VASAVI ATTENDANCE MONITORING SYSTEM

A

Report

Submitted in partial fulfilment of the

Requirements for the award of the Degree of

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

By

V AKHILA <1602-18-737-063>

Under the guidance of

B LEELAVATHY

Department of Information Technology

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31

2020

BONAFIDE CERTIFICATE

This is to certify that this project entitles “**VASAVI ATTENDANCE MONITORING SYSTEM”** is a bonafide mini project work of Ms**. V AKHILA** bearing the hall ticket number **1602-18-737-063** who carried out the project under my supervision in the year **2020** certified further my best knowledge.

Signature of the examiner

B LEELAVATHY

Assistant professor

Department of Information Technology

ABSTRACT:

This project is going to maintain attendance of students of Vasavi College of Engineering. It generates attendance of a student based on their daily presence (class to class). The staff/faculty will enter the attendance status of a student with respect to their subject into the database. They will be provided with login credentials for marking the attendance. Students will also be provided with login credentials with which they can access their attendance. A weekly report will be generated based on attendance. Message will be generated and sent to their respective phone numbers weekly once.

INTRODUCTION:

1. Requirements about project domain in general

Aim:

To create a **Java GUI based Student Registration form** which takes the values like: student ID, student name, father name, email, phone number, date-of-birth, gender, course, branch, year, sem, password (for website access) from the user. These values are to be updated in the database using **JDBC connectivity.**

2. Information about the project

The project aims at providing a platform made from Java GUI, to the user (teacher/student) where he/she can access attendance of students. Teachers can enter the attendance details of a student and students can access their regular attendance period to period and day to day. The main objective of the project is to understand the procedure of Java Database Connectivity.

3. Architecture and Technology used

Technology:

Java Eclipse, Oracle 11g Database, Java SE version 7, SQL*Plus.

Java AWT:

Java AWT (Abstract Window Toolkit) is *an API to develop GUI or window-based applications* in java.

Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system. AWT is heavyweight i.e. its components are using the resources of OS.

The java.awt package provides classes for AWT API such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.

SQL:

Structure Query Language(SQL) is a database query language used for storing and managing data in Relational DBMS. SQL was the first commercial language introduced for E.F Codd's

Relational model of database. Today almost all RDBMS use SQL as the standard database query language. SQL is used to perform all types of data operations in RDBMS.

Java-SQL Connectivity using JDBC:

Java Database Connectivity is an application programming interface (API) for the programming language Java, which defines how a client may access a database. It is a Java-based data access technology used for Java database connectivity. It is part of the Java Standard Edition platform, from Oracle Corporation. It provides methods to query and update data in a database and is oriented towards relational databases.

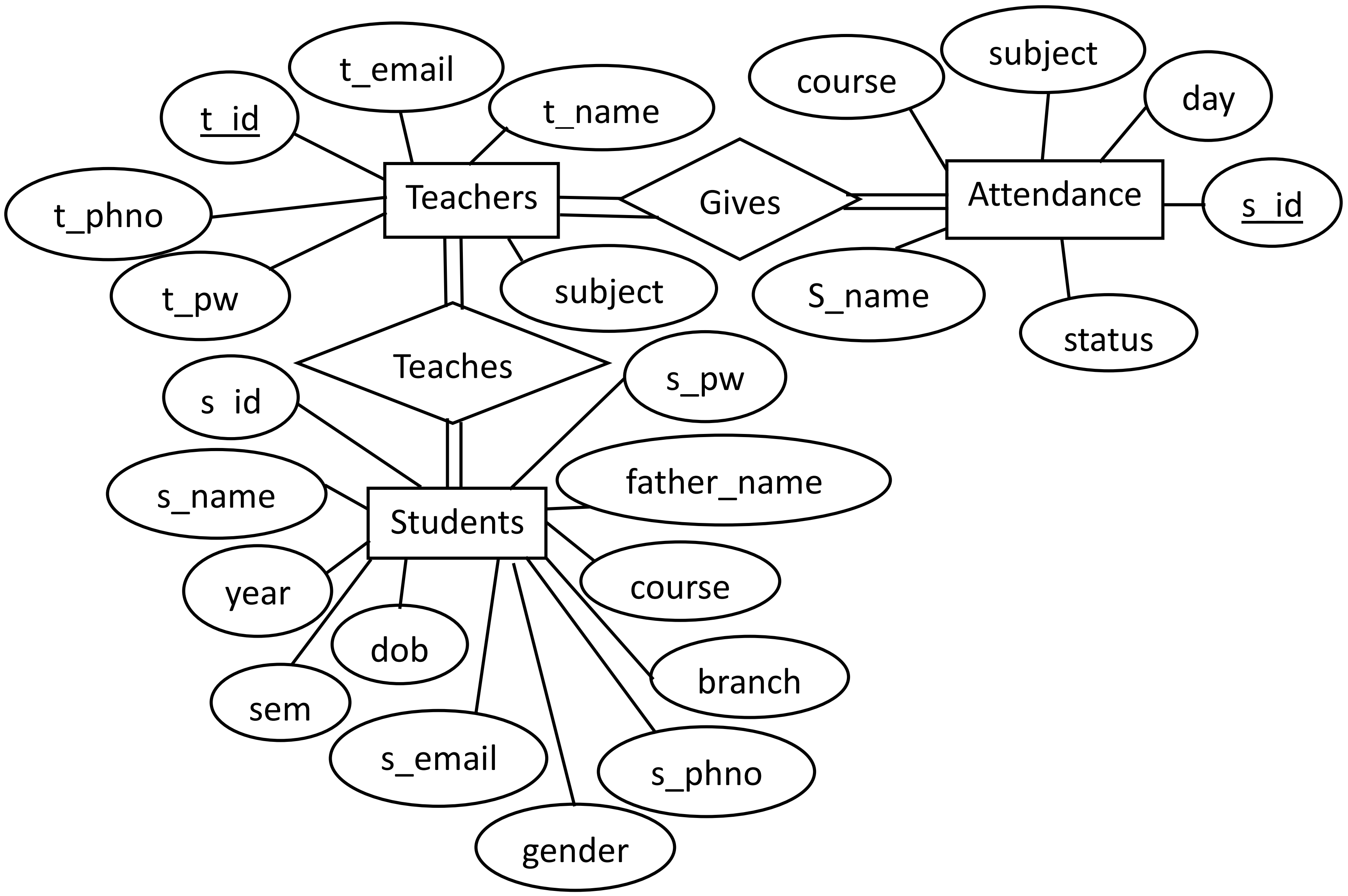
4. Design

Requirement Analysis:

Table Name	**Attributes**
Students	➢ s_id - varchar(10) **(Primary Key)** ➢ s_name - char(30) ➢ s_email - varchar(20) ➢ s_phno – number(10) ➢ s_pw - varchar(10) ➢ father_name - char(30) ➢ sem - number(2) ➢ course - char(5) ➢ dob - varchar(10) ➢ gender - char(5) ➢ year – varchar(2) ➢ branch – char(10)
Teachers	➢ t_id - varchar(10) **(Primary Key)** ➢ t_email - varchar(20) ➢ t_pw - varchar(10) ➢ t_name - varchar(30) ➢ t_phno - number(10) ➢ subject – char(10)
Attendance	➢ course - char(5)

	➢ subject - char(10) ➢ day - date ➢ s_id - varchar(10) **(Foreign Key)** ➢ s_name – char(30) ➢ status - char(5)
Teaches	➢ t_id - varchar(10) **(Foreign Key)** ➢ s_id - varchar(10) **(Foreign Key)** ➢ subject – char(10)
Student_Attendance	➢ t_id - varchar(10) **(Foreign Key)** ➢ s_id - varchar(10) **(Foreign Key)** ➢ status - char(5) ➢ day – date

Entity-Relation Diagram:



Mapping Cardinalities and Constraints:

One teacher can give attendance to many students. Therefore, teachers and attendance have many to many mapping cardinalities.

One teacher can teach to many students. Therefore, teachers and students have many to many mapping cardinalities.

Teachers and students completely participate in the relationships. They have total participation, which is indicated by the two lines.

