RAMAIAH UNIVERSITY OF APPLIED SCIENCES

ASSIGNMENT

Course Code	CSC311A
Course Name	Database Systems
Programme	B.Tech
Department	CSE
Faculty	Ami Rai E.

Name of the Student	Shikhar singh
Reg. No	17ETCS002168
Semester/Year	06/2017
Course Leader/s	Ami Rai E.

Declaration Sheet			
Student Name	Shikhar singh		
Reg. No	17ETCS002168		
Programme	B. Tech	Semester/Year	06/2017
Course Code	CSC311A		
Course Title	Database Systems		
Course Date		to	
Course Leader	Ami Rai E.		

Declaration

The assignment submitted herewith is a result of my own investigations and that I have conformed to the guidelines against plagiarism as laid out in the Student Handbook. All sections of the text and results, which have been obtained from other sources, are fully referenced. I understand that cheating and plagiarism constitute a breach of University regulations and will be dealt with accordingly.

Signature of the Student		Date	
Submission date stamp (by Examination & Assessment Section)			

Signature of the Course Leader and date	Signature of the Reviewer and date

Contents

List of Figures

Solution to Question No. 1:

1.1 Functional and Data Requirements:

Requirement Tag	FR1
Requirement Description	The system should allow the user to register in the system using his student ID
Dependent on Requirements	-
User/System interacting with the requirement	Staff

Requirement Tag	DR1
Item Name	Student Name
Item Description (Where/How used)	The student will be entering his details
Item type	Char
User/System interacting with the item	student
Constraints (if any)	The value should be less than 50 characters.

Requirement Tag	DR2
Item Name	student ID
Item Description (Where/How used)	The student will be entering his student Id.
Item type	Integer
User/System interacting with the item	Student
Constraints (if any)	The value should be an integer number.

Requirement Tag	DR3
Item Name	Password
Item Description (Where/How used)	The student will be entering his password
Item type	Char
User/System interacting with the item	Employee

Constraints (if any)	The value should be combination of characters, digits and special characters.

Requirement Tag	FR2
Requirement Description	The system should allow the registered user to login in the system, using his user ID and password
Dependent on Requirements	FR1
User/System interacting with the requirement	student

Requirement Tag	DR1
Item Name	Student ID
Item Description (Where/How used)	The verification of the student ID from the database.
Item type	Integer
User/System interacting with the item	student
Constraints (if any)	The value should be an integer number.

Requirement Tag	DR2
Item Name	Password
Item Description (Where/How used)	The verification of the student password as in the feed of the database.
Item type	Char
User/System interacting with the item	student
Constraints (if any)	The value should combination of characters, digits and special characters.

Requirement Tag	FR3
Requirement Description	The student should be able to book his project.
Dependent on Requirements	FR1- FR2
User/System interacting with the requirement	student

Requirement Tag	DR1
Item Name	Department Name

Item Description (Where/How used)	Used to distinguish between different Department verbally
Item type	Char
User/System interacting with the item	Head of department.

Requirement Tag	DR2
Item Name	Department Id
Item Description (Where/How used)	Used to distinguish between different Department
Item type	char
User/System interacting with the item	Head of department.

Requirement Tag	FR4
Requirement Description	The Student should be able to join a team.
Dependent on Requirements	
User/System interacting with the requirement	Head of department.

Requirement Tag	DR1
Item Name	Team Name
Item Description (Where/How used)	Used to distinguish between different teams verbally
Item type	Char
User/System interacting with the item	Team leader

Requirement Tag	DR2
Item Name	Team Id
Item Description (Where/How used)	Used to distinguish between different teams
Item type	Integer
User/System interacting with the item	Team leader

Requirement Tag	FR5
Requirement Description	The Project leader should be able to control or take projects.

Dependent on Requirements	FR4
User/System interacting with the requirement	Project leader

Requirement Tag	DR1
Item Name	Project ID
Item Description (Where/How used)	Used to distinguish between different projects
Item type	Integer
User/System interacting with the item	Project leader
Constraints (if any)	The value should be integer.

Requirement Tag	DR2
Item Name	Project Name
Item Description (Where/How used)	Used to distinguish between different projects verbally
Item type	Character
User/System interacting with the item	Project leader

Requirement Tag	DR3
Item Name	Category of project
Item Description (Where/How used)	Used to represent the types of the project
Item type	Character
User/System interacting with the item	Project leader

Requirement Tag	FR6
Requirement Description	The Mentor should be able to control or take projects.
Dependent on Requirements	FR5
User/System interacting with the requirement	Mentor

Requirement Tag	DR1
Item Name	Mentor Name

Item Description (Where/How used)	Used to distinguish between different Mentors
Item type	character
User/System interacting with the item	Mentor

Requirement Tag	DR2
Item Name	Mentor ID
Item Description (Where/How used)	Used to distinguish between different Mentors
Item type	character
User/System interacting with the item	Mentor

An Entity-Relationship Diagram can be used to give a better understanding of the Database and the Relationship between various entities.

