L1- CORE JAVA LAB ASSESSMENT

Emp No: 12228

Emp Name: Nagarjun N S

Batch : Java Batch-5

Duration : 4 Hours

Please add the below notes to all the Questions. Instructions:

1. Analyze the given problem using W3H technique.
2. Write an algorithm and draw the flowchart for the application.
3. Prepare UML diagrams for the given scenario [Class Diagram, Use Case Diagram & Sequence Diagram].
4. Create a console-based Java application and connect with MySQL database using JDBC API for the given scenario.
5. Perform all the CRUD operations using Java DAO pattern.
6. Make the application as user interactive.
7. Maintain the Code Quality and the Coding Standard .

Q4) Visitors Management System

W3H:

PROJECT NAME: Visitors Management System	
What? 1	How? 2
1. What are the modules are required? **Ans:** a) Admin (System) Module b) Employee Module c) Visitors Module **2. What are the responsibilities of Admin (System)?** **Ans:** a) Admin (System) will handle all the CRUD operations of the visitors database and also on the employee's database they want to visit. **3. What can the visitor do?** **Ans:** a) The visitor can able to enter the employee's name and other personal details of the employee they wish to see. b) Visitors can able to see the Employee is present or not. c) Visitors can easily select the date and time to visit the employees' of the office. **4. What can the employees see?** **Ans:** a) Employees can be able to see their list of visitors. b) They can see the appointed date and time. c) They can see the purpose of the visit written in a short description. **5. What are the fields required for visitors?** **Ans:** Visitor_ID, Visitors_Name, Employee_name, Visiting_Purpose, Visiting_DateTime, Visitor_PhoneNo, Visitor_Address. **6. What if the employee is unavailable on that day?** **Ans:** The employee can be able to update the next available date and this information will be updated to the system and a SMS will be sent to the visitor	**1) Admin (System):** **a) Login:** **Method 1:** The admin can login using their Email and password. **Method 2:** The admin can login using their username and password. **Method 3:** The admin can login to their account using social media accounts. **b) Verifying Appointment:** **Method 1:** Admin can verify visitors by sending an OTP to confirm the appointment. **Method 2:** Admin can verify visitors with their basic details like Name, email, Employee name, ph.no, address to confirm their appointment. **c) Update / delete / view visitor details:** **Method 1:** Admin can be able to Update / delete / view visitor details by providing separate buttons. **Method 2:** Admin can be able to Update / delete / view visitor details by using the drop-down list. **2) Visitor:** **a) Login:** **Method 1:** The Visitor can login by using their Email and password. **Method 2:** The Visitor can login by using mobile number and Password. **Method 3:** The Visitor can login to their account using their visiting id. **b) CRUD Operations:** **Method 1: Visitor can perform CRUD Operations for their account with separate buttons.** **Method 2:** Visitor can be able to perform CRUD Operations for their account using a drop-down list and a confirm button. **c) Filling Visiting Form:** **Method 1:** Visitor can enter their basic details like name, Purpose of the visit, Visiting Date and Time, ph no, address and the name of the employee they want to visit in the fields. **Method 2:** Visitor can enter their name and purpose and the name of the employee.
1) Admin (System): **a) Login:** **Method 1:** The admin can login using their Email and password. (This method is used because the originality of the admin is verified). **b) Verifying Appointment:** **Method 1:** Admin can verify visitors by sending an OTP to confirm the appointment. (This method is used because the user will be verified as well as the date and time also acknowledged by confirming the OTP) **c) Update / delete / view visitor details:** **Method 1:** Admin can be able to Update / delete / view visitor details by providing separate buttons. (Providing separate buttons with separate functionalities will help to use the application easily.) **2) Visitor:** **a) Login:** **Method 1:** The Visitor can login by using their Email and password. (Login to the system by using their Email and password will help to identify the originality of the user) **b) CRUD Operations:** **Method 1: Visitor can perform CRUD Operations for their account with separate buttons.** (Providing separate buttons with separate functionalities will help to use the application easily.) **c) Filling Visiting Form:** **Method 1:** Visitor can enter their basic details like name, Purpose of the visit, Visiting Date and Time, ph no, address and the name of the employee they want to visit in the fields. (By filling the above details in the online form will pave a way for the clear understanding of the visit and these are all the basic details to get from a visitor.)	**1) Admin:** **a) Login:** **Method 2:** The admin can login using their username and password. (Admin can't login using their username and password as the originality is not verified.) **Method 3:** The admin can login to their account using social media accounts. (Using Social media accounts to login will not be a feasible way and it is less secure). **b) Verifying Appointment:** **Method 2:** Admin can verify visitors with their basic details like Name, email, Employee Name, ph.no, address to confirm their appointment. (If a visitor just filled in all the details and didn't verify their appointment then there is a cause for spamming or any other unauthorized access.) **c) Update / delete / view visitor details:** **Method 2:** Admin can be able to Update / delete / view visitor details by using the drop-down list. (The drop-down list feature is not feasible, and it is hard to use, and it is time-consuming function too). **2) Visitor:** **a) Login:** **Method 2:** The visitor can login by using mobile number and password. (User login using mobile number and account number will not be secure) **Method 3:** The Visitor can login to their account using their visiting id. (We cannot authorize the visitor who only has the visiting_ID and it is not a feasible way to authorize) **b) CRUD Operations:** **Method 2:** Visitor can be able to perform CRUD Operations for their account using a drop-down list and a confirm button. (The drop-down list feature is not feasible, and it is hard to use, and it is time-consuming function too). **c) Filling Visiting Form:** **Method 2:** Visitor can enter their name and purpose and the name of the employee. (If visitor enters only their name, purpose and name of the employee, these details are not enough for records if anything goes wrong in the visit. So, it is not feasible.)
Why? 3	Why not? 4

ALGORITHM & FLOW CHART:

ALGORITHM:

<u>Visitor Algorithm:</u>

Step 1: Start

Step 2: Go to Login page

Step 3: Enter the credentials

Step 4: After completing step 3, click login

Step 5: Go to homepage

Step 6: Select “Visit Employee”