DDL Commands:

Table creation:

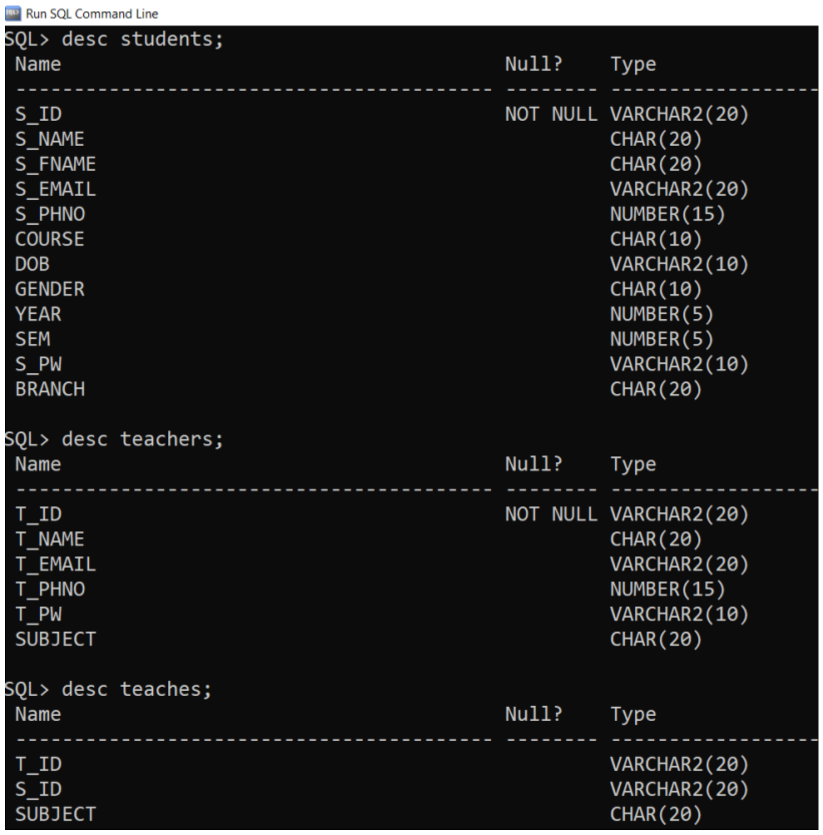
Create table **teachers**(t_id varchar(10), t_email varchar(20), t_password varchar(10), t_name varchar(30), t_phno number(10), subject(20));

Create table **attendance**(course char(5), subject char(10), day date, s_id varchar(10), s_name char(30), status char(5));

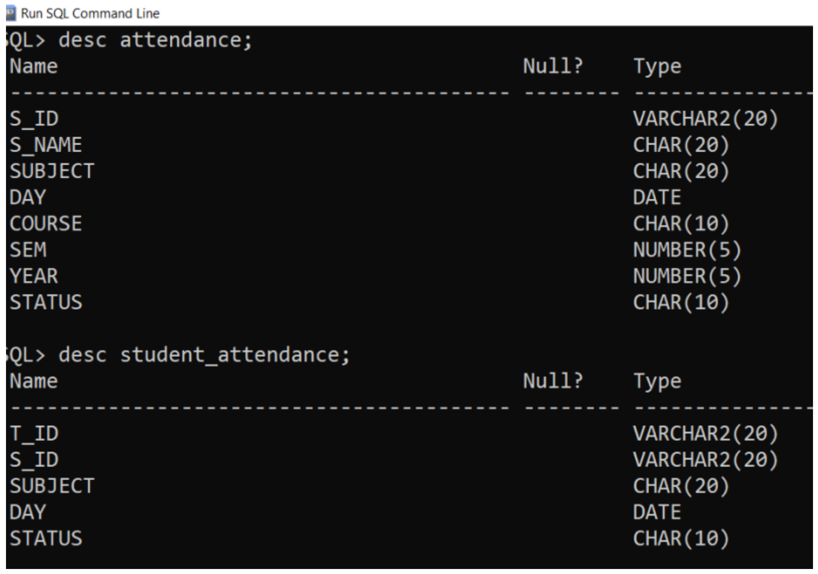
Create table **students**(s_id varchar(10), s_name char(30), s_email varchar(20), s_phno number(10), s_password varchar(10) unique, father_name char(30), branch char(10), sem number(2), course char(5), dob varchar(10), gender char(5), year varchar(2));

Create table **teaches**(t_id varchar(10), s_id varchar(10), subject char(20));

Create table **student_attendance**(t_id varchar(10), s_id varchar(10), status char(5), day date);



```
Run SQL Command Line
SQL> desc students;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 S_ID                                      NOT NULL VARCHAR2(20)
 S_NAME                                             CHAR(20)
 S_FNAME                                            CHAR(20)
 S_EMAIL                                            VARCHAR2(20)
 S_PHNO                                             NUMBER(15)
 COURSE                                             CHAR(10)
 DOB                                                VARCHAR2(10)
 GENDER                                             CHAR(10)
 YEAR                                               NUMBER(5)
 SEM                                                NUMBER(5)
 S_PW                                               VARCHAR2(10)
 BRANCH                                             CHAR(20)

SQL> desc teachers;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 T_ID                                      NOT NULL VARCHAR2(20)
 T_NAME                                             CHAR(20)
 T_EMAIL                                            VARCHAR2(20)
 T_PHNO                                             NUMBER(15)
 T_PW                                               VARCHAR2(10)
 SUBJECT                                            CHAR(20)

SQL> desc teaches;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 T_ID                                               VARCHAR2(20)
 S_ID                                               VARCHAR2(20)
 SUBJECT                                            CHAR(20)
```



```
Run SQL Command Line
SQL> desc attendance;
Name                                      Null?    Type
----------------------------------------- -------- --------------
S_ID                                               VARCHAR2(20)
S_NAME                                             CHAR(20)
SUBJECT                                            CHAR(20)
DAY                                                DATE
COURSE                                             CHAR(10)
SEM                                                NUMBER(5)
YEAR                                               NUMBER(5)
STATUS                                             CHAR(10)

SQL> desc student_attendance;
Name                                      Null?    Type
----------------------------------------- -------- --------------
T_ID                                               VARCHAR2(20)
S_ID                                               VARCHAR2(20)
SUBJECT                                            CHAR(20)
DAY                                                DATE
STATUS                                             CHAR(10)
```

DML Commands:

Insert into **teachers** values('&t_id','&t_email','&t_pw','&t_name',&t_phno,’&subject’);

Insert into **students** values('&s_id', '&s_name', '&s_email', &s_phno, '&s_pw', '&father_name', &sem, '&course', '&dob', '&gender', '&year',’&branch’);

Insert into **attendance** values('&s_id','&s_name','&course','&subject','&day','&status');

Insert into **student_attendance** values ('&s_id','&t_id','&status','&day',’&subject’);

Insert into **teaches** values('&t_id,'&s_id',’&subject’);

Adding foreign keys to the tables:

Alter table **student_attendance** add foreign key(s_id) references students on delete cascade;

Alter table **student_attendance** add foreign key(t_id) references teachers on delete cascade;

Alter table **teaches** add foreign key(s_id) references students on delete cascade;

Alter table **teaches** add foreign key(t_id) references teachers on delete cascade;

Alter table **attendance** add foreign key(s_id) references students on delete cascade;

5. Implementation

Front end programs and connectivity:

Insert Students:

```
import java.awt.*;

import java.awt.event.*;

import java.sql.*;

public class InsertStudents extends Frame

{

        Button insertStudentsButton;

        TextField s_idText, s_nameText, s_fnameText, s_emailText, s_phnoText, courseText, dobText,
genderText, yearText, semText, s_pwText, branchText;

        TextArea errorText;

        Connection connection;
```

```
    Statement statement;

    public InsertStudents()

    {

        try

        {

            Class.forName("oracle.jdbc.driver.OracleDriver");

        }

        catch (Exception e)

        {

            System.err.println("Unable to find and load driver");

            System.exit(1);

        }

        connectToDB();

    }


    public void connectToDB()

  {

        try

        {

         connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","akhila","vasavi");

         statement = connection.createStatement();


        }

        catch (SQLException connectException)
```

```
                {
                  System.out.println(connectException.getMessage());
                  System.out.println(connectException.getSQLState());
                  System.out.println(connectException.getErrorCode());
                  System.exit(1);
                }
    }
        public void buildGUI()
        {
                //Handle Insert Account Button
                insertStudentsButton = new Button("Insert Students");
                insertStudentsButton.addActionListener(new ActionListener()
                {
                        public void actionPerformed(ActionEvent e)
                        {
                                try
                                {
String query= "INSERT INTO students VALUES('" + s_idText.getText() + "', " + "'" +
s_nameText.getText() + "','" + s_fnameText.getText() + "','" + s_emailText.getText() + "'," +
s_phnoText.getText() + ", " + "'" + courseText.getText() + "','" + dobText.getText() + "','" +
genderText.getText() + "'," + yearText.getText() + ", " + semText.getText() + ",'" + s_pwText.getText() +
"','" + branchText.getText() + "')";
                                                                int i = statement.executeUpdate(query);
                                  errorText.append("\nInserted " + i + " rows successfully");
                                }
                                catch (SQLException insertException)
                                {
                                  displaySQLErrors(insertException);
                                }
                        }
                });



                s_idText = new TextField(15);
                s_nameText = new TextField(15);
                s_fnameText = new TextField(15);
                s_emailText = new TextField(15);
                s_phnoText = new TextField(15);
                courseText = new TextField(15);
                dobText = new TextField(15);
                genderText = new TextField(15);
                yearText = new TextField(15);
                semText = new TextField(15);
                s_pwText = new TextField(15);
                branchText = new TextField(15);




                errorText = new TextArea(10, 20);
                errorText.setEditable(false);
```

```
Panel first = new Panel();

first.setLayout(new GridLayout(12, 2));

first.add(new Label("Student ID:"));

first.add(s_idText);

first.add(new Label("Name:"));

first.add(s_nameText);

first.add(new Label("Father's Name:"));

first.add(s_fnameText);

first.add(new Label("Email:"));

first.add(s_emailText);

first.add(new Label("Phno.:"));

first.add(s_phnoText);

first.add(new Label("Course:"));

first.add(courseText);

first.add(new Label("DOB:"));

first.add(dobText);

first.add(new Label("Gender:"));

first.add(genderText);

first.add(new Label("Year:"));

first.add(yearText);

first.add(new Label("Sem:"));

first.add(semText);

first.add(new Label("Password:"));
```

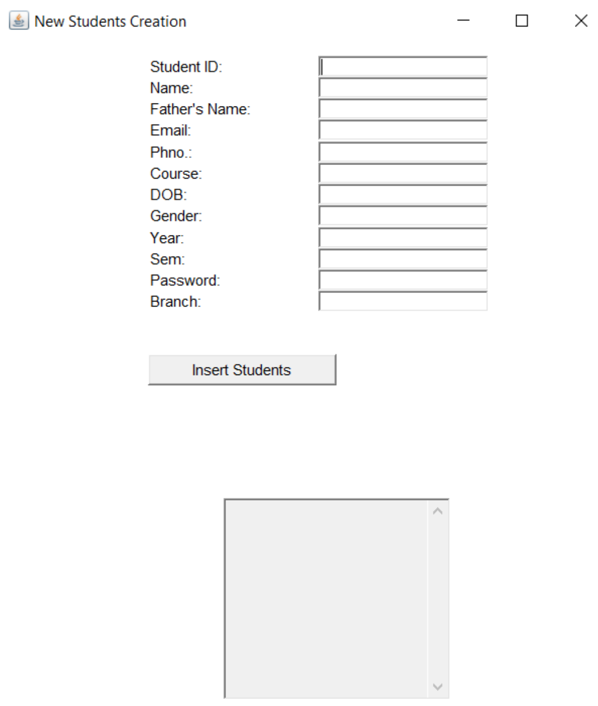
```
            first.add(s_pwText);
            first.add(new Label("Branch:"));
            first.add(branchText);
            first.setBounds(125,40,270,210);

            Panel second = new Panel(new GridLayout(4, 1));
            second.add(insertStudentsButton);
    second.setBounds(125,220,150,100);

            Panel third = new Panel();
            third.add(errorText);
            third.setBounds(125,320,300,200);

            setLayout(null);

            add(first);
            add(second);
            add(third);

            setTitle("New Students Creation");
            setSize(500, 600);
            setVisible(true);
     }
```

```
private void displaySQLErrors(SQLException e)
{
        errorText.append("\nSQLException: " + e.getMessage() + "\n");
        errorText.append("SQLState:     " + e.getSQLState() + "\n");
        errorText.append("VendorError:  " + e.getErrorCode() + "\n");
}




public static void main(String[] args)
{
        InsertStudents s = new InsertStudents();

        s.addWindowListener(new WindowAdapter(){
          public void windowClosing(WindowEvent e)
          {
                System.exit(0);
          }
        });

        s.buildGUI();
}
```

```
}
```

OUTPUT SCREENSHOTS:

Java GUI Screenshot:



Program:

Delete Students:

```
import java.awt.*;

import java.awt.event.*;

import java.sql.*;

public class DeleteStudents extends Frame

{

        Button deleteStudentsButton;

        List StudentsIDList;

        TextField S_IDText,

        S_NAMEText,
S_FNAMEText,
S_EMAILText,
S_PHNOText,
COURSEText,
DOBText,
GENDERText,
YEARText,
SEMText,
S_PWText,
BRANCHText;
TextArea errorText;
Connection connection;
Statement statement;
ResultSet rs;

public DeleteStudents()
{
        try
        {
                Class.forName("oracle.jdbc.driver.OracleDriver");
        }
        catch (Exception e)
        {
                System.err.println("Unable to find and load driver");
```

```
                        System.exit(1);
                    }
                    connectToDB();
                }


                public void connectToDB()
            {
                    try
                    {
                      connection = 
        DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","akhila","vasavi");
                      statement = connection.createStatement();


                    }
                    catch (SQLException connectException)
                    {
                      System.out.println(connectException.getMessage());
                      System.out.println(connectException.getSQLState());
                      System.out.println(connectException.getErrorCode());
                      System.exit(1);
                    }
            }


                private void loadStudents()
                {
```

```
          try
          {
            rs = statement.executeQuery("SELECT * FROM Students");
            while (rs.next())
            {
                    StudentsIDList.add(rs.getString("S_ID"));
            }
          }
          catch (SQLException e)
          {
            displaySQLErrors(e);
          }
}

public void buildGUI()
{
    StudentsIDList = new List(10);
          loadStudents();
          add(StudentsIDList);

          //When a list item is selected populate the text fields
          StudentsIDList.addItemListener(new ItemListener()
          {
                    public void itemStateChanged(ItemEvent e)
```

```
{
    try
    {
        rs = statement.executeQuery("SELECT * FROM Students");
        while (rs.next())
        {
            if (rs.getString("S_ID").equals(StudentsIDList.getSelectedItem()))
            break;
        }
        if (!rs.isAfterLast())
        {
            S_IDText.setText(rs.getString("S_ID"));
            S_NAMEText.setText(rs.getString("S_NAME"));
            S_FNAMEText.setText(rs.getString("S_FNAME"));
            S_EMAILText.setText(rs.getString("S_EMAIL"));
            S_PHNOText.setText(rs.getString("S_PHNO"));
            COURSEText.setText(rs.getString("COURSE"));
            DOBText.setText(rs.getString("DOB"));
            GENDERText.setText(rs.getString("GENDER"));
            YEARText.setText(rs.getString("YEAR"));
            SEMText.setText(rs.getString("SEM"));
            S_PWText.setText(rs.getString("S_PW"));
            BRANCHText.setText(rs.getString("BRANCH"));
    } }
```

```
                catch (SQLException selectException)
                {
                        displaySQLErrors(selectException);
                }
        }
});



//Handle Delete Sailor Button
deleteStudentsButton = new Button("Delete Students");
deleteStudentsButton.addActionListener(new ActionListener()
{
        public void actionPerformed(ActionEvent e)
        {
                try
                {
                        Statement statement = connection.createStatement();
                        int i = statement.executeUpdate("DELETE FROM Students WHERE S_ID
= "
                                        + StudentsIDList.getSelectedItem());
                        errorText.append("\nDeleted " + i + " rows successfully");

                        S_IDText.setText(null);
                        S_NAMEText.setText(null);
                        S_FNAMEText.setText(null);
```

```
                    S_EMAILText.setText(null);
                    S_PHNOText.setText(null);
                    COURSEText.setText(null);
                    DOBText.setText(null);
                    GENDERText.setText(null);
                    YEARText.setText(null);
                    SEMText.setText(null);
                    S_PWText.setText(null);
                    BRANCHText.setText(null);
                    StudentsIDList.removeAll();
                    loadStudents();
                }
                catch (SQLException insertException)
                {
                    displaySQLErrors(insertException);
                }
        }
});


S_IDText = new TextField(15);
S_NAMEText = new TextField(15);
S_FNAMEText = new TextField(15);
S_EMAILText = new TextField(15);
```