The entity relationship diagram for the given problem can be seen in the figure 1.

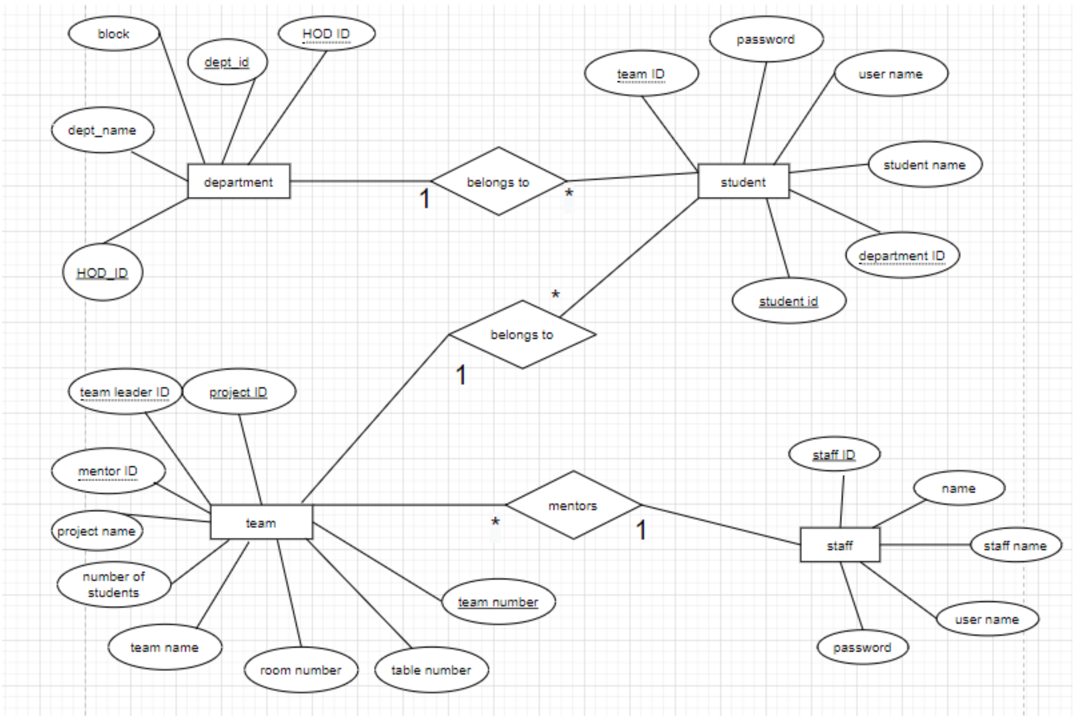
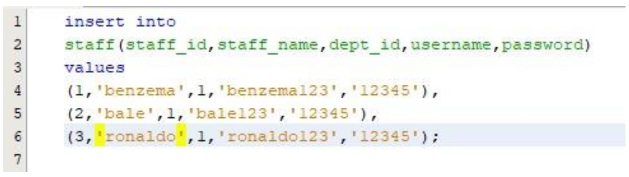


Figure 1: ER diagram

1.2 Implementation of database Tables:

The tables were created using GUI interface provided by netbeans and populated using SQL commands. The code snippets of commands used are given below:

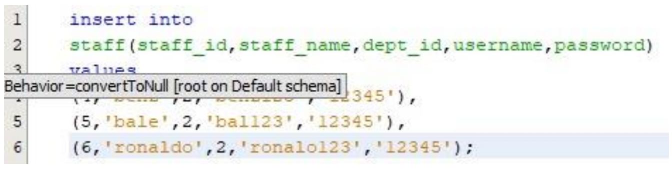
1. Populating staff table



```
insert into
staff(staff_id,staff_name,dept_id,username,password)
values
(1,'benzema',1,'benzema123','12345'),
(2,'bale',1,'bale123','12345'),
(3,'ronaldo',1,'ronaldo123','12345');
```

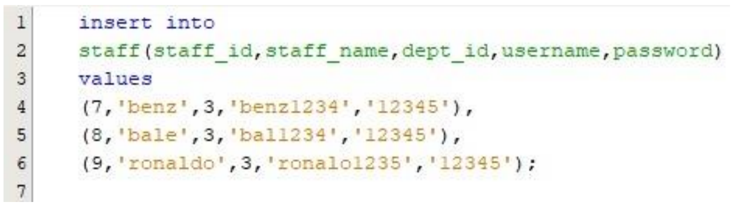
Figure 2: sql command to populate staff table

