DBMS Mini Project
Title: Hackathon Contest

JAVA AWT BASED- Hackathon Contest- 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

T. Pavan Kumar <1602-18-737-087>

Department of Information Technology

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31

2019-20

BONAFIDE CERTIFICATE

This to Certify that the project report titled “ Hackathon Contest” project work of Mr.T.Pavan Kumar bearing Roll.no:1602-18-737-087 who carried out this project under my supervision in the IV semester for the academic year 2019-2020.

Signature Signature

external examine internal examine

Abstract

A hackathon is basically an event, typically lasting several days, in which a large number of people meet to engage in collaborative computer programming in order to solve a real time problem or a simulated problem or a case study usually by building web and mobile services. Now, to facilitate hackathons, one must ensure the smooth management of the event such as gathering the solutions and validating the strength of the solutions provided by the students. Finally, the best possible solution is awarded a prize by the experts/panel of judges. Also the teams must be permitted to participate in the event by their respective colleges/universities. At the end of the hackathon, all the participants are given a certificate of participation. So, this project basically deals with managing a hackathon efficiently. It’s implemented using SQL(back end) and JAVA(front end).

REQUIREMENT ANALYSIS

List of tables:

- HACKATHON
- STUDENTS
- COLLEGES
- EXPERT
- RESULTS
- PARTICIPATE
- STUDY
- PRESENTS
- REWARDS

List of attributes with their domain types:

HACKATHON :

Team id:team_id -Number()

duration:duration-varchar()

type-varchar(20)

STUDENTS:

student id: sid -number(10)

student name: sname-varchar(20)

branch-varchar(15)

COLLEGES:

college id: cid-number(5)

college address-varchar(5)

college name: cname -varchar(20)

DBMS Mini Project
Title: Hackathon Contest

EXPERT :

expert id: eid-number(5)

expert name=ename-varchar(15)

qualification-varchar(30)

RESULTS :

student id:sid -number(5)

score-number(20)

certificate_status-varchar2(20)

Date: day-date

PARTICIPATE :

Date : day-date

PRESENTS:

Date : day-date

REWARDS:

Date : day-date

Roll No:1602-18-737-087
Name: T. Pavan Kumar

ARCHITECTURE AND TECHNOLOGY USED:

SOFTWARE USED:

Java Eclipse, Oracle 11g Database, Java SE version 8, SQL Plus.

Java SWING:

Swing is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) – an API for providing a graphical user interface (GUI) for Java programs. Swing was developed to provide a more sophisticated set of GUI components than the earlier AWT. Swing provides a look and feel that emulates the look and feel of several platforms, and also supports a pluggable look and feel that allows applications to have a look and feel unrelated to the underlying platform. It has more powerful and flexible components than AWT. In addition to familiar components such as buttons, check boxes and labels, Swing provides several advanced components such as tabbed panel, scroll panes, trees, tables, and lists.

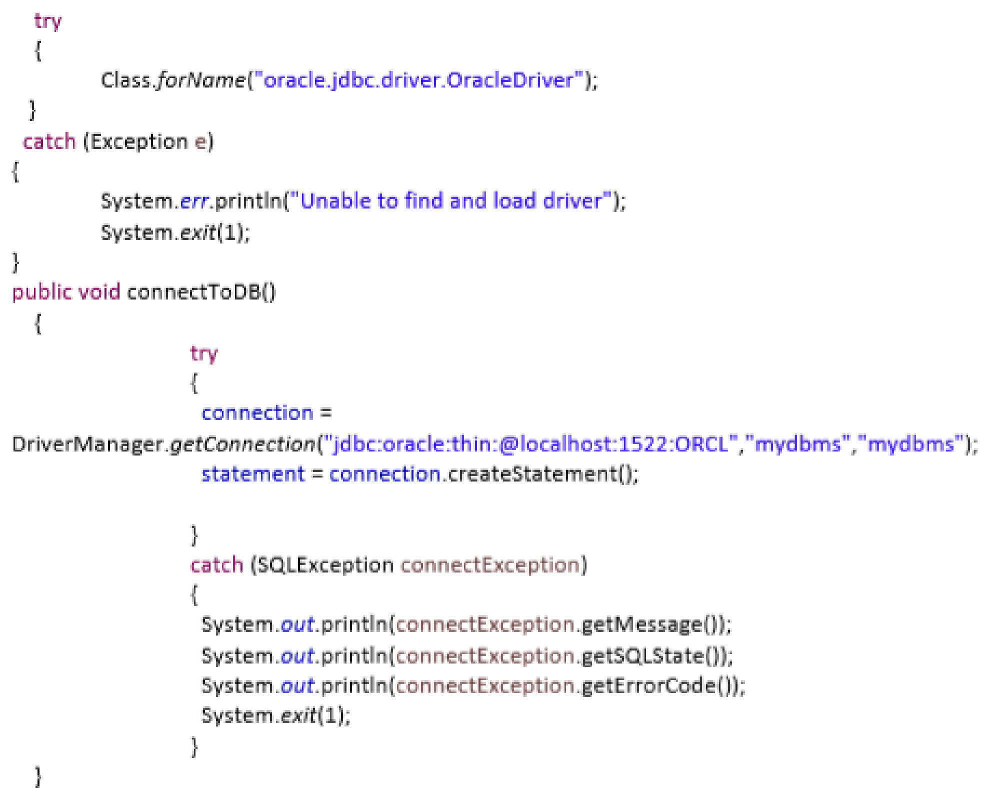
SQL:

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:



```
    try
    {
            Class.forName("oracle.jdbc.driver.OracleDriver");
    }
   catch (Exception e)
  {
            System.err.println("Unable to find and load driver");
            System.exit(1);
  }
  public void connectToDB()
    {
                    try
                    {
                     connection =
  DriverManager.getConnection("jdbc:oracle:thin:@localhost:1522:ORCL","mydbms","mydbms");
                     statement = connection.createStatement();

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

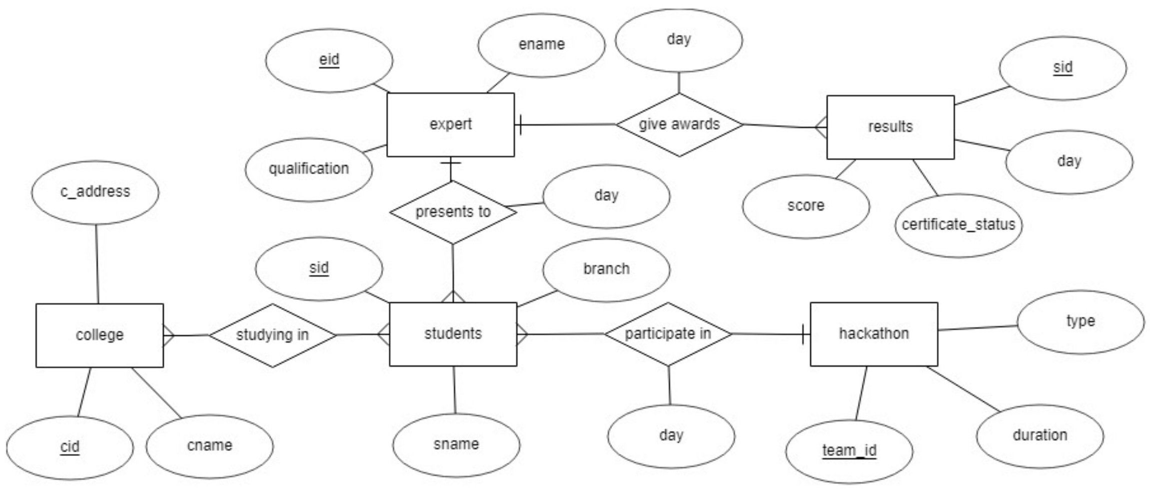
Thus, the connection from Java to Oracle database is performed and therefore, can be used for updating tables in the database directly.

DDL COMMADS:

```
SQL> create table hackathon(
  2  team_id number(10) primary key,

  3  duration varchar2(20),

  4  type char(50));


Table created.

