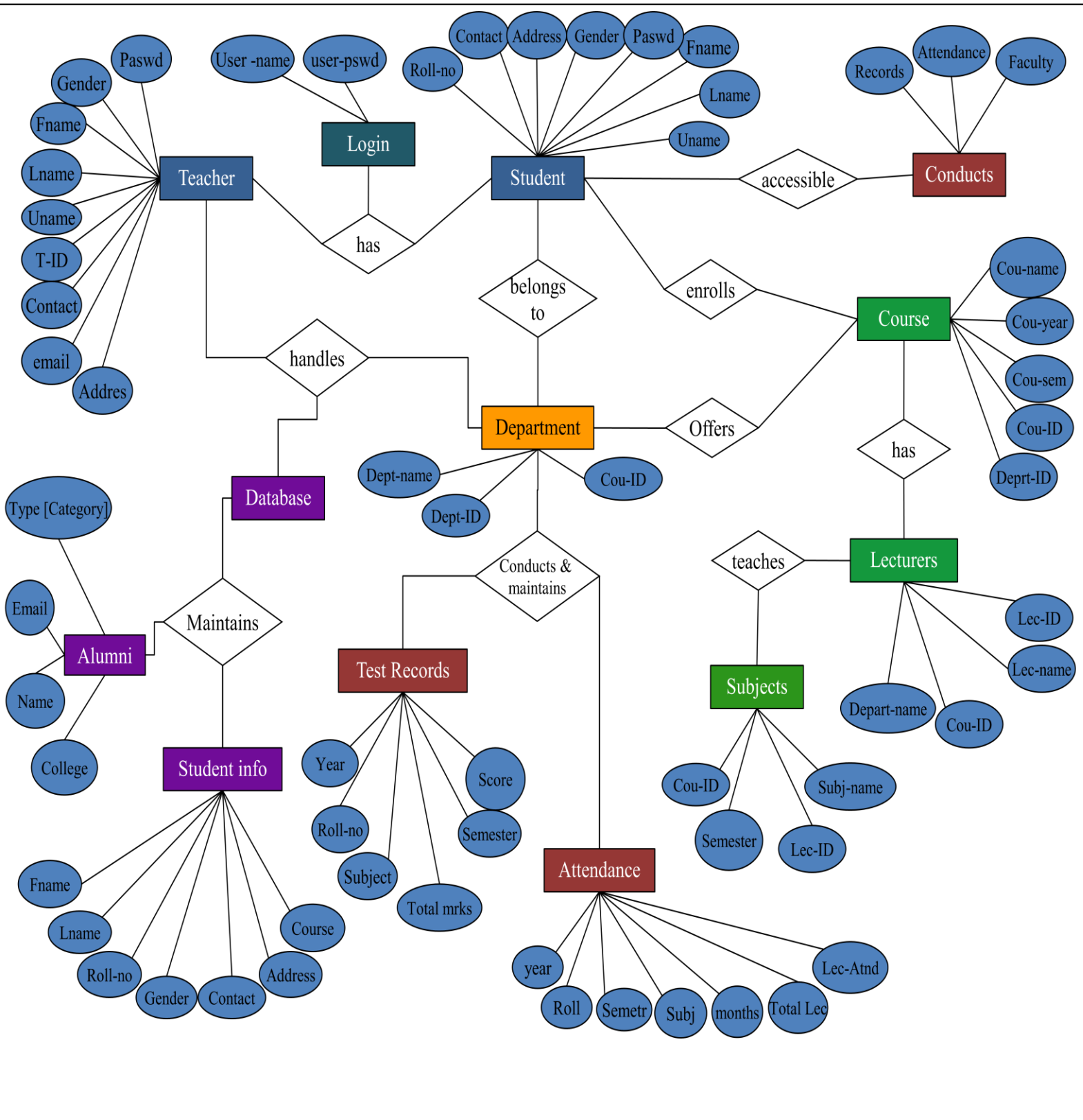
Minimum Hardware Configuration	Minimum Hardware Configuration
Processor: Intel(R) Pentium(R) Dual CPU E2200 @2.2.GHz 2.20 GHz	Processor: intel Core i3
Mememory(RAM): 2 GB	Mememory(RAM): 2 GB
Disk Space: 200 MB of free disk space	Disk Space: 500 GB of free disk space

## 2. User Software Requirements:

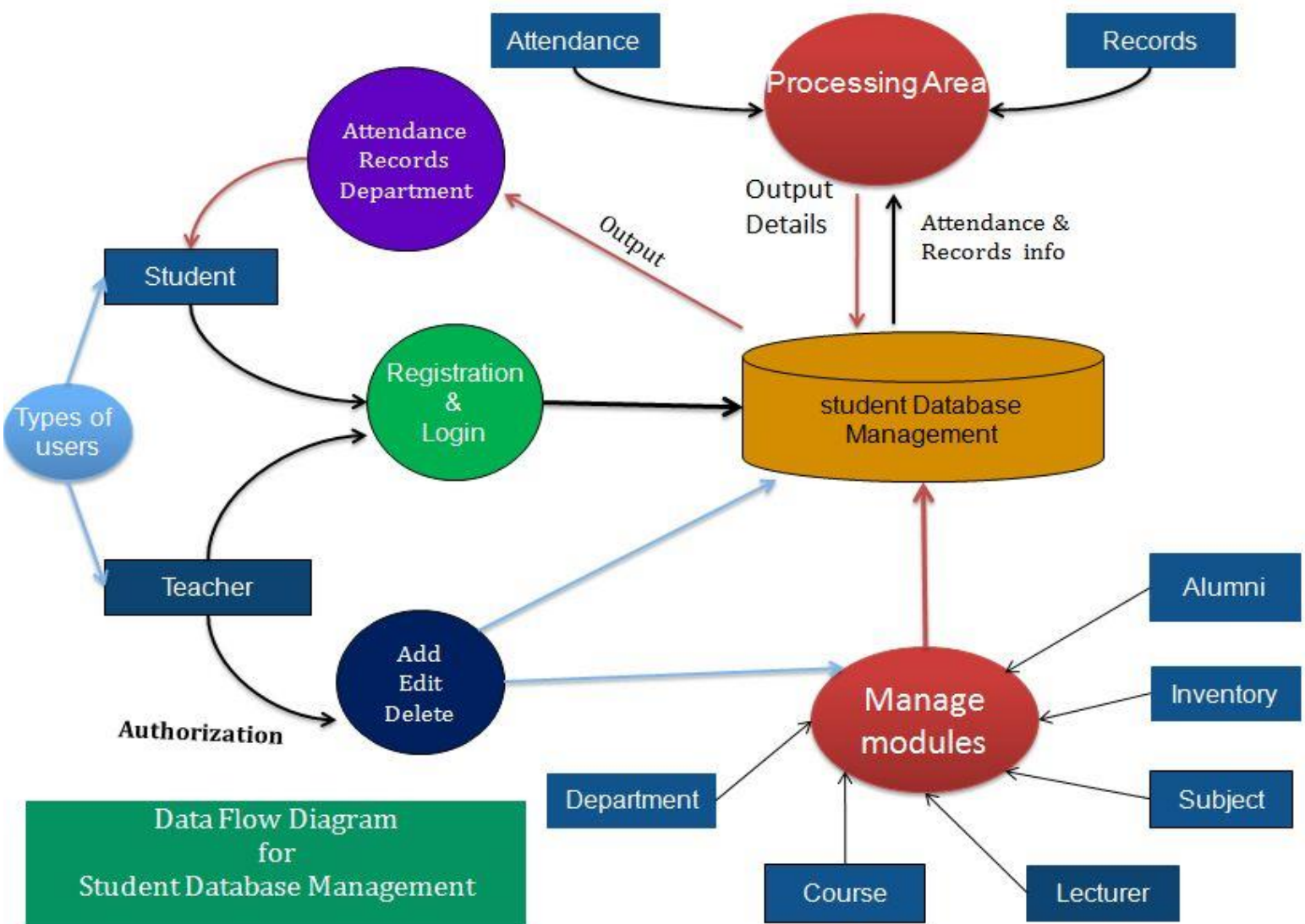
- jdk-8u162-windows-x64 (JAVA SE 8.01 version)
- Wamp Server2.2e
- Microsoft visual c++ 2015
- Direct

# DESIGN:

## 1. ER DIAGRAM FOR SDM:



## 2. DATA FLOW DIAGRAM (DFD) FOR SDM:





## **FUTURE ENHANCEMENT**

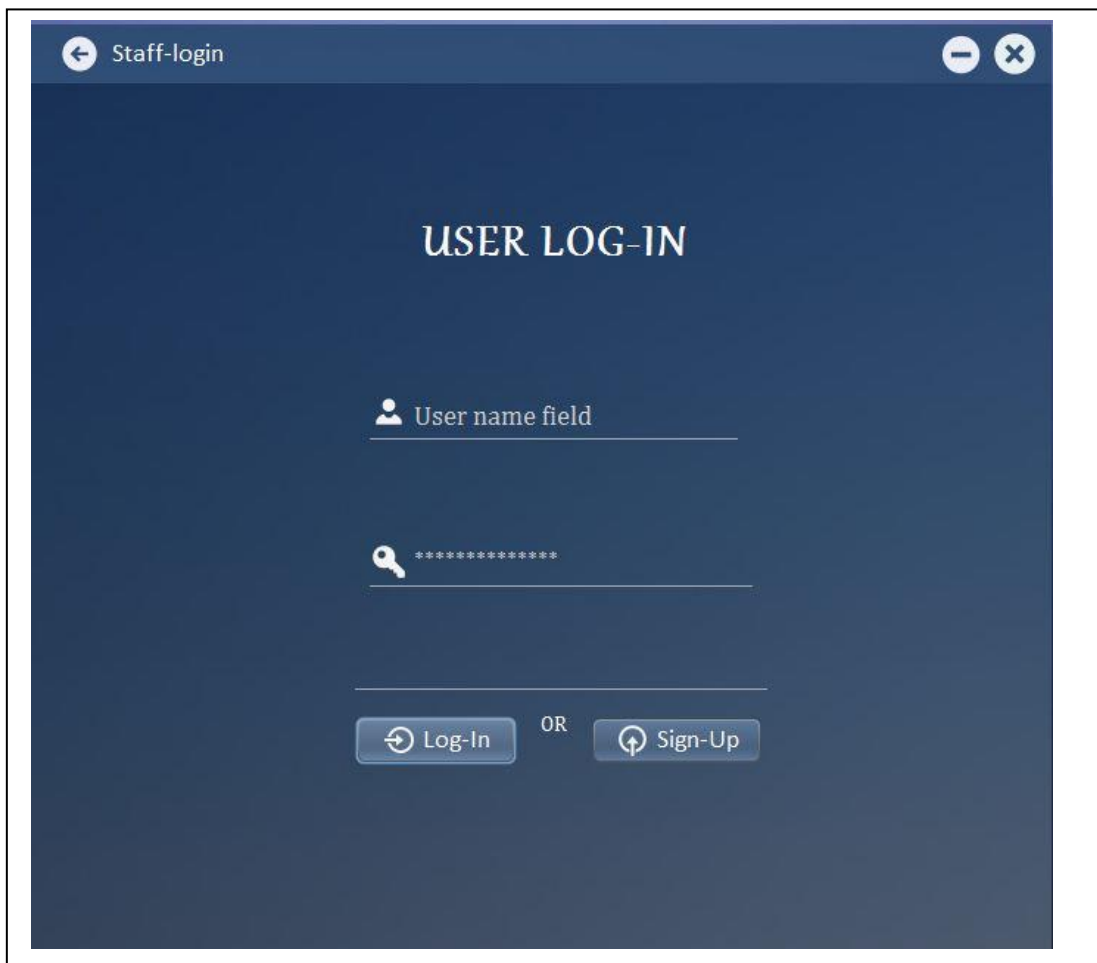
SDM works like a component which can access all the databases and stored them securely and if required to view the response is fast and easy.

- Easy implementation environment
- Generate report flexibly
- Individual Attendance system to check if he's eligible for the coming test or not.

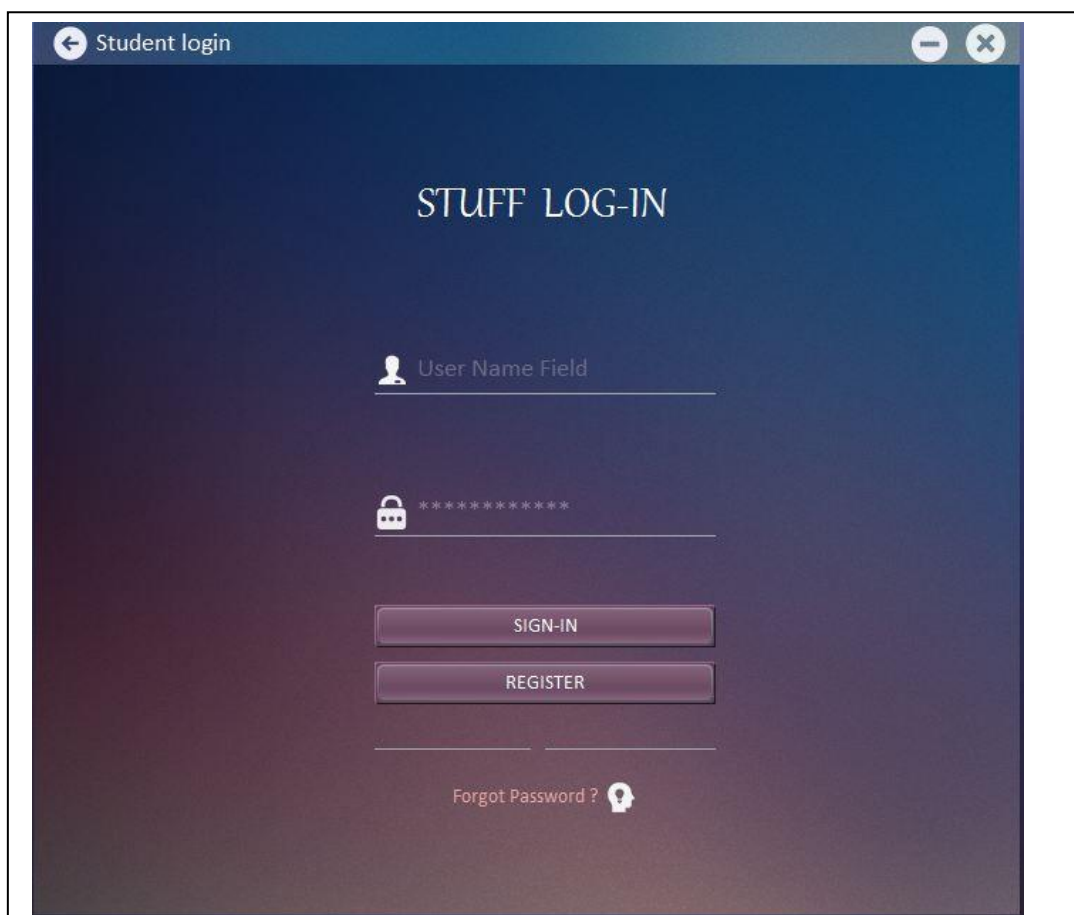
The project has a very vast scope in future though it has been not tested yet. The project can be implemented on intranet in future. Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion. With the proposed system of SDM its ready and fully functional the users can manage and hence run the entire work in a much better, accurate and error free manner since its an automation system.

## SCREENSHOT OF DEVELOPED SYSTEM

### 1.Login page:



A screenshot of a web application window titled "Staff-login". The window has a dark blue header bar with a back arrow icon on the left and window control buttons (minimize, maximize, close) on the right. The main content area is a dark blue gradient. At the top, the text "USER LOG-IN" is displayed in white, bold, uppercase letters. Below this, there are two input fields. The first field is labeled "User name field" and has a person icon to its left. The second field is for a password, indicated by a key icon and a series of asterisks. Below the password field, there are two buttons: "Log-In" with a circular arrow icon and "Sign-Up" with a circular arrow icon, separated by the word "OR".



