**JAVA AWT BASED- REGISTRATION FORM- SQL CONNECTIVITY USING JDBC**

*A*

*Report*

*Submitted in partial fulfilment of the Requirements for the award of the Degree of* **BACHELOR OF ENGINEERING**

IN

## INFORMATION TECHNOLOGY

By

B. Mahathi

1602-19-737-080



**Department of Information Technology**

**Vasavi College of Engineering(Autonomous)**

**(Affiliated to Osmania University)**

**Ibrahimbagh, Hyderabad-31**

**2020-2021**

**BONOFIED CERTIFICATE**

This is to certify that the project report titled “**BODY** **MASS INDEX CALCULATOR**” project work of B. Mahathi bearing hall- ticket number 1602-19-737-080 who carried out this project under my supervision the IV semester for the academic year 2020-2021.

Signature Signature Internal Examiner External Examiner

ABSTRACT:

Body Mass Index is calculated using weight and height of a particular person and it is also considered according to the age. We use body mass index to know whether we are wealthy enough or not. In recent years many are suffering from malnutrition due to the lack of food and some people are lack of awareness on food and fitness. The body mass index tells us our body condition and also suggests few ways to be nutrient. If a person’s BMI value is below 18.5 then the person is mal nutrient and underweight. If a person has weighted between 18.5 to 24.9 then the person is healthy enough. If the person has bmi between 25 to 29.9 then the person is overweighted and the last is if a person with bmi between 30 to 39.9 then he is obessed. So, the BMI states our body condition and suggests ways to develop ourselves.

AIM:

To create a **Java GUI based Body Mass Index Calculator** which takes the values like Name, gender, weight, height, age details and it will provide us the value of BMI and health condition of the user. These values are to be updated in the database using **JDBC connectivity.**

**SOFTWARE USED:**

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 (MySql, Oracle, Infomix, Sybase, MS Access) 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** (**JDBC**) 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.

The connection to the database can be performed using Java programming (JDBC API) as:

**private void** connToDb(){

**try** { Class.*forName*("oracle.jdbc.driver.OracleDriver"); connection =

DriverManager.*getConnection*(“jdbc:oracle:thin:@localhost: 1521:xe","mahathi","2507");

statement = connection.createStatement();

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

}

**catch** (Exception e)

{

System.*err*.println("Unable to find and load driver"); System.*exit*(1);

}

Thus, the connection from Java to Oracle database is performed and therefore, can be used for updating tables in the database directly

**Requirement Analysis:**

List of tables:

User\_details, Report, Suggestions, User\_report, Reports\_check

List of attributes with their domain types:

User\_details:  
 User\_id-number()

User\_name-varchar2()

Age-number()

Gender-varchar2()

Report:

Report\_id-number()

Weight-number()

Height-number()

BMI-number()

Suggestions:

S\_id-number()

Fitness\_sugg-varchar2()

Food-sugg-varchar2()

User\_report:

User\_id-number()

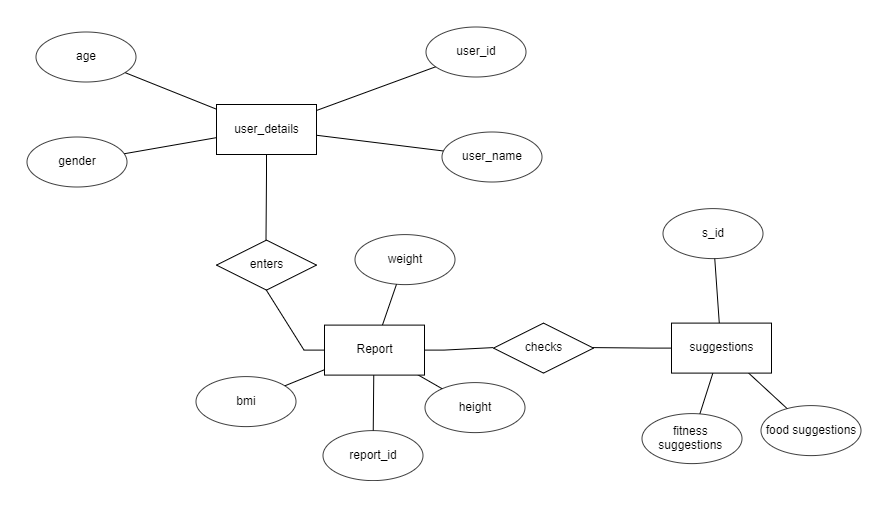
Report\_id-number()