SQL> create table students(

  2  sid number(5) primary key,

  3  sname varchar2(20),

  4  branch varchar2(20));


Table created.

SQL> create table colleges(

  2  c_address varchar2(20),

  3  cname varchar2(20),

  4  cid number(10)) primary key;


Table created.

SQL> create table expert(

  2  eid number(10) primary key,

  3  ename varchar2(20),

  4  qualification varchar2(20));


Table created.

SQL> create table results(

  2  sid number(10) primary key,

  3  day date,

  4  certificate_status varchar2(10),
  5  score number(20));


Table created.

SQL> ed

Wrote file afiedt.buf


  1  create table participate(

  2  team_id number(10),

  3  sid number(10),

  4  primary key(team_id,sid),

  5  foreign key(team_id)references hackathon(team_id),

  6* foreign key(sid)references students(sid))

SQL> /


Table created.

SQL> create table study(

  2  sid number(10),

  3  cid number(10),

  4  primary key(sid,cid),

  5  foreign key(sid)references students(sid),

  6  foreign key(cid)references colleges(cid));


Table created.

SQL> create table presents(

  2  sid number(10),

  3  eid number(10),

  4  primary key(sid,eid),
  5  foreign key(sid)references students(sid),

  6  foreign key(eid)references expert(eid));


Table created.

SQL> alter table participate add(day date);


Table altered.


SQL> alter table presents add(day date);


Table altered.

SQL> create table rewards(

  1  day date,

  2  eid number(10),

  3  sid number(10),

  4  primary key(eid,sid),

  5  foreign key(eid)refereces expert(eid),

  6  foreign key(sid)references results(sid));


Table created.
```

DBMS Mini Project
Title: Hackathon Contest

ER DIAGRAM:



Database Design:

```
SQL> select * from tab;


TNAME                    TABTYPE  CLUSTERID

------------------------------ ------- ----------

COLLEGES                   TABLE

EXPERT                   TABLE

HACKATHON                  TABLE

PARTICIPATE                TABLE
```

Roll No:1602-18-737-087
Name: T. Pavan Kumar

```
PRESENTS                    TABLE

RESULTS                   TABLE

REWARDS                    TABLE

STUDENTS                    TABLE

STUDY                   TABLE


9 rows selected.

SQL> desc hackathon;

 Name                                      Null?    Type

 ----------------------------------------- -------- ----------------------------

 TEAM_ID                                    NOT NULL NUMBER(10)

 DURATION                                            VARCHAR2(20)

 TYPE                                           CHAR(50)


SQL> desc students;

 Name                                      Null?    Type

 ----------------------------------------- -------- ----------------------------

 SID                                       NOT NULL NUMBER(5)

 SNAME                                             VARCHAR2(20)

 BRANCH                                             VARCHAR2(20)


SQL> desc colleges;

 Name                                      Null?    Type

 ----------------------------------------- -------- ----------------------------

 C_ADDRESS                                            VARCHAR2(20)

 CNAME                                             VARCHAR2(20)

 CID                                       NOT NULL NUMBER(10)
```

DBMS Mini Project
Title: Hackathon Contest

```
SQL> desc expert;

 Name                                      Null?    Type

 ----------------------------------------- -------- ----------------------------

 EID                                       NOT NULL NUMBER(10)

 ENAME                                              VARCHAR2(20)

 QUALIFICATION                                      VARCHAR2(20)


SQL> desc rewards;

 Name                                      Null?    Type

 ----------------------------------------- -------- ----------------------------

 EID                                       NOT NULL NUMBER(10)

 SID                                       NOT NULL NUMBER(10)

 DAY                                                DATE


SQL> desc participate;

 Name                                      Null?    Type

 ----------------------------------------- -------- ----------------------------

 TEAM_ID                                   NOT NULL NUMBER(10)

 SID                                       NOT NULL NUMBER(10)

 DAY                                                DATE


SQL> desc study;

 Name                                      Null?    Type

 ----------------------------------------- -------- ----------------------------

 SID                                       NOT NULL NUMBER(10)

 CID                                       NOT NULL NUMBER(10)
```

Roll No:1602-18-737-087
Name: T. Pavan Kumar

```
SQL> desc presents;
 Name                          Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                          NOT NULL NUMBER(10)
 EID                          NOT NULL NUMBER(10)
 DAY                                   DATE
```

```
SQL> desc results;
 Name                          Null?    Type
 ----------------------------------------- -------- ----------------------------
 SID                          NOT NULL NUMBER(10)
 DAY                                   DATE
in CERTIFICATE_STATUS                          VARCHAR2(10)
 SCORE                                  NUMBER(20)
```

Implementation:

Program:

User Interface:

```
import java.awt.*;

import java.awt.event.*;

import javax.swing.*;

import college.*;

import expert.*;

import hackathon.*;

import results.*;

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

         AddCollege addC;

         UpdateCollege upC;

         DeleteCollege delC;

         AddExpert addE;

         UpdateExpert upE;

         DeleteExpert delE;

         AddHackathon addH;

         UpdateHackathon upH;

         DeleteHackathon delH;

         AddResults addR;

         DeleteResults delR;

         UpdateResults upR;

         AddStudents addS;

         UpdateStudents upS;

         DeleteStudents 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 AddCollege();addC.buildGUI();

			upC = new UpdateCollege();  upC.buildGUI();

			delC = new DeleteCollege();		delC.buildGUI();

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

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

			delE=new DeleteExpert();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 AddStudents();addS.buildGUI();
upS = new UpdateStudents();upS.buildGUI();
delS = new DeleteStudents();delS.buildGUI();
//add all the panels to the home panel which has a cardlayout
home.add(welcome, "Welcome");
home.add(addC, "Add College");
home.add(upC, "Update College");
home.add(delC, "Delete College");
home.add(addE, "Add Expert");
home.add(upE, "Update Expert");
home.add(delE,"Delete Expert");
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 Students");
home.add(upS,"Delete Students");
// 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 College = new Menu("College");
MenuItem item1, item2, item3;
College.add(item1 = new MenuItem("Add College"));
College.add(item2 = new MenuItem("View College"));
College.add(item3 = new MenuItem("Delete College"));
mbar.add(College);

Menu Expert = new Menu("expert");
MenuItem item4, item5, item6;
Expert.add(item4 = new MenuItem("Add Expert"));
Expert.add(item5 = new MenuItem("View Expert"));
Expert.add(item6 = new MenuItem("Delete Expert"));
mbar.add(Expert);

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 Students = new Menu("Students");

MenuItem item13, item14, item15;

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

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

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

mbar.add(Students);


// 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 College"))

                    {

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

        }


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

                    {

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

                            upC.loadColleges();