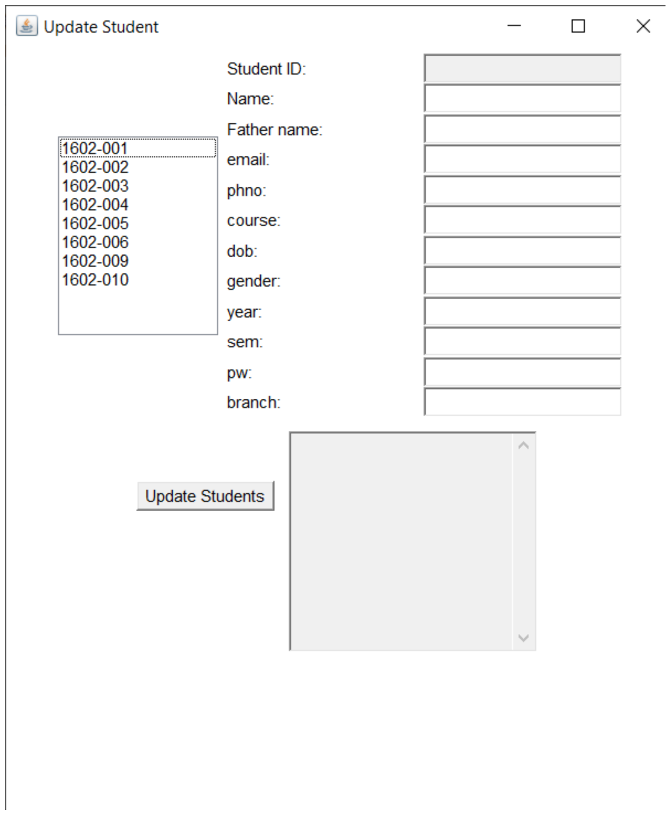
```
S_PHNOText = new TextField(15);
COURSEText  = new TextField(15);
DOBText    = new TextField(15);
GENDERText = new TextField(15);
YEARText = new TextField(15);
SEMText = new TextField(15);
S_PWText = new TextField(15);
BRANCHText = new TextField(15);

errorText = new TextArea(10, 40);
errorText.setEditable(false);

Panel first = new Panel();
first.setLayout(new GridLayout(4, 2));
first.add(new Label("Student ID:"));
first.add(S_IDText);
first.add(new Label("Name:"));
first.add(S_NAMEText);
first.add(new Label("Father name:"));
first.add(S_FNAMEText);
first.add(new Label("email:"));
first.add(S_EMAILText);
first.add(new Label("phno:"));
first.add(S_PHNOText);
first.add(new Label("course:"));

first.add(COURSEText);

first.add(new Label("dob:"));

first.add(DOBText);

first.add(new Label("gender:"));

first.add(GENDERText);

first.add(new Label("year:"));

first.add(YEARText);

first.add(new Label("sem:"));

first.add(SEMText);

first.add(new Label("pw:"));

first.add(S_PWText);

first.add(new Label("branch:"));

first.add(BRANCHText);


Panel second = new Panel(new GridLayout(4, 1));

second.add(deleteStudentsButton);


Panel third = new Panel();

third.add(errorText);


add(first);

add(second);

add(third);
```

```
        setTitle("Remove Students");

        setSize(450, 600);

        setLayout(new FlowLayout());

        setVisible(true);


}








private void displaySQLErrors(SQLException e)

{

        errorText.append("\nSQLException: " + e.getMessage() + "\n");

        errorText.append("SQLState:     " + e.getSQLState() + "\n");

        errorText.append("VendorError:  " + e.getErrorCode() + "\n");

}





public static void main(String[] args)

{

        DeleteStudents dels = new DeleteStudents();
```

```
                    dels.addWindowListener(new WindowAdapter(){
                         public void windowClosing(WindowEvent e)
                         {
                              System.exit(0);
                         }
                        });

                        dels.buildGUI();
                    }
          }
```

OUTPUT SCREENSHOTS:

Java GUI Screenshot:



```
Update Student

Student ID:
Name:
Father name:
email:
phno:
course:
dob:
gender:
year:
sem:
pw:
branch:

1602-001
1602-002
1602-003
1602-004
1602-005
1602-006
1602-009
1602-010

Update Students
```

Program:

Update Students:

```
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class UpdateStudents extends Frame
{
        Button updateStudentsButton;
        List StudentsIDList;
        TextField S_IDText,
        S_NAMEText,
        S_FNAMEText,
        S_EMAILText,
        S_PHNOText,
        COURSEText,
        DOBText,
        GENDERText,
        YEARText,
        SEMText,
        S_PWText,
        BRANCHText;
        TextArea errorText;
        Connection connection;
        Statement statement;
        ResultSet rs;
```

```
    public UpdateStudents()
    {
        try
        {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        }
        catch (Exception e)
        {
            System.err.println("Unable to find and load driver");
            System.exit(1);
        }
        connectToDB();
    }

    public void connectToDB()
  {
        try
        {
         connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","akhila","vasavi");
         statement = connection.createStatement();


        }
        catch (SQLException connectException)
```

```
                {
                  System.out.println(connectException.getMessage());
                  System.out.println(connectException.getSQLState());
                  System.out.println(connectException.getErrorCode());
                  System.exit(1);
                }
}

      private void loadStudents()
      {
                try
                {
                  rs = statement.executeQuery("SELECT S_ID FROM Students");
                  while (rs.next())
                  {
                          StudentsIDList.add(rs.getString("S_ID"));
                  }
                }
                catch (SQLException e)
                {
                  displaySQLErrors(e);
                }
      }
```

```
    public void buildGUI()
    {
      StudentsIDList = new List(10);
          loadStudents();
          add(StudentsIDList);

          //When a list item is selected populate the text fields
          StudentsIDList.addItemListener(new ItemListener()
          {
                public void itemStateChanged(ItemEvent e)
                {
                      try
                      {
                            rs = statement.executeQuery("SELECT * FROM Students where S_ID
="+StudentsIDList.getSelectedItem());
                            rs.next();
                            S_IDText.setText(rs.getString("S_ID"));
                            S_NAMEText.setText(rs.getString("S_NAME"));
                            S_FNAMEText.setText(rs.getString("S_FNAME"));
                            S_EMAILText.setText(rs.getString("S_EMAIL"));
                            S_PHNOText.setText(rs.getString("S_PHNO"));
                            COURSEText.setText(rs.getString("COURSE"));
                            DOBText.setText(rs.getString("DOB"));
                            GENDERText.setText(rs.getString("GENDER"));
                            YEARText.setText(rs.getString("YEAR"));
                SEMText.setText(rs.getString("SEM"));
                S_PWText.setText(rs.getString("S_PW"));
                BRANCHText.setText(rs.getString("BRANCH"));
            }
            catch (SQLException selectException)
            {
                displaySQLErrors(selectException);
            }
        }
});



//Handle Update Sailor Button
updateStudentsButton = new Button("Update Students");
updateStudentsButton.addActionListener(new ActionListener()
{
        public void actionPerformed(ActionEvent e)
        {
            try
            {
                Statement statement = connection.createStatement();
                int i = statement.executeUpdate("UPDATE Students "
                + "SET S_NAME='" + S_NAMEText.getText() + "', "
                + "S_FNAME=" + S_FNAMEText.getText() + ", "
```

```
                                        + "S_EMAIL ="+ S_EMAILText.getText() +" S_PHNO='" +
S_PHNOText.getText() + "', "
                                                    + "COURSE=" + COURSEText.getText() + ", "
                                                    + "DOB ="+ DOBText.getText() +" GENDER='" +
GENDERText.getText() + "', "
                                                                + "YEAR=" + YEARText.getText() + ", "
                                                                + "SEM ="+ SEMText.getText() + " S_PW='" +
S_PWText.getText() + "', "
                                                                            + "BRANCH=" +
BRANCHText.getText()  + " WHERE S_ID = "
                                        + StudentsIDList.getSelectedItem());
                                        errorText.append("\nUpdated " + i + " rows successfully");
                                        StudentsIDList.removeAll();
                                        loadStudents();
                                   }
                                   catch (SQLException insertException)
                                   {
                                        displaySQLErrors(insertException);
                                   }
                              }
                    });

                    S_IDText = new TextField(15);
                    S_IDText.setEditable(false);
                    S_NAMEText = new TextField(15);
                    S_FNAMEText = new TextField(15);
```