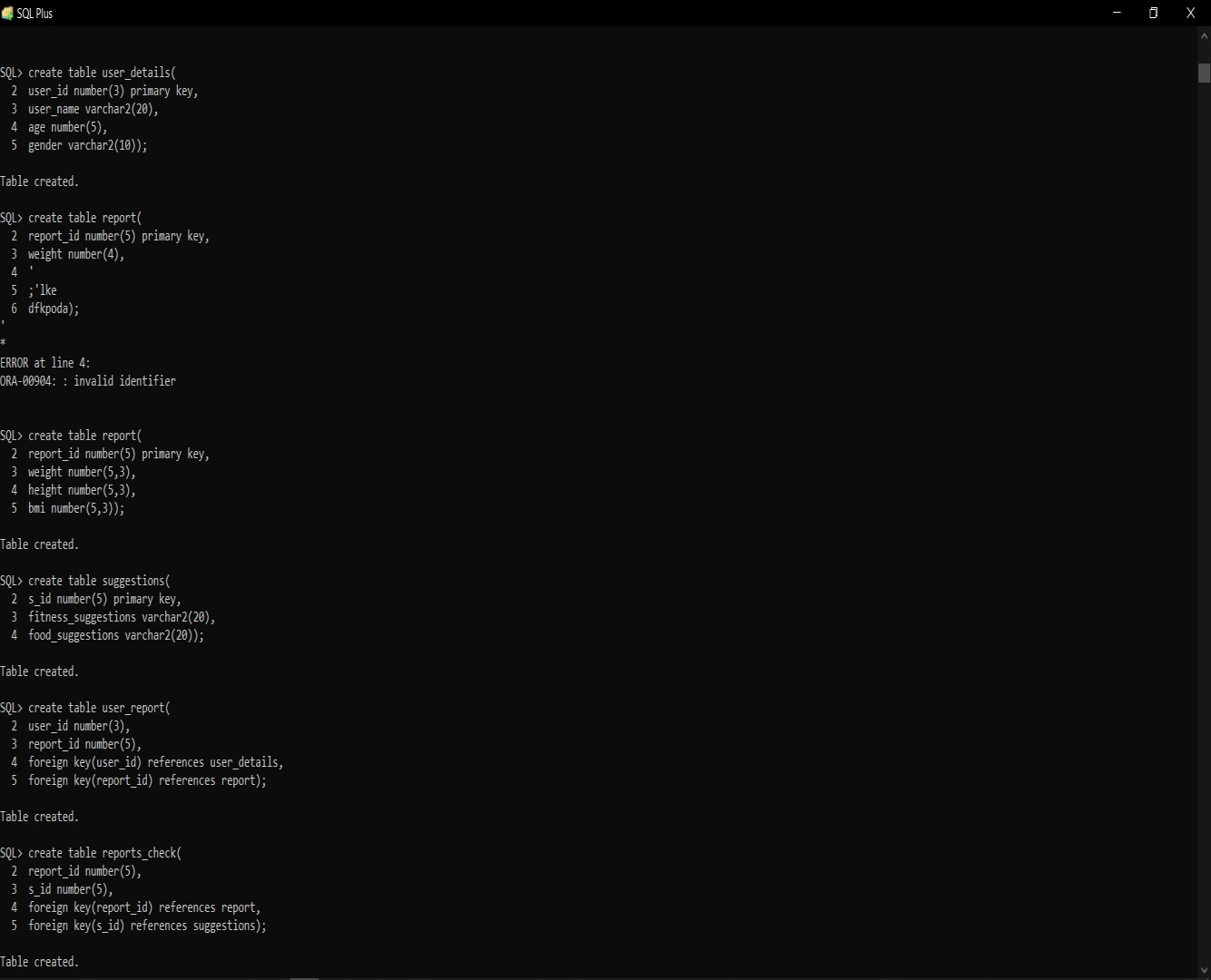
Reports\_check:

Report\_id-number()

S\_id-number()