Here, department id is 1 for CSE, 2 for ECE and 3 for EEE. Figure 2 shows the sql command used to populate staff table for cse department. Similarly, data for other departments was populated. Figure 3 and figure 4 shows the same.



```
insert into
staff(staff_id,staff_name,dept_id,username,password)
values
Behavior=convertToNull [root on Default schema]
345'),
(5,'bale',2,'bal123','12345'),
(6,'ronaldo',2,'ronalo123','12345');
```

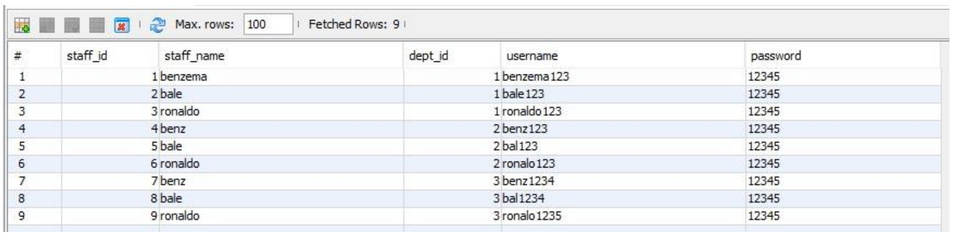
Figure 3: sql command to populate staff table



```
insert into
staff(staff_id,staff_name,dept_id,username,password)
values
(7,'benz',3,'benz1234','12345'),
(8,'bale',3,'bal1234','12345'),
(9,'ronaldo',3,'ronalo1235','12345');
```

Figure 4: sql command to populate staff table

the staff table after being populated looks like this:

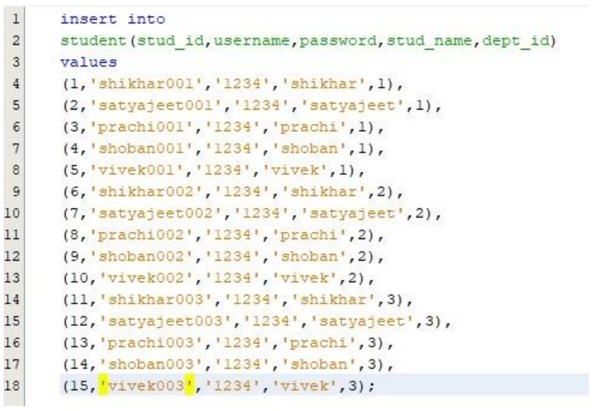


Max. rows: 100 | Fetched Rows: 9

#	staff_id	staff_name	dept_id	username	password
1	1	benzema	1	benzema123	12345
2	2	bale	1	bale123	12345
3	3	ronaldo	1	ronaldo123	12345
4	4	benz	2	benz123	12345
5	5	bale	2	bal123	12345
6	6	ronaldo	2	ronalo123	12345
7	7	benz	3	benz1234	12345
8	8	bale	3	bal1234	12345
9	9	ronaldo	3	ronalo1235	12345

Figure 5: staff table with data

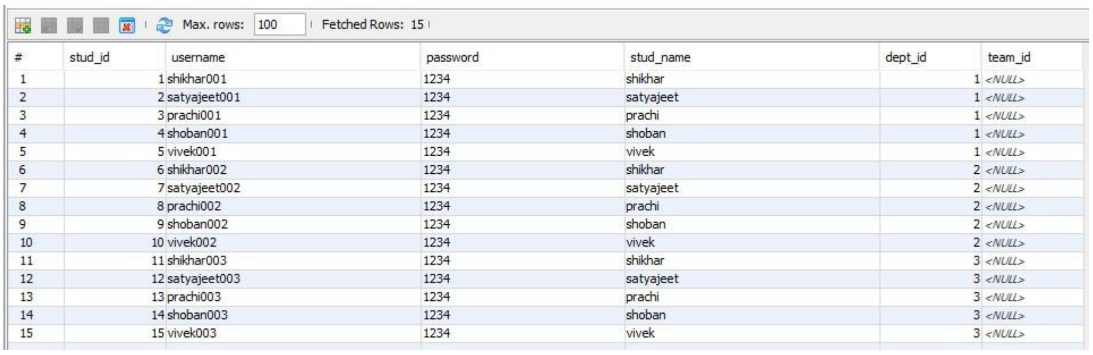
2. Populating student table :



```
insert into
student(stud_id,username,password,stud_name,dept_id)
values
(1,'shikhar001','1234','shikhar',1),
(2,'satyajeet001','1234','satyajeet',1),
(3,'prachi001','1234','prachi',1),
(4,'shoban001','1234','shoban',1),
(5,'vivek001','1234','vivek',1),
(6,'shikhar002','1234','shikhar',2),
(7,'satyajeet002','1234','satyajeet',2),
(8,'prachi002','1234','prachi',2),
(9,'shoban002','1234','shoban',2),
(10,'vivek002','1234','vivek',2),
(11,'shikhar003','1234','shikhar',3),
(12,'satyajeet003','1234','satyajeet',3),
(13,'prachi003','1234','prachi',3),
(14,'shoban003','1234','shoban',3),
(15,'vivek003','1234','vivek',3);
```