Step 7: Fill the form

Step 8: Confirm the visit with OTP

Step 9: Logout

Step 10: Stop

<u>Admin Algorithm:</u>

Step 1: Start

Step 2: Go to Login page

Step 3: Enter the credentials

Step 4: After completing step 3, click login

Step 5: Go to homepage

Step 6: Click on Visitor’s list

Step 7: Send the approved Visitor’s details to the relevant Employees

Step 8: Logout

Step 9: Stop

<u>Employee Algorithm:</u>

Step 1: Start

Step 2: Go to Login page

Step 3: Enter the credentials

Step 4: After completing step 3, click login

Step 5: Go to homepage

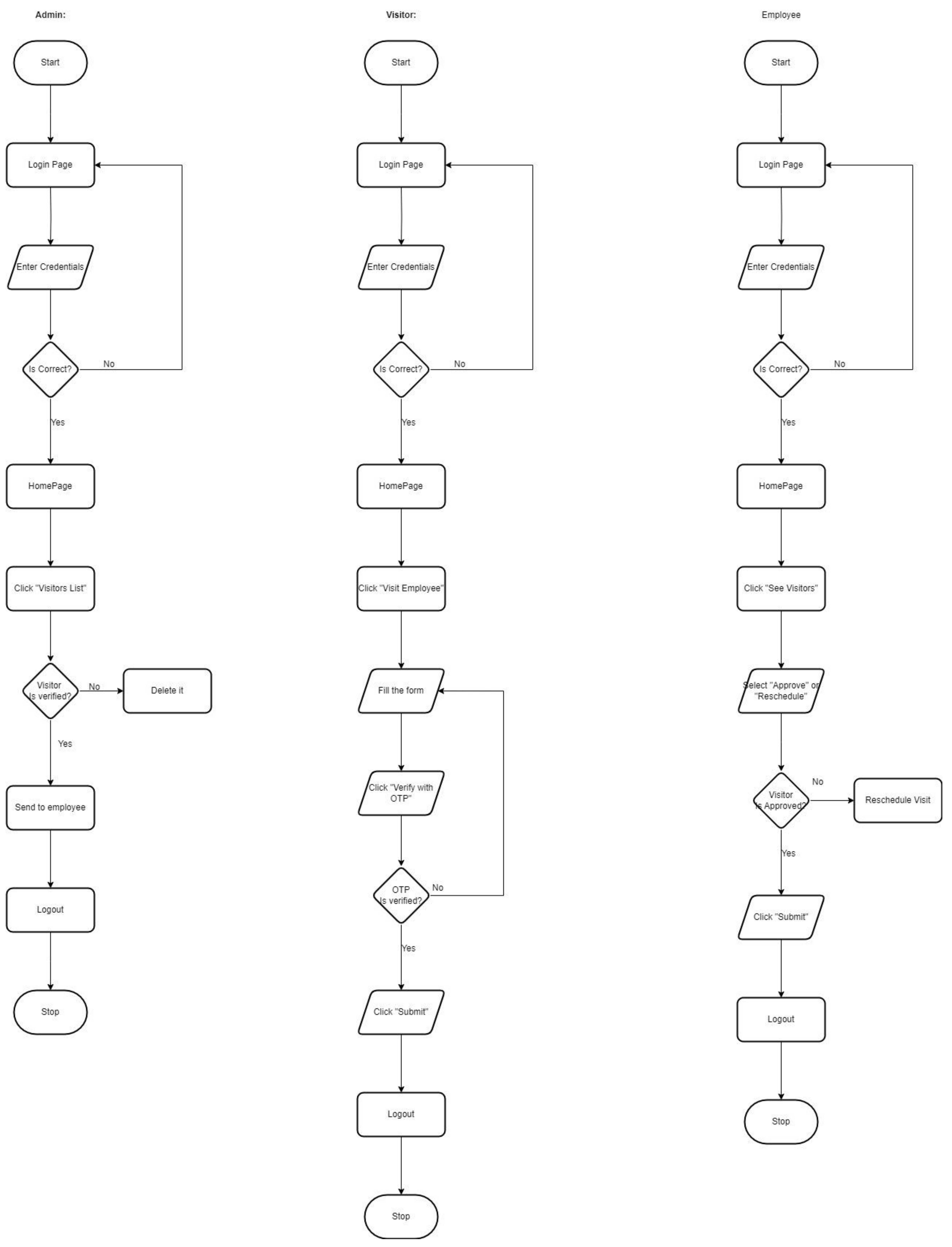
Step 6: Select “See Visitors” tab

Step 7: Click “Approve” button to accept the appointment, or else click “Reschedule” to inform the availability to the visitor on some other days

Step 8: Logout

Step 9: Stop

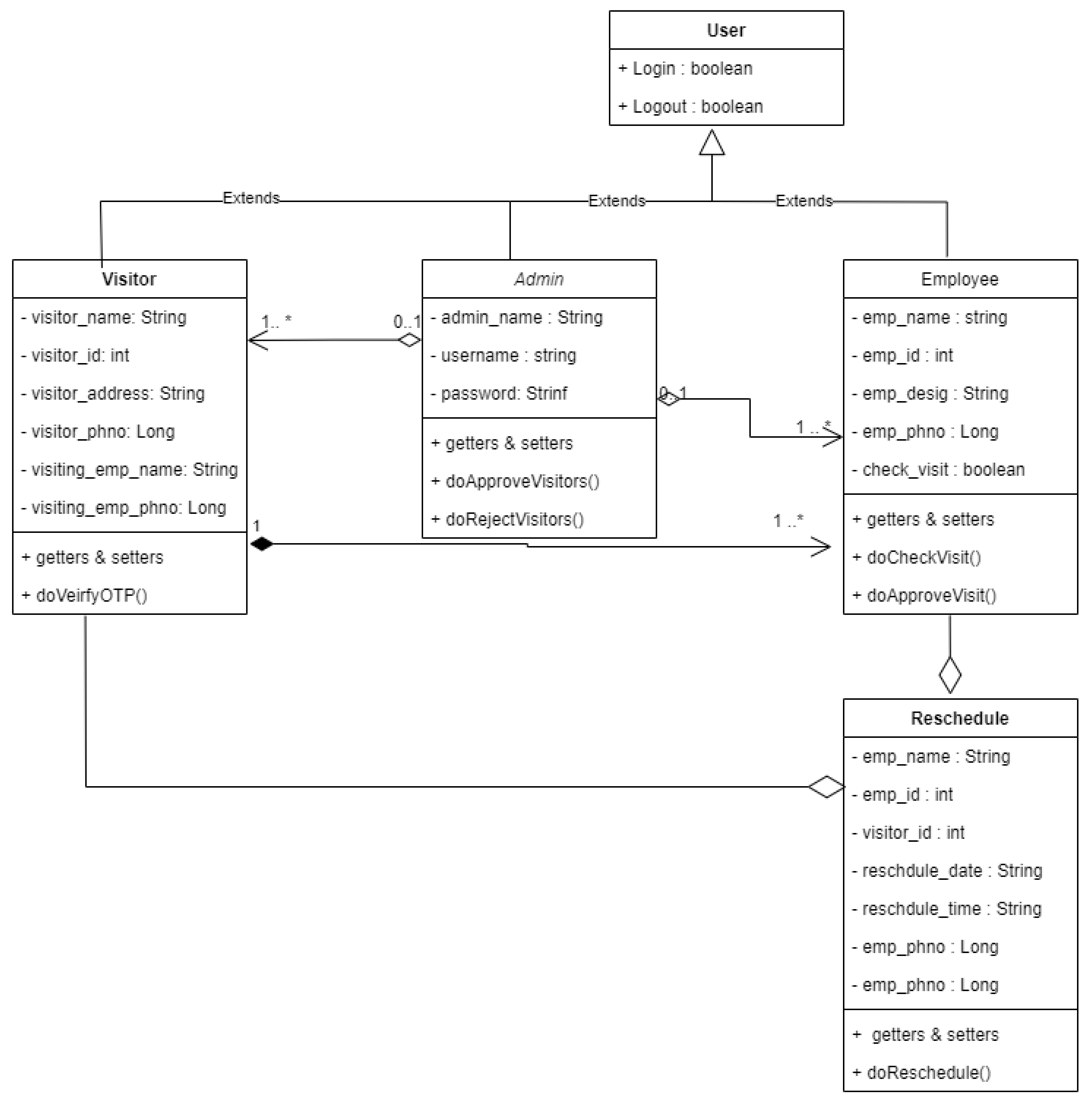
FLOWCHART:


Admin:
Start
Login Page
Enter Credentials
Is Correct?
No
Yes
HomePage
Click "Visitors List"
Visitor Is verified?
No
Delete it
Yes
Send to employee
Logout
Stop
Visitor:
Start
Login Page
Enter Credentials
Is Correct?
No
Yes
HomePage
Click "Visit Employee"
Fill the form
Click "Verify with OTP"
OTP Is verified?
No
Yes
Click "Submit"
Logout
Stop
Employee
Start
Login Page
Enter Credentials
Is Correct?
No
Yes
HomePage
Click "See Visitors"
Select "Approve" or "Reschedule"
Visitor Is Approved?
No
Reschedule Visit
Yes
Click "Submit"
Logout
Stop

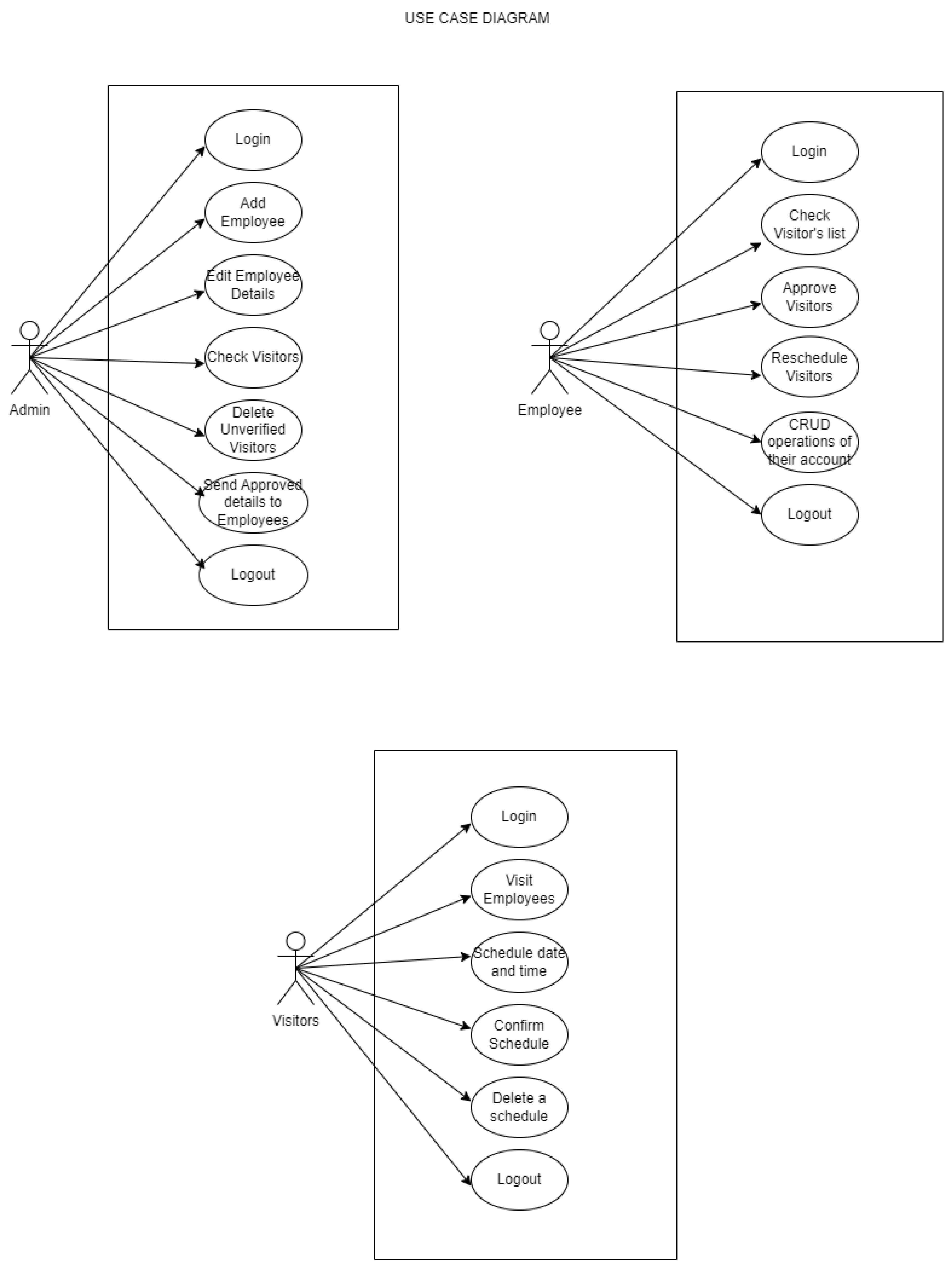
UML DIAGRAMS:

CLASS DIAGRAM:

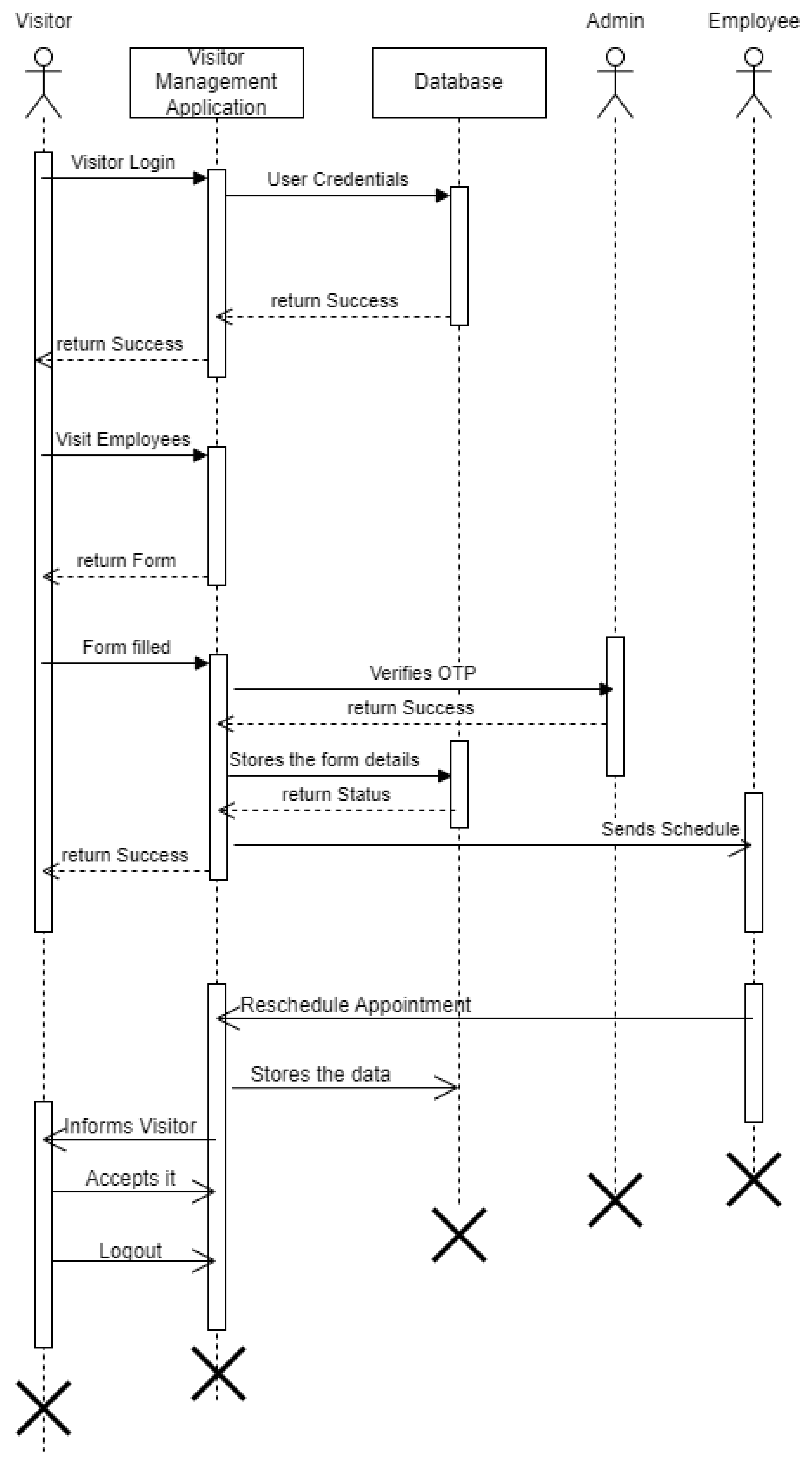
VISITOR MANAGEMENT CLASS DIAGRAM



USE-CASE DIAGRAM:



SEQUENCE DIAGRAM:



FOLDER STRUCTURE:

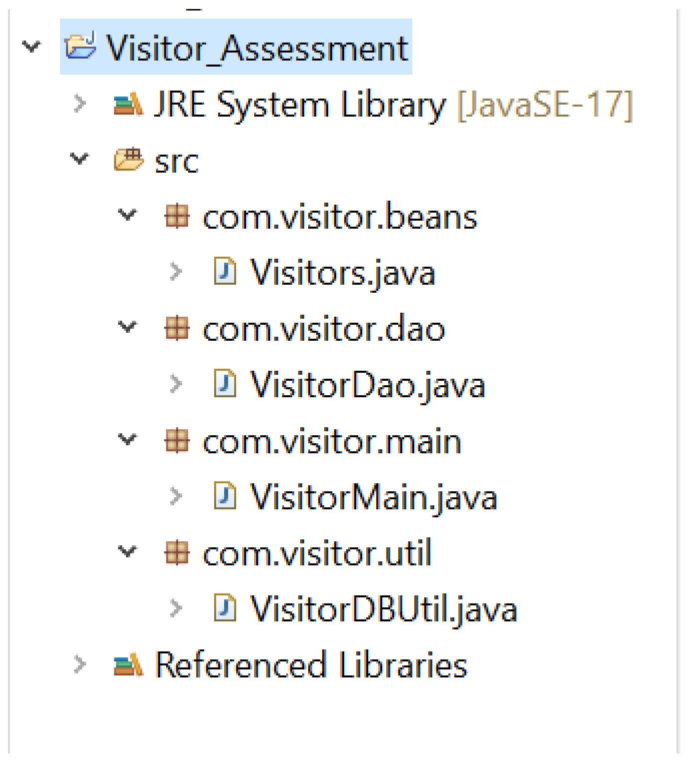
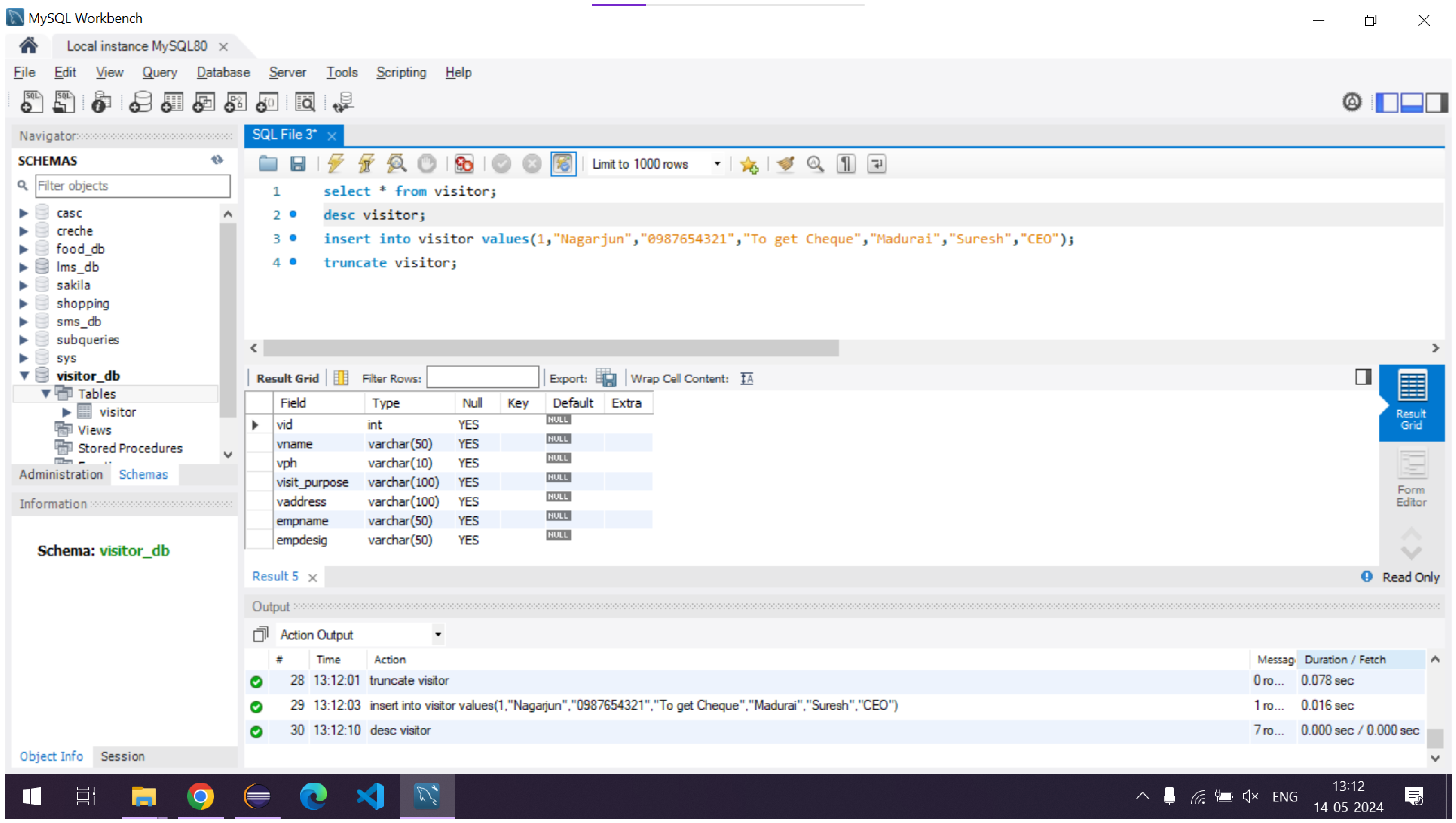


TABLE STRUCTURE:



```
select * from visitor;
desc visitor;
insert into visitor values(1,"Nagarjun","0987654321","To get Cheque","Madurai","Suresh","CEO");
truncate visitor;
```

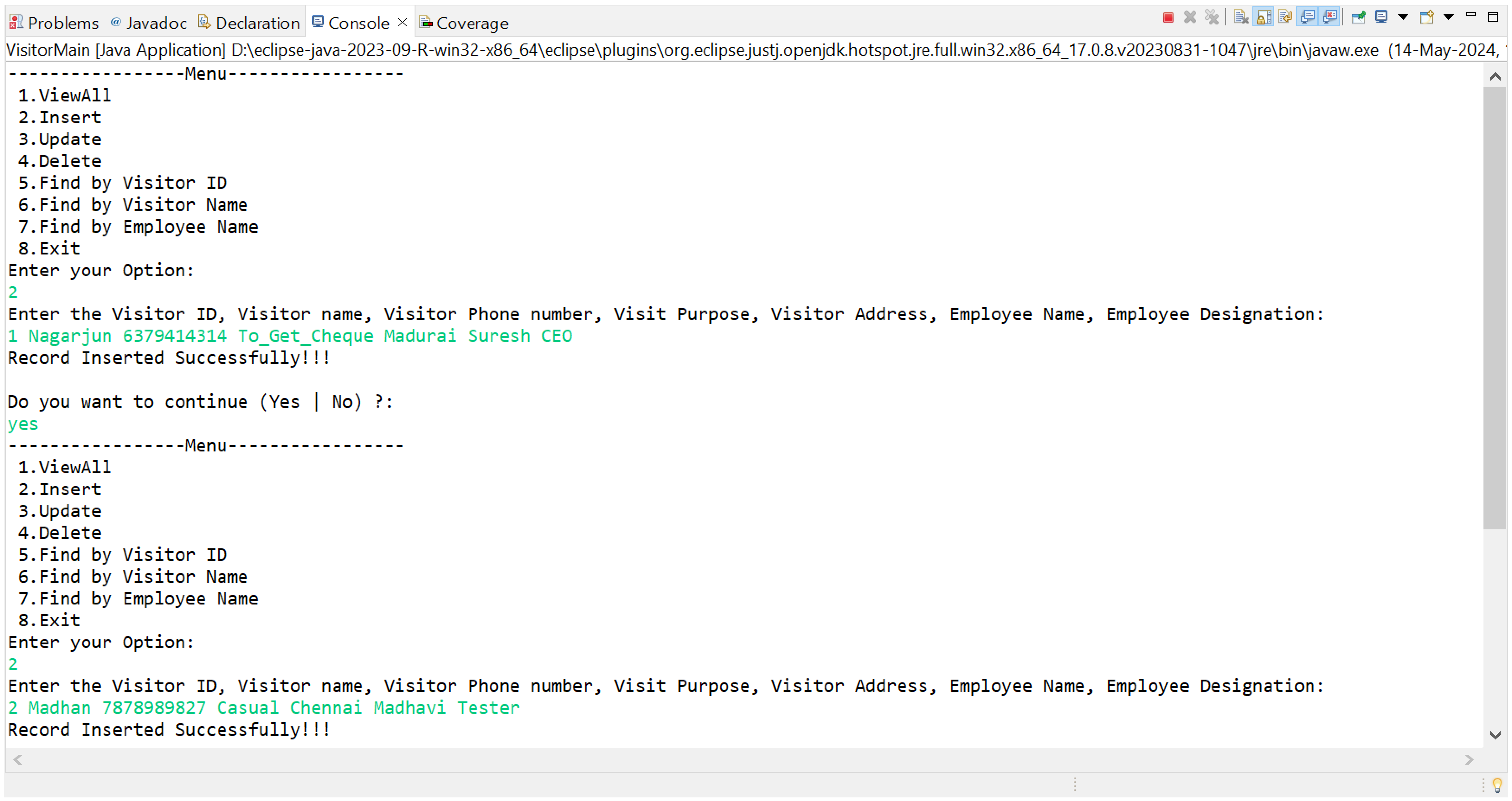
Field	Type	Null	Key	Default	Extra
vid	int	YES		NULL	
vname	varchar(50)	YES		NULL	
vph	varchar(10)	YES		NULL	
visit_purpose	varchar(100)	YES		NULL	
vaddress	varchar(100)	YES		NULL	
empname	varchar(50)	YES		NULL	
empdesig	varchar(50)	YES		NULL	