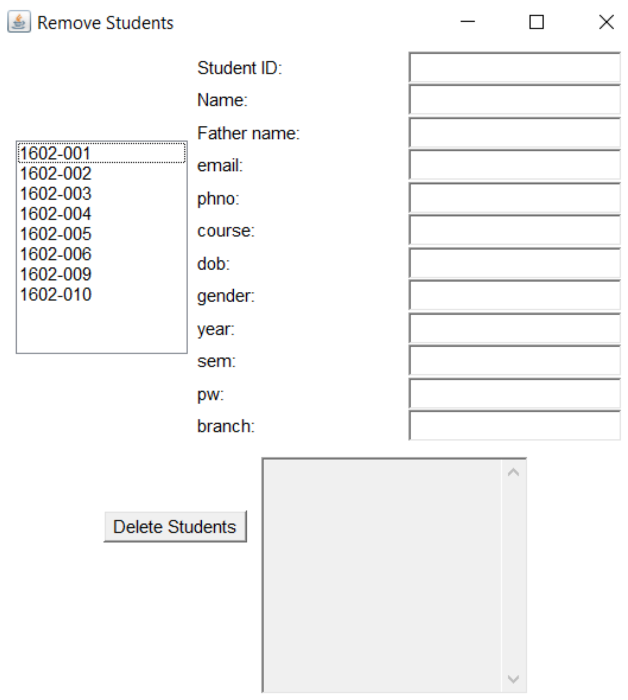
```
S_EMAILText = new TextField(15);
S_PHNOText = new TextField(15);
COURSEText  = new TextField(15);
DOBText    = new TextField(15);
GENDERText = new TextField(15);
YEARText = new TextField(15);
SEMText = new TextField(15);
S_PWText = new TextField(15);
BRANCHText = new TextField(15);

errorText = new TextArea(10, 40);
errorText.setEditable(false);

Panel first = new Panel();
first.setLayout(new GridLayout(4, 2));
first.add(new Label("Sailor ID:"));
first.add(S_IDText);
first.add(new Label("Name:"));
first.add(S_NAMEText);
first.add(new Label("Father name:"));
first.add(S_FNAMEText);
first.add(new Label("email:"));
first.add(S_EMAILText);
first.add(new Label("phno:"));
first.add(S_PHNOText);

first.add(new Label("course:"));

first.add(COURSEText);

first.add(new Label("dob:"));

first.add(DOBText);

first.add(new Label("gender:"));

first.add(GENDERText);

first.add(new Label("year:"));

first.add(YEARText);

first.add(new Label("sem:"));

first.add(SEMText);

first.add(new Label("pw:"));

first.add(S_PWText);

first.add(new Label("branch:"));

first.add(BRANCHText);


Panel second = new Panel(new GridLayout(4, 1));

second.add(updateStudentsButton);


Panel third = new Panel();

third.add(errorText);


add(first);

add(second);
```

```
        add(third);

        setTitle("Update Student");
        setSize(500, 600);
        setLayout(new FlowLayout());
        setVisible(true);

    }

    private void displaySQLErrors(SQLException e)
    {
        errorText.append("\nSQLException: " + e.getMessage() + "\n");
        errorText.append("SQLState:     " + e.getSQLState() + "\n");
        errorText.append("VendorError:  " + e.getErrorCode() + "\n");
    }

    public static void main(String[] args)
    {
        UpdateStudents ups = new UpdateStudents();

        ups.addWindowListener(new WindowAdapter(){
          public void windowClosing(WindowEvent e)
          {
                System.exit(0);
```

```
                 }
                });

                ups.buildGUI();
        }
}
```

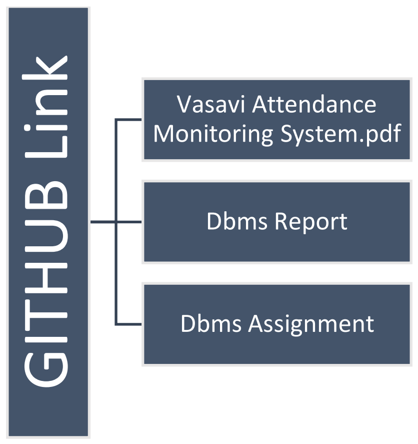
OUTPUT SCREENSHOTS:

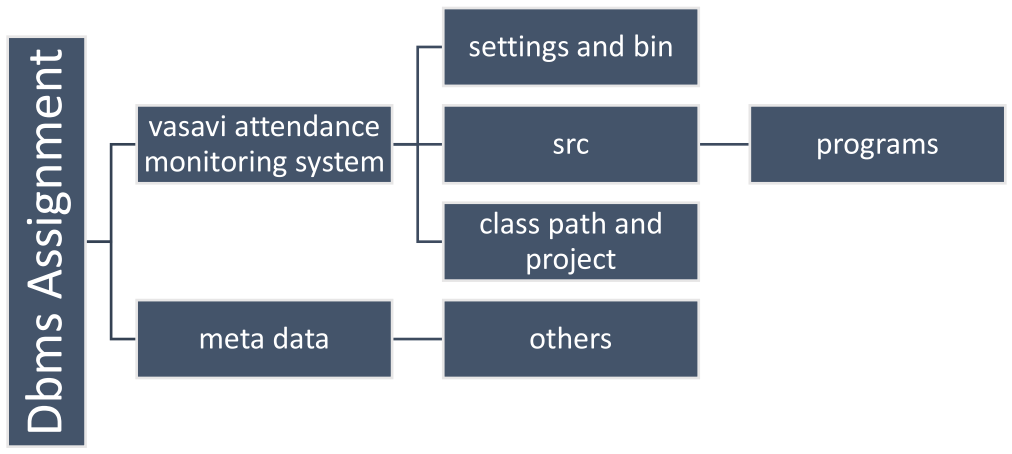
Java GUI Screenshot:



GITHUB LINK:

https://github.com/v-akhila02/akhila

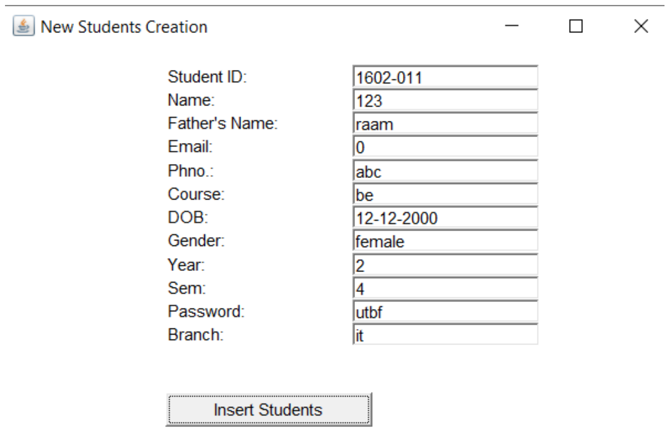




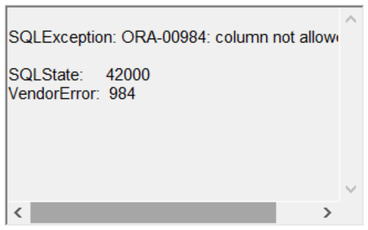
6. TESTING:

This section of the report deals with the testing of the connection between java GUI and the database established previously.

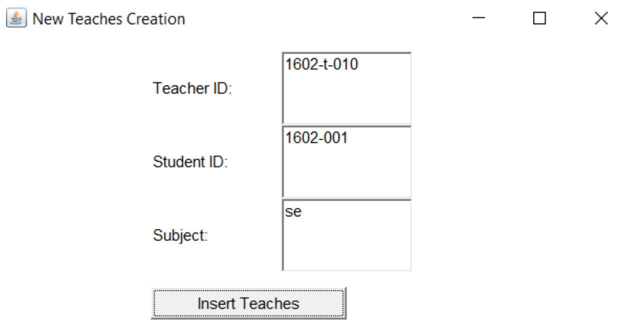
1. Testing for incorrect format/data type of details entered when inserting values into the database of user table using the GUI designed.