                    }


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

                    {

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

                            delC.loadColleges();

                    }


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

                    {

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

                    }

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

                    {

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

                            upE.loadExperts();
```

```
		}
		else if(arg.equals("Delete Expert"))
		{
			cardLO.show(home, "Delete Expert");
			delE.loadExperts();
		}
		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 Students"))

            {

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

            }

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

            {

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

                delS.loadStudents();

            }

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

            {

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

                upS.loadStudents();

            }


        }

        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();
```

```
            }


}
```

GUI For Insert in College Table:

```
package college;


import java.awt.*;


import java.awt.event.*;

import java.sql.*;

public class AddCollege extends Panel{

        /**

         *

         */

        private static final long serialVersionUID = 5726382096160244564L;

        Button AddCollegeButton;

        TextField cidText,cnameText,addressText;

        TextArea errorText;

        Connection connection;

        Statement statement;

        public AddCollege()

        {

                try

                {

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

                }
```

```
                    catch (Exception e)

                    {

                              System.err.println("Unable to find and l"

                                                  + ""

                                                  + ""

                                                  + ""

                                                  + "oad driver");

                              System.exit(1);

                    }

                    connectToDB();


          }


          public void connectToDB()

   {

                    try

                    {

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

                      statement = connection.createStatement();

                      statement.executeUpdate("commit");


                    }

                    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

               AddCollegeButton = new Button("Add College");

               AddCollegeButton.addActionListener(new ActionListener()

               {

                      public void actionPerformed(ActionEvent e)

                      {

                             try

                             {


                               String query= "INSERT INTO
colleges(C_Address,CNAME,CID) VALUES('"+ addressText.getText() + "', " + "'" +
cnameText.getText() +"',"+cidText.getText()+")";

                               int i = statement.executeUpdate(query);

                               statement.executeUpdate("commit");

                               errorText.append("\nInserted " + i + " rows successfully");

                             }

                             catch (SQLException insertException)

                             {

                               displaySQLErrors(insertException);
```

```
                    }

                }

            });

            cidText=new TextField(15);

            cnameText = new TextField(15);

            addressText = 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("College ID:"));

            first.add(cidText);

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

            first.add(cnameText);

            first.add(new Label("College Address:"));

            first.add(addressText);

            first.setBounds(125,90,200,100);


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

            second.add(AddCollegeButton);

    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);

                setSize(500, 600);

                setVisible(true);

                System.out.println("hello");

        }




        private void displaySQLErrors(SQLException e)

        {

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

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

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

        }

}
```

GUI For Update in College Table:

```
package college;
```

```
import java.awt.*;


import java.awt.event.*;

import java.sql.*;

@SuppressWarnings("serial")

public class UpdateCollege extends Panel{

        Button updateCollegeButton;

        List collegeIDList;

        TextField cidText, cnameText, c_addressText;

        TextArea errorText;

        Connection connection;

        Statement statement;

        ResultSet rs;


        public UpdateCollege()

        {

                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","pavan","pavan");

			  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 loadColleges()

		{

			//try

		//	{

				try {
```

```
                                        collegeIDList.removeAll();
 rs = statement.executeQuery("SELECT CID FROM colleges");
 while (rs.next())
 {
                                        collegeIDList.add(rs.getString("CID"));
 }
                              } catch (SQLException e) {
                                        // TODO Auto-generated catch block
                                        e.printStackTrace();
                                        errorText.append("\nSQLException: " + e.getMessage() +
"\n");
                                        errorText.append("SQLState:     " + e.getSQLState() + "\n");
                                        errorText.append("VendorError:  " + e.getErrorCode() +
"\n");
                              }
                    //}
                    //catch (SQLException e)
                    //{
                     // displaySQLErrors(e);
                    //}
          }

          public void buildGUI()
          {
             collegeIDList = new List(10);
                    loadColleges();
```

```
		add(collegeIDList);


		collegeIDList.addItemListener(new ItemListener()

		{

			public void itemStateChanged(ItemEvent e)

			{

				try

				{

					rs = statement.executeQuery("SELECT * FROM
colleges where CID ="+collegeIDList.getSelectedItem());

					rs.next();

					cidText.setText(rs.getString("CID"));

					cnameText.setText(rs.getString("CNAME"));

					c_addressText.setText(rs.getString("c_address"));

				}

				catch (SQLException selectException)

				{

					displaySQLErrors(selectException);

				}

			}

		});



		//Handle Update Sailor Button

		updateCollegeButton = new Button("Update College");

		updateCollegeButton.addActionListener(new ActionListener()
		{
			public void actionPerformed(ActionEvent e)
			{
				try
				{
					Statement statement =
connection.createStatement();
					int i = statement.executeUpdate("UPDATE colleges
"
					+ "SET c_address ='"+ c_addressText.getText() + "',
"
					+ " name='" + cnameText.getText() + "' WHERE cid
= "
					+ collegeIDList.getSelectedItem());
					errorText.append("\nUpdated " + i + " rows
successfully");
					i = statement.executeUpdate("commit");
					loadColleges();
				}
				catch (SQLException insertException)
				{
					displaySQLErrors(insertException);
				}
			}
		});


		cidText = new TextField(15);
	//	cidText.setEditable(false);
```

```
		cnameText = new TextField(15);

		c_addressText = 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("College ID:"));

		first.add(cidText);

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

		first.add(cnameText);

		first.add(new Label("Collge Address"));

		first.add(c_addressText);


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

		second.add(updateCollegeButton);


		Panel third = new Panel();

		third.add(errorText);

		add(first);

		add(second);

		add(third);


		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");

	}



}
```

GUI For Delete in College Table:

```
package college;


import java.awt.*;


import java.awt.event.*;

import java.sql.*;




@SuppressWarnings("serial")

public class DeleteCollege extends Panel {

	//private static final List collegesIDList = null;

	Button deleteCollegeButton;

	List collegesIDList;
```

```
        TextField cidText, cnameText, c_addressText;

        TextArea errorText;

        Connection connection;

        Statement statement;

        ResultSet rs;


        public DeleteCollege()