	#	Time	Action	Messag	Duration / Fetch
✓	28	13:12:01	truncate visitor	0 ro...	0.078 sec
✓	29	13:12:03	insert into visitor values(1,"Nagarjun","0987654321","To get Cheque","Madurai","Suresh","CEO")	1 ro...	0.016 sec
✓	30	13:12:10	desc visitor	7 ro...	0.000 sec / 0.000 sec

ENG 13:12 14-05-2024

SCREENSHOTS:

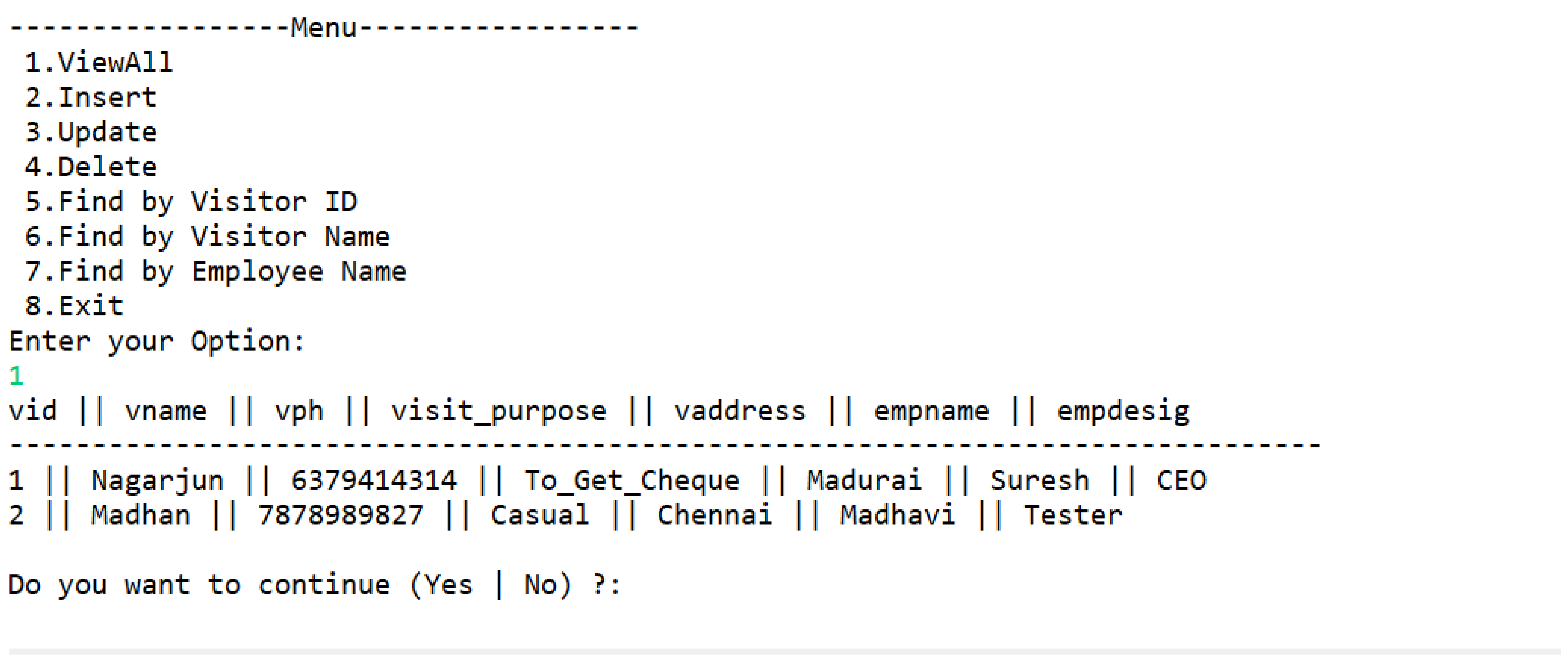
INSERTION:



```
VisitorMain [Java Application] D:\eclipse-java-2023-09-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe  (14-May-2024,
-----------------Menu-----------------
 1.ViewAll
 2.Insert
 3.Update
 4.Delete
 5.Find by Visitor ID
 6.Find by Visitor Name
 7.Find by Employee Name
 8.Exit
Enter your Option:
2
Enter the Visitor ID, Visitor name, Visitor Phone number, Visit Purpose, Visitor Address, Employee Name, Employee Designation:
1 Nagarjun 6379414314 To_Get_Cheque Madurai Suresh CEO
Record Inserted Successfully!!!

Do you want to continue (Yes | No) ?:
yes
-----------------Menu-----------------
 1.ViewAll
 2.Insert
 3.Update
 4.Delete
 5.Find by Visitor ID
 6.Find by Visitor Name
 7.Find by Employee Name
 8.Exit
Enter your Option:
2
Enter the Visitor ID, Visitor name, Visitor Phone number, Visit Purpose, Visitor Address, Employee Name, Employee Designation:
2 Madhan 7878989827 Casual Chennai Madhavi Tester
Record Inserted Successfully!!!
```

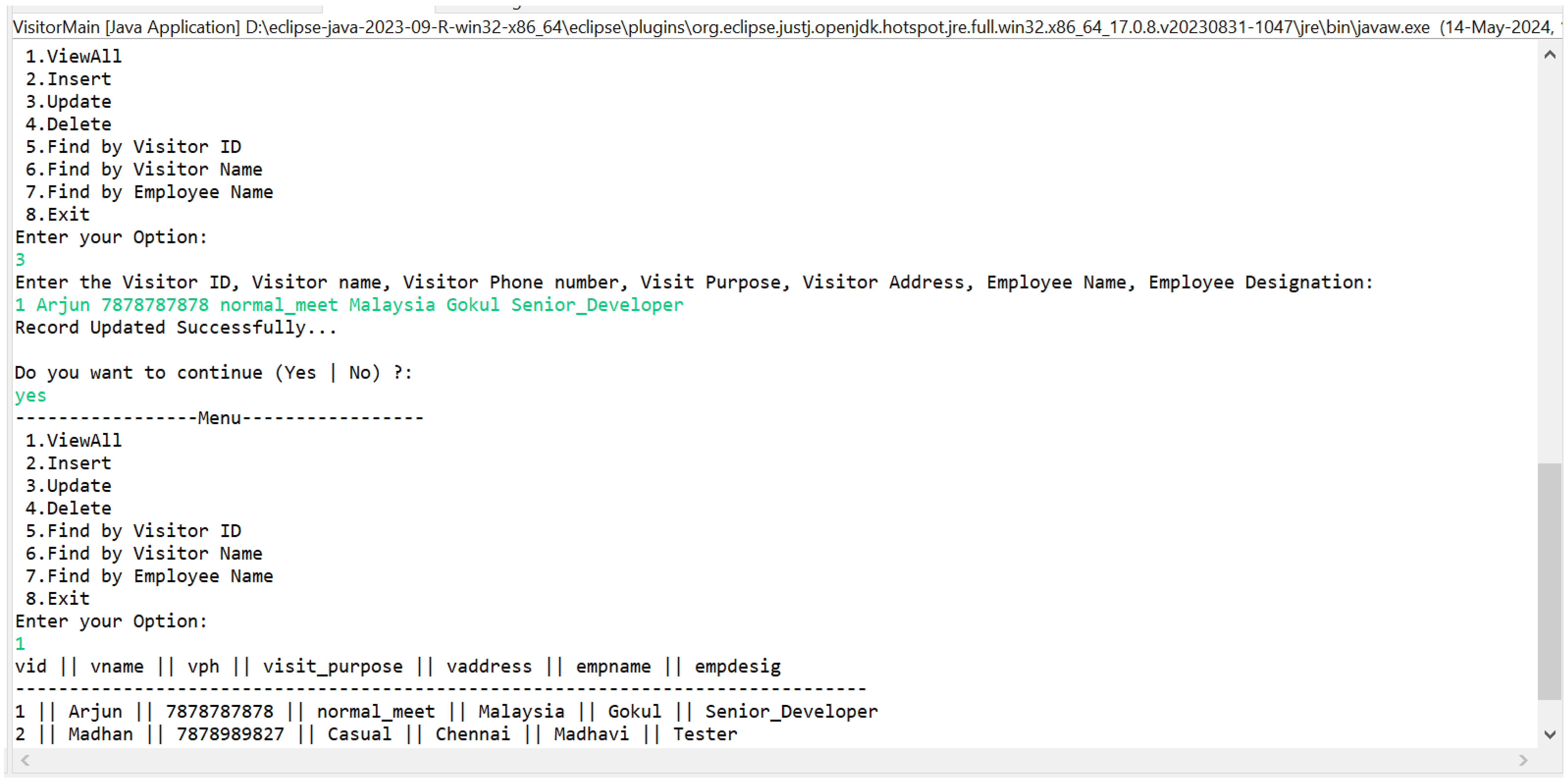
VIEW ALL:



```
-----------------Menu-----------------
 1.ViewAll
 2.Insert
 3.Update
 4.Delete
 5.Find by Visitor ID
 6.Find by Visitor Name
 7.Find by Employee Name
 8.Exit
Enter your Option:
1
vid || vname || vph || visit_purpose || vaddress || empname || empdesig
-------------------------------------------------------------------------------
1 || Nagarjun || 6379414314 || To_Get_Cheque || Madurai || Suresh || CEO
2 || Madhan || 7878989827 || Casual || Chennai || Madhavi || Tester

Do you want to continue (Yes | No) ?:
```

UPDATE:

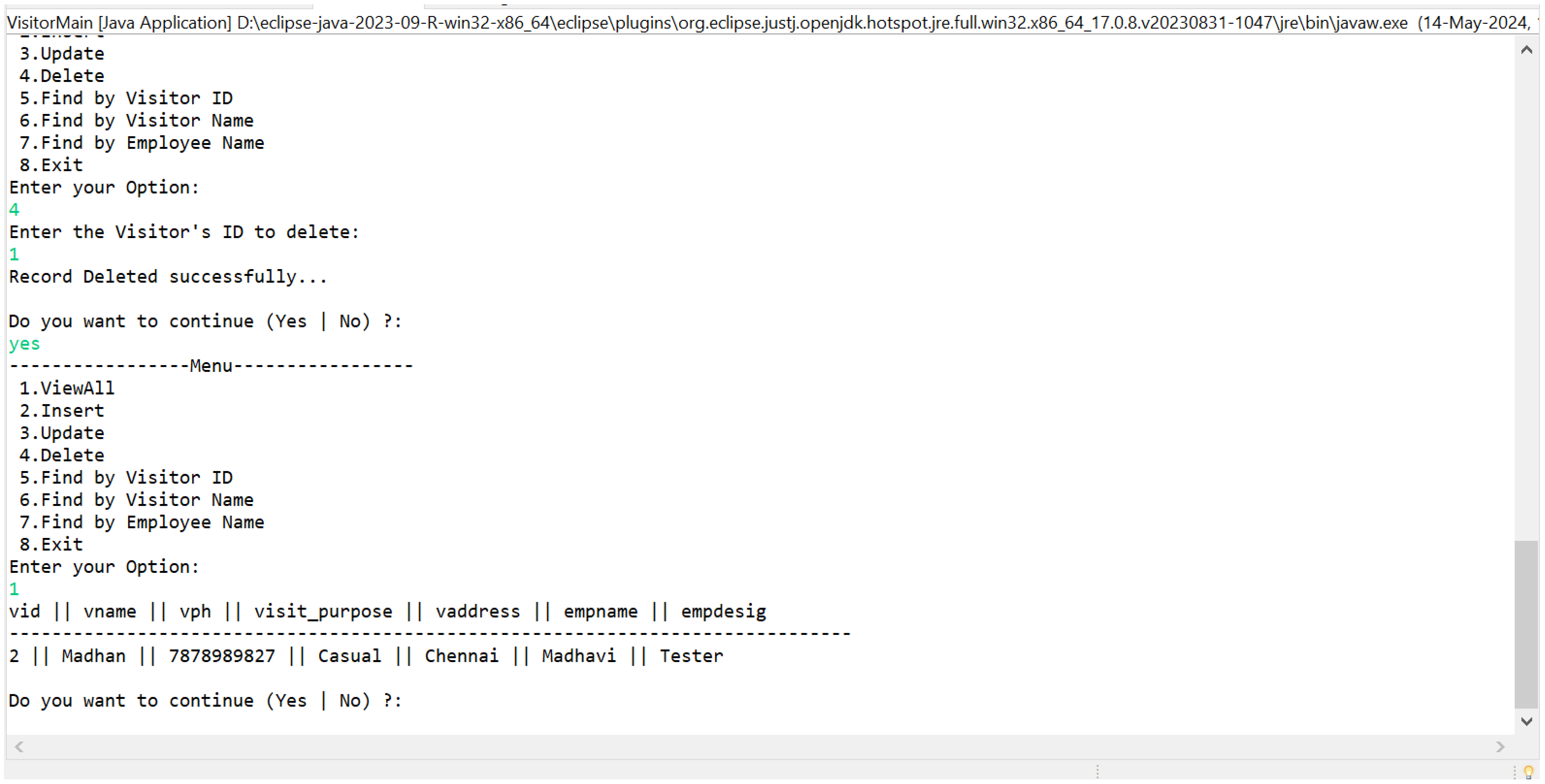


VisitorMain [Java Application] D:\eclipse-java-2023-09-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe (14-May-2024,

```
 1.ViewAll
 2.Insert
 3.Update
 4.Delete
 5.Find by Visitor ID
 6.Find by Visitor Name
 7.Find by Employee Name
 8.Exit
Enter your Option:
3
Enter the Visitor ID, Visitor name, Visitor Phone number, Visit Purpose, Visitor Address, Employee Name, Employee Designation:
1 Arjun 7878787878 normal_meet Malaysia Gokul Senior_Developer
Record Updated Successfully...

Do you want to continue (Yes | No) ?:
yes
-----------------Menu-----------------
 1.ViewAll
 2.Insert
 3.Update
 4.Delete
 5.Find by Visitor ID
 6.Find by Visitor Name
 7.Find by Employee Name
 8.Exit
Enter your Option:
1
vid || vname || vph || visit_purpose || vaddress || empname || empdesig
-------------------------------------------------------------------------------
1 || Arjun || 7878787878 || normal_meet || Malaysia || Gokul || Senior_Developer
2 || Madhan || 7878989827 || Casual || Chennai || Madhavi || Tester
```

DELETE:



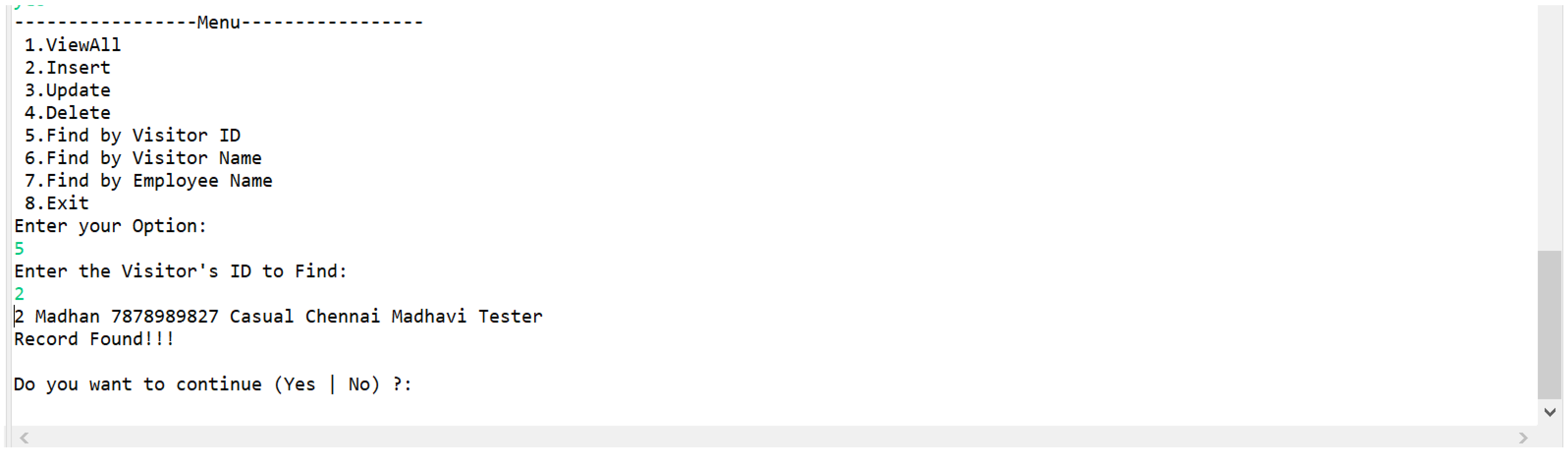
VisitorMain [Java Application] D:\eclipse-java-2023-09-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe (14-May-2024,

```
 3.Update
 4.Delete
 5.Find by Visitor ID
 6.Find by Visitor Name
 7.Find by Employee Name
 8.Exit
Enter your Option:
4
Enter the Visitor's ID to delete:
1
Record Deleted successfully...

Do you want to continue (Yes | No) ?:
yes
-----------------Menu-----------------
 1.ViewAll
 2.Insert
 3.Update
 4.Delete
 5.Find by Visitor ID
 6.Find by Visitor Name
 7.Find by Employee Name
 8.Exit
Enter your Option:
1
vid || vname || vph || visit_purpose || vaddress || empname || empdesig
-------------------------------------------------------------------------------
2 || Madhan || 7878989827 || Casual || Chennai || Madhavi || Tester

Do you want to continue (Yes | No) ?:
```