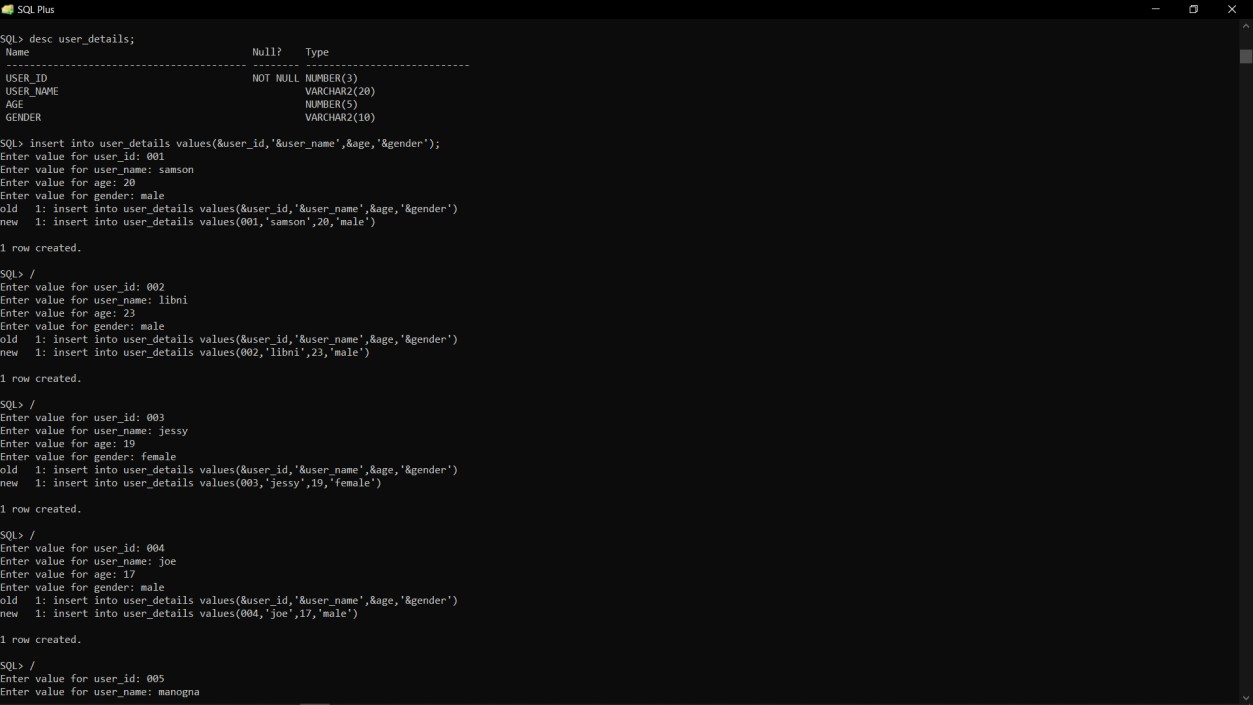
**Entity-Relationship Diagram:**

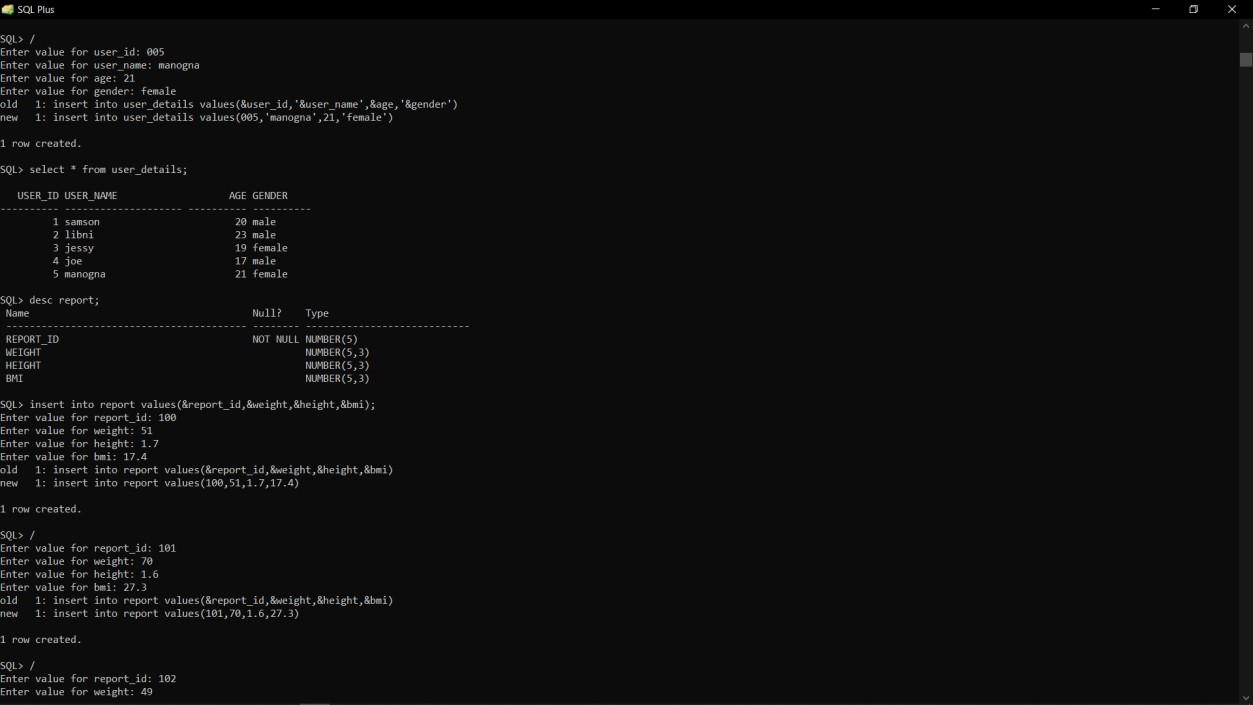
****

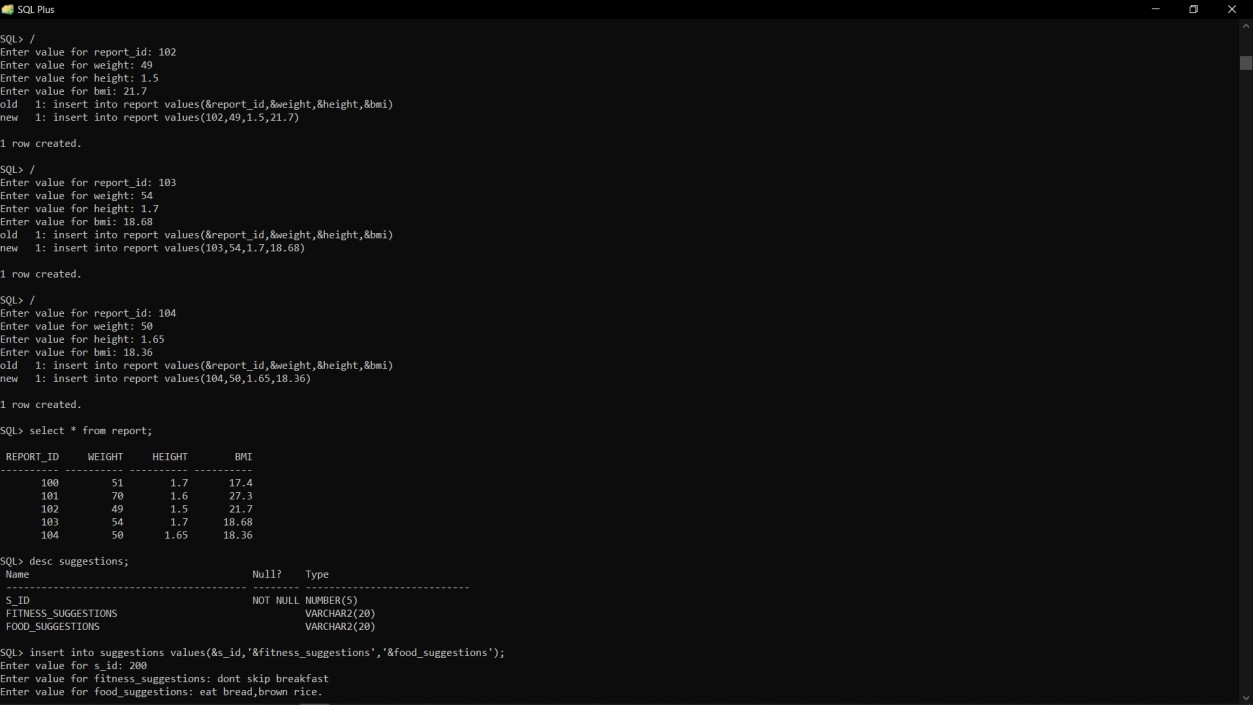


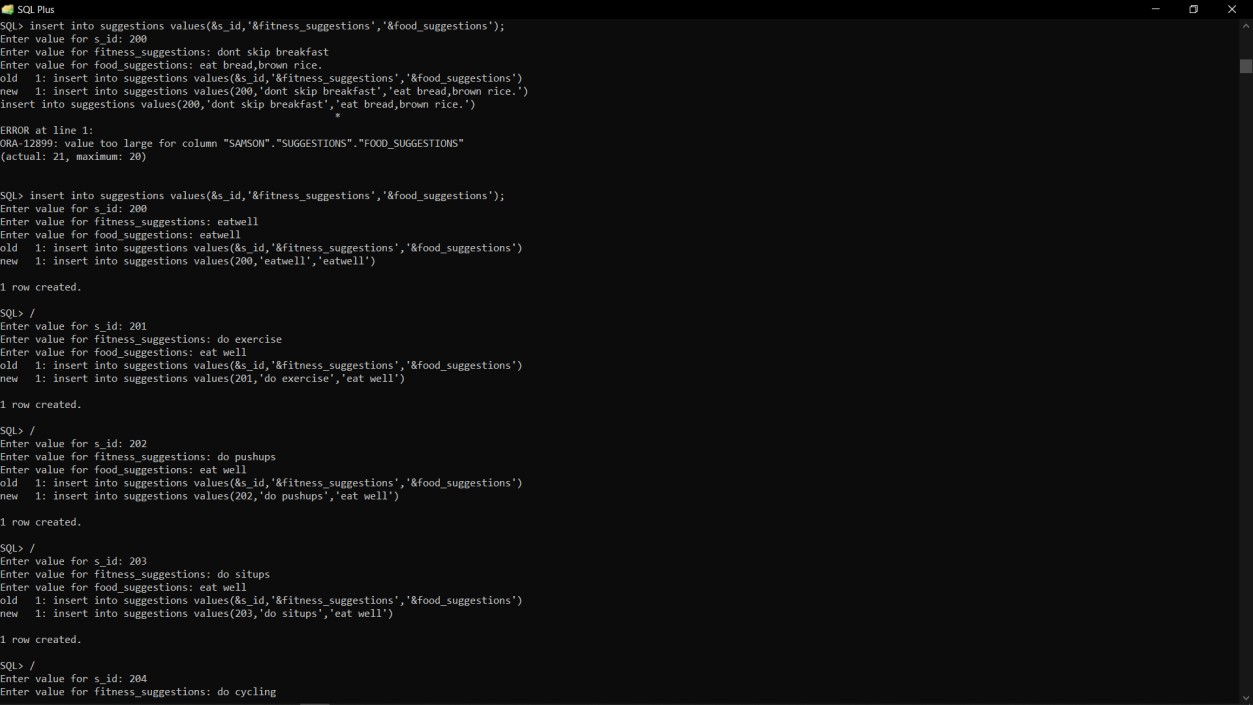
DCL COMMANDS

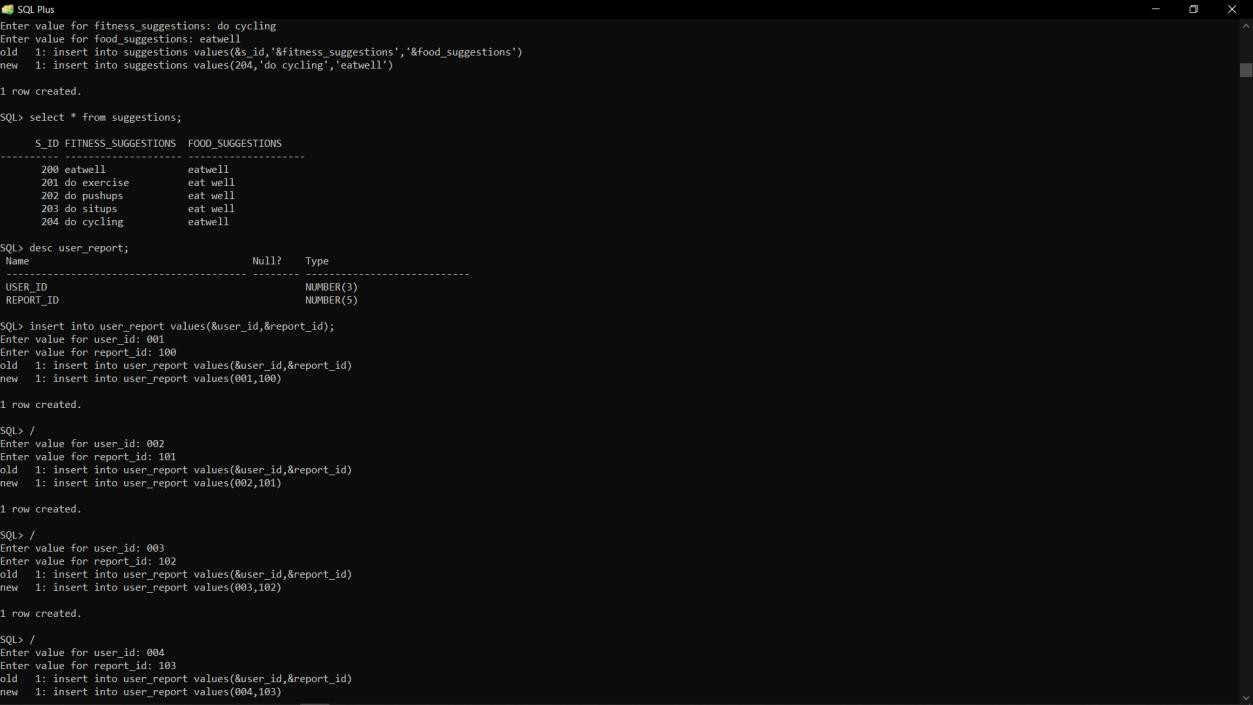
DML COMMANDS

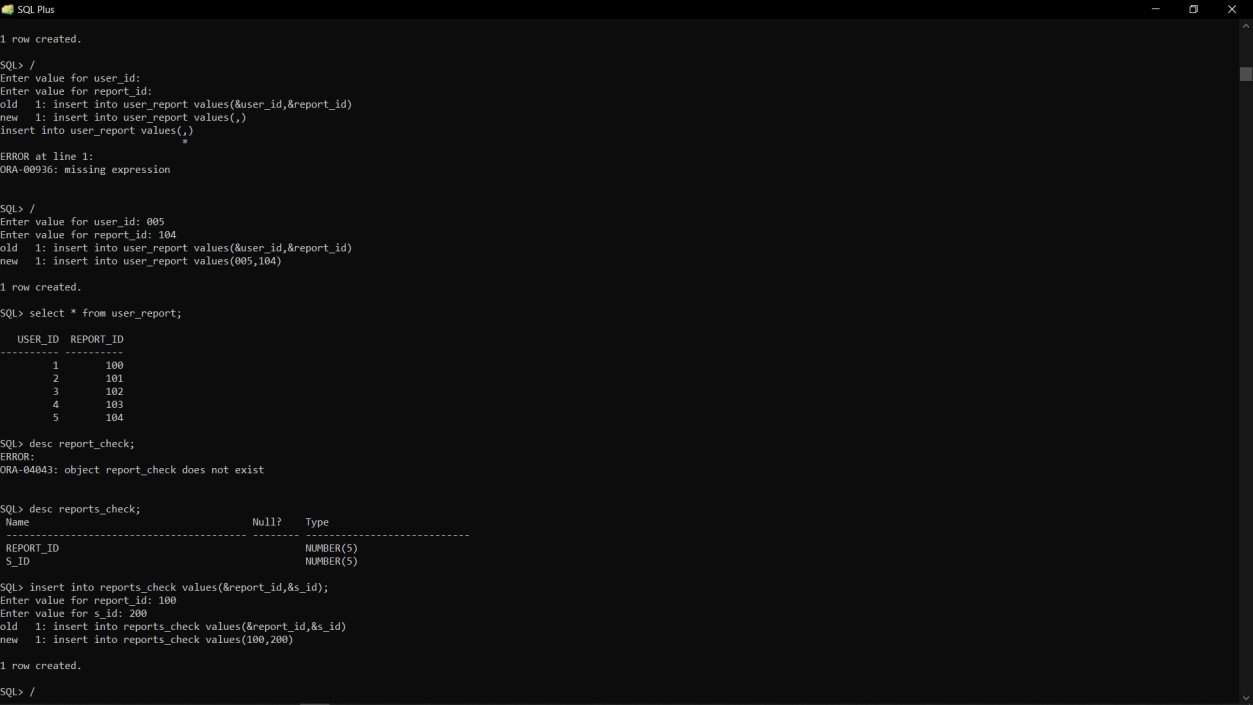


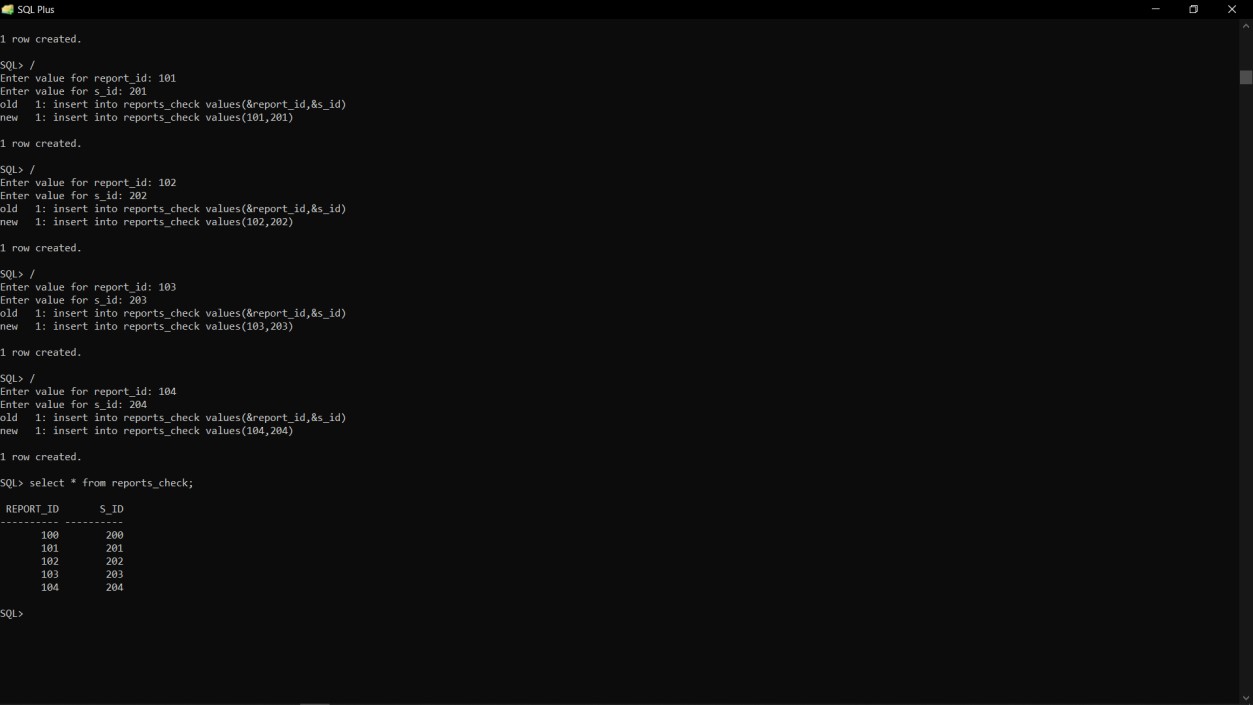




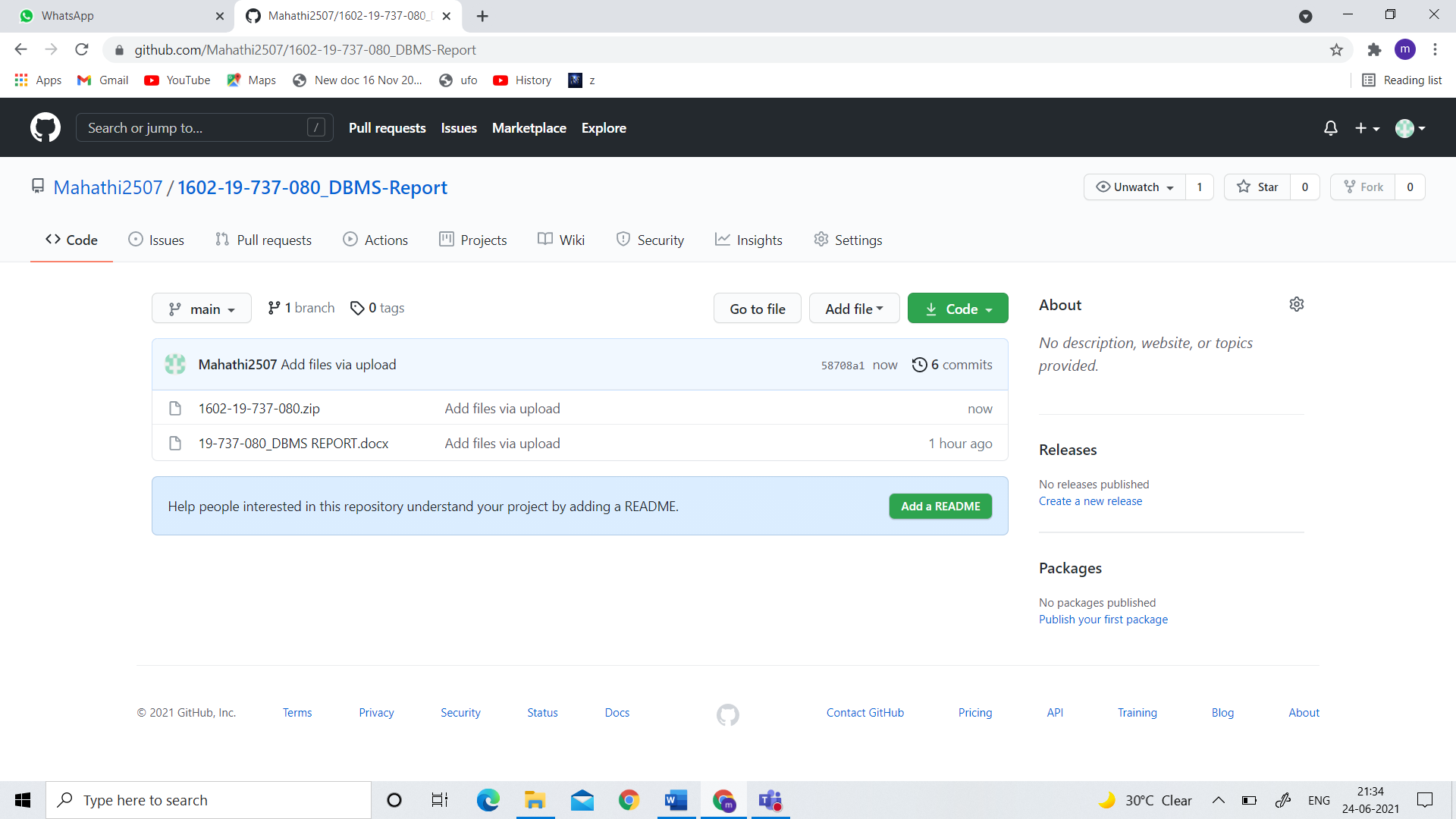








GITHUB:



**Program:**

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

import college.\*;

import report.\*;

import user\_report.\*;

import reports\_checks.\*;