A screenshot of a web application window titled "Student login". The window has a dark blue header bar with a back arrow icon on the left and window control buttons (minimize, maximize, close) on the right. The main content area is a dark blue gradient. At the top, the text "STUFF LOG-IN" is displayed in white, bold, uppercase letters. Below this, there are two input fields. The first field is labeled "User Name Field" and has a person icon to its left. The second field is for a password, indicated by a padlock icon and a series of asterisks. Below the password field, there are two buttons: "SIGN-IN" and "REGISTER", both in white, bold, uppercase letters. At the bottom, there is a link that says "Forgot Password ?" with a lightbulb icon.

## 2.Registration:

Back

USER-REGISTRATION

Upload

First Name :

Last Name :

User name :

Roll no :

Gender :  
☐ Male ☐ Female

Contact no :

Course (Honour) :  
Computer Science

Address :

Birth-Date :

Password :

REGISTER

Back

REGISTRATION - FORM

First Name :

Last Name :

User name :

Contact no :

Teacher's ID :

Email :

Gender : ☒ Male ☐ Female

Address :

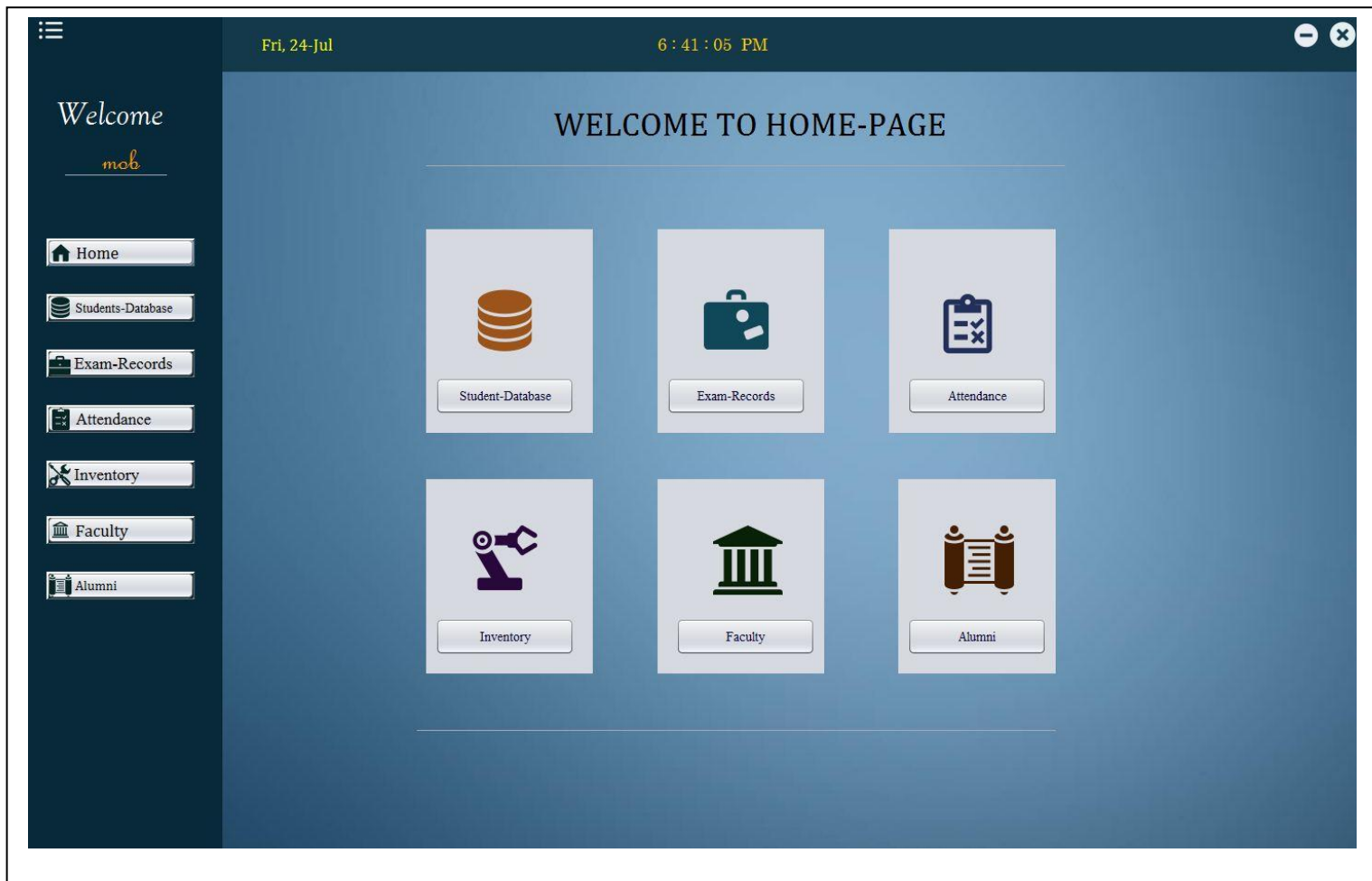
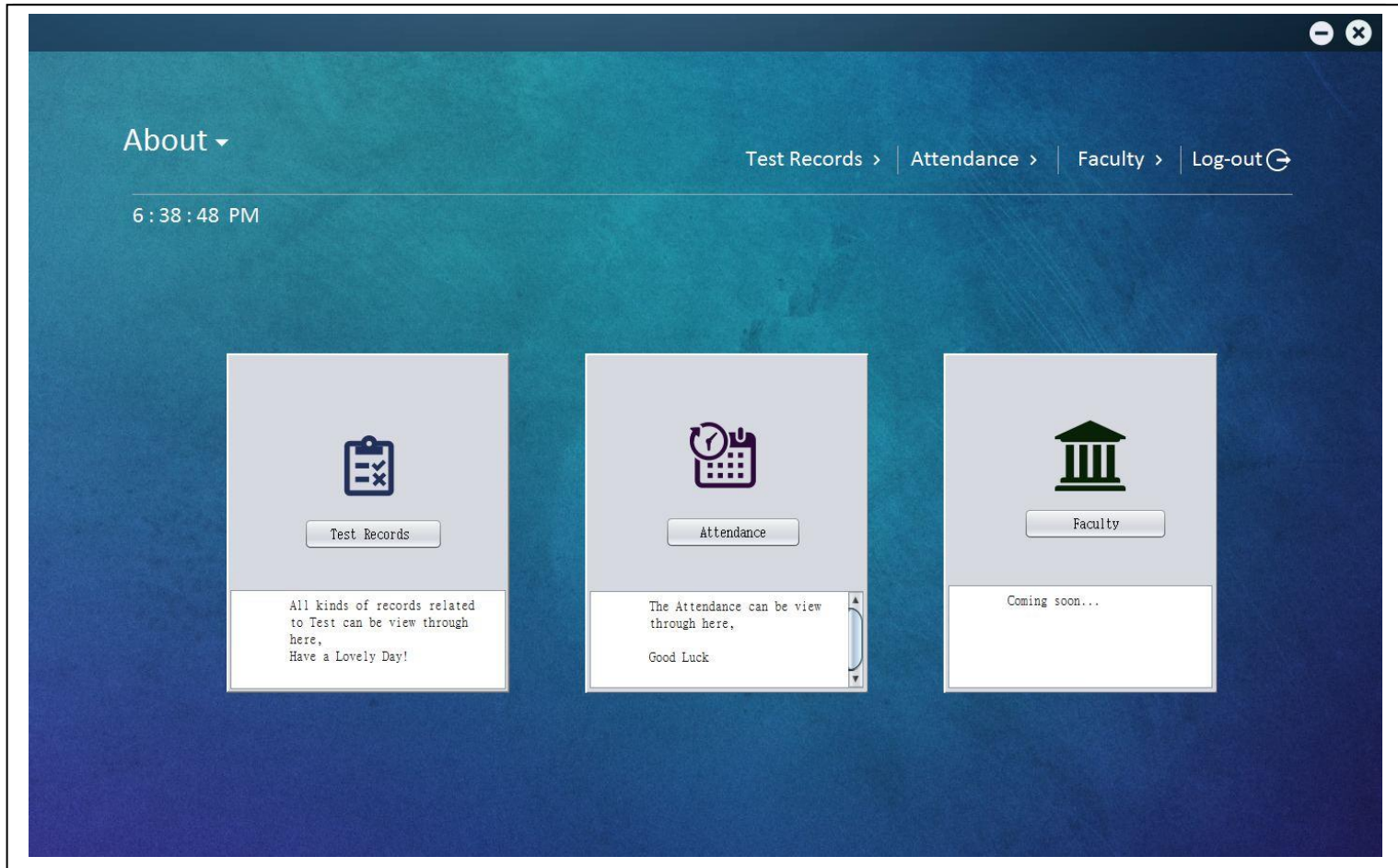
Password :

Re-Type Password :

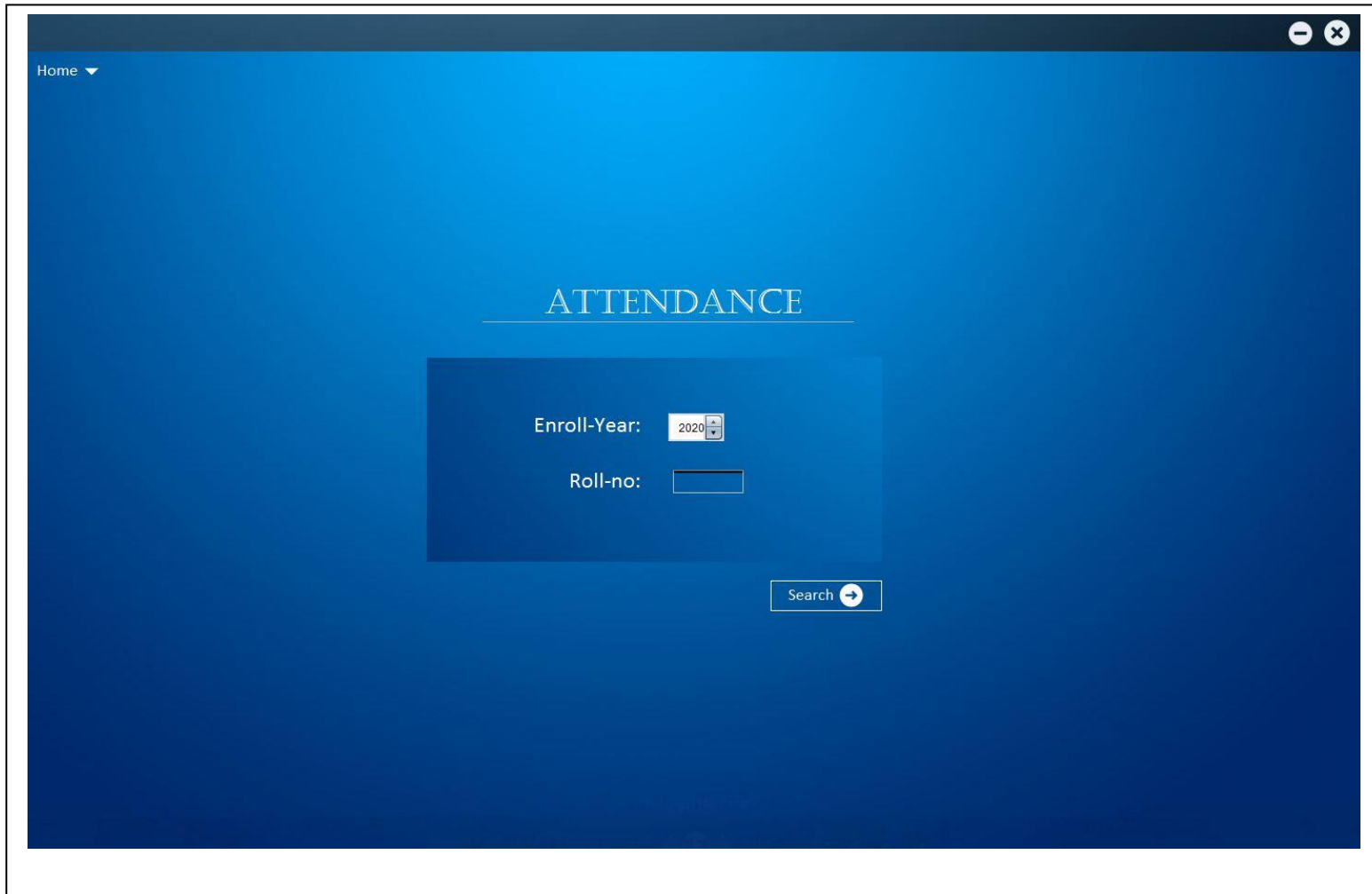
Upload

Sign-Up

#### 4.Home page for both users:



## 5.Attendance and records from student home page view



A screenshot of a web application interface for a student's home page. The background is a dark blue gradient. In the top-left corner, there is a 'Home' link with a downward arrow. In the top-right corner, there are standard window control buttons (minimize, maximize, close). The word 'ATTENDANCE' is centered in a large, white, serif font, underlined. Below it, there is a dark blue rectangular box containing two input fields: 'Enroll-Year:' with a dropdown menu showing '2020' and 'Roll-no:' with a text input field. To the right of these fields is a 'Search' button with a right-pointing arrow icon.

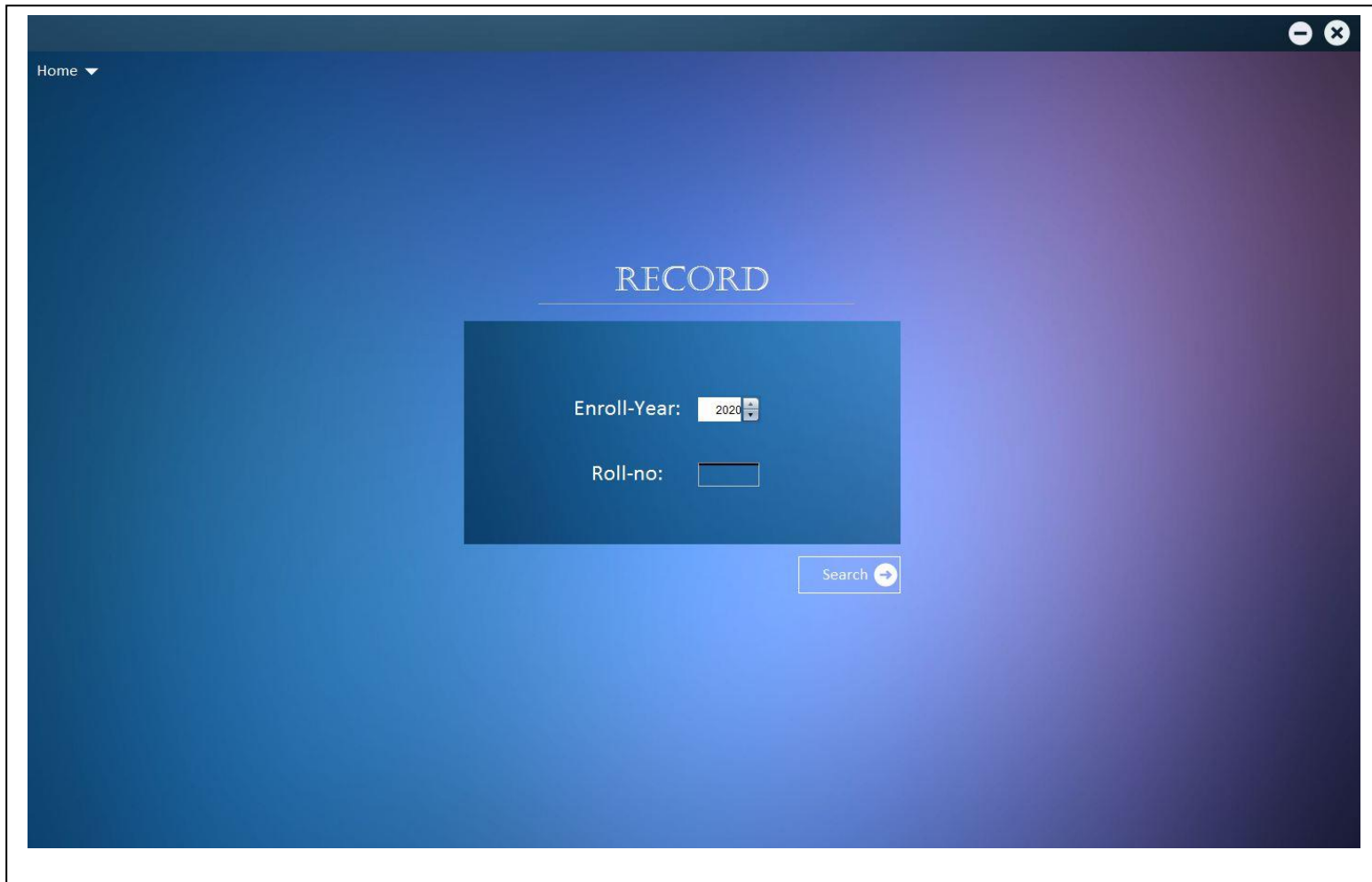
Home ▾

# ATTENDANCE

Enroll-Year: 2020

Roll-no:

Search →



A screenshot of a web application interface for a student's home page, similar to the one above but with a different background color (purple/blue gradient). The layout is identical: 'Home' link, window controls, the word 'RECORD' in a large, white, serif font, underlined, followed by a dark blue box with 'Enroll-Year:' (dropdown showing '2020') and 'Roll-no:' (text input) fields, and a 'Search' button with a right-pointing arrow icon.