New Students Creation

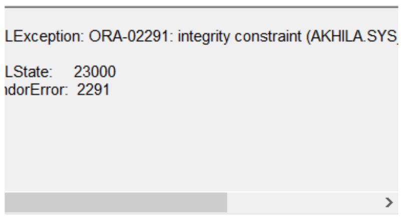
Student ID:	1602-011
Name:	123
Father's Name:	raam
Email:	0
Phno.:	abc
Course:	be
DOB:	12-12-2000
Gender:	female
Year:	2
Sem:	4
Password:	utbf
Branch:	it


```
SQLException: ORA-00984: column not allow

SQLState:    42000
VendorError:  984
```

2. Testing for inserting values into child table those of which are not present in the parent table.

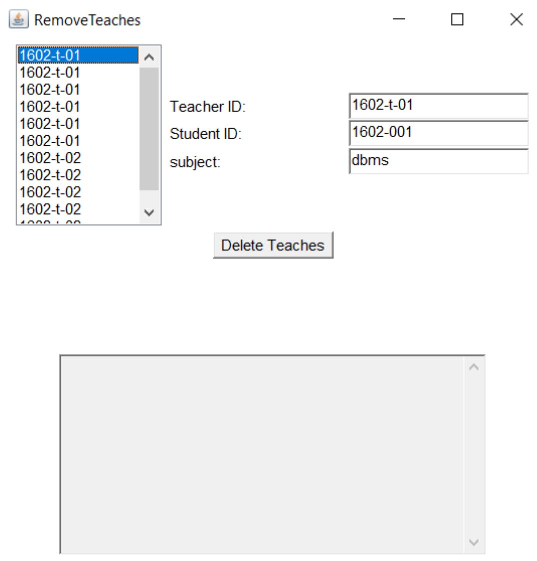

New Teaches Creation

Teacher ID:	1602-t-010
Student ID:	1602-001
Subject:	se

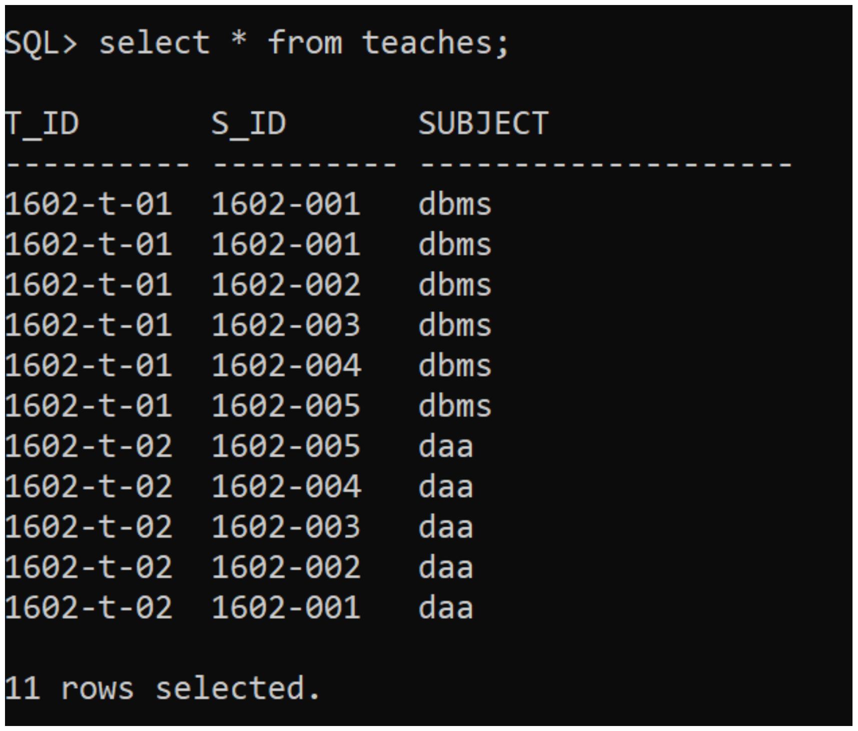

```
LException: ORA-02291: integrity constraint (AKHILA.SYS

LState:    23000
ndorError:  2291
```

3. Deleting a value that doesn't exist in the data base.

The GUI provides a list of values that are present in the data base, making it easier for user to access values. So, the problem of deleting the values that doesn't exist in the data base will not arise.



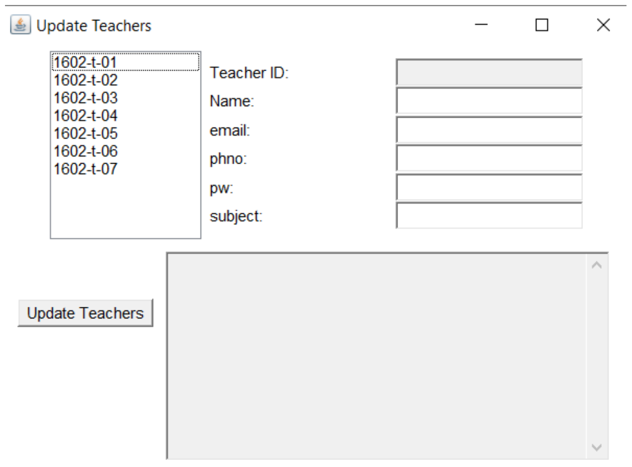
```
RemoveTeaches

1602-t-01
1602-t-01
1602-t-01
1602-t-01
1602-t-01
1602-t-01
1602-t-02
1602-t-02
1602-t-02
1602-t-02

Teacher ID:   1602-t-01
Student ID:   1602-001
subject:      dbms
```



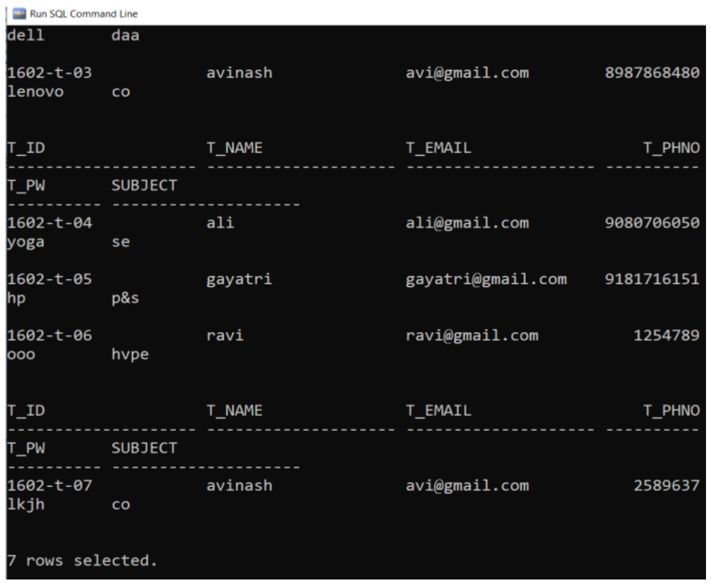
```
SQL> select * from teaches;

T_ID       S_ID       SUBJECT
---------- ---------- --------------------
1602-t-01  1602-001   dbms
1602-t-01  1602-001   dbms
1602-t-01  1602-002   dbms
1602-t-01  1602-003   dbms
1602-t-01  1602-004   dbms
1602-t-01  1602-005   dbms
1602-t-02  1602-005   daa
1602-t-02  1602-004   daa
1602-t-02  1602-003   daa
1602-t-02  1602-002   daa
1602-t-02  1602-001   daa

11 rows selected.
```

3. Deleting a value that doesn't exist in the data base.

Same like the delete GUI, update GUI also provides with the list of values.