import students.\*;

@SuppressWarnings("serial")

public class FrontPage extends JFrame implements ActionListener{

String msg = "";

Label ll;

CardLayout cardLO;

//Create Panels for each of the menu items, welcome screen panel and home screen panel with CardLayout

Adduser\_details addC;

Updateuser\_details upC;

Deleteuser\_details delC;

Addreport addE;

Updatereport upE;

Deletereport delE;

AddHackathon addH;

UpdateHackathon upH;

DeleteHackathon delH;

AddResults addR;

DeleteResults delR;

UpdateResults upR;

Addsuggestionss addS;

Updatesuggestionss upS;

Deletesuggestionss delS;

Panel home,welcome;

public FrontPage()

{

cardLO = new CardLayout();

//Create an empty home panel and set its layout to card layout

home = new Panel();

home.setLayout(cardLO);

ll = new Label();

ll.setAlignment(Label.CENTER);

ll.setText("Welcome to Hackathon Contest");

//Create welcome panel and add the label to it

welcome = new Panel();

welcome.add(ll);

//create panels for each of our menu items and build them with respective components

addC=new Adduser\_details();addC.buildGUI();

upC = new Updateuser\_details(); upC.buildGUI();

delC = new Deleteuser\_details(); delC.buildGUI();

addE = new Addreport();addE.buildGUI();

upE = new Updatereport();upE.buildGUI();

delE=new Deletereport();delE.buildGUI();

addH=new AddHackathon();addH.buildGUI();

upH=new UpdateHackathon();upH.buildGUI();

delH=new DeleteHackathon();delH.buildGUI();

addR=new AddResults();addR.buildGUI();

delR=new DeleteResults();delR.buildGUI();

upR=new UpdateResults();upR.buildGUI();

addS=new Addsuggestionss();addS.buildGUI();

upS = new Updatesuggestionss();upS.buildGUI();

delS = new Deletesuggestionss();delS.buildGUI();