Figure 6: sql command to populate student table.

the student table after being populated looks like this:

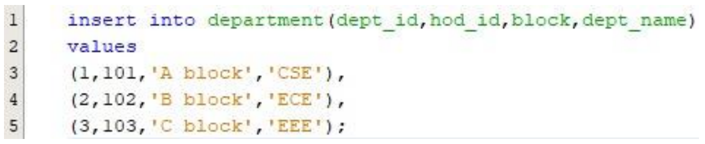


Max. rows: 100 Fetched Rows: 15

#	stud_id	username	password	stud_name	dept_id	team_id
1	1	shikhar001	1234	shikhar	1	<NULL>
2	2	satyajeet001	1234	satyajeet	1	<NULL>
3	3	prachi001	1234	prachi	1	<NULL>
4	4	shoban001	1234	shoban	1	<NULL>
5	5	vivek001	1234	vivek	1	<NULL>
6	6	shikhar002	1234	shikhar	2	<NULL>
7	7	satyajeet002	1234	satyajeet	2	<NULL>
8	8	prachi002	1234	prachi	2	<NULL>
9	9	shoban002	1234	shoban	2	<NULL>
10	10	vivek002	1234	vivek	2	<NULL>
11	11	shikhar003	1234	shikhar	3	<NULL>
12	12	satyajeet003	1234	satyajeet	3	<NULL>
13	13	prachi003	1234	prachi	3	<NULL>
14	14	shoban003	1234	shoban	3	<NULL>
15	15	vivek003	1234	vivek	3	<NULL>

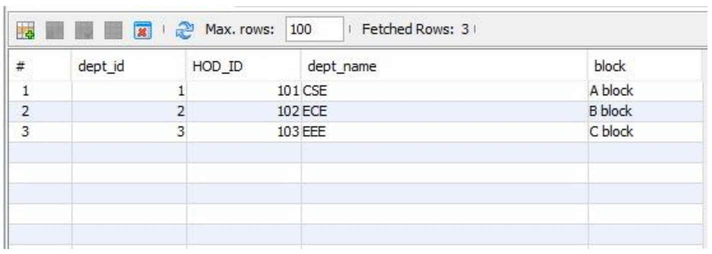
Figure 7: student table with data

3. Populating department table:



```
insert into department(dept_id,hod_id,block,dept_name)
values
(1,101,'A block','CSE'),
(2,102,'B block','ECE'),
(3,103,'C block','EEE');
```

Figure 8: sql command to populate department table



Max. rows: 100 Fetched Rows: 3

#	dept_id	HOD_ID	dept_name	block
1	1	101	CSE	A block
2	2	102	ECE	B block
3	3	103	EEE	C block

Figure 9: department table with data

1.3 Implementation of GUI:

1. Login-screen interface for both student and staff.

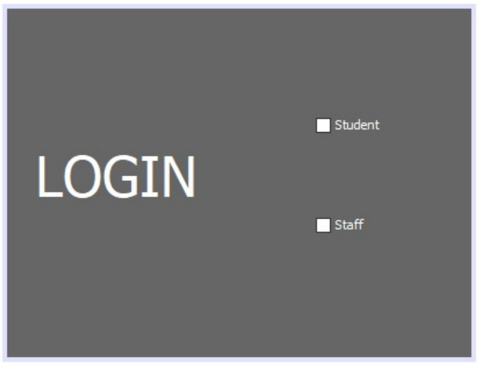
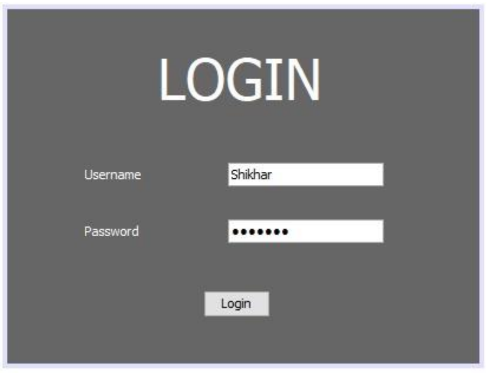