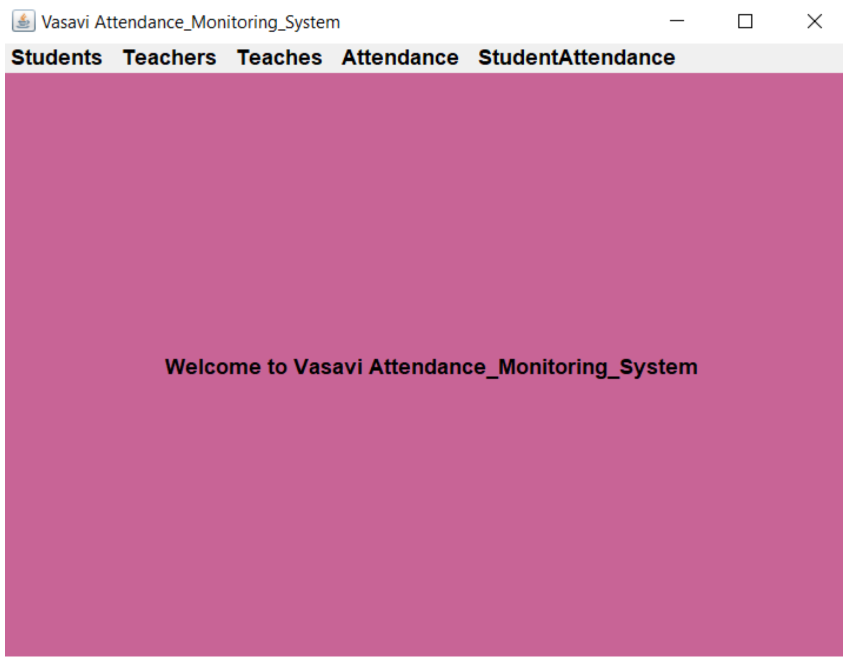
```
Update Teachers

1602-t-01
1602-t-02
1602-t-03
1602-t-04
1602-t-05
1602-t-06
1602-t-07

Teacher ID:
Name:
email:
phno:
pw:
subject:
```

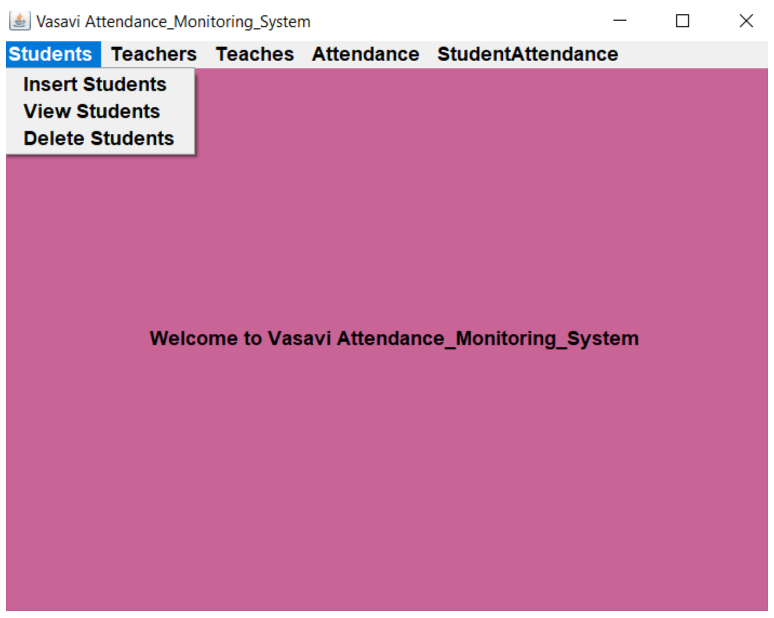


```
Run SQL Command Line
dell       daa

1602-t-03            avinash              avi@gmail.com        8987868480
lenovo     co


T_ID                 T_NAME               T_EMAIL                   T_PHNO
-------------------- -------------------- -------------------- ----------
T_PW       SUBJECT
---------- --------------------
1602-t-04            ali                  ali@gmail.com        9080706050
yoga       se

1602-t-05            gayatri              gayatri@gmail.com    9181716151
hp         p&s

1602-t-06            ravi                 ravi@gmail.com          1254789
ooo        hvpe


T_ID                 T_NAME               T_EMAIL                   T_PHNO
-------------------- -------------------- -------------------- ----------
T_PW       SUBJECT
---------- --------------------
1602-t-07            avinash              avi@gmail.com           2589637
lkjh       co


7 rows selected.
```

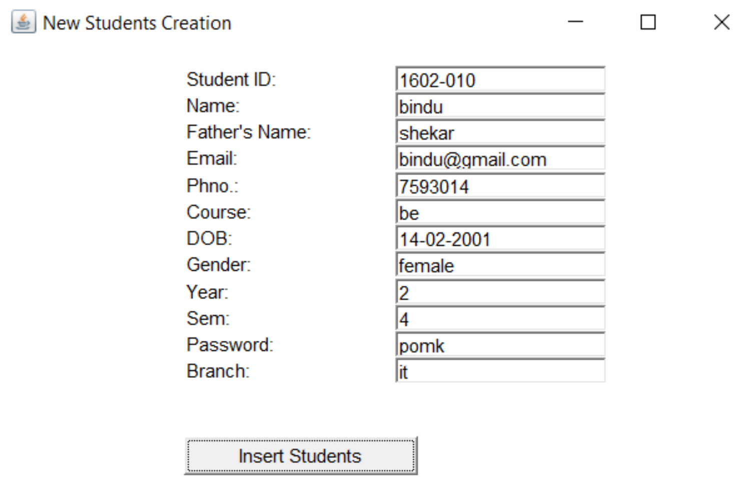
RESULTS:

Main GUI:

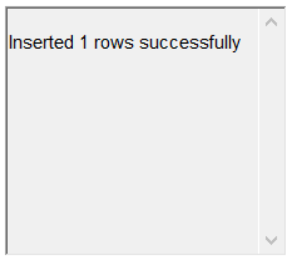

Vasavi Attendance_Monitoring_System
Students Teachers Teaches Attendance StudentAttendance
Welcome to Vasavi Attendance_Monitoring_System


Vasavi Attendance_Monitoring_System
Students Teachers Teaches Attendance StudentAttendance
Insert Students
View Students
Delete Students
Welcome to Vasavi Attendance_Monitoring_System

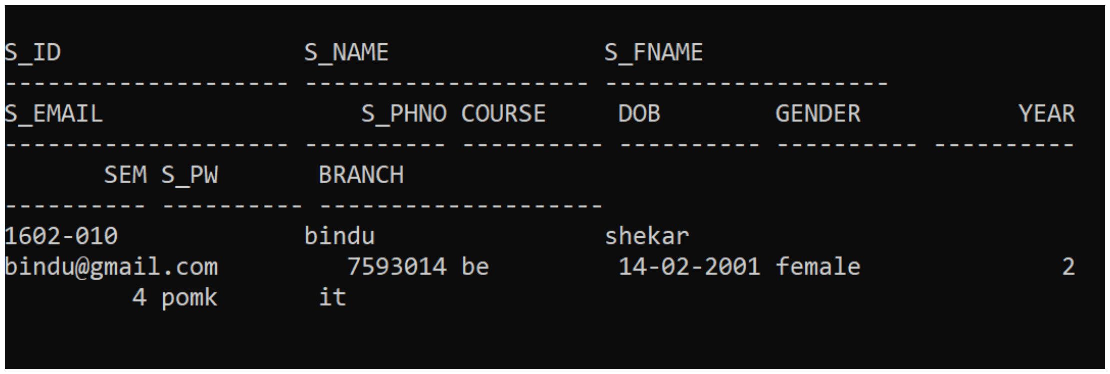
Inserting in to students Table:



Student ID:	1602-010
Name:	bindu
Father's Name:	shekar
Email:	bindu@gmail.com
Phno.:	7593014
Course:	be
DOB:	14-02-2001
Gender:	female
Year:	2
Sem:	4
Password:	pomk
Branch:	it

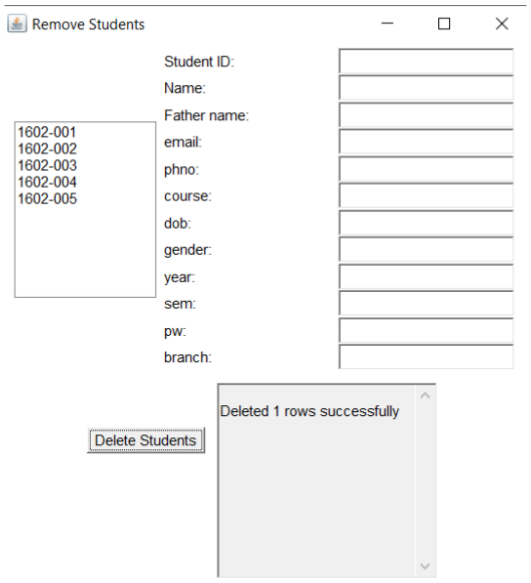


```
Inserted 1 rows successfully
```



```
S_ID                 S_NAME               S_FNAME
-------------------- -------------------- --------------------
S_EMAIL                  S_PHNO COURSE     DOB        GENDER           YEAR
-------------------- ---------- ---------- ---------- ---------- ----------
       SEM S_PW       BRANCH
---------- ---------- --------------------
1602-010             bindu                shekar
bindu@gmail.com         7593014 be         14-02-2001 female              2
         4 pomk       it
```