Home ▾

# RECORD

Enroll-Year: 2020

Roll-no:

Search →

## 6.Profile page and Faculty:

### MY PROFILE



#### PERSONAL INFORMATION

mob

phy

mob123@gmail.com

#### TEACHER

☒ Male

123

☐ Female

2343512354

Earth Blue planet

Status

Thoughts dont Speak!  
oh yeah

Save

### PRIVACY SECURITY

### FACULTY

DEPARRTMENT

COURSE

LECTURER

SUBJECT



## 7.Stundets bio and Filtering section:

☰

Welcome  
mob

Home

Students-Database

Exam-Records

Attendance

Inventory


Faculty

Alumni

Fri, 24-Jul

6 : 43 : 28 PM

Search Students :



UPDATE

DELETE

saitama

Anime

12

☒ Male ☐ Female

2147483647

Physics

Earth-Tokyo

FName	LName	Rollno	Gender	Contact	Course	Address
saitama	Anime	12	Male	2147483647	Physics	Earth-Tokyo

Search By :  Roll no :

Filter By : ☒ Check individual Attendance%

Return

ATTENDACE %

ROLL-NO:

FILTER

Total Lectured

Lectured Attended

Add

Update

Delete

Lect	Lec_Attended	Overall_Attended
79		88.76
54		80.6
50		90.91

## 8.Database

phpMyAdmin interface showing the database structure for 'db'.

Database: db

Table	Action	Rows	Type	Collation	Size	Overhead
attendance	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	16 K1B	-
batch	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 K1B	-
course	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	16 K1B	-
date	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 K1B	-
dept	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	16 K1B	-
inventory	Browse Structure Search Insert Empty Drop	5	InnoDB	latin1_swedish_ci	16 K1B	-
lec	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 K1B	-
records	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	16 K1B	-
stdreg	Browse Structure Search Insert Empty Drop	1	InnoDB	latin1_swedish_ci	16 K1B	-
subject	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 K1B	-
tchreg	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	96 K1B	-
test	Browse Structure Search Insert Empty Drop	1	InnoDB	latin1_swedish_ci	16 K1B	-
<b>12 tables</b>	<b>Sum</b>	<b>30</b>	<b>InnoDB</b>	<b>latin1_swedish_ci</b>	<b>272 K1B</b>	<b>0 B</b>

Buttons: Check All / Uncheck All, With selected: [v]

Print view, Data Dictionary, Create table

Name: [ ] Number of columns: [ ]

phpMyAdmin interface showing the query results for the 'tchreg' table.

Database: db, Table: tchreg

Showing rows 0 - 2 (~3 total), Query took 0.0022 sec

```
SELECT *
FROM `tchreg`
LIMIT 0, 30
```

Buttons: Profiling [Inline], [Edit], [Explain SQL], [Create PHP Code], [Refresh]

Show: Start row: 0, Number of rows: 30, Headers every: 100 rows

Sort by key: None

Options: [v]

	SI	Fname	Lname	Uname	Contact	T_Id	Email	Gender	Add	password	Image	Thought	Sec1	Sec2
<input type="checkbox"/> Edit Copy Delete	1	mob	phy	mob	2343512354	123	mob123@gmail.com	Male	Earth Blue planet	692fa2f8a89357ac4f8a89faa61e0368	[BLOB - 12.3 KiB]	Thoughts dont Speak oh yeah	2510c39011c5be704182423e3a695e91	e358efa489
<input type="checkbox"/> Edit Copy Delete	2	siatama	tarnani	saitarna	123141242	21	saitarna@gmail.com	Male	Planets	3bccba45b8189496b768338ccce44e37	[BLOB - 6.7 KiB]	NULL	funny	strong
<input type="checkbox"/> Edit Copy Delete	4	helo	helo	helo	3425234154	12	ewwa	Male	ear	202cb962ac59075b964b07152d2234b70	[BLOB - 49.6 KiB]	NULL		NULL

Buttons: Check All / Uncheck All, With selected: [v], Change, Delete, Export

Show: Start row: 0, Number of rows: 30, Headers every: 100 rows

Query results operations: [v]

Buttons: Print view, Print view (with full texts), Export, Display chart, Create view

## ATTENDANCE RECORD

Roll_no	Semester	Subject	Month	Total_Lect	Lec_Attended	Overall_Atten...
12	3rd Sem	Computer S...	September	67	54	80.6
12	3rd Sem	Physics	March	56	34	60.71



## CONCLUSION

The Student Database Management system is a user-friendly way that seamlessly deals and maintains all kinds of details or information regarding student's info, academic progress and the institution. The software can be used in educational institution and colleges where the manual system work done will be upgraded to automation system and not replaces any working agents in that field but instead helps him/her and provide more simplicity and ease of doing work.

It is easy to use and simple to understand, the system is robust and flexible that can quickly perform any operations within a second. It keeps the data or files in ordered, consistent state and improves time flexibility and helps the user with easy navigations and in a systematic way.

Although all the objectives have been met, the system will always have a room for improvement to have better functionalities and more flexibility.

**{ CODE }**

### **1.Connection ( 'MyConnection.java' ):**

```
package Connection;
import java.sql.Connection;
import java.sql.DriverManager;

public class MyConnection {
public static Connection getConnection(){
Connection con = null;
try {
Class.forName("com.mysql.jdbc.Driver");
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/db","root","");
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
return con;
}
}
```

---

### **2. Log-in ( Student section- 'Login.java' ):**

```
package LoginPage;

import Connection.MyConnection;
import Registration.stu_reg;
import Security.loading;
import java.awt.Color;
import java.awt.Cursor;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JFrame;
import javax.swing.JOptionPane;

public Login() {
initComponents();
jButton1.setBackground(new Color(0,0,0,0));
jButton3.setBackground(new Color(0,0,0,0));
uname.setBackground(new Color(0,0,0,0));
pass.setBackground(new Color(0,0,0,0));
}
```

```
/*Action Command : */
```

```
private void unameFocusGained(java.awt.event.FocusEvent evt) {
uname.setText("");
}

private void passFocusGained(java.awt.event.FocusEvent evt) {
pass.setText("");
}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
teacher_login t = new teacher_login();
t.setVisible(true);
setVisible(false);
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
stu_reg st = new stu_reg();
st.setVisible(true);
setVisible(false);
}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
PreparedStatement ps = null;
ResultSet rs = null;
Connection con = null;
String j=String.valueOf(pass.getPassword());

if(uname.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Fill up your Username!");
}
else if( j.equals(""))
{
JOptionPane.showMessageDialog(null, "Enter your Password");
} else {
String sq= "SELECT * FROM `stdreg` WHERE `Uname`=? AND `password`=?" ;
try {
ps= MyConnection.getConnection().prepareStatement(sq);
ps.setString(1, uname.getText());
ps.setString(2, Registration.teach_reg.encrypt(j));
rs=ps.executeQuery();
if(rs.next()){
JOptionPane.showMessageDialog(null,"Successfull, WELCOME!");
loading l = new loading();
l.str();
l.setVisible(true);
this.setVisible(false);
} else {
JOptionPane.showMessageDialog(null,"Please re-check the values!");
}

} catch (SQLException ex) {

JOptionPane.showMessageDialog(null, ex);
}
}

private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
jButton1.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void jLabel7MouseClicked(java.awt.event.MouseEvent evt) {
```

```

jLabel7.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void jLabel7MouseClicked(java.awt.event.MouseEvent evt) {
this.setVisible(false);
}
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
setState(JFrame.ICONIFIED);
}
private void jButton3MouseMoved(java.awt.event.MouseEvent evt) {
jButton3.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}
}

```

---

### 3. Teacher Login section ( 'teacher\_login.java' ):

```

package LoginPage;

import Connection.MyConnection;
import Registration.teach_reg;
import Security.Progressbar;
import java.awt.Color;
import java.awt.Cursor;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JFrame;
import javax.swing.JOptionPane;

public teacher_login() {
initComponents();
pas.setBackground(new Color(0,0,0,0));
uname.setBackground(new Color(0,0,0,0));
jButton5.setBackground(new Color(0,0,0,0));
jButton3.setBackground(new Color(0,0,0,0));
minus.setBackground(new Color(0,0,0,0));
can.setBackground(new Color(0,0,0,0));
}

/*Action Command*/

private void unameFocusGained(java.awt.event.FocusEvent evt) {
uname.setText("");
}
private void pasFocusGained(java.awt.event.FocusEvent evt) {
pas.setText("");
}

```

```

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
teach_reg tr = new teach_reg();
tr.setVisible(true);
setVisible(false);
}
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
PreparedStatement ps = null;
ResultSet rs = null;
Connection con = null;
String hs = String.valueOf(pas.getPassword());
if(uname.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Fill up your Username");
}
else if(hs.equals(""))
{
JOptionPane.showMessageDialog(null, "Enter your Password");
}
else{
String sq ="SELECT * FROM `tchreg` WHERE Uname=? AND password=?";
try {
ps=MyConnection.getConnection().prepareStatement(sq);
ps.setString(1, uname.getText());
ps.setString(2, teach_reg.encrypt(hs));
rs=ps.executeQuery();

if (rs.next()){
JOptionPane.showMessageDialog(null, "Successfull, Welcome!");
Progressbar pg = new Progressbar();
pg.Setep();
pg.setVisible(true);
this.setVisible(false);
} else{
JOptionPane.showMessageDialog(null, "Please Re-check the Values!");
}
}
catch (SQLException ex) {
Logger.getLogger(teacher_login.class.getName()).log(Level.SEVERE, null, ex);
}
}
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
Login j = new Login();
j.setVisible(true);
setVisible(false);
}
private void jButton3MouseMoved(java.awt.event.MouseEvent evt) {
jButton3.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}
private void jButton3MouseClicked(java.awt.event.MouseEvent evt) {
Security.forgetpswd pa = new Security.forgetpswd();
pa.setVisible(true);
this.setVisible(false);
}
private void jButton5MouseMoved(java.awt.event.MouseEvent evt) {
jButton5.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

```

```

private void minusMouseClicked(java.awt.event.MouseEvent evt) {
minus.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}
private void canMouseClicked(java.awt.event.MouseEvent evt) {
can.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}
private void minusActionPerformed(java.awt.event.ActionEvent evt) {
setState(JFrame.ICONIFIED);
}
private void canActionPerformed(java.awt.event.ActionEvent evt) {
this.setVisible(false);
}
private void jButton4MouseClicked(java.awt.event.MouseEvent evt) {
jButton4.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}
private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
jButton1.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}
}

```

---

#### 4. Student Registration ( 'stu\_reg.java'):

```

package Registration;

import Connection.MyConnection;
import LoginPage.Login;
import java.awt.Color;
import java.awt.Cursor;
import java.awt.Image;
import java.awt.event.KeyEvent;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStream;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.text.SimpleDateFormat;
import javax.swing.ImageIcon;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;

public class stu_reg extends javax.swing.JFrame {

    PreparedStatement ps = null;
    Connection con = null;
    ResultSet rs = null;
    String filename=null;
    byte[] person_image=null;

    public stu_reg() {

```

```

initComponents();
jPanel1.setBackground(new Color(0,0,0,0));
uname.setBackground(new Color(0,0,0,0));

mi.setBackground(new Color(0,0,0,0));
ca.setBackground(new Color(0,0,0,0));
Back.setBackground(new Color(0,0,0,0));

fname.setBackground(new Color(0,0,0,0));
roll.setBackground(new Color(0,0,0,0));
contact.setBackground(new Color(0,0,0,0));
lname.setBackground(new Color(0,0,0,0));
add.setBackground(new Color(0,0,0,0));

password.setBackground(new Color(0,0,0,0));
bdate.setBackground(new Color(0,0,0,0));
deps();
}

```

### /\***Methods**\*/

```

public void deps(){
String o ="Select * FROM `dept`";
try {
ps=MyConnection.getConnection().prepareStatement(o);
rs=ps.executeQuery();
while(rs.next()){
String k=rs.getString("Dept_name");
Dept.addItem(k);
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

```

### /\***Action Command**\*/

```

private void fnameFocusGained(java.awt.event.FocusEvent evt) {
fname.setText("");
}

private void lnameFocusGained(java.awt.event.FocusEvent evt) {
lname.setText("");
}

private void rollFocusGained(java.awt.event.FocusEvent evt) {
roll.setText("");
}

private void contactFocusGained(java.awt.event.FocusEvent evt) {
contact.setText("");
}

private void addFocusGained(java.awt.event.FocusEvent evt) {
add.setText("");
}

```

```

private void uploadActionPerformed(java.awt.event.ActionEvent evt) {

JFileChooser chooser = new JFileChooser();
chooser.showOpenDialog(null);
File f =chooser.getSelectedFile();
filename =f.getAbsolutePath();
ImageIcon imageIcon = new ImageIcon(new
ImageIcon(filename).getImage().getScaledInstance(picture.getWidth(),picture.getHeight(),
Image.SCALE_SMOOTH));
picture.setIcon(imageIcon);
try {
File image = new File(filename);
FileInputStream fis = new FileInputStream(image);
ByteArrayOutputStream bos = new ByteArrayOutputStream();
byte[] buf = new byte[1024];
for(int readNum;(readNum=fis.read(buf))!=-1;){
bos.write(buf,0,readNum);
}
person_image=bos.toByteArray();

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
String psa = String.valueOf(password.getPassword());
String sq ="INSERT INTO `stdreg`(`Fname`,`Lname`,`Uname`,`Rollno`,`Gender`,`Contact`,`Course`,`Address`,`Bdate`,`password`,`Image`) VALUES (?,?,?,?,?,?,?,?,?,?)";

if(fname.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Add your first name");
}
else if(lname.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Add your last name");
}
else if(uname.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Add your user name");
}
else if(roll.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Add your roll no");
}
else if(contact.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Add your Contact no");
}
else if(add.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Add your address location");
}
else if(psa.equals(""))
{

```



```

JOptionPane.showMessageDialog(null, "Add your password");
}
else{
try {

con=DriverManager.getConnection("jdbc:mysql://localhost:3306/db","root","");
ps=(PreparedStatement) con.prepareStatement(sq);

ps.setString(1, fname.getText());
ps.setString(2, lname.getText());
ps.setString(3, uname.getText());
ps.setString(4, roll.getText());
String gender = null;
if(male.isSelected()){
gender="Male";
}
if(female.isSelected()){
gender="Female";
}
ps.setString(5, gender);

ps.setString(6, contact.getText());

String cours;
cours=Dept.getSelectedItem().toString();
ps.setString(7, cours);

ps.setString(8, add.getText());

SimpleDateFormat df = new SimpleDateFormat("yyyy-MM-dd");
String date = df.format(bdate.getDate());
ps.setString(9, date);

ps.setString(10, teach_reg.encrypt(psa));
ps.setBytes(11, person_image);
ps.executeUpdate();
JOptionPane.showMessageDialog(null, "New Student added Successfully!");

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

private void BackActionPerformed(java.awt.event.ActionEvent evt) {
Login loh = new Login();
loh.setVisible(true);
setVisible(false);
}

private void contactKeyPressed(java.awt.event.KeyEvent evt) {
String can = contact.getText();
int length = can.length();
char a =evt.getKeyChar();

if(evt.getKeyChar()>='0' && evt.getKeyChar()<='9'){
if(length < 10){