Figure 10: login screen 1

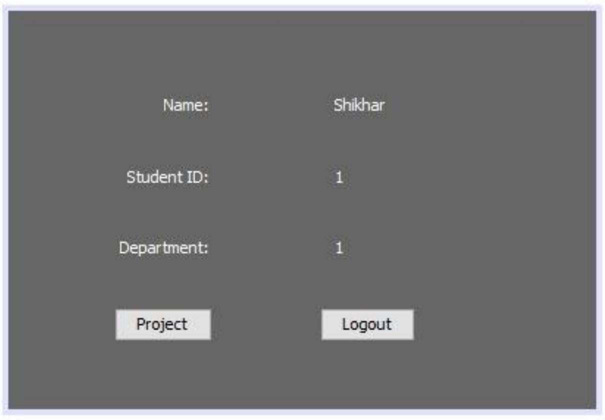

LOGIN

Username: Shikhar

Password: •••••••

Figure 11: login screen after filling data

2. Home page for both student and staff

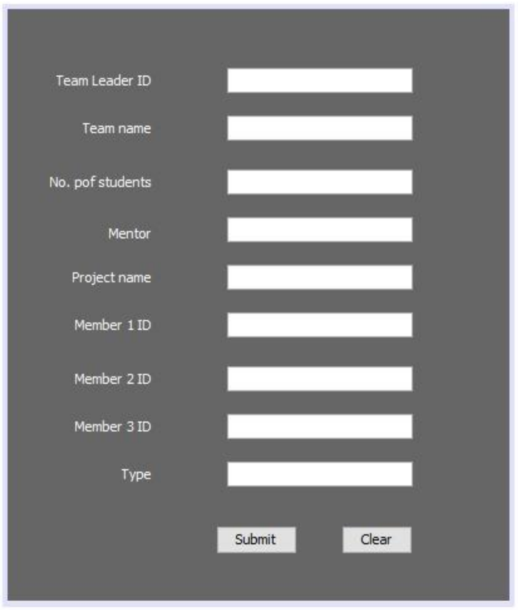

Name: Shikhar

Student ID: 1

Department: 1

Figure 12: student and staff home page

3. GUI for team registration.


Team Leader ID

Team name

No. pof students

Mentor

Project name

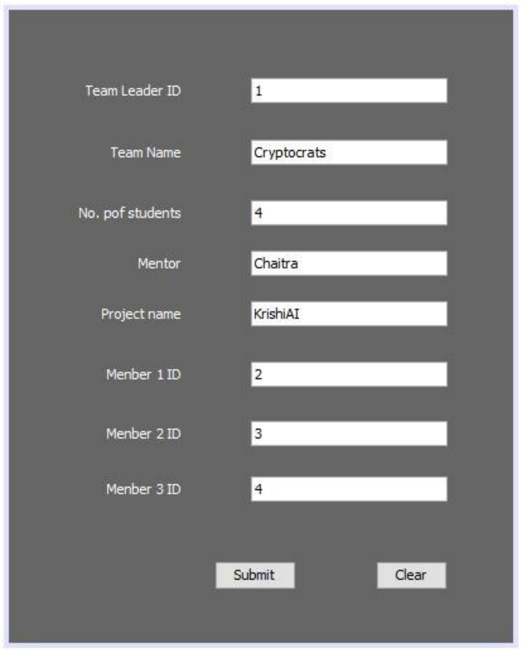
Member 1 ID

Member 2 ID

Member 3 ID

Type

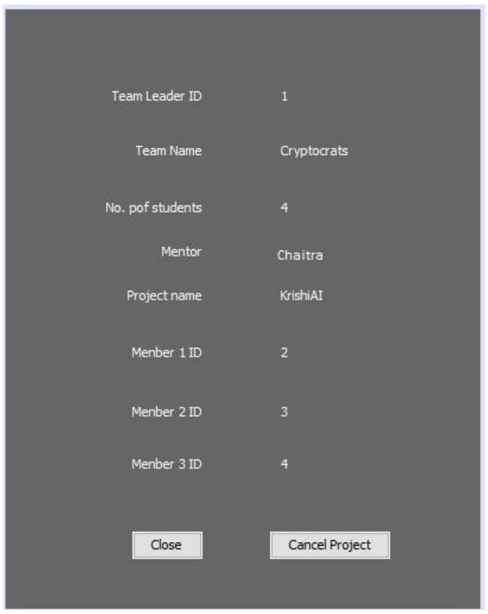
Figure 13: team registration



Team Leader ID	1
Team Name	Cryptocrats
No. pof students	4
Mentor	Chaitra
Project name	KrishiAI
Menber 1 ID	2
Menber 2 ID	3
Menber 3 ID	4