Deleting from students Table:

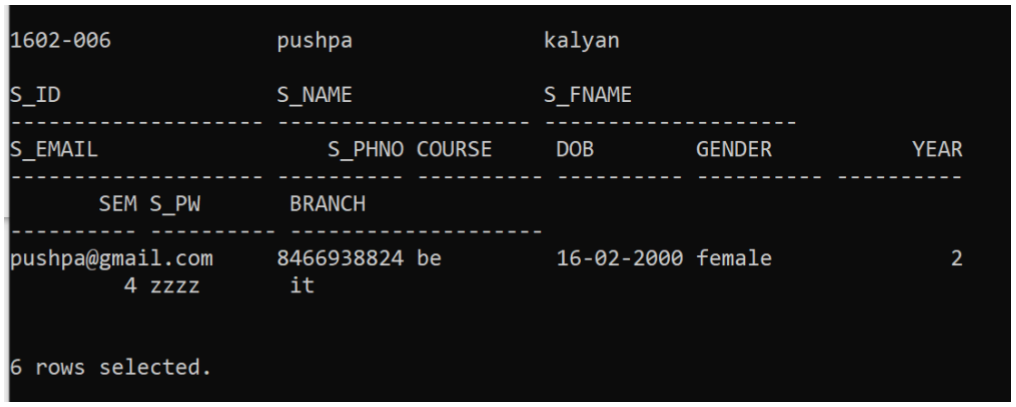


Remove Students

Student ID:
Name:
Father name:
email:
phno:
course:
dob:
gender:
year:
sem:
pw:
branch:

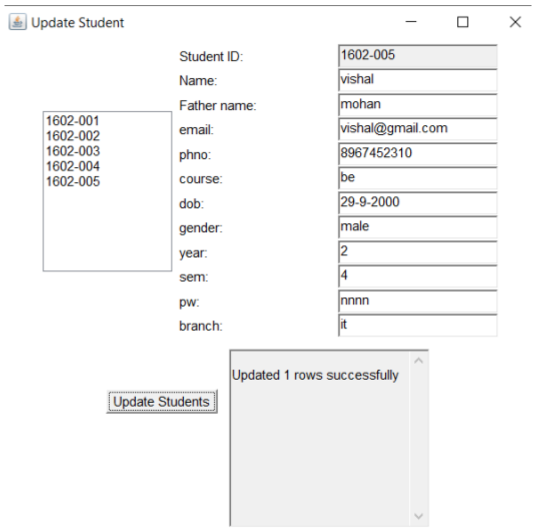
1602-001
1602-002
1602-003
1602-004
1602-005

Deleted 1 rows successfully



```
1602-006             pushpa               kalyan

S_ID                 S_NAME               S_FNAME
-------------------- -------------------- --------------------
S_EMAIL                  S_PHNO COURSE     DOB        GENDER           YEAR
-------------------- ---------- ---------- ---------- ---------- ----------
       SEM S_PW       BRANCH
---------- ---------- --------------------
pushpa@gmail.com     8466938824 be         16-02-2000 female              2
         4 zzzz       it


6 rows selected.
```

Updating from students Table:

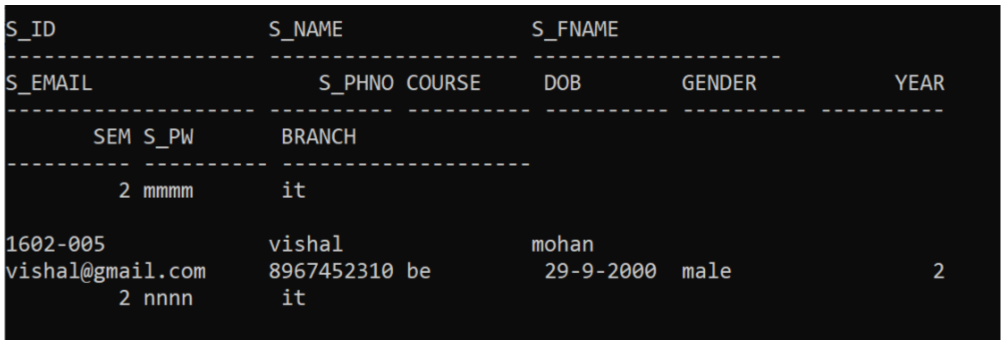


Update Student

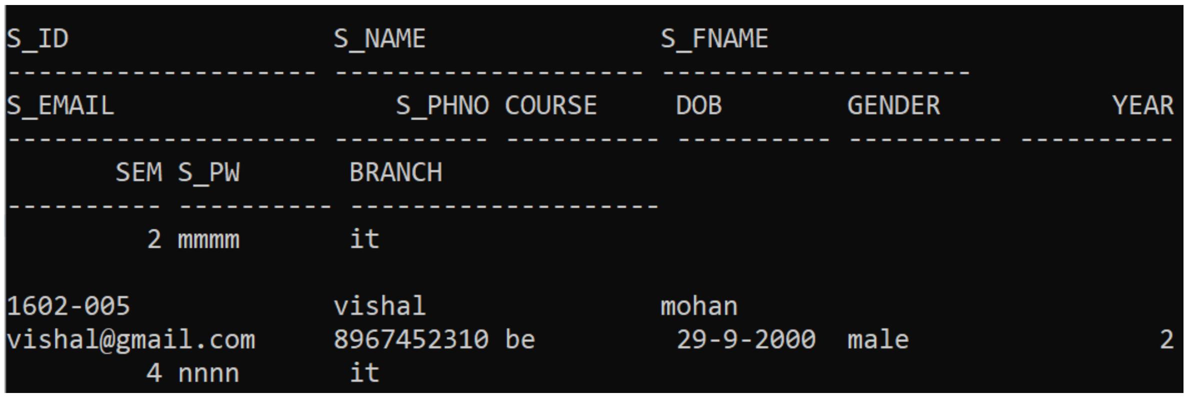
Student ID:	1602-005
Name:	vishal
Father name:	mohan
email:	vishal@gmail.com
phno:	8967452310
course:	be
dob:	29-9-2000
gender:	male
year:	2
sem:	4
pw:	nnnn
branch:	it

1602-001
1602-002
1602-003
1602-004
1602-005

Updated 1 rows successfully



```
S_ID                 S_NAME               S_FNAME
-------------------- -------------------- --------------------
S_EMAIL                  S_PHNO COURSE     DOB        GENDER           YEAR
-------------------- ---------- ---------- ---------- ---------- ----------
       SEM S_PW       BRANCH
---------- ---------- --------------------
         2 mmmm       it

1602-005             vishal               mohan
vishal@gmail.com     8967452310 be         29-9-2000  male                2
         2 nnnn       it
```



```
S_ID                 S_NAME               S_FNAME
-------------------- -------------------- --------------------
S_EMAIL                  S_PHNO COURSE     DOB        GENDER           YEAR
-------------------- ---------- ---------- ---------- ---------- ----------
       SEM S_PW       BRANCH
---------- ---------- --------------------
         2 mmmm       it

1602-005             vishal               mohan
vishal@gmail.com     8967452310 be         29-9-2000  male                2
         4 nnnn       it
```

DISCUSSION & FUTURE WORK:

The application done till now is to store all the information related to the network connection of our college. Furthermore, other programming languages can also be used along with database by connecting SQL with it. This application can be extended further more to store network connections of other colleges, organizations etc

CONCLUSION:

Thus, a Java AWT based registration form is created which is connected to the Oracle 11g database. Therefore, all the entries in the form are directly updated on the register table created in the database.

REFERENCES:

https://www.oracle.com/technetwork/java/javase/documentation/index.html

https://nptel.ac.in/courses/106105175/

https://google.github.io/styleguide/javaguide.html

https://nptel.ac.in/courses/106105191/