        {

                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","pavan","pavan");
```

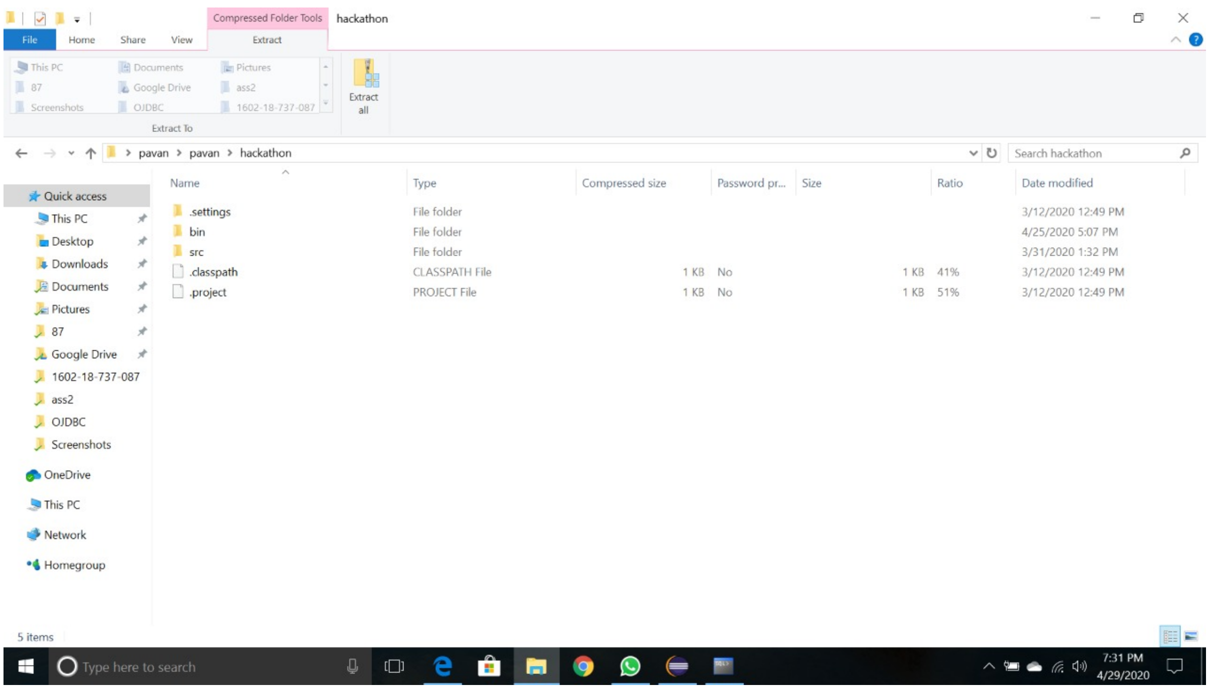
```
                        statement = connection.createStatement();


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


            public void loadColleges()
            {
                      try
                      {



                                collegesIDList.removeAll();
                        rs = statement.executeQuery("SELECT * FROM colleges");
                        while (rs.next())
                        {
                                collegesIDList.add(rs.getString("CID"));
                        }
                      }
                      catch (SQLException e)
                      {
                                e.printStackTrace();
                      errorText.append("\nSQLException: " + e.getMessage() + "\n");
                      errorText.append("SQLState:     " + e.getSQLState() + "\n");
                      errorText.append("VendorError:  " + e.getErrorCode() + "\n");
                      }
            }


            public void buildGUI()
            {
               collegesIDList = new List(10);
                      loadColleges();
                      add(collegesIDList);


                      //When a list item is selected populate the text fields
                      collegesIDList.addItemListener(new ItemListener()
                      {


                                public void itemStateChanged(ItemEvent e)
                                {
                                        try
                                        {
                                                rs = statement.executeQuery("SELECT * FROM
            colleges");
                                                while (rs.next())
                                                {
                                                    if
            (rs.getString("CID").equals(collegesIDList.getSelectedItem()))

                                                    break;

                                                }

                                                if (!rs.isAfterLast())

                                                {

                                                    cidText.setText(rs.getString("CID"));

                                                    cnameText.setText(rs.getString("CNAME"));


            c_addressText.setText(rs.getString("C_Address"));

                                                }

                                        }

                                        catch (SQLException selectException)

                                        {

                                                displaySQLErrors(selectException);

                                        }

                                }

                      });




                      deleteCollegeButton = new Button("Delete College");

                      deleteCollegeButton.addActionListener(new ActionListener()

                      {

                                public void actionPerformed(ActionEvent e)

                                {

                                        try
```

```
                    {


                        Statement statement =
connection.createStatement();
                        int i = statement.executeUpdate("DELETE FROM
colleges WHERE CID = "
                            + collegesIDList.getSelectedItem());
                        errorText.append("\nDeleted " + i + " rows
successfully");
                        cidText.setText(null);
                        cnameText.setText(null);
                        c_addressText.setText(null);
                        statement.executeUpdate("commit");
                        //collegesIDList.removeAll();
                        loadColleges();
                    }
                    catch (SQLException insertException)
                    {
                        displaySQLErrors(insertException);
                    }
                }
            });

            cidText = new TextField(15);
            cnameText = new TextField(15);
            c_addressText = new TextField(15);
```

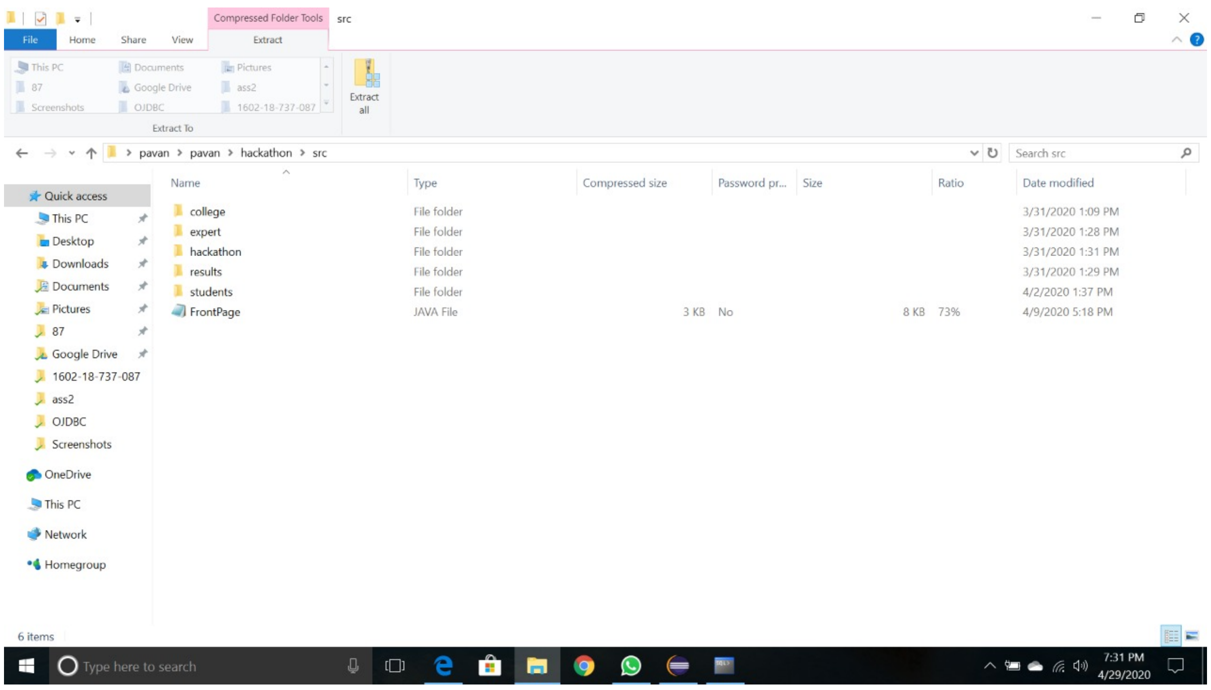
```
errorText = new TextArea(10, 40);

errorText.setEditable(false);


Panel first = new Panel();

first.setLayout(new GridLayout(4, 2));

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

first.add(cidText);

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

first.add(cnameText);

first.add(new Label("College Address:"));

first.add(c_addressText);


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

second.add(deleteCollegeButton);


Panel third = new Panel();

third.add(errorText);


add(first);

add(second);

add(third);