Figure 14: team registration

4. GUI for viewing project details

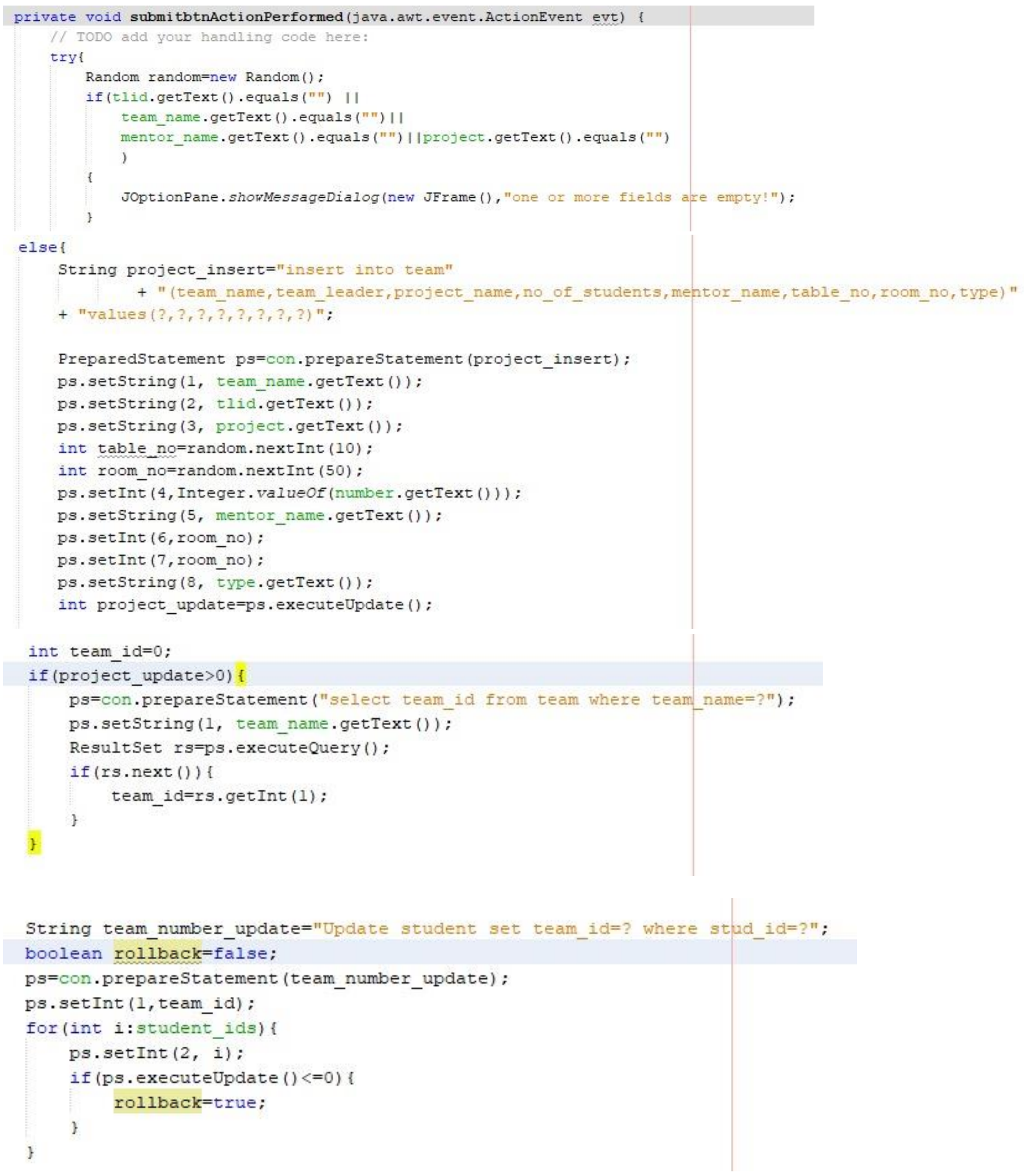


Team Leader ID	1
Team Name	Cryptocrats
No. pof students	4
Mentor	Chaitra
Project name	KrishiAI
Menber 1 ID	2
Menber 2 ID	3
Menber 3 ID	4