//add all the panels to the home panel which has a cardlayout

home.add(welcome, "Welcome");

home.add(addC, "Add user\_details");

home.add(upC, "Update user\_details");

home.add(delC, "Delete user\_details");

home.add(addE, "Add report");

home.add(upE, "Update report");

home.add(delE,"Delete report");

home.add(addH,"Add Hackathon");

home.add(upH,"Update Hackathon");

home.add(delH,"Delete Hackathon");

home.add(addR,"Add Results");

home.add(upR,"Update Results");

home.add(delR,"Delete Results");

home.add(addS,"Add Sttudents");

home.add(upS,"Update suggestionss");

home.add(upS,"Delete suggestionss");

// add home panel to main frame

add(home);

// create menu bar and add it to frame

MenuBar mbar = new MenuBar();

setMenuBar(mbar);

// create the menu items and add it to Menu

Menu user\_details = new Menu("user\_details");

MenuItem item1, item2, item3;

user\_details.add(item1 = new MenuItem("Add user\_details"));

user\_details.add(item2 = new MenuItem("View user\_details"));

user\_details.add(item3 = new MenuItem("Delete user\_details"));

mbar.add(user\_details);

Menu report = new Menu("report");

MenuItem item4, item5, item6;

report.add(item4 = new MenuItem("Add report"));

report.add(item5 = new MenuItem("View report"));

report.add(item6 = new MenuItem("Delete report"));

mbar.add(report);

Menu Hackathon = new Menu("Hackathon");

MenuItem item7, item8, item9;

Hackathon.add(item7 = new MenuItem("Add Hackathon"));

Hackathon.add(item8 = new MenuItem("View Hackathon"));

Hackathon.add(item9 = new MenuItem("Delete Hackathon"));

mbar.add(Hackathon);

Menu Results = new Menu("Results");

MenuItem item10, item11, item12;

Results.add(item10 = new MenuItem("Add Results"));

Results.add(item11 = new MenuItem("View Results"));

Results.add(item12 = new MenuItem("Delete Results"));

mbar.add(Results);

Menu suggestionss = new Menu("suggestionss");

MenuItem item13, item14, item15;

suggestionss.add(item13 = new MenuItem("Add suggestionss"));

suggestionss.add(item14 = new MenuItem("View suggestionss"));

suggestionss.add(item15 = new MenuItem("Delete suggestionss"));

mbar.add(suggestionss);

// register listeners

item1.addActionListener(this);

item2.addActionListener(this);

item3.addActionListener(this);

item4.addActionListener(this);

item5.addActionListener(this);

item6.addActionListener(this);

item7.addActionListener(this);

item8.addActionListener(this);

item9.addActionListener(this);

item10.addActionListener(this);

item11.addActionListener(this);

item12.addActionListener(this);

item13.addActionListener(this);

item14.addActionListener(this);

item15.addActionListener(this);

// Anonymous inner class which extends WindowAdaptor to handle the Window event: windowClosing

addWindowListener(new WindowAdapter(){

public void windowClosing(WindowEvent we)

{

quitApp();

}

});

//Frame properties

setTitle("Hackathon Contest");

setSize(500, 600);

setVisible(true);

}

public void actionPerformed(ActionEvent ae)

{

String arg = ae.getActionCommand();

if(arg.equals("Add user\_details"))

{

cardLO.show(home, "Add user\_details");

}

else if(arg.equals("View user\_details"))

{

cardLO.show(home, "Update user\_details");

upC.loaduser\_detailss();

}

else if(arg.equals("Delete user\_details"))

{

cardLO.show(home, "Delete user\_details");

delC.loaduser\_detailss();

}

else if(arg.equals("Add report"))

{

cardLO.show(home, "Add report");

}

else if(arg.equals("View report"))

{

cardLO.show(home, "Update report");

upE.loadreports();

}

else if(arg.equals("Delete report"))

{

cardLO.show(home, "Delete report");

delE.loadreports();

}

else if(arg.equals("Add Hackathon"))

{

cardLO.show(home, "Add Hackathon");

}

else if(arg.equals("View Hackathon"))

{

cardLO.show(home, "Update Hackathon");

upH.loadHackathons();

}

else if(arg.equals("Delete Hackathon"))

{

cardLO.show(home, "Delete Hackathon");

delH.loadHackathons();

}

else if(arg.equals("Add Results"))

{

cardLO.show(home, "Add Results");

}

else if(arg.equals("Delete Results"))

{

cardLO.show(home, "Delete Results");

delR.loadResults();

}

else if(arg.equals("View Results"))

{

cardLO.show(home, "Update Results");

upR.loadResults();

}

else if(arg.equals("Add suggestionss"))

{

cardLO.show(home, "Add suggestionsts");

}

else if(arg.equals("Delete suggestionss"))

{

cardLO.show(home, "Delete suggestionss");

delS.loadsuggestionss();

}

else if(arg.equals("View suggestionss"))

{

cardLO.show(home, "Update suggestionss");

upS.loadsuggestionss();

}

}

private void quitApp () {

try {

//Show a Confirmation Dialog.

int reply = JOptionPane.showConfirmDialog (this,

"Are you really want to exit\nFrom Hackathon Contest?",

"Contest - Exit", JOptionPane.YES\_NO\_OPTION, JOptionPane.PLAIN\_MESSAGE);

//Check the User Selection.

if (reply == JOptionPane.YES\_OPTION) {

setVisible (false); //Hide the Frame.

dispose(); //Free the System Resources.