```

```

contact.setEditable(true);
}else{
contact.setEditable(false);
}
}else{
if(evt.getExtendedKeyCode()==KeyEvent.VK_BACK_SPACE){
contact.setEditable(true);
} else{
contact.setEditable(false);
}
}
}

private void miActionPerformed(java.awt.event.ActionEvent evt) {
setState(stu_reg.ICONIFIED);
}

private void caActionPerformed(java.awt.event.ActionEvent evt) {
this.setVisible(false);
}

private void BackMouseClicked(java.awt.event.MouseEvent evt) {
Back.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void caMouseClicked(java.awt.event.MouseEvent evt) {
ca.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void miMouseClicked(java.awt.event.MouseEvent evt) {
mi.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void uploadMouseClicked(java.awt.event.MouseEvent evt) {
upload.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}
}

```

---

## 5. Teacher Registration ( 'teach\_reg.java' ):

```

package Registration;

import LoginPage.teacher_login;
import java.awt.Color;
import java.awt.Cursor;
import java.awt.Image;
import java.awt.event.KeyEvent;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileInputStream;

```

```

import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.ImageIcon;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;

/**
 *
 * @author bei
 */
public class teach_reg extends javax.swing.JFrame {
    Connection con =null;
    PreparedStatement ps =null;
    ResultSet rs = null;
    String gender;
    String filename=null;
    byte[] person_image=null;

    public teach_reg() {
        initComponents();
        Id.setBackground(new Color(0,0,0,0));

        bak.setBackground(new Color(0,0,0,0));
        minu.setBackground(new Color(0,0,0,0));
        canc.setBackground(new Color(0,0,0,0));

        add.setBackground(new Color(0,0,0,0));
        fname.setBackground(new Color(0,0,0,0));
        lname.setBackground(new Color(0,0,0,0));
        uname.setBackground(new Color(0,0,0,0));
        contact.setBackground(new Color(0,0,0,0));
        male.setBackground(new Color(0,0,0,0));
        female.setBackground(new Color(0,0,0,0));
        add.setBackground(new Color(0,0,0,0));
        email.setBackground(new Color(0,0,0,0));
        repass.setBackground(new Color(0,0,0,0));
        pas.setBackground(new Color(0,0,0,0));
    }

    /*Action Command*/

    private void lnameFocusGained(java.awt.event.FocusEvent evt) {
        lname.setText(null);
    }

    private void unameFocusGained(java.awt.event.FocusEvent evt) {
        uname.setText(null);
    }

```

```

private void contactFocusGained(java.awt.event.FocusEvent evt) {
contact.setText(null);
}

private void addFocusGained(java.awt.event.FocusEvent evt) {
add.setText(null);
}

private void pasFocusGained(java.awt.event.FocusEvent evt) {
pas.setText(null);
}

private void repassFocusGained(java.awt.event.FocusEvent evt) {
repass.setText(null);
}
private void fnameFocusGained(java.awt.event.FocusEvent evt) {
fname.setText(null);
}
private void bakActionPerformed(java.awt.event.ActionEvent evt) {
teacher_login tl = new teacher_login();
tl.setVisible(true);
setVisible(false);
}
private void emailFocusGained(java.awt.event.FocusEvent evt) {
email.setText(null);
}
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
String s =String.valueOf(pas.getPassword());
String r = String.valueOf(repass.getPassword());

String sq ="INSERT INTO `tchreg`(`Fname`,`Lname`,`Uname`,`Contact`,`T_Id`,`Email`,`Gender`,`Add`,`password`,`Image`) VALUES (?,?,?,?,?,?,?,?,?)";

if(pic.equals("")){
JOptionPane.showMessageDialog(null, "Your image is missing");
}
else if(fname.getText().equals("")){
JOptionPane.showMessageDialog(null, "Fill up the first name");
}

else if(lname.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Fill up the last name");
}
else if (uname.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Fill up the user name");
}
else if(contact.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Fill up the contact no");
}
else if(Id.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Fill up the ID no");
}
else if(email.getText().equals(""))

```

```

{
JOptionPane.showMessageDialog(null, "Fill up ur emial Account");
}
else if(add.getText().equals(""))
{
JOptionPane.showMessageDialog(null, "Fill up ur address location");
}

else if(s.equals(""))
{
JOptionPane.showMessageDialog(null, "Fill up ur password");
}
else if(!s.equals(r))
{
JOptionPane.showMessageDialog(null, "Retype up ur password");
}

else{

try {

con=DriverManager.getConnection("jdbc:mysql://localhost:3306/db","root","");
ps=(PreparedStatement) con.prepareStatement(sq);

ps.setString(1, fname.getText());
ps.setString(2, lname.getText());
ps.setString(3, uname.getText());
ps.setString(4, contact.getText());
ps.setString(5, Id.getText());

ps.setString(6, email.getText());

String Gender = null;
if(male.isSelected()){
Gender="Male";
}
if(female.isSelected()){
Gender="Female";
}
ps.setString(7, Gender);
ps.setString(8, add.getText());
ps.setString(9, encrypt(s));
ps.setBytes(10, person_image);

if(ps.executeUpdate(>0){
JOptionPane.showMessageDialog(null, "Successfully inserted into Database!");
}else{
JOptionPane.showMessageDialog(null, "Fill all the Forms!");
}

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

```

```

private void UploadActionPerformed(java.awt.event.ActionEvent evt) {
JFileChooser chooser = new JFileChooser();
chooser.showOpenDialog(null);
File f =chooser.getSelectedFile();
filename =f.getAbsolutePath();
ImageIcon imageIcon = new ImageIcon(new
ImageIcon(filename).getImage().getScaledInstance(pic.getWidth(),pic.getHeight(),
Image.SCALE_SMOOTH));
pic.setIcon(imageIcon);
try {
File image = new File(filename);
FileInputStream fis = new FileInputStream(image);
ByteArrayOutputStream bos = new ByteArrayOutputStream();
byte[] buf = new byte[1024];
for(int readNum;(readNum=fis.read(buf))!=-1;){
bos.write(buf,0,readNum);
}
person_image=bos.toByteArray();

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void contactKeyPressed(java.awt.event.KeyEvent evt) {
String can = contact.getText();
int length = can.length();
char a =evt.getKeyChar();

if(evt.getKeyChar()>='0' && evt.getKeyChar()<='9'){
if(length < 10){
contact.setEditable(true);
}else{
contact.setEditable(false);
}
}else{
if(evt.getExtendedKeyCode()==KeyEvent.VK_BACK_SPACE){
contact.setEditable(true);
}else{
contact.setEditable(false);
}
}
}

private void bakMouseMoved(java.awt.event.MouseEvent evt) {
bak.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void minuMouseMoved(java.awt.event.MouseEvent evt) {
minu.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void cancMouseMoved(java.awt.event.MouseEvent evt) {
canc.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

```

```

private void UploadMouseClicked(java.awt.event.MouseEvent evt) {
Upload.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void minuActionPerformed(java.awt.event.ActionEvent evt) {
setState(teach_reg.ICONIFIED);
}

private void cancActionPerformed(java.awt.event.ActionEvent evt) {
this.setVisible(false);
}

```

### **/\*The code for encrypting the database method\*/**

```

public static String encrypt(String input)
{
try {
MessageDigest md = MessageDigest.getInstance("MD5");
byte[] messageDigest = md.digest(input.getBytes());
BigInteger no = new BigInteger(1, messageDigest);
String hashtext = no.toString(16);
while (hashtext.length() < 32) {
hashtext = "0" + hashtext;
}
return hashtext;
}
catch (NoSuchAlgorithmException e) {
throw new RuntimeException(e);
}
}

```

---

## **6. Loading screen for Staff ('Progressbar.java'):**

```

package Security;

import Home_pag1.Home1;
import javax.swing.plaf.basic.BasicProgressbarUI;

public class Progressbar extends javax.swing.JFrame implements Runnable{

Thread th;
public Progressbar() {
initComponents();
th=new Thread((Runnable)this);
}
public void Setep(){
this.setVisible(true);
th.start();
}
public void run(){
try {
for(int i=0; i<=100; i++){

```

```

pp.setText(+i+" %");
load.setValue(i);
load.setStringPainted(true);

load.setUI(new BasicProgressBarUI());
if(i == 99){
Home1 h = new Home1();
h.show();
setVisible(false);
}Thread.sleep(40);
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

```

---

## 7. Loading Screen for Student Section ( 'loading.java' ):

```

package Security;

import Home_pag1.Home;
import javax.swing.JOptionPane;

public class loading extends javax.swing.JFrame implements Runnable{

Thread t;

public loading() {
initComponents();
t=new Thread((Runnable)this);
}

public void str(){
this.setVisible(true);
t.start();
}
public void run(){
try {
for(int x=0; x<=100; x++){

io.setText("Please wait, Loading ---");

if(x == 99){
Home s = new Home();
s.show();
this.setVisible(false);
}
}Thread.sleep(100);

} catch (InterruptedException e) {

```



```

JOptionPane.showMessageDialog(null, e);
}

}

```

---

## 8. Security Section A ( ‘privacys.java’ ):

```

package Security;

import Registration.teach_reg;
import java.awt.Color;
import java.awt.Cursor;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import javax.swing.JOptionPane;

public class privacys extends javax.swing.JFrame {

    public privacys() {
        initComponents();
        first.setBackground( new Color(0,0,0,0));
        second.setBackground( new Color(0,0,0,0));
        seid.setBackground( new Color(0,0,0,0));

    }

    Connection con=null;
    ResultSet rs = null;
    PreparedStatement ps = null;

    private void mouMouseClicked(java.awt.event.MouseEvent evt) {
        mou.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
    }

    private void mouMouseClicked(java.awt.event.MouseEvent evt) {
        this.setVisible(false);
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

        String u ="UPDATE `tchreg` SET `Sec1`=?,`Sec2`=? WHERE `T_Id`=?" ;
        try {
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/db","root","");
            ps=con.prepareStatement(u);
            ps.setString(1, teach_reg.encrypt(first.getText()));
            ps.setString(2, teach_reg.encrypt(second.getText()));
            ps.setString(3, seid.getText());
            if(ps.executeUpdate()>0){

```

```

JOptionPane.showMessageDialog(null, "Security has been successfully recorded");
this.setVisible(false);
}

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

```

## 9. Security Section B ( 'forgetpswd.java' ):

```

package Security;

import Registration.teach_reg;
import java.awt.Color;
import java.awt.Cursor;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import javax.swing.JOptionPane;

public class forgetpswd extends javax.swing.JFrame {

    public forgetpswd() {
        initComponents();
        firsts.setBackground(new Color(0,0,0,0));
        use.setBackground(new Color(0,0,0,0));
        seconds.setBackground(new Color(0,0,0,0));
    }

    Connection con=null;
    ResultSet rs =null;
    PreparedStatement ps =null;

    private void mousMouseClicked(java.awt.event.MouseEvent evt) {
        mous.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

        String j = "SELECT * FROM `tchreg` WHERE `T_Id`=? AND `Sec1`=? AND `Sec2`=? ";
        try {
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/db","root","");
            ps=con.prepareStatement(j);
            ps.setString(1,use.getText());
            ps.setString(2, teach_reg.encrypt(firsts.getText()));
            ps.setString(3, teach_reg.encrypt(seconds.getText()));
            rs=ps.executeQuery();
            if(rs.next()){
                JOptionPane.showMessageDialog(null, "VERIFIED !");
                Reset r = new Reset();
                r.setVisible(true);
            }
        } catch (Exception e) {
            JOptionPane.showMessageDialog(null, e);
        }
    }
}

```

```

this.setVisible(false);
}else{
JOptionPane.showMessageDialog(null, "The Values does not match!");
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void mousMouseClicked(java.awt.event.MouseEvent evt) {
LoginPage.teacher_login l = new LoginPage.teacher_login();
l.setVisible(true);
this.setVisible(false);
}

```

## 10. Security Section C ( ‘Reset.java’ ):

```

package Security;

import Registration.teach_reg;
import java.awt.Color;
import java.awt.Cursor;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import javax.swing.JOptionPane;

public class Reset extends javax.swing.JFrame {

    public Reset() {
        initComponents();
        repass.setBackground(new Color(0,0,0,0));
        pass.setBackground(new Color(0,0,0,0));
    }

    Connection con=null;
    ResultSet rs =null;
    PreparedStatement ps =null;

    private void canMouseMoved(java.awt.event.MouseEvent evt) {
        can.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
    }

    private void confActionPerformed(java.awt.event.ActionEvent evt) {
        String j =String.valueOf(pass.getPassword());
        String l = String.valueOf(repass.getPassword());
        if(j.equals("")){
            JOptionPane.showMessageDialog(null, "Fill up your password");
        }

        else if(l.equals(""))
        {
            JOptionPane.showMessageDialog(null, "Enter you confirm password");
        } else{
            if(!j.equals(l)){

```

```

JOptionPane.showMessageDialog(null, "Password does not match please re-check!");
} else if(j.equals(1)){

String ud="UPDATE `tchreg` SET `password`=? WHERE `T_Id`=?";

try {
Class.forName("com.mysql.jdbc.Driver");
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/db","root","");
ps=con.prepareStatement(ud);

ps.setString(1, teach_reg.encrypt(j));
ps.setString(2, forgetpswd.use.getText());
ps.executeUpdate();
JOptionPane.showMessageDialog(null, "Your Password has been updated Successfully!");
LoginPage.teacher_login tl = new LoginPage.teacher_login();
tl.setVisible(true);
this.setVisible(false);

}
catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}

} else{
JOptionPane.showMessageDialog(null, "NO value enter in the field!");
}
}
}

private void canMouseClicked(java.awt.event.MouseEvent evt) {
LoginPage.teacher_login tr = new LoginPage.teacher_login();
tr.setVisible(true);
this.setVisible(false);
}

```

---

## 11. Student Home page ( 'Home.java' ):

```

package Home_pag1;

import Connection.MyConnection;
import LoginPage.Login;
import java.awt.Color;
import java.awt.Cursor;
import static java.lang.Thread.sleep;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.text.SimpleDateFormat;
import java.time.Year;

```

```

import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Date;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.RowFilter;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableRowSorter;

public class Home extends javax.swing.JFrame {

    public Home() {
        initComponents();
        setTime();
        Find();
        DisplayRecord();
        Sum();
        deptvw();

        hidea();
        bakk.setBackground(new Color(0,0,0,0));
        text.setBackground(new Color(0,0,0,0));
        bakk1.setBackground(new Color(0,0,0,0));
        log.setBackground(new Color(0,0,0,0));
        roll.setBackground(new Color(0,0,0,0));
        action.setBackground(new Color(0,0,0,0));
        jButton4.setBackground(new Color(0,0,0,0));
        jButton5.setBackground(new Color(0,0,0,0));
        roll.setBackground(new Color(0,0,0,0));
        dds.setBackground(new Color(0,0,0,0));
        setExtendedState(JFrame.MAXIMIZED_BOTH);
    }

```

### **/\*Methods\*/**

```

public void hidea(){
    jPanel2.hide();
}

public void showab(){
    jPanel2.show();
}

public void setTime(){
    Thread ti = new Thread()
    {
        public void run()
        {
            try {
                while(true){
                    Date d = new Date();
                    SimpleDateFormat sf = new SimpleDateFormat("h : mm : ss a");
                    String dt = sf.format(d);
                    jLabel4.setText(dt);
                    sleep(1000);
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
}