Figure 15: Gui for viewing project and cancelation option

1.4 Connection of front end with database:

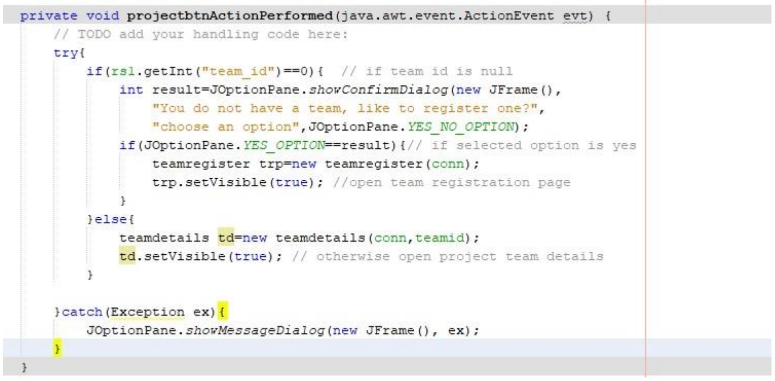
1. Snippet for Submit button on the registration page: passes all the entered values into the database.



```
private void submitbtnActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try{
        Random random=new Random();
        if(tlid.getText().equals("") ||
            team_name.getText().equals("")||
            mentor_name.getText().equals("")||project.getText().equals("")
            )
        {
            JOptionPane.showMessageDialog(new JFrame(),"one or more fields are empty!");
        }
 else{
    String project_insert="insert into team"
            + "(team_name,team_leader,project_name,no_of_students,mentor_name,table_no,room_no,type)"
    + "values(?,?,?,?,?,?,?,?)";

    PreparedStatement ps=con.prepareStatement(project_insert);
    ps.setString(1, team_name.getText());
    ps.setString(2, tlid.getText());
    ps.setString(3, project.getText());
    int table_no=random.nextInt(10);
    int room_no=random.nextInt(50);
    ps.setInt(4,Integer.valueOf(number.getText()));
    ps.setString(5, mentor_name.getText());
    ps.setInt(6,room_no);
    ps.setInt(7,room_no);
    ps.setString(8, type.getText());
    int project_update=ps.executeUpdate();

  int team_id=0;
  if(project_update>0){
      ps=con.prepareStatement("select team_id from team where team_name=?");
      ps.setString(1, team_name.getText());
      ResultSet rs=ps.executeQuery();
      if(rs.next()){
          team_id=rs.getInt(1);
      }
  }


  String team_number_update="Update student set team_id=? where stud_id=?";
  boolean rollback=false;
  ps=con.prepareStatement(team_number_update);
  ps.setInt(1,team_id);
  for(int i:student_ids){
      ps.setInt(2, i);
      if(ps.executeUpdate()<=0){
          rollback=true;
      }
  }
```

Figure 16: code for submit button on registration screen

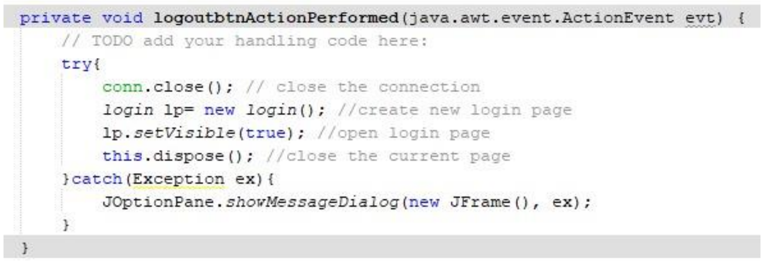
2. Snippet for project button action in student home page



```
private void projectbtnActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try{
        if(rs1.getInt("team_id")==0){  // if team id is null
            int result=JOptionPane.showConfirmDialog(new JFrame(),
                "You do not have a team, like to register one?",
                "choose an option",JOptionPane.YES_NO_OPTION);
            if(JOptionPane.YES_OPTION==result){// if selected option is yes
                teamregister trp=new teamregister(conn);
                trp.setVisible(true); //open team registration page
            }
        }else{
            teamdetails td=new teamdetails(conn,teamid);
            td.setVisible(true); // otherwise open project team details
        }

    }catch(Exception ex){
        JOptionPane.showMessageDialog(new JFrame(), ex);
    }
}
```

Figure 17: code for student home page screen

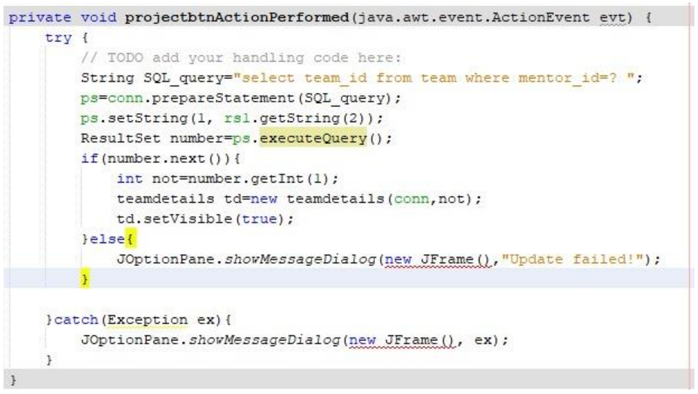
3. Snippet for logout in student home page screen



```java
private void logoutbtnActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try{
        conn.close(); // close the connection
        login lp= new login(); //create new login page
        lp.setVisible(true); //open login page
        this.dispose(); //close the current page
    }catch(Exception ex){
        JOptionPane.showMessageDialog(new JFrame(), ex);
    }
}
```