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");
        }



}
```

GitHub links and folder structure:



Name	Type	Compressed size	Password pr...	Size	Ratio	Date modified
.settings	File folder					3/12/2020 12:49 PM
bin	File folder					4/25/2020 5:07 PM
src	File folder					3/31/2020 1:32 PM
.classpath	CLASSPATH File	1 KB	No	1 KB	41%	3/12/2020 12:49 PM
.project	PROJECT File	1 KB	No	1 KB	51%	3/12/2020 12:49 PM

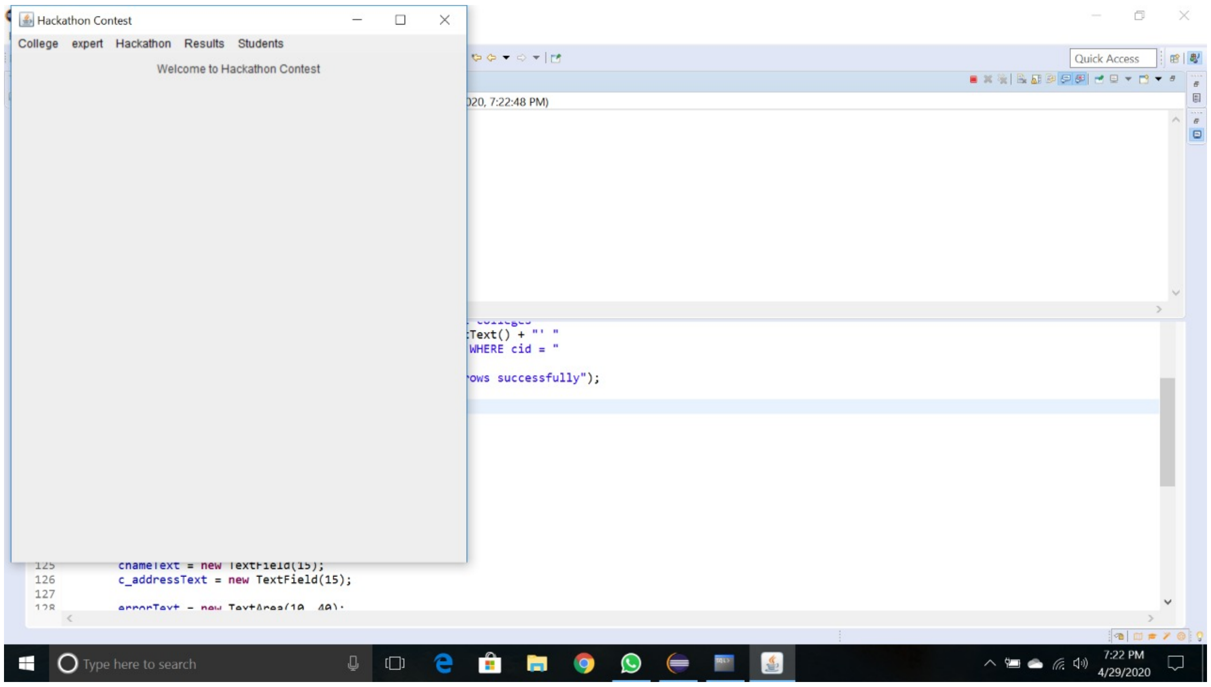


Name	Type	Compressed size	Password pr...	Size	Ratio	Date modified
college	File folder					3/31/2020 1:09 PM
expert	File folder					3/31/2020 1:28 PM
hackathon	File folder					3/31/2020 1:31 PM
results	File folder					3/31/2020 1:29 PM
students	File folder					4/2/2020 1:37 PM
FrontPage	JAVA File	3 KB	No	8 KB	73%	4/9/2020 5:18 PM

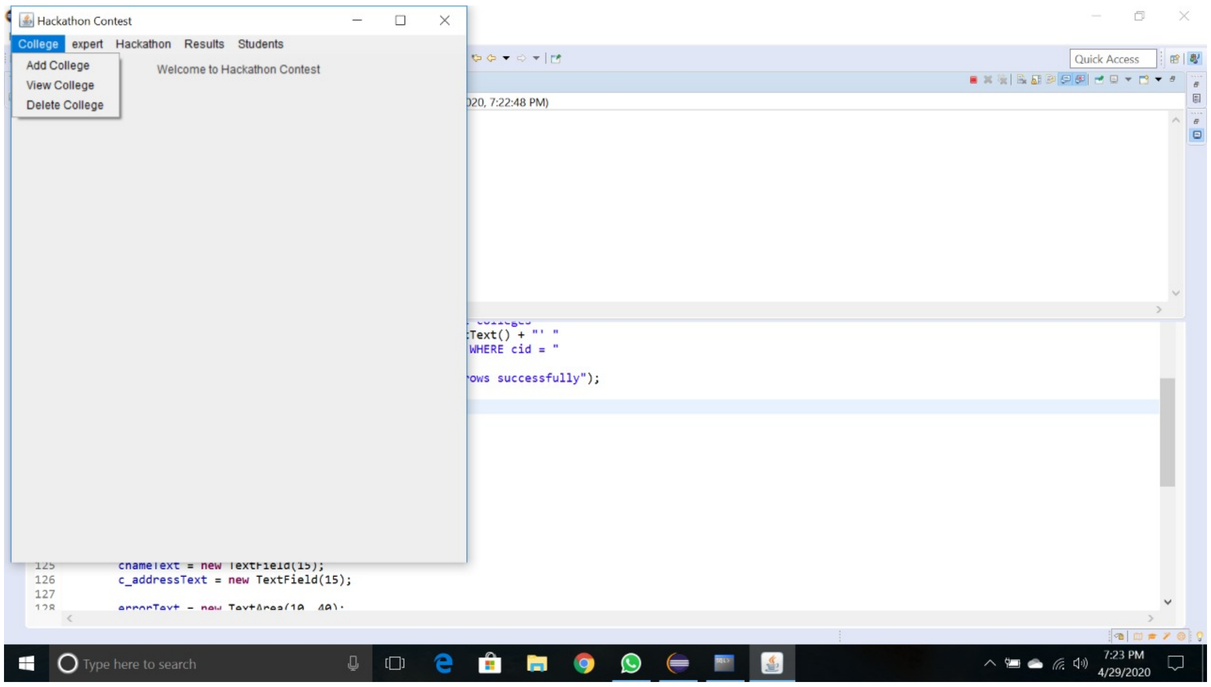
Testing:

Java GUI Testing:

DBMS Mini Project
Title: Hackathon Contest



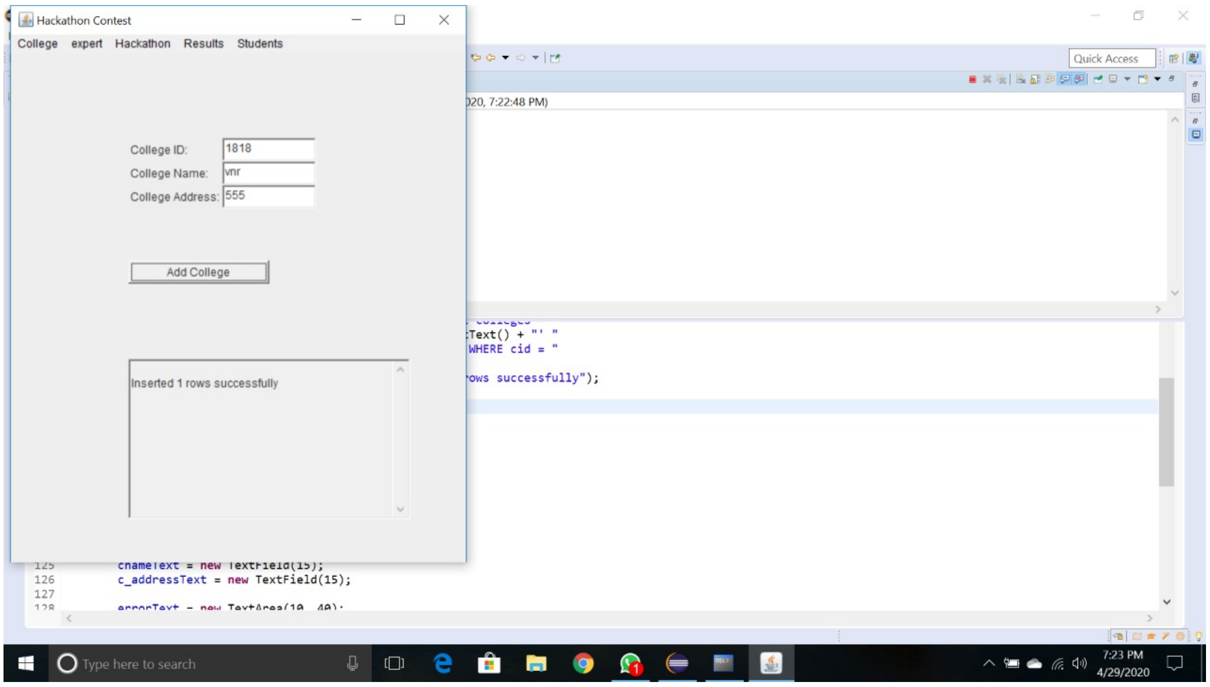
```
020, 7:22:48 PM)
:Text() + "' "
WHERE cid = "
rows successfully");
        cnameText = new TextField(15);
        c_addressText = new TextField(15);