```

```

}
}
};
ti.start();
}

```

```

PreparedStatement ps;
ResultSet rs;
Connection con;
DefaultTableModel model;

```

### **/\*Action Commands\*/**

```

private void jComboBox1ItemStateChanged(java.awt.event.ItemEvent evt) {
String s=jComboBox1.getSelectedItem().toString();
Filt(s);
}

```

```

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(3);
}

```

```

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(2);
}

```

```

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(1);
}

```

```

private void actionActionPerformed(java.awt.event.ActionEvent evt) {

```

```

String t="SELECT * FROM `records` WHERE Year=? AND roll_no=? ";
try {
ps=MyConnection.getConnection().prepareStatement(t);

```

```

DateTimeFormatter yr = DateTimeFormatter.ofPattern("yyyy");
String y = yr.format(Year.of(yearr.getYear()));
ps.setString(1, y);
ps.setString(2, dds.getText());
rs=ps.executeQuery();

```

```

if(rs.next()){
JOptionPane.showMessageDialog(null,"Matched, Welcome!");
jTabbedPane1.setSelectedIndex(5);
DisplayRecord();

```

```

}else{
JOptionPane.showMessageDialog(null,"Invalid Roll-no plz Re-check!");
}

```

```

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

```

```

private void jComboBox2ItemStateChanged(java.awt.event.ItemEvent evt) {

```

```

String o = jComboBox2.getSelectedItem().toString();
Filt2(o);
}
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
this.dispose();
}

private void jButton4MouseClicked(java.awt.event.MouseEvent evt) {
jButton4.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void jButton5MouseClicked(java.awt.event.MouseEvent evt) {
jButton5.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
setState(Home.ICONIFIED);
}

private void logActionPerformed(java.awt.event.ActionEvent evt) {

String t ="SELECT * FROM `attendance` WHERE Year=? AND Roll_no=? ";
try {
ps=MyConnection.getConnection().prepareStatement(t);

DateTimeFormatter yr = DateTimeFormatter.ofPattern("yyyy");
String y = yr.format(Year.of(year.getYear()));
ps.setString(1, y);
ps.setString(2, roll.getText());
rs=ps.executeQuery();

if (rs.next()){
JOptionPane.showMessageDialog(null,"Matched, Welcome!");
jTabbedPane1.setSelectedIndex(4);
Find();

} else{
JOptionPane.showMessageDialog(null,"Recheck your roll no and Enroll year!");
}

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void babbyMouseClicked(java.awt.event.MouseEvent evt) {
babby.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void recordMouseClicked(java.awt.event.MouseEvent evt) {
record.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void attendMouseClicked(java.awt.event.MouseEvent evt) {
attend.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

```

```

private void fakultyMouseClicked(java.awt.event.MouseEvent evt) {
fakulty.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void boboMouseClicked(java.awt.event.MouseEvent evt) {
bobo.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void poMouseClicked(java.awt.event.MouseEvent evt) {
po.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void recordMouseClicked(java.awt.event.MouseEvent evt) {
jTabbedPane1.setSelectedIndex(1);
}

private void babby1MouseClicked(java.awt.event.MouseEvent evt) {
babby1.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void babbyMouseClicked(java.awt.event.MouseEvent evt) {
jTabbedPane1.setSelectedIndex(0);
}

private void actionMouseClicked(java.awt.event.MouseEvent evt) {
action.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void logMouseClicked(java.awt.event.MouseEvent evt) {
log.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void babby1MouseClicked(java.awt.event.MouseEvent evt) {
jTabbedPane1.setSelectedIndex(0);
}

private void babby2MouseClicked(java.awt.event.MouseEvent evt) {
babby2.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void babby2MouseClicked(java.awt.event.MouseEvent evt) {
jTabbedPane1.setSelectedIndex(0);
}

private void bakkMouseClicked(java.awt.event.MouseEvent evt) {
bakk.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void bakkActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(2);
}

private void bakk1MouseClicked(java.awt.event.MouseEvent evt) {
bakk1.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void bakk1ActionPerformed(java.awt.event.ActionEvent evt) {

```



```

jTabbedPane1.setSelectedIndex(1);
}

private void attendMouseClicked(java.awt.event.MouseEvent evt) {
jTabbedPane1.setSelectedIndex(2);
}

private void fakultyMouseClicked(java.awt.event.MouseEvent evt) {
jTabbedPane1.setSelectedIndex(3);
}

private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
jButton1.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
jButton2.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void jButton3MouseClicked(java.awt.event.MouseEvent evt) {
jButton3.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void boboMouseClicked(java.awt.event.MouseEvent evt) {
Login l = new Login();
l.setVisible(true);
this.setVisible(false);
}

private void poMouseClicked(java.awt.event.MouseEvent evt) {
showab();
}

private void jLabel26MouseClicked(java.awt.event.MouseEvent evt) {
hidea(); }

```

### **/\*Methods for calling, connecting, displaying & Filtering the Database\*/**

```

public ArrayList<heo> dd(){
ArrayList<heo> ros = new ArrayList<>();
String qa="SELECT * FROM `dept`";
try {
ps=MyConnection.getConnection().prepareStatement(qa);
rs=ps.executeQuery();
heo hs;

while(rs.next()){
hs=new heo(rs.getInt("Dept"),rs.getString("Dept_name"));
ros.add(hs);
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
return(ros);
}

public void deptvw(){
ArrayList<heo> List = dd();

```

```

model = new DefaultTableModel();
model.setColumnIdentifiers(new Object[]{"Dept_ID", "Dept_name"});

```

```

Object[] row = new Object[2];
for (int i=0; i<List.size();i++){
row[0]=List.get(i).getDid();
row[1]=List.get(i).getDname();
model.addRow(row);
}
facul.setModel(model);
}

```

```

public int Sum(){

double total=0,t = 0;
for(int i=0; i<jTable1.getRowCount();i++)
{
double to=Double.parseDouble(jTable1.getValueAt(i, 6).toString());
total+=to;
double f=jTable1.getRowCount();
t=total/f;
}
perc.setText(String.valueOf(String.format("%.2f",t)+"%"));
return(0);
}

```

```

public void Filt2(String q){
TableRowSorter<DefaultTableModel> t = new TableRowSorter<DefaultTableModel>(model);
jTable2.setRowSorter(t);

```

```

if (q != "Show All Sem")
{
t.setRowFilter(RowFilter.regexFilter(q));
Sum();
} else
{
jTable1.setRowSorter(t);
Sum();
}
}

```

```

public void Filt(String q){
TableRowSorter<DefaultTableModel> t = new TableRowSorter<DefaultTableModel>(model);
jTable1.setRowSorter(t);

```

```

if(q != "Show All Sem")
{
t.setRowFilter(RowFilter.regexFilter(q));

```

```

Sum();
}else

```

```

{
jTable1.setRowSorter(t);

```

```

Sum();
}

```

```

}

public ArrayList<rec> Forum(String jas){
ArrayList<rec> record = new ArrayList<>();
try {
String r ="SELECT * FROM `records` WHERE roll_no=?";
ps=MyConnection.getConnection().prepareStatement(r);
ps.setString(1, dds.getText());
rs=ps.executeQuery();
rec a;
while(rs.next()){
a = new rec (rs.getInt("SI"), rs.getInt("Year"), rs.getInt("roll_no"), rs.getString("Subject"),
rs.getString("Semester"), rs.getInt("T_marks"), rs.getInt("Score") );
record.add(a);
}

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
return(record);
}

public void DisplayRecord(){
ArrayList<rec> j =Forum(dds.getText());
model = new DefaultTableModel();
model.setColumnIdentifiers(new Object[]{"Roll_no","Subject","Semester","Total_Marks","Score"});
Object[] d = new Object[5];
for(int i =0; i<j.size();i++){

d[0]=j.get(i).getRoll();
d[1]=j.get(i).getSubject();
d[2]=j.get(i).getSemester();
d[3]=j.get(i).getTotal();
d[4]=j.get(i).getScore();

model.addRow(d);
}
jTable2.setModel(model);
}

public ArrayList<user> List(String val){
ArrayList<user> listuser = new ArrayList<>();

try {
String s="SELECT * FROM `attendance` WHERE Roll_no =?";
ps=MyConnection.getConnection().prepareStatement(s);
ps.setString(1, roll.getText());
rs=ps.executeQuery();
user a;
while(rs.next()){
a = new user(rs.getInt("Roll_no"), rs.getString("Semester"), rs.getString("Subject"), rs.getString("Month"),
rs.getInt("Total_Lec"), rs.getInt("Lec_Attended"), rs.getFloat("Overall_Attendance"));
listuser.add(a);
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

```

```

    }
    return(listuser);
    }

    public void Find(){

        ArrayList<user> us = List(roll.getText());
        model = new DefaultTableModel();
        model.setColumnIdentifiers(new
        Object[] { "Roll_no", "Semester", "Subject", "Month", "Total_Lect", "Lec_Attended", "Overall_Attended" });
        Object[] row = new Object[7];
        for(int i=0; i<us.size();i++){
            row[0]=us.get(i).getRoll();
            row[1]=us.get(i).getSem();
            row[2]=us.get(i).getSubject();
            row[3]=us.get(i).getMonth();
            row[4]=us.get(i).getTl();
            row[5]=us.get(i).getLt();
            row[6]=us.get(i).getOa();

            model.addRow(row);

        }
        jTable1.setModel(model);

    }

```

---

## 12. Staff Home page ( 'Home1.java' ):

```

package Home_pag1;

import Connection.MyConnection;
import LoginPage.teacher_login;
import java.awt.Color;
import java.awt.Cursor;
import java.awt.Image;
import static java.lang.Thread.sleep;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.MessageFormat;
import java.text.SimpleDateFormat;
import java.time.Month;
import java.time.Year;

```

```

import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Date;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.RowFilter;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableModel;
import javax.swing.table.TableRowSorter;
import net.proteanit.sql.DbUtils;

public class Home1 extends javax.swing.JFrame {

    /*Variables */

    PreparedStatement ps = null;
    ResultSet rs = null;
    Connection con = null;
    DefaultTableModel model;
    String filename = null;
    byte[] person_image = null;

    /*Calling Function Methods to home */

    public Home1() {

        initComponents();
        setExtendedState(JFrame.MAXIMIZED_BOTH);
        deptview();
        setTime();
        FindUser();
        atnd();
        Recs();
        setTime();
        setDate();
        newma();
        DisplayRecord();
        course();
        lec();
        sub();
        Invs();
        FilterInventory();
        bat();
        plp();
        com();
        le();
        nam();
        roy();
        ser();
        cous();
        subj();
        ro();
        View();
        noac();
    }

```

```

depname();
newman();
pep();
noac();
menuhide();

exitt.setBackground(new Color(0,0,0,0));
minim.setBackground(new Color(0,0,0,0));
proid.setBackground(new Color(0,0,0,0));
ac.setBackground(new Color(0,0,0,0));
si.setBackground(new Color(0,0,0,0));

}

/* Methods & Functions */

public ArrayList<st> studenta() {
    ArrayList<st> n = new ArrayList<>();
    try {
        String i = "SELECT * FROM `stdreg`";
        ps = MyConnection.getConnection().prepareStatement(i);
        rs = ps.executeQuery();
        st s;
        while (rs.next()) {
            s = new st(rs.getString("Fname"), rs.getString("Lname"), rs.getString("Uname"), rs.getInt("Rollno"),
                rs.getString("Gender"),rs.getInt("Contact"), rs.getString("Course"), rs.getString("Address"),
                rs.getDate("Bdate"), rs.getString("password"), rs.getBytes("Image"));
            n.add(s);
        }

    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, e);
    }
    return (n);
}

public void View() {
    ArrayList<st> m = studenta();

    DefaultTableModel model = (DefaultTableModel) info.getModel();
    model.setColumnIdentifiers(new Object[]{"FName", "LName", "Rollno",
        "Gender","Contact","Course","Address"});
    Object[] row = new Object[7];
    for (int i = 0; i < m.size(); i++) {
        row[0] = m.get(i).getFn();
        row[1] = m.get(i).getLn();
        row[2] = m.get(i).getRollno();
        row[3] = m.get(i).getGen();
        row[4] = m.get(i).getContact();
        row[5] = m.get(i).getCou();
        row[6] = m.get(i).getAdd();

        model.addRow(row);
    }
    info.setModel(model);
}

```

```
private void ro() {
int row = lecta.getRowCount();
txt.setText(Integer.toString(row));
}
```

```
public void plp(){
    nameto.setText(teacher_login.uname.getText());
}
```

```
public void setTime(){
Thread ti = new Thread()
{
public void run()
{
try {
while(true){
Date d = new Date();
SimpleDateFormat sf = new SimpleDateFormat("h : mm : ss a");
String dt = sf.format(d);
tim.setText(dt);
sleep(1000);
}
} catch (Exception e) {
e.printStackTrace();
}
};
ti.start();
}
```

```
public void setDate(){
Thread ti = new Thread()
{
public void run()
{
try {
while(true){
Date da = new Date();
SimpleDateFormat saf = new SimpleDateFormat("EEE, dd-MMM");
String dta = saf.format(da);
Dat.setText(dta);
sleep(1000);
}
} catch (Exception e) {
e.printStackTrace();
}
};
ti.start();
}
```

```
private void roy(){
int row=tablecourse.getRowCount();
ID2.setText(Integer.toString(row));
}
```

```

private void com() {
String id = "SELECT * FROM `tchreg`";

try {
ps = MyConnection.getConnection().prepareStatement(id);
rs = ps.executeQuery();
while (rs.next()) {
String j = rs.getString("T_Id");
sid.addItem(j);
}

} catch (Exception e) {
}
}

private void subj() {
String n = "SELECT * FROM `course`";

try {
ps = MyConnection.getConnection().prepareStatement(n);
rs = ps.executeQuery();

while (rs.next()) {
String nas = rs.getString("Course_ID");
subcid.addItem(nas);
}

} catch (Exception e) {
}
}

private void pep() {
String n = "SELECT * FROM `dept`";
try {
ps = MyConnection.getConnection().prepareStatement(n);
rs = ps.executeQuery();

while (rs.next()) {
String nas = rs.getString("Dept");
pepsi.addItem(nas);
}

} catch (Exception e) {
}
}

private void newman() {
String n = "SELECT * FROM `dept`";
try {
ps = MyConnection.getConnection().prepareStatement(n);
rs = ps.executeQuery();

while (rs.next()) {
String dsa = rs.getString("Dept_name");
inde.addItem(dsa);
}
}
}

```



```

}

} catch (Exception e) {
}
}

private void ser() {
String n = "SELECT * FROM `dept`";
try {
ps = MyConnection.getConnection().prepareStatement(n);
rs = ps.executeQuery();

while (rs.next()) {
String dsa = rs.getString("Dept_name");
cop.addItem(dsa);
}

} catch (Exception e) {
}
}

private void newma() {
String n = "SELECT * FROM `dept`";
try {
ps = MyConnection.getConnection().prepareStatement(n);
rs = ps.executeQuery();

while (rs.next()) {
String dsu = rs.getString("Dept_name");
inde1.addItem(dsu);
}

} catch (Exception e) {
}
}

private void Recs() {
String n = "SELECT * FROM `dept`";
try {
ps = MyConnection.getConnection().prepareStatement(n);
rs = ps.executeQuery();

while (rs.next()) {
String r = rs.getString("Dept_name");
subject.addItem(r);
}

} catch (Exception e) {
}
}

private void cous() {
String n = "SELECT * FROM `course`";
try {
ps = MyConnection.getConnection().prepareStatement(n);
rs = ps.executeQuery();

while (rs.next()) {
String na = rs.getString("Course_ID");
refid1.addItem(na);
}
}