Figure 18: Code for logout in student home page screen

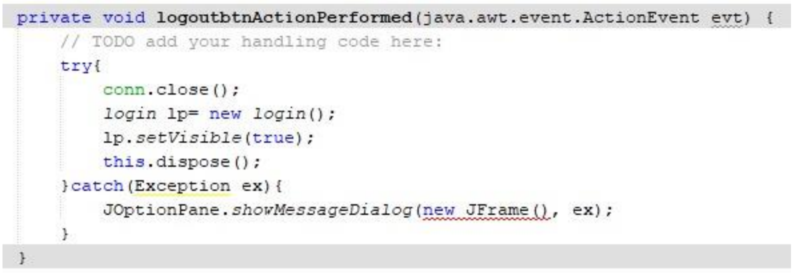
4. Snippet for Project button on staff home page



```java
private void projectbtnActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        // TODO add your handling code here:
        String SQL_query="select team_id from team where mentor_id=? ";
        ps=conn.prepareStatement(SQL_query);
        ps.setString(1, rsl.getString(2));
        ResultSet number=ps.executeQuery();
        if(number.next()){
            int not=number.getInt(1);
            teamdetails td=new teamdetails(conn,not);
            td.setVisible(true);
        }else{
            JOptionPane.showMessageDialog(new JFrame(),"Update failed!");
        }

    }catch(Exception ex){
        JOptionPane.showMessageDialog(new JFrame(), ex);
    }
}
```

Figure 19: project button on staff home page

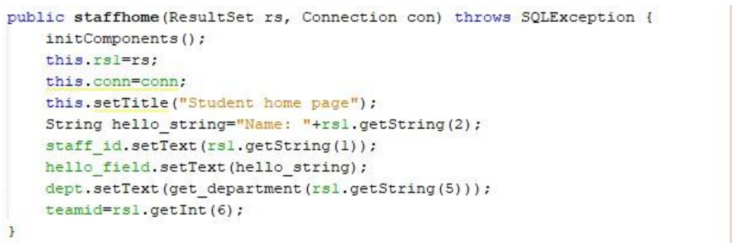
5. Snippet for logout button for staff home page



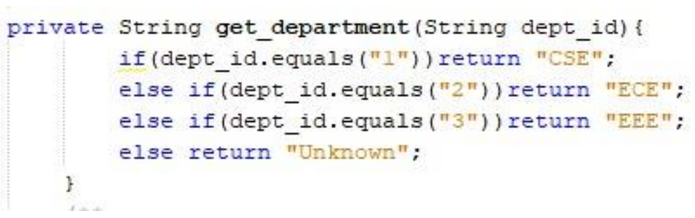
```java
private void logoutbtnActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try{
        conn.close();
        login lp= new login();
        lp.setVisible(true);
        this.dispose();
    }catch(Exception ex){
        JOptionPane.showMessageDialog(new JFrame(), ex);
    }
}
```

Figure 20: logout button for staff home page

6. Snippet for staff home page constructor



```java
public staffhome(ResultSet rs, Connection con) throws SQLException {
    initComponents();
    this.rsl=rs;
    this.conn=conn;
    this.setTitle("Student home page");
    String hello_string="Name: "+rsl.getString(2);
    staff_id.setText(rsl.getString(1));
    hello_field.setText(hello_string);
    dept.setText(get_department(rsl.getString(5)));
    teamid=rsl.getInt(6);
}
```



```
private String get_department(String dept_id){
        if(dept_id.equals("1"))return "CSE";
        else if(dept_id.equals("2"))return "ECE";
        else if(dept_id.equals("3"))return "EEE";
        else return "Unknown";
    }
```

Figure 21: staff home page constructor

1.5 Conclusion:

CONCLUSION:

A database management system is important because it manages data efficiently and allows users to perform multiple tasks with ease. A database management system stores, organizes and manages a large amount of information within a single software application.
The user interface (UI) is a critical part of any software product. When it is done well, users do not even notice it. When it is done poorly, users cannot get past it to efficiently use a product. To increase the chances of success when creating user interfaces, most designers follow interface design principles. Interface design principles represent high-level concepts that are used to guide software design.
The UI design principals are:

- Place users in control of the interface
- Make it comfortable to interact with a product
- Reduce cognitive load
- Make user interfaces consistent

LIMITATION:

the limitation of this application comes directly from the structure of SQL. SQL database is prone to SQL injections which can result in corruption of the database hence ruining the backbone of the project. Also, using the wild card operator, i.e. * a person can view all the confidential data including passwords.
Another limitation is that the application stores passwords without hashing which is harmful in case of cyber-attacks as the passwords are stored in plaintext.

IMPROVEMENT:

The database can be changed from SQL to NoSQL databases like firebase or mongo dB, which provides more useful and modern features like easy scalability, more security, support for real-time changes etc. Also the passwords should be passed through hash functions which provides proper encoding to the passwords before storing.

Bibliography

1. https://xd.adobe.com/ideas/process/ui-design/4-golden-rules-ui-design/
2. https://www.manomayasoft.com/blog/item/210-what-is-the-importance-of-a-database-management-system
3. https://www.javaworld.com/article/3388036/what-is-jdbc-introduction-to-java-database-connectivity.html