        errorText = new TextArea(10, 40);
```



```
020, 7:22:48 PM)
:Text() + "' "
WHERE cid = "
rows successfully");
        cnameText = new TextField(15);
        c_addressText = new TextField(15);

        errorText = new TextArea(10, 40);
```

Roll No:1602-18-737-087
Name: T. Pavan Kumar

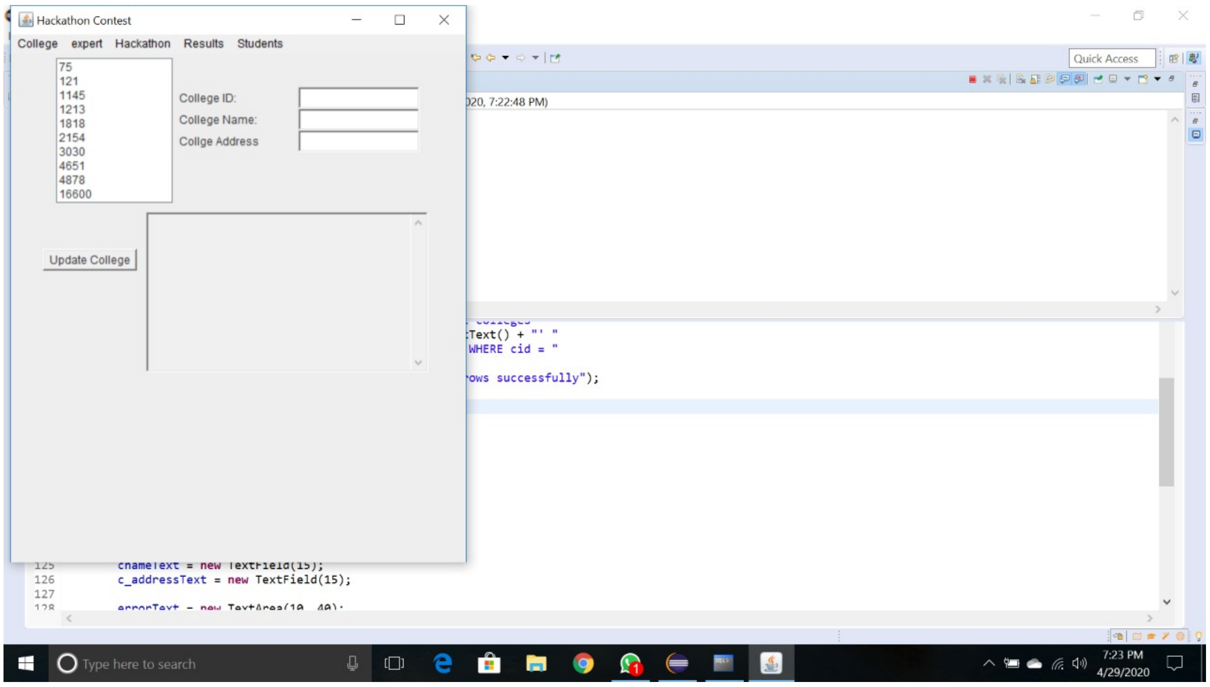
DBMS Mini Project
Title: Hackathon Contest



College ID: 1818
College Name: vnr
College Address: 555

Inserted 1 rows successfully

```
:Text() + "' "
 WHERE cid = "
ows successfully");
        cnameText = new TextField(15);
        c_addressText = new TextField(15);
        errorText = new TextArea(10, 40);
```



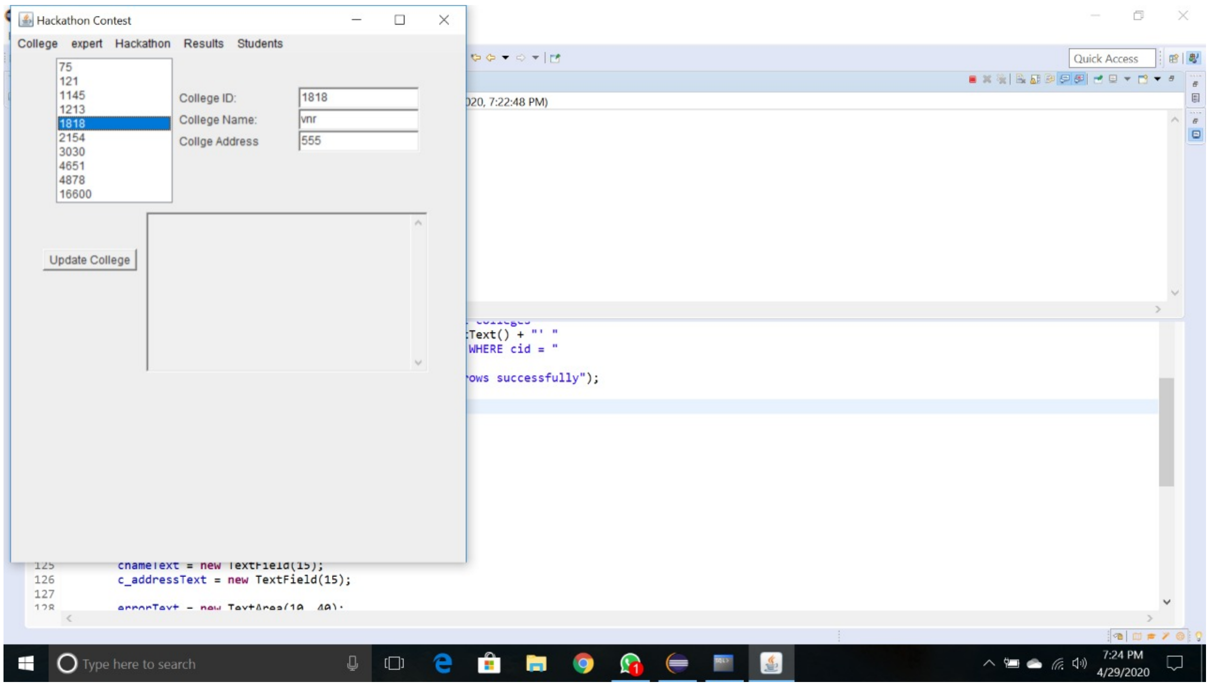
```
75
121
1145
1213
1818
2154
3030
4651
4878
16600
```

College ID:
Collge Name:
Collge Address