```

```

    } catch (Exception e) {
    }
}

private void depname() {
String n = "SELECT * FROM `dept`";
try {
ps = MyConnection.getConnection().prepareStatement(n);
rs = ps.executeQuery();

while (rs.next()) {
String dsa = rs.getString("Dept_name");
dn1.addItem(dsa);
}

} catch (Exception e) {
}
}

private void le() {
String l = "SELECT * FROM `tchreg`";
try {
ps = MyConnection.getConnection().prepareStatement(l);
rs = ps.executeQuery();

while (rs.next()) {
String j = rs.getString("T_Id");
lid1.addItem(j);
}

} catch (Exception e) {
}
}

private void nam() {
String n = "SELECT * FROM `tchreg`";
try {
ps = MyConnection.getConnection().prepareStatement(n);
rs = ps.executeQuery();

while (rs.next()) {
String nd = rs.getString("Fname");
lname.addItem(nd);
}

} catch (Exception e) {
}
}

public void atnd(){
String n = "SELECT * FROM `dept`";
try {
ps = MyConnection.getConnection().prepareStatement(n);
rs = ps.executeQuery();

while (rs.next()) {

```

```

String jh = rs.getString("Dept_name");
sub.addItem(jh);
}
} catch (Exception e) {
}
}

public void Filt(String query){
TableRowSorter<DefaultTableModel> t = new TableRowSorter<DefaultTableModel>(model);
jTable2.setRowSorter(t);

if (query != "Show All ")
{
t.setRowFilter(RowFilter.regexFilter(query));
}else
{
jTable2.setRowSorter(t);
}
}

public void Fils(String query){
TableRowSorter<DefaultTableModel> t = new TableRowSorter<DefaultTableModel>(model);
vivo.setRowSorter(t);
t.setRowFilter(RowFilter.regexFilter(query));

}

private void studentActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(1);
}

private void popMenuKeyPressed(javax.swing.event.MenuKeyEvent evt) {
JOptionPane.showMessageDialog(null, "Hello");
}

private void student1ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(8);
}

private void student2ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(3);
}

private void student3ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(2);
}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(9);
}

private void searchKeyReleased(java.awt.event.KeyEvent evt) {
String ser = search.getText();
String sql ="SELECT * FROM stdreg WHERE Rollno=?";
try {

ps = MyConnection.getConnection().prepareStatement(sql);
ps.setString(1, ser);

```

```

rs = ps.executeQuery();
if(rs.next()){
String st = rs.getString("Fname");
fn.setText(st);

String sa = rs.getString("Lname");
ln.setText(sa);

String r = rs.getString("Rollno");
rolno.setText(r);

String sex = rs.getString("Gender");
if(sex.equals("Male")){
male.setSelected(true);
}else{
female.setSelected(true);
}

String c = rs.getString("Contact");
contact.setText(c);

String j= rs.getString("Course");
cop.addItem(j);

String ad = rs.getString(" Address");
address.setText(ad);

byte[] img =rs.getBytes("Image");
ImageIcon imageIcon = new ImageIcon(new
ImageIcon(img).getImage().getScaledInstance(pic.getWidth(),pic.getHeight(), Image.SCALE_SMOOTH));
pic.setIcon(imageIcon);
person_image = img;
}

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void infoMouseClicked(java.awt.event.MouseEvent evt) {
int i =info.getSelectedRow();
TableModel model = info.getModel();
fn.setText(model.getValueAt(i, 0).toString());
ln.setText(model.getValueAt(i, 1).toString());
rolno.setText(model.getValueAt(i, 2).toString());

String gen = model.getValueAt(i, 3).toString();
if(gen.equals("Male")){
male.setSelected(true);
}
else {
female.setSelected(true);
}
contact.setText(model.getValueAt(i, 4).toString());

```

```

cop.setSelectedItem(info.getValueAt(i, 5).toString());
address.setText(model.getValueAt(i, 6).toString());

byte[] iag =(studenta().get(i).getImge());
ImageIcon imageIcon = new ImageIcon(new
ImageIcon(iag).getImage().getScaledInstance(pic.getWidth(),pic.getHeight(), Image.SCALE_SMOOTH));
pic.setIcon(imageIcon);
}

private void jButton12ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(4);
}

private void jButton13ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(0);
}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
String s = ("INSERT INTO `attendance`(`Year`, `Roll_no`, `Semester`, `Subject`, `Month`, `Total_Lec`,
`Lec_Attended`, `Overall_Attendance`) VALUES (?, ?, ?, ?, ?, ?, ?)");
try {

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/db", "root", "");
ps = (PreparedStatement) con.prepareStatement(s);

DateTimeFormatter yr = DateTimeFormatter.ofPattern("yyyy");
String y = yr.format(Year.of(year.getYear()));
ps.setString(1, y);

ps.setString(2, roll.getText());

String se;
se = sem.getSelectedItemAt().toString();
ps.setString(3, se);

String su;
su = sub.getSelectedItemAt().toString();
ps.setString(4, su);

DateTimeFormatter mn = DateTimeFormatter.ofPattern("MMMM");
String m = mn.format(Month.values()[month.getMonth()]);

ps.setString(5, m);
ps.setString(6, tl.getText());
ps.setString(7, la.getText());

Float n1, n2, res;
n1 = Float.parseFloat(tl.getText());
n2 = Float.parseFloat(la.getText());
res = n2 / n1 * 100;
String formattedString = String.format("%.02f", res);
result.setText(String.valueOf(formattedString));
ps.setString(8, result.getText());

if (ps.executeUpdate() > 0);
JOptionPane.showMessageDialog(null, "New Student added Successfully!");
}

```

```

    } catch (SQLException ex) {
JOptionPane.showMessageDialog(null, ex);
    }
    DefaultTableModel model = (DefaultTableModel) jTable2.getModel();
    model.setRowCount(0);
    FindUser();
    }

    private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
int de =jTable2.getSelectedRow();
String d =(jTable2.getModel().getValueAt(de, 0).toString());
String del = "DELETE FROM `attendance` WHERE SI="+d;
try {
ps = MyConnection.getConnection().prepareStatement(del);

ps.executeUpdate();
JOptionPane.showMessageDialog(null, "Deleted Successfully!");

    } catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
    }

    FindUser();
    }

    private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {

        int roww = jTable2.getSelectedRow();
String attenda =(jTable2.getModel().getValueAt(roww, 0).toString());
String up = "UPDATE `attendance` SET
`Year`=?, `Roll_no`=?, `Semester`=?, `Subject`=?, `Month`=?, `Total_Lec`=?, `Lec_Attended`=?, `Overall_Attenda
nce`=? WHERE SI=" +attenda;
try {
con = DriverManager.getConnection("jdbc:mysql://localhost:3306/db", "root", "");
ps = (PreparedStatement) con.prepareStatement(up);

DateTimeFormatter yr = DateTimeFormatter.ofPattern("yyyy");
String y = yr.format(Year.of(year.getYear()));
ps.setString(1, y);

ps.setString(2, roll.getText());

String se;
se = sem.getSelectedItem().toString();
ps.setString(3, se);

String su;
su = sub.getSelectedItem().toString();
ps.setString(4, su);

DateTimeFormatter mn = DateTimeFormatter.ofPattern("MMMM");
String m = mn.format(Month.values()[month.getMonth()]);

ps.setString(5, m);
ps.setString(6, tl.getText());
ps.setString(7, la.getText());

```

```

Float n1, n2, res;
n1 = Float.parseFloat(tl.getText());
n2 = Float.parseFloat(la.getText());
res = n2 / n1 * 100;
String formattedString = String.format("%.02f", res);
result.setText(String.valueOf(formattedString));
ps.setString(8, result.getText());

if (ps.executeUpdate() > 0);
JOptionPane.showMessageDialog(null, "Update Successfully!");

} catch (SQLException ex) {
JOptionPane.showMessageDialog(null, ex);
}
DefaultTableModel model = (DefaultTableModel) jTable2.getModel();
model.setRowCount(0);
FindUser();
}

private void jTable2MouseClicked(java.awt.event.MouseEvent evt) {

int i = jTable2.getSelectedRow();
TableModel model = jTable2.getModel();

roll.setText(model.getValueAt(i, 2).toString());
sem.setSelectedItem(jTable2.getValueAt(i, 3).toString());
sub.setSelectedItem(jTable2.getValueAt(i, 4).toString());

tl.setText(model.getValueAt(i, 6).toString());
la.setText(model.getValueAt(i, 7).toString());

}

private void jButton16ActionPerformed(java.awt.event.ActionEvent evt) {
String es = "INSERT INTO `records`(`Year`, `roll_no`, `Subject`, `Semester`, `T_marks`, `Score`) VALUES
(?,?,?, ?,?,?)";
try {
ps = MyConnection.getConnection().prepareStatement(es);

DateTimeFormatter f = DateTimeFormatter.ofPattern("yyyy");
String yer = f.format(Year.of(yr.getYear()));
ps.setString(1, yer);

ps.setString(2, rol.getText());

String s;
s = subject.getSelectedItem().toString();
ps.setString(3, s);

String se;
se = seme.getSelectedItem().toString();
ps.setString(4, se);

ps.setString(5, t.getText());

ps.setString(6, score.getText());

```

```

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Inserted Successfully!");
}

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
model = (DefaultTableModel) jrecord.getModel();
model.setRowCount(0);
DisplayRecord();
}

private void jButton15ActionPerformed(java.awt.event.ActionEvent evt) {

int i = jrecord.getSelectedRow();
String ip = (jrecord.getModel().getValueAt(i, 0).toString());

String up = "UPDATE `records` SET Year=?, roll_no=?,Subject=?,`Semester`=?,`T_marks`=?,`Score`=?
WHERE `SI`=" + ip;
try {
ps = MyConnection.getConnection().prepareStatement(up);

String se;
se = seme.getSelectedItemAt().toString();
ps.setString(4, se);

ps.setString(5, t.getText());

ps.setString(6, score.getText());

DateTimeFormatter f = DateTimeFormatter.ofPattern("yyyy");
String yer = f.format(Year.of(yr.getYear()));
ps.setString(1, yer);

ps.setString(2, rol.getText());

String s;
s = subject.getSelectedItemAt().toString();
ps.setString(3, s);

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Updated Successfully!");
}

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
model = (DefaultTableModel) jrecord.getModel();
model.setRowCount(0);
DisplayRecord();

}

private void jButton14ActionPerformed(java.awt.event.ActionEvent evt) {

```



```

int row = jrecord.getSelectedRow();
String dele = (jrecord.getModel().getValueAt(row, 0).toString());

String del = "DELETE FROM `records` WHERE `SI`=" + dele;
try {
ps = MyConnection.getConnection().prepareStatement(del);

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Deleted Successfully!");
}

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}

DisplayRecord();
}

private void jrecordMouseClicked(java.awt.event.MouseEvent evt) {

int i = jrecord.getSelectedRow();
TableModel model = jrecord.getModel();
rol.setText(model.getValueAt(i, 2).toString());
subject.setSelectedItem(jrecord.getValueAt(i, 3).toString());
seme.setSelectedItem((jrecord.getValueAt(i, 4).toString()));
t.setText(model.getValueAt(i, 5).toString());
score.setText(model.getValueAt(i, 6).toString());

}

private void addActionPerformed(java.awt.event.ActionEvent evt) {
int pos = JOptionPane.showConfirmDialog(null, "Are you sure you want to Add a new Course?", "Add",
JOptionPane.YES_NO_OPTION);
if (pos == 0) {

String cou = "INSERT INTO `course`(`Dept`, `Course_ID`, `Course_name`, `Year`, `Course_Semester`)
VALUES (?, ?, ?, ?, ?)";
try {
ps = MyConnection.getConnection().prepareStatement(cou);
String iw;
iw=pepsi.getSelectedItem().toString();
ps.setString(1, iw);

ps.setString(2, ID1.getText());

ps.setString(3, name1.getText());

String y;
y = year1.getSelectedItem().toString();
ps.setString(4, y);

String se;
se = semester1.getSelectedItem().toString();
ps.setString(5, se);

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "New Course Added Successfully!");
}

```

```

course();
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}
}

private void editActionPerformed(java.awt.event.ActionEvent evt) {

int io = JOptionPane.showConfirmDialog(null, "Are you sure you want to Edit", "Edit",
JOptionPane.YES_NO_OPTION);
if (io == 0) {
int row = tablecourse.getSelectedRow();
String j = (tablecourse.getModel().getValueAt(row, 0).toString());

String u = "UPDATE `course` SET `Course_ID`=?,`Course_name`=?,`Year`=?,`Course_Semester`=? WHERE
`Dept`=" + j;
try {
ps = MyConnection.getConnection().prepareStatement(u);

ps.setString(1, ID1.getText());

ps.setString(2, name1.getText());

String y;
y = year1.getSelectedItem().toString();
ps.setString(3, y);

String se;
se = semester1.getSelectedItem().toString();
ps.setString(4, se);

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Updated Succesfully!");
}
course();
} catch (Exception e) {
JOptionPane.showConfirmDialog(null, e);
}
}
}

private void deleteActionPerformed(java.awt.event.ActionEvent evt) {

int psa = JOptionPane.showConfirmDialog(null, "Are you sure you want to Delete this Course?", "Delete",
JOptionPane.YES_NO_OPTION);
if (psa == 0) {

int row = tablecourse.getSelectedRow();
String l = (tablecourse.getModel().getValueAt(row, 0).toString());

String d = " DELETE FROM `course` WHERE `Dept`=" + l;
try {
ps = MyConnection.getConnection().prepareStatement(d);

if (ps.executeUpdate() > 0);

```

```

{
JOptionPane.showMessageDialog(null, "Deleted Successfully!");
course();
}

} catch (Exception e) {
JOptionPane.showConfirmDialog(null, e);
}
}

}

private void tablecourseMouseClicked(java.awt.event.MouseEvent evt) {

int i = tablecourse.getSelectedRow();
TableModel model = tablecourse.getModel();
name1.setText(model.getValueAt(i, 2).toString());
pepsi.setSelectedItem(tablecourse.getValueAt(i, 0).toString());
ID1.setText(model.getValueAt(i, 1).toString());
year1.setSelectedItem(tablecourse.getValueAt(i, 3).toString());
semester1.setSelectedItem(tablecourse.getValueAt(i, 4).toString());

}

private void jButton23ActionPerformed(java.awt.event.ActionEvent evt) {

String l = "INSERT INTO `lec`(`Lec_ID`, `Lec_name`, `Course_ID`, `Department`) VALUES (?, ?, ?, ?)";
try {
ps = MyConnection.getConnection().prepareStatement(l);

String i;
i = lid1.getSelectedItem().toString();
ps.setString(1, i);

String n;
n = lname.getSelectedItem().toString();
ps.setString(2, n);

String d;
d = refid1.getSelectedItem().toString();
ps.setString(3, d);

String p;
p = dn1.getSelectedItem().toString();
ps.setString(4, p);

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Inserted Successfully!");

}

lec();

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

```

```

}

private void jButton24ActionPerformed(java.awt.event.ActionEvent evt) {
int od = JOptionPane.showConfirmDialog(null, "Are you sure you want to Delete?", "Delete",
JOptionPane.YES_NO_OPTION);

if (od == 0) {
int row = lecta.getSelectedRow();
String vl = (lecta.getModel().getValueAt(row, 0).toString());

String l = "DELETE FROM `lec` WHERE `Lec_ID`=" + vl;
try {
ps = MyConnection.getConnection().prepareStatement(l);

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Deleted Successfully!");
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}

}

private void jButton27ActionPerformed(java.awt.event.ActionEvent evt) {
String s = "INSERT INTO `subject`(`Lec_ID`, `Subject`, `Cour_ID`, `Semester`) VALUES (?, ?, ?, ?)";
try {
ps = MyConnection.getConnection().prepareStatement(s);

String l;
l = sid.getSelectedItem().toString();
ps.setString(1, l);

ps.setString(2, ssub.getText());

String sci;
sci = subcid.getSelectedItem().toString();
ps.setString(3, sci);

String sc;
sc = ssem.getSelectedItem().toString();
ps.setString(4, sc);

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Inserted Succesfully!");
}
} catch (Exception e) {
}

private void jButton28ActionPerformed(java.awt.event.ActionEvent evt) {

int op = JOptionPane.showConfirmDialog(null, "Are you sure you want to Delete?", "Delete",

```

```

JOptionPane.YES_NO_OPTION);
if (op == 0) {

String s = "DELETE FROM `subject` WHERE `Lec_ID`= ? AND `Subject`= ?";

try {
ps = MyConnection.getConnection().prepareStatement(s);

String l2;
l2 = sid.getSelectedItem().toString();
ps.setString(1, l2);

ps.setString(2, ssub.getText());

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Deleted Succesfully!");
}
sub();

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void jButton28ActionPerformed(java.awt.event.ActionEvent evt) {

int op = JOptionPane.showConfirmDialog(null, "Are you sure you want to Delete?", "Delete",
JOptionPane.YES_NO_OPTION);
if (op == 0) {

String s = "DELETE FROM `subject` WHERE `Lec_ID`= ? AND `Subject`= ?";

try {
ps = MyConnection.getConnection().prepareStatement(s);

String l2;
l2 = sid.getSelectedItem().toString();
ps.setString(1, l2);

ps.setString(2, ssub.getText());

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Deleted Succesfully!");
}
sub();
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void stabMouseClicked(java.awt.event.MouseEvent evt) {

int i = stab.getSelectedRow();
TableModel model = stab.getModel();
sid.setSelectedItem(stab.getValueAt(i, 0).toString());

```

```

ssub.setText(model.getValueAt(i, 1).toString());
subcid.setSelectedItem(stab.getValueAt(i, 2).toString());
ssem.setSelectedItem(stab.getValueAt(i, 3).toString());
}

private void lectaMouseClicked(java.awt.event.MouseEvent evt) {

int is = lecta.getSelectedRow();
lid1.setSelectedItem(lecta.getValueAt(is, 0).toString());
lname.setSelectedItem(lecta.getValueAt(is, 1).toString());
refid1.setSelectedItem(lecta.getValueAt(is, 2).toString());
dn1.setSelectedItem(lecta.getValueAt(is, 3).toString());

}

private void jButton17ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(5);
}

private void jButton18ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(6);
}

private void jButton19ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(7);
}

private void jButton29ActionPerformed(java.awt.event.ActionEvent evt) {

int row = JOptionPane.showConfirmDialog(null, "Are you sure you want to Add this item?", "Add",
JOptionPane.YES_NO_OPTION);
if (row == 0) {

String in = "INSERT INTO `inventory`(`Item Description`, `Quantity`, `Unit Price(rs)`, `Department`)
VALUES (?, ?, ?, ?)";
try {
ps = MyConnection.getConnection().prepareStatement(in);

ps.setString(1, item.getText());
ps.setString(2, quantity.getText());
ps.setString(3, unit.getText());
String k;
k = inde.getSelectedItem().toString();
ps.setString(4, k);

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Item added Successfully!");
}
Invs();
} catch (Exception e) {
}
}

private void jButton31ActionPerformed(java.awt.event.ActionEvent evt) {

int rem = JOptionPane.showConfirmDialog(null, "Are you sure you want to Delete this item?", "Delete",

```

```

JOptionPane.YES_NO_OPTION);
if (rem == 0) {

int row = intable.getSelectedRow();
String de = (intable.getModel().getValueAt(row, 0).toString());

String d = "DELETE FROM `inventory` WHERE `S.I.`=" + de;
try {
ps = MyConnection.getConnection().prepareStatement(d);

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Item Deleted Successfully!");
}
Invs();
}
catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void jButton30ActionPerformed(java.awt.event.ActionEvent evt) {

int dfs = JOptionPane.showConfirmDialog(null, "Are you sure you want to Edit this item?", "Edit",
JOptionPane.YES_NO_OPTION);
if (dfs == 0) {

int ws = intable.getSelectedRow();
String de = (intable.getModel().getValueAt(ws, 0).toString());

String d = "UPDATE `inventory` SET `Item Description`=?, `Quantity`=?, `Unit Price(rs)`=?, `Department`=?
WHERE `S.I.`=" + de;
try {
ps = MyConnection.getConnection().prepareStatement(d);

ps.setString(1, item.getText());
ps.setString(2, quantity.getText());
ps.setString(3, unit.getText());

String yo;
yo = inde.getSelectedItem().toString();
ps.setString(4, yo);

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "Item Edited Successfully!");
}
Invs();
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void intableMouseClicked(java.awt.event.MouseEvent evt) {

int i = intable.getSelectedRow();
TableModel model = intable.getModel();

```

```

item.setText(model.getValueAt(i, 1).toString());
quantity.setText(model.getValueAt(i, 2).toString());
unit.setText(model.getValueAt(i, 3).toString());
inde.setSelectedItem(intable.getValueAt(i,4).toString());
}

private void jButton32ActionPerformed(java.awt.event.ActionEvent evt) {

String j = "INSERT INTO `batch` (`Name`, `email`, `alumni type (work field)`, `College`) VALUES (?, ?, ?, ?)";
try {
ps = MyConnection.getConnection().prepareStatement(j);
ps.setString(1, ns.getText());
ps.setString(2, te.getText());
ps.setString(3, stf.getText());
ps.setString(4, college.getText());

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "person Inserted Successfully!");
}
bat();
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void jButton34ActionPerformed(java.awt.event.ActionEvent evt) {

int io = JOptionPane.showConfirmDialog(null, "Are you sure you want to Delete this Batch?", "Delete",
JOptionPane.YES_NO_OPTION);
if (io == 0) {
int ip = jTable6.getSelectedRow();
String b = (jTable6.getModel().getValueAt(ip, 0).toString());
String dd = "DELETE FROM `batch` WHERE `SI`=" + b;
try {
ps = MyConnection.getConnection().prepareStatement(dd);
if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "person Deleted Successfully!");
}
bat();
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}
}

private void jTable6MouseClicked(java.awt.event.MouseEvent evt) {

int i = jTable6.getSelectedRow();
TableModel model = jTable6.getModel();
ns.setText(model.getValueAt(i, 1).toString());
te.setText(model.getValueAt(i, 2).toString());
stf.setText(model.getValueAt(i, 3).toString());
college.setText(model.getValueAt(i, 4).toString());

}

private void jButton33ActionPerformed(java.awt.event.ActionEvent evt) {

```



```

int io = JOptionPane.showConfirmDialog(null, "Are you sure you want to Edit this Batch?", "Edit",
JOptionPane.YES_NO_OPTION);
if (io == 0) {

int a = jTable6.getSelectedRow();
String b = (jTable6.getModel().getValueAt(a, 0).toString());
String dd = "UPDATE `batch` SET `Name`=?,`email`=?,`alumni type (work field)`=?,`College`=? WHERE
`SI`=" + b;
try {
ps = MyConnection.getConnection().prepareStatement(dd);
ps.setString(1, ns.getText());
ps.setString(2, te.getText());
ps.setString(3, stf.getText());
ps.setString(4, college.getText());

if (ps.executeUpdate() > 0) {
JOptionPane.showMessageDialog(null, "person Updated Successfully!");
}
bat();

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void reeKeyReleased(java.awt.event.KeyEvent evt) {

String sql = "SELECT * FROM `records` WHERE `roll_no`=?" ;
try {

ps = MyConnection.getConnection().prepareStatement(sql);
ps.setString(1, ree.getText());
rs = ps.executeQuery();
DefaultTableModel model =(DefaultTableModel)jrecord.getModel();
model.setRowCount(0);
jrecord.setModel(DbUtils.resultSetToTableModel(rs));

if (rs.next()){

String aa = rs.getString("roll_no");
rol.setText(aa);
String ab = rs.getString("T_marks");
t.setText(ab);
String ac = rs.getString("Score");
score.setText(ac);
}

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void jButton35ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(11);
model = (DefaultTableModel) vivo.getModel();

```

```

model.setRowCount(0);
FilterInventory();
}

private void jButton44ActionPerformed(java.awt.event.ActionEvent evt) {

int dal = JOptionPane.showConfirmDialog(null, "Are you sure you want to Delete?", "Delete",
JOptionPane.YES_NO_OPTION);
if (dal == 0) {

String d = " DELETE FROM `stdreg` WHERE `Rollno`=?";
try {
ps = MyConnection.getConnection().prepareStatement(d);
ps.setString(1, rolno.getText());
if (ps.executeUpdate() > 0);
{
JOptionPane.showMessageDialog(null, "Deleted Successfully!");
}
DefaultTableModel model = (DefaultTableModel) info.getModel();
model.setRowCount(0);
View();
} catch (Exception e) {
JOptionPane.showConfirmDialog(null, e);
}
}
}

private void jButton43ActionPerformed(java.awt.event.ActionEvent evt) {

int t = JOptionPane.showConfirmDialog(null, "Are you sure you want to Edit the changes?", "Edit",
JOptionPane.YES_NO_OPTION);
if (t == 0) {
//int row = info.getSelectedRow();
//String j = (info.getModel().getValueAt(row, 2).toString());
String val = rolno.getText();
String u = "UPDATE `stdreg` SET `Fname`=?,`Lname`=?,`Gender`=?,`Contact`=?,`Course`=?,`Address`=?
WHERE `Rollno`=" +val;
try {
ps = MyConnection.getConnection().prepareStatement(u);

ps.setString(1, fn.getText());
ps.setString(2, ln.getText());

String gender = null;
if (male.isSelected()) {
gender = "Male";
}
if (female.isSelected()) {
gender = "Female";
}
ps.setString(3, gender);

ps.setString(4, contact.getText());

```

```

String cours;
cours = cop.getSelectedItem().toString();
ps.setString(5, cours);

ps.setString(6, address.getText());

ps.executeUpdate();
JOptionPane.showMessageDialog(null, "Updated Succesfully!");

DefaultTableModel model = (DefaultTableModel) info.getModel();
model.setRowCount(0);
View();

} catch (Exception e) {
JOptionPane.showConfirmDialog(null, e);
}
}

}

public ArrayList<heo> dd(){
ArrayList<heo> ros = new ArrayList<>();
String qa="SELECT * FROM `dept`";
try {
ps=MyConnection.getConnection().prepareStatement(qa);
rs=ps.executeQuery();

heo hs;
while(rs.next()){
hs=new heo(rs.getInt("Dept"),rs.getString("Dept_name"));
ros.add(hs);
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
return(ros);
}

public void deptview(){
ArrayList<heo> List = dd();
model = new DefaultTableModel();
model.setColumnIdentifiers(new Object[]{"Dept_ID", "Dept_name"});

Object[] row = new Object[2];
for(int i=0; i<List.size();i++){
row[0]=List.get(i).getDid();
row[1]=List.get(i).getDname();

model.addRow(row);
}
table_dept.setModel(model);
}

private void jButton25ActionPerformed(java.awt.event.ActionEvent evt) {
String ui="INSERT INTO `dept`(`Dept`, `Dept_name`) VALUES (?,?)";
try {

```

```

ps=MyConnection.getConnection().prepareStatement(ui);
ps.setString(1, did.getText());
ps.setString(2, dname.getText());
ps.executeUpdate();
JOptionPane.showMessageDialog(null, "Inserted Successfully!");

deptview();
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void jButton26ActionPerformed(java.awt.event.ActionEvent evt) {

int ood = JOptionPane.showConfirmDialog(null, "Are you sure you want to Delete?", "Delete",
JOptionPane.YES_NO_OPTION);

if (ood == 0) {
String d = " DELETE FROM `dept` WHERE `Dept`=?";
try {
ps = MyConnection.getConnection().prepareStatement(d);
ps.setString(1, did.getText());
if (ps.executeUpdate() > 0);
{

JOptionPane.showMessageDialog(null, "Deleted Successfully!");

}
DefaultTableModel model = (DefaultTableModel)table_dept.getModel();
model.setRowCount(0);
deptview();

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void table_deptMouseClicked(java.awt.event.MouseEvent evt) {
int i=table_dept.getSelectedRow();
TableModel model =table_dept.getModel();
did.setText(model.getValueAt(i, 0).toString());
dname.setText(model.getValueAt(i, 1).toString());

}

private void jButton42ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(10);
}
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(8);
}

private void jButton40ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(1);
}

```

```

private void nnnnKeyReleased(java.awt.event.KeyEvent evt) {
String j ="SELECT * FROM `attendance` WHERE `Roll_no`=?";
try {
ps=MyConnection.getConnection().prepareStatement(j);
ps.setString(1, nnnn.getText());
rs=ps.executeQuery();
jTable2.setModel(DbUtils.resultSetToTableModel(rs));

} catch (Exception e) {
}
}

private void jButton39ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(9);
}

private void jButton48ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(4);
}

private void jButton47ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(8);
}

private void jButton46ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(2);
}

private void jButton45ActionPerformed(java.awt.event.ActionEvent evt) {
jTabbedPane1.setSelectedIndex(1);
}

private void prMouseClicked(java.awt.event.MouseEvent evt) {
noac();
}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
int p=JOptionPane.showConfirmDialog(null, "The Profile will be change are you sure?","change",
JOptionPane.YES_NO_OPTION);
if(p==0){
String pp ="UPDATE `tchreg` SET `Fname`=?,`Lname`=?,`Contact`=?,`Gender`=?,`Add`=?,`Thought`=?
WHERE `T_Id`=? AND `Email`=?";
try {
ps=MyConnection.getConnection().prepareStatement(pp);
ps.setString(1, proFname.getText());
ps.setString(2, proLname.getText());
ps.setString(3, procon.getText());

String se = null;
If (prom.isSelected()){
se="Male";
}
if (prof.isSelected()){
se="Female";
}
ps.setString(4, se);
ps.setString(5, proadd.getText());

```

```

ps.setString(6, thought.getText());
ps.setString(7, proid.getText());
ps.setString(8, proGmail.getText());

if(ps.executeUpdate() > 0){
JOptionPane.showMessageDialog(null, "Your Profile has been updated Successfully");
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null,e);
}
}
}

private void aboutMouseClicked(java.awt.event.MouseEvent evt) {
about.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void aboutMouseClicked(java.awt.event.MouseEvent evt) {
ac();
}

private void privacyMouseClicked(java.awt.event.MouseEvent evt) {
privacy.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void privacyMouseClicked(java.awt.event.MouseEvent evt) {
Security.privacys p = new Security.privacys();
p.setVisible(true);
}

private void searchKeyPressed(java.awt.event.KeyEvent evt) {
try {
int i =Integer.parseInt(search.getText());
showa.setText("");
} catch (NumberFormatException e) {
showa.setText("Please enter valid Rollno!");
}
}

private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {
DefaultTableModel model = (DefaultTableModel)jrecord.getModel();
model.setRowCount(0);
DisplayRecord();
}

private void jButton9ActionPerformed(java.awt.event.ActionEvent evt) {
DefaultTableModel model = (DefaultTableModel)jTable2.getModel();
model.setRowCount(0);
FindUser();
}

private void SearchActionPerformed(java.awt.event.ActionEvent evt) {
String query= inde1.getSelectedItem().toString();
Fils(query);
}

private void checkMouseClicked(java.awt.event.MouseEvent evt) {

```

```

check.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void checkMouseClicked(java.awt.event.MouseEvent evt) {
    Filtration.attend_filter fi = new Filtration.attend_filter();
    fi.setVisible(true);
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    MessageFormat header = new MessageFormat("Score Records");

    MessageFormat footer = new MessageFormat("Page{0}");
    try{
        jrecord.print(JTable.PrintMode.FIT_WIDTH, header, footer);
    }
    catch(Exception e){
        JOptionPane.showMessageDialog(null,"Cannot Print "+e.getMessage());
    }
}

private void jButton10ActionPerformed(java.awt.event.ActionEvent evt) {
    jTabbedPane1.setSelectedIndex(4);
}

private void jButton20ActionPerformed(java.awt.event.ActionEvent evt) {
    jTabbedPane1.setSelectedIndex(4);
}

private void jButton11ActionPerformed(java.awt.event.ActionEvent evt) {
    jTabbedPane1.setSelectedIndex(4);
}

private void jButton21ActionPerformed(java.awt.event.ActionEvent evt) {
    jTabbedPane1.setSelectedIndex(4);
}

private void voActionPerformed(java.awt.event.ActionEvent evt) {

    MessageFormat header = new MessageFormat("Inventory List");
    MessageFormat footer = new MessageFormat("Page{0}");
    try{
        vivo.print(JTable.PrintMode.FIT_WIDTH, header, footer);
    }
    catch(Exception e){
        JOptionPane.showMessageDialog(null,"Cannot Print "+e.getMessage());
    }
}

private void minimMouseClicked(java.awt.event.MouseEvent evt) {
    minim.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void exittMouseClicked(java.awt.event.MouseEvent evt) {
    exitt.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void minimActionPerformed(java.awt.event.ActionEvent evt) {

```

```

setState(Home1.ICONIFIED);
}

private void exittActionPerformed(java.awt.event.ActionEvent evt) {
System.exit(0);
}

private void menuuMouseClicked(java.awt.event.MouseEvent evt) {
menuu.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void menuuMouseClicked(java.awt.event.MouseEvent evt) {
menushow();
}

private void jPanel2MouseClicked(java.awt.event.MouseEvent evt) {
menuhide();
}

private void acActionPerformed(java.awt.event.ActionEvent evt) {
menuhide();
jTabbedPane1.setSelectedIndex(12);
String o = "SELECT * FROM `tchreg` WHERE `Uname`=?" ;
try {
ps=MyConnection.getConnection().prepareStatement(o);
ps.setString(1, teacher_login.uname.getText());
rs=ps.executeQuery();
if(rs.next()){
String ad=rs.getString("Fname");
proFname.setText(ad);

String a1=rs.getString("Lname");
proLname.setText(a1);
String a2=rs.getString("Email");
proGmail.setText(a2);
String a3=rs.getString("T_Id");
proid.setText(a3);

String a4=rs.getString("Contact");
procon.setText(a4);

String a5=rs.getString("Add");
proadd.setText(a5);

String a6=rs.getString("Gender");

if(a6.equals("Male"))
{
prom.setSelected(true);
}else{
prof.setSelected(true);
}
byte[] img =rs.getBytes("Image");
ImageIcon imageIcon = new ImageIcon(new
ImageIcon(img).getImage().getScaledInstance(propic.getWidth(),propic.getHeight(),
Image.SCALE_SMOOTH));
propic.setIcon(imageIcon);

```



```

person_image = img;

String th = rs.getString("Thought");
thought.setText(th);
}

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
}

private void siActionPerformed(java.awt.event.ActionEvent evt) {
teacher_login tl = new teacher_login();
tl.setVisible(true);
this.setVisible(false);
}

private void acMouseClicked(java.awt.event.MouseEvent evt) {
ac.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void siMouseClicked(java.awt.event.MouseEvent evt) {
si.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

public void menushow(){
list.show();
}
public void menuhide(){
list.hide();
}

private void inventfil(String que) {
TableRowSorter<DefaultTableModel> tr = new TableRowSorter<DefaultTableModel>(model);
intable.setRowSorter(tr);

if (que != "Show all") {
tr.setRowFilter(RowFilter.regexFilter(que));

} else {
intable.setRowSorter(tr);
}
}

private void bat() {
String b = "SELECT * FROM `batch`";
try {
ps = MyConnection.getConnection().prepareStatement(b);
rs = ps.executeQuery();
jTable6.setModel(DbUtils.resultSetToTableModel(rs));
} catch (Exception e) {
}
}

private void sub() {

try {

```

```

ps = MyConnection.getConnection().prepareStatement("SELECT * FROM `subject`");
rs = ps.executeQuery();
stab.setModel(DbUtils.resultSetToTableModel(rs));
} catch (Exception e) {
}
}

private void lec() {
String s = "SELECT * FROM `lec`";
try {
ps = MyConnection.getConnection().prepareStatement(s);
rs = ps.executeQuery();
lecta.setModel(DbUtils.resultSetToTableModel(rs));
} catch (Exception e) {
}
}

private void course() {
String g = "SELECT * FROM `course`";
try {
ps = MyConnection.getConnection().prepareStatement(g);
rs = ps.executeQuery();
tablecourse.setModel(DbUtils.resultSetToTableModel(rs));
} catch (Exception e) {
}
}

private void filter(String query) {
TableRowSorter<DefaultTableModel> tr = new TableRowSorter<DefaultTableModel>(model);
jTable2.setRowSorter(tr);

if (query != "Show All") {
tr.setRowFilter(RowFilter.regexFilter(query));

} else {
jTable2.setRowSorter(tr);
}
}

public ArrayList<inve> inventshow() {
ArrayList<inve> lista = new ArrayList<>();
try {
String s = "SELECT * FROM `inventory`";
ps = MyConnection.getConnection().prepareStatement(s);
rs = ps.executeQuery();
inve ia;

while (rs.next()) {
ia = new inve(rs.getInt("S.I."), rs.getString("Item Description"), rs.getInt("Quantity"), rs.getInt("Unit Price(rs)"), rs.getString("Department"));
lista.add(ia);
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
return (lista);
}

```

```

public void Invs() {
    ArrayList<inve> in = inventshow();
    model = new DefaultTableModel();
    model.setColumnIdentifiers(new Object[]{"S.I.", "Item Description", "Quantity", "Unit-Price",
    "Department"});
    Object[] row = new Object[5];
    for (int i = 0; i < in.size(); i++) {
        row[0] = in.get(i).getSi();
        row[1] = in.get(i).getItem();
        row[2] = in.get(i).getQuantity();
        row[3] = in.get(i).getPrice();
        row[4] = in.get(i).getDept();

        model.addRow(row);
    }
    intable.setModel(model);
}

public ArrayList<inve> Filterinve() {
    ArrayList<inve> lista = new ArrayList<>();
    try {
        String s = "SELECT * FROM `inventory`";
        ps = MyConnection.getConnection().prepareStatement(s);
        rs = ps.executeQuery();
        inve ia;

        while (rs.next()) {
            ia = new inve(rs.getInt("S.I."), rs.getString("Item Description"), rs.getInt("Quantity"), rs.getInt("Unit
            Price(rs)"), rs.getString("Department"));
            lista.add(ia);
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, e);
    }
    return (lista);
}

public void FilterInventory() {
    ArrayList<inve> in = Filterinve();
    model = new DefaultTableModel();
    model.setColumnIdentifiers(new Object[]{"S.I.", "Item Description", "Quantity", "Unit-Price",
    "Department"});
    Object[] row = new Object[5];
    for (int i = 0; i < in.size(); i++) {
        row[0] = in.get(i).getSi();
        row[1] = in.get(i).getItem();
        row[2] = in.get(i).getQuantity();
        row[3] = in.get(i).getPrice();
        row[4] = in.get(i).getDept();

        model.addRow(row);
    }

    vivo.setModel(model);
}

```

```

public ArrayList<att> ListUser() {
    ArrayList<att> UserLists = new ArrayList<>();
    try {
        String sq = " SELECT * FROM `attendance` ";
        ps = MyConnection.getConnection().prepareStatement(sq);
        rs = ps.executeQuery();

        att a;

        while (rs.next()) {
            a = new att(rs.getInt("SI"),rs.getInt("Year"), rs.getInt("Roll_no"), rs.getString("Semester"),
            rs.getString("Subject"), rs.getString("Month"), rs.getInt("Total_Lec"), rs.getInt("Lec_Attended"),
            rs.getFloat("Overall_Attendance"));
            UserLists.add(a);

        }

    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, e);
    }
    return (UserLists);
}

public void FindUser() {
    ArrayList<att> us = ListUser();
    model = new DefaultTableModel();
    model.setColumnIdentifiers(new Object[]{"SI","Year", "Roll_no", "Semester", "Subject", "Month",
    "Total_Lect", "Lec_Attended", "Overall_Attended"});
    Object[] row = new Object[9];
    for (int i = 0; i < us.size(); i++) {
        row[0] = us.get(i).getSI();
        row[1] = us.get(i).getYear();
        row[2] = us.get(i).getRoll();
        row[3] = us.get(i).getSem();
        row[4] = us.get(i).getSubject();
        row[5] = us.get(i).getMonth();
        row[6] = us.get(i).getTl();
        row[7] = us.get(i).getLt();
        row[8] = us.get(i).getOa();

        model.addRow(row);
    }
    jTable2.setModel(model);
}

public ArrayList<rec> List() {
    ArrayList<rec> record = new ArrayList<>();
    try {
        String r = "SELECT * FROM `records`";
        ps = MyConnection.getConnection().prepareStatement(r);
        rs = ps.executeQuery();

        rec a;
        while (rs.next()) {
            a = new rec(rs.getInt("SI"), rs.getInt("Year"), rs.getInt("roll_no"), rs.getString("Subject"),
            rs.getString("Semester"), rs.getInt("T_marks"), rs.getInt("Score"));

```

```

record.add(a);
}

} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
return (record);
}

public void DisplayRecord() {
ArrayList<rec> j = List();
model = new DefaultTableModel();
model.setColumnIdentifiers(new Object[]{"SI", "Year", "Roll_no", "Subject", "Semester", "Total_Marks",
"Score"});
Object[] d = new Object[7];
for (int i = 0; i < j.size(); i++) {
d[0] = j.get(i).getSi();
d[1] = j.get(i).getYear();
d[2] = j.get(i).getRoll();
d[3] = j.get(i).getSubject();
d[4] = j.get(i).getSemester();

d[5] = j.get(i).getTotal();
d[6] = j.get(i).getScore();

model.addRow(d);
}
jrecord.setModel(model);
}

public void ac(){
pro.show();
abou.setVisible(false);
abou.setVisible(false);
}
public void noac(){
pro.hide();
abou.setVisible(true);
abou.setEnabled(true);
}
}
}

```

---

### 13. Filtering and Sorting Section ( 'attend\_filter.java' ):

```

package Filtration;

import Connection.MyConnection;

```

```

import Home_pag1.user;
import java.awt.Color;
import java.awt.Cursor;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.text.MessageFormat;
import java.util.ArrayList;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.RowFilter;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableRowSorter;

public attend_filter() {
initComponents();
Finda();
Sum();
ROLL.setBackground( new Color(0,0,0,0));
}

/*Variables*/

PreparedStatement ps = null;
ResultSet rs = null;
Connection con=null;
DefaultTableModel model;

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
String ta ="SELECT * FROM `attendance` WHERE Roll_no=? ";
try {
ps=MyConnection.getConnection().prepareStatement(ta);
ps.setString(1, ROLL.getText());
rs=ps.executeQuery();
if(rs.next()){
JOptionPane.showMessageDialog(null,"Data Found!");
jTabbedPane1.setSelectedIndex(1);
Finda();
Sum();

} else{
JOptionPane.showMessageDialog(null,"Data does not exist! please re-check");
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null, "");
}
}

private void jcomItemStateChanged(java.awt.event.ItemEvent evt) {
String se =jcom.getSelectedItem().toString();
Filter(se);
}

private void moMouseClicked(java.awt.event.MouseEvent evt) {
mo.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

```

```

private void moMouseClicked(java.awt.event.MouseEvent evt) {
this.setVisible(false);
}

private void lefMouseClicked(java.awt.event.MouseEvent evt) {
lef.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
}

private void lefMouseClicked(java.awt.event.MouseEvent evt) {
jTabbedPane1.setSelectedIndex(0);
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
MessageFormat header = new MessageFormat("Attendance Record");
MessageFormat footer = new MessageFormat("Page{0}");
try{
jTable1.print(JTable.PrintMode.FIT_WIDTH, header, footer);
}
catch(Exception e){
JOptionPane.showMessageDialog(null,"Cannot Print "+e.getMessage());
}
}

public void Filter(String qe){
TableRowSorter<DefaultTableModel> t = new TableRowSorter<DefaultTableModel>(model);
jTable1.setRowSorter(t);

if(qe != "Show All Sem")
{
t.setRowFilter(RowFilter.regexFilter(qe));
Sum();
} else
{
jTable1.setRowSorter(t);
Sum();
}
}

public ArrayList<user> Lista(String val){
ArrayList<user> listuser = new ArrayList<>();

try {
String s="SELECT * FROM `attendance` WHERE `Roll_no`=?";
ps=MyConnection.getConnection().prepareStatement(s);
ps.setString(1, ROLL.getText());
rs=ps.executeQuery();
user a;
while(rs.next()){
a = new user(rs.getInt("Roll_no"), rs.getString("Semester"), rs.getString("Subject"), rs.getString("Month"),
rs.getInt("Total_Lec"), rs.getInt("Lec_Attended"), rs.getFloat("Overall_Attendance"));
listuser.add(a);
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null, e);
}
return(listuser);
}

```

```

public void Finda(){

ArrayList<user> us = Lista(ROLL.getText());
model = new DefaultTableModel();
model.setColumnIdentifiers(new
Object[]{"Roll_no","Semester","Subject","Month","Total_Lect","Lec_Attended","Overall_Attended"});
Object[] row = new Object[7];
for(int i=0; i<us.size();i++){
row[0]=us.get(i).getRoll();
row[1]=us.get(i).getSem();
row[2]=us.get(i).getSubject();
row[3]=us.get(i).getMonth();
row[4]=us.get(i).getTl();
row[5]=us.get(i).getLt();
row[6]=us.get(i).getOa();
model.addRow(row);
}
 jTable1.setModel(model);
}

public int Sum(){
double total=0,t = 0;
for(int i=0; i<jTable1.getRowCount();i++)
{
double to=Double.parseDouble(jTable1.getValueAt(i, 6).toString());
total+=to;
double f=jTable1.getRowCount();
t=total/f;
}
over.setText(String.valueOf(String.format("%.2f",t)+"%"));
return(0);
}

```

## 14. Libraries { Imported }:

- jCalendar-1.4.jar
- Absolute Layout –AbsoluteLayout.jar
- MySQL JDBC Driver –mysql-Connector-java-5.1.23-bin.jar
- Swingx-a;-6.4.jar
- rx2xml.jar

---

X-----X



## BIBLIOGRAPHY

Books referred:

Java: The Complete Reference, Ninth Edition **Authors:** [Herbert Schildt](#)

url link for this book in PDF format :

<https://medium.com/@abwehrpotentia/java-the-complete-reference-ninth-edition-by-herbert-schildt-available-in-hardcover-800b1d15766c>

<http://www.sietk.org/downloads/javabook.pdf>

Designing Section -background and interfaces :

<https://netbeans.org/kb/docs/java/quickstart-gui.html>

<https://netbeans.org/kb/docs/java/gui-functionality.html>

<https://stackoverflow.com> (Recommend this website)

<https://stackoverflow.com/questions/14553024/jtabbedPane-in-jpanel>

<https://stackoverflow.com/questions/6660908/how-to-make-jframe-transparent>

Best website for HD Abstract Wallpaper for cool design background:

<https://www.wallpaperflare.com/untitled-simple-background-blue->

Encrypting and printing section:

<https://examples.javacodegeeks.com/core-java/security/encrypt-decrypt-with-salt/>

<https://www.javaguides.net/2020/02/java-sha-512-hash-with-salt-example.html>

<https://docs.oracle.com/javase/tutorial/uiswing/components/table.html#printing>

Sorting , Filtering and printing

<https://docs.oracle.com/javase/tutorial/uiswing/components/table.html#sorting><https://docs.oracle.com/javase/8/docs/api/javax/swing/DefaultRowSorter.html>

Mysql and netbeans connection:

<https://netbeans.org/kb/docs/ide/mysql.html>

<https://stackoverflow.com/questions/6081307/how-to-connect-netbeans-to-mysql-database>

Inserting, updating and displaying values in netbeans jTable, searching records and display image file :

[http://www.java2s.com/Tutorials/Java/JDBC\\_How\\_to/Statement/Display\\_Records\\_From\\_MySQL\\_Database\\_using\\_JTable.htm](http://www.java2s.com/Tutorials/Java/JDBC_How_to/Statement/Display_Records_From_MySQL_Database_using_JTable.htm)

<https://www.youtube.com/watch?v=vFI6m64F6GU>

<http://1bestcsharp.blogspot.com/2015/11/java-jtable-mysql-data-search-filter.html> (Searching records into jTable)

-----Thank You -----