System.out.println ("Thanks for Using Hackathon Contest\nAuthor - thalari pavan kumar");

System.exit (0); //Close the Application.

}

else if (reply == JOptionPane.NO\_OPTION) {

setDefaultCloseOperation(JFrame.DO\_NOTHING\_ON\_CLOSE);

}

}

catch (Exception e) {}

}

public static void main(String ... args)

{

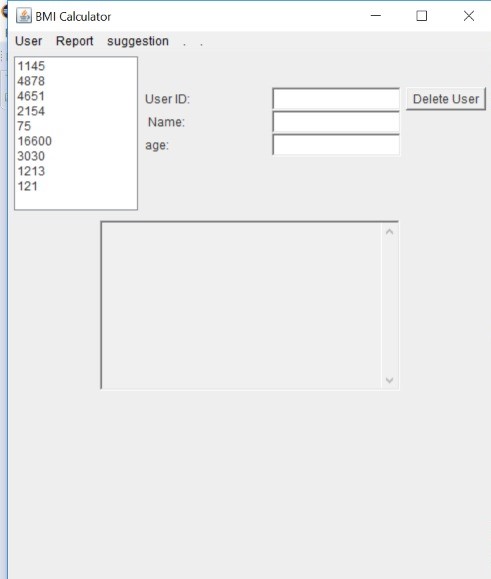
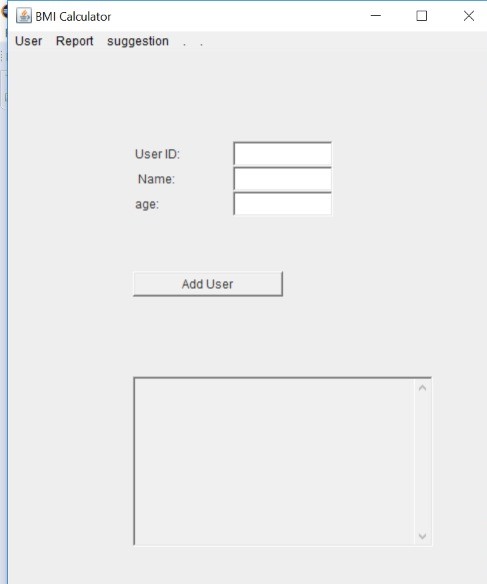
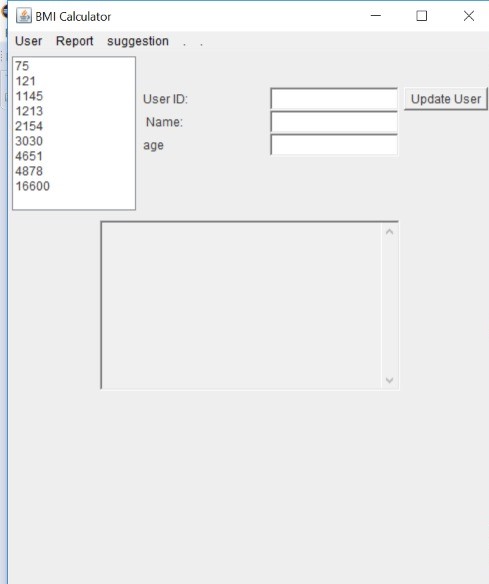
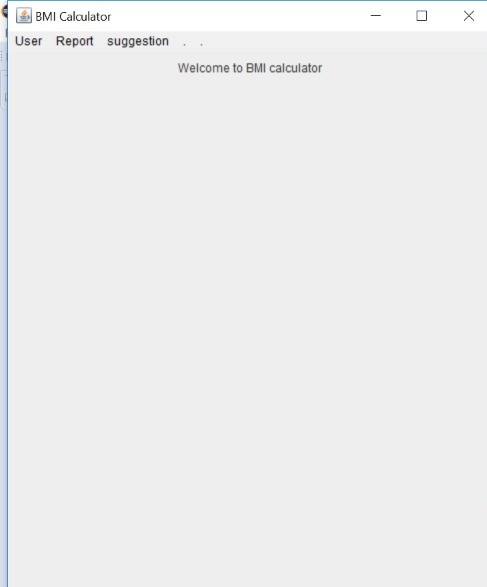
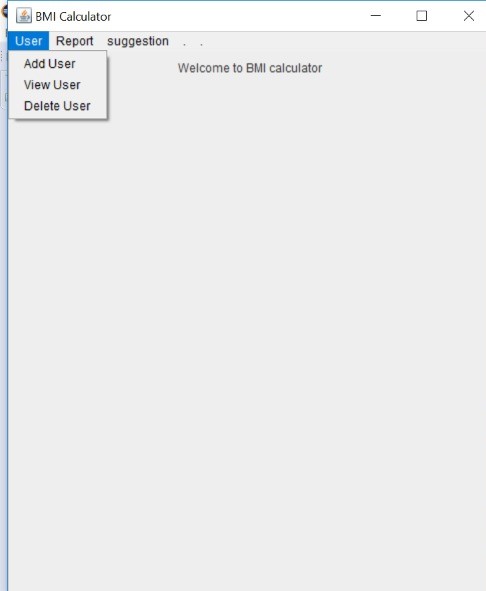
new FrontPage();

}

}

**Output Screenshots:**

**Java GUI screenshot:**

****

**CONCLUSION:**

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

**REFERENCES:**

<https://www.decodejava.com/what-is-jdbc.htm>

https://docs.oracle.com/javase/8/docs/api/

<https://www.tutorialspoint.com/swing/index.htm>

GITHUB LINK:

https://github.com/Mahathi2507/1602-19-737-080\_DBMS-Report.git