```
:Text() + "' "
 WHERE cid = "
ows successfully");
        cnameText = new TextField(15);
        c_addressText = new TextField(15);
        errorText = new TextArea(10, 40);
```

Roll No:1602-18-737-087
Name: T. Pavan Kumar

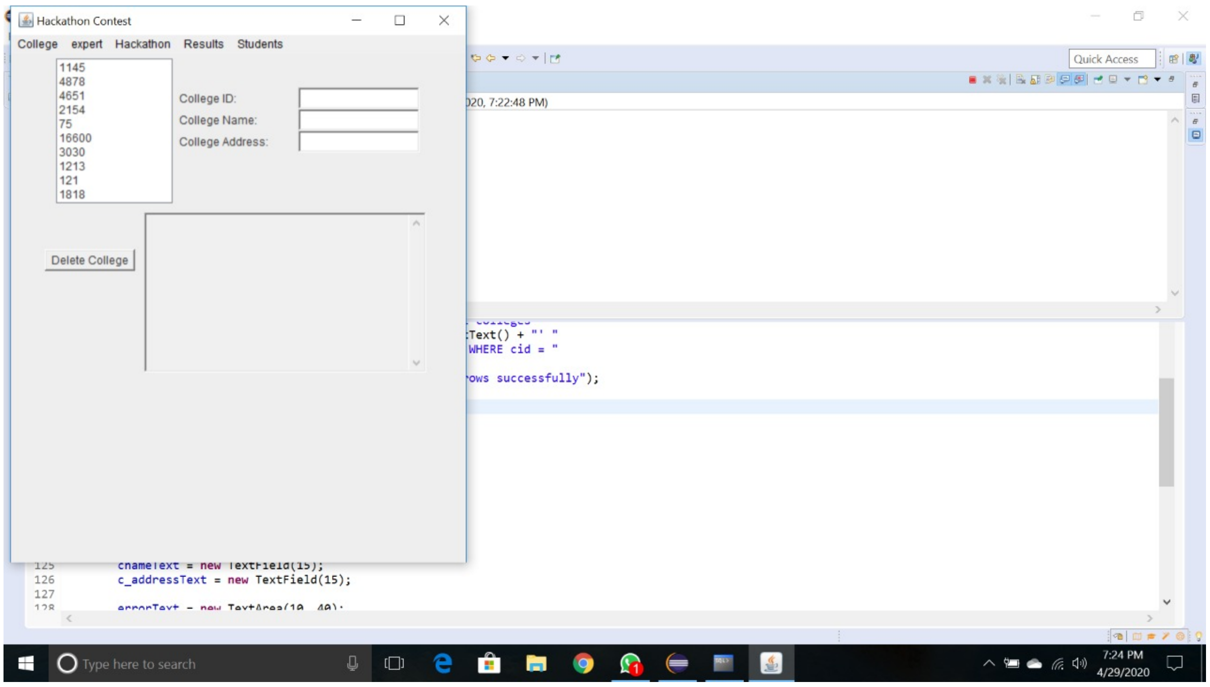
DBMS Mini Project
Title: Hackathon Contest



```
75
121
1145
1213
1818
2154
3030
4651
4878
16600
```

College ID: 1818
College Name: vnr
Collge Address 555

```
020, 7:22:48 PM)
:Text() + "' "
 WHERE cid = "
rows successfully");
        c_addressText = new TextField(15);
```



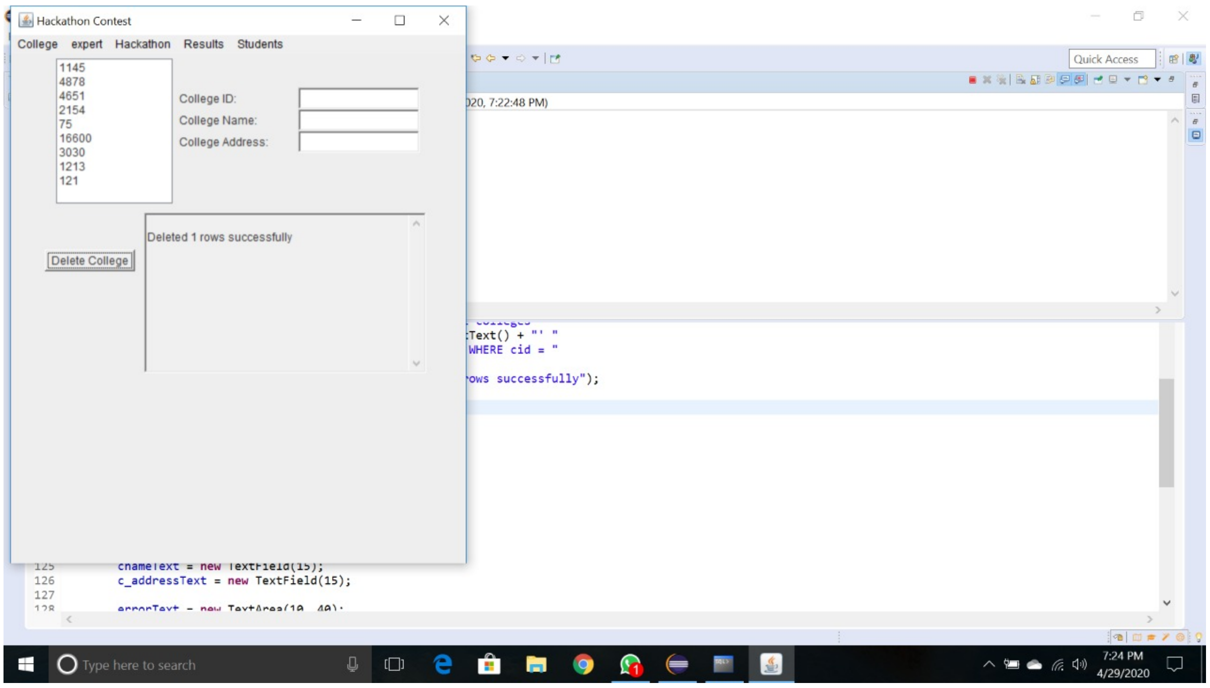
```
1145
4878
4651
2154
75
16600
3030
1213
121
1818
```

College ID:
College Name:
College Address:

```
020, 7:22:48 PM)
:Text() + "' "
 WHERE cid = "
rows successfully");
        c_addressText = new TextField(15);
```

Roll No:1602-18-737-087
Name: T. Pavan Kumar

DBMS Mini Project
Title: Hackathon Contest



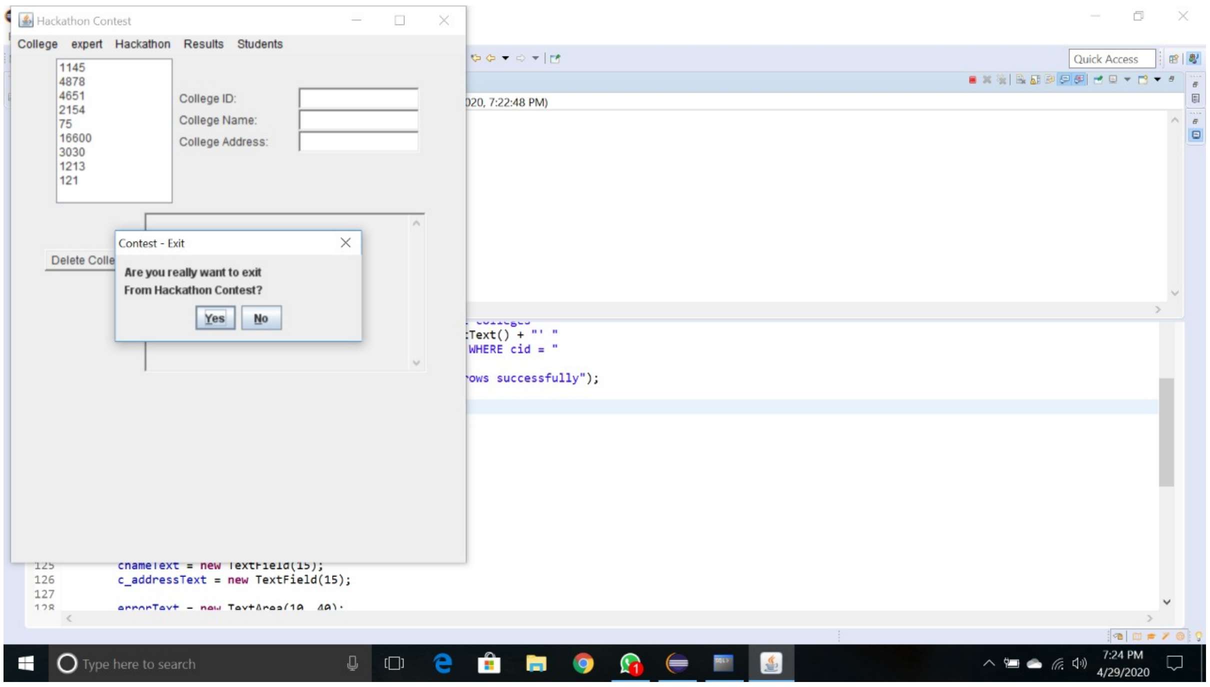
```
1145
4878
4651
2154
75
16600
3030
1213
121
```

```
Deleted 1 rows successfully
```

```
020, 7:22:48 PM)
```

```
:Text() + "' "
 WHERE cid = "

rows successfully");
```

```
        cnameText = new TextField(15);
        c_addressText = new TextField(15);

        errorText = new TextArea(10, 40);
```



```
1145
4878
4651
2154
75
16600
3030
1213
121
```

Are you really want to exit
From Hackathon Contest?

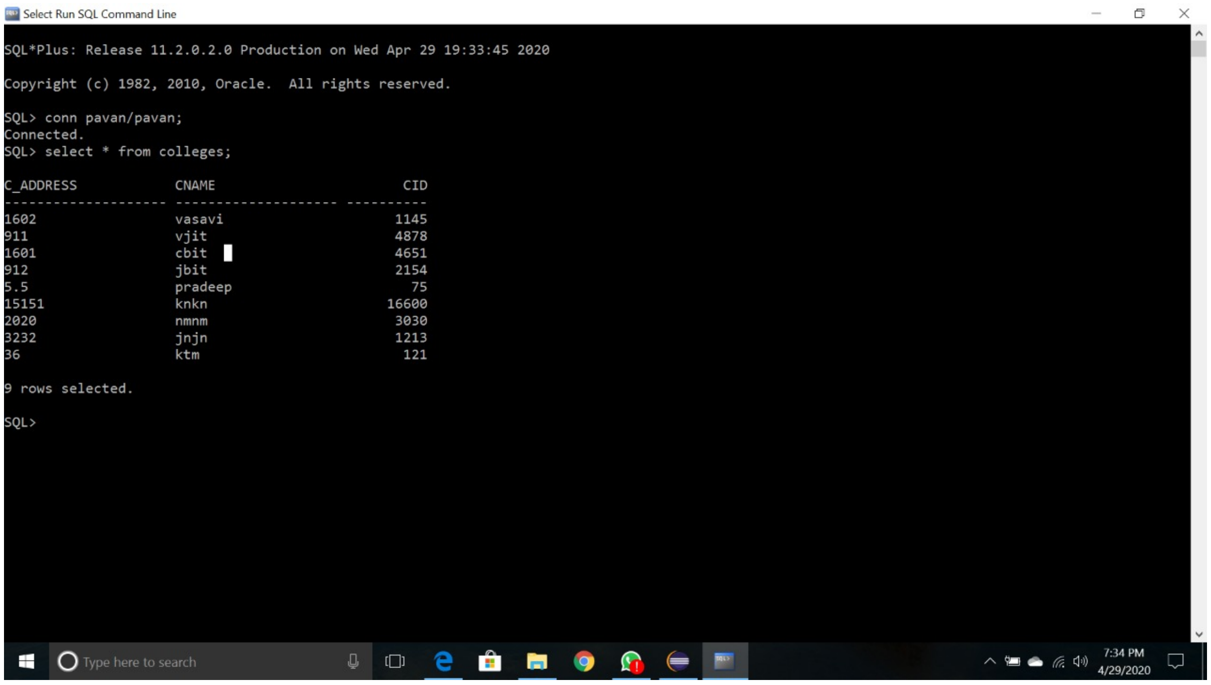
```
020, 7:22:48 PM)
```

```
:Text() + "' "
 WHERE cid = "

rows successfully");
```

```
        cnameText = new TextField(15);
        c_addressText = new TextField(15);

        errorText = new TextArea(10, 40);
```

Roll No:1602-18-737-087
Name: T. Pavan Kumar

The data entered in the above form is updated in the “college” table of the Oracle database 11g as:



```
SQL*Plus: Release 11.2.0.2.0 Production on Wed Apr 29 19:33:45 2020

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> conn pavan/pavan;
Connected.
SQL> select * from colleges;

C_ADDRESS            CNAME                       CID
-------------------- -------------------- ----------
1602                 vasavi                     1145
911                  vjit                       4878
1601                 cbit                       4651
912                  jbit                       2154
5.5                  pradeep                      75
15151                knkn                      16600
2020                 nmnm                       3030
3232                 jnjn                       1213
36                   ktm                         121

9 rows selected.

SQL>
```

Results:

I successfully completed this MINI PROJECT “Hackathon Contest”.

Discussion and Future work

While doing this project I got new ideas I understood how to work on projects. Now to further extend this project I want to create a android app by which I can control my project on my hand and connect to it. This project efficiently stores the data in tables and we can manipulate it easily by friendly userinterface References:

https://www.javatpoint.com/

Roll No:1602-18-737-087
Name: T. Pavan Kumar

DBMS Mini Project
Title: Hackathon Contest

http://www.sqlines.com/articles/java/sql_server_jdbc_connection

https://docs.oracle.com/javase/7/docs/index.html

Roll No:1602-18-737-087
Name: T. Pavan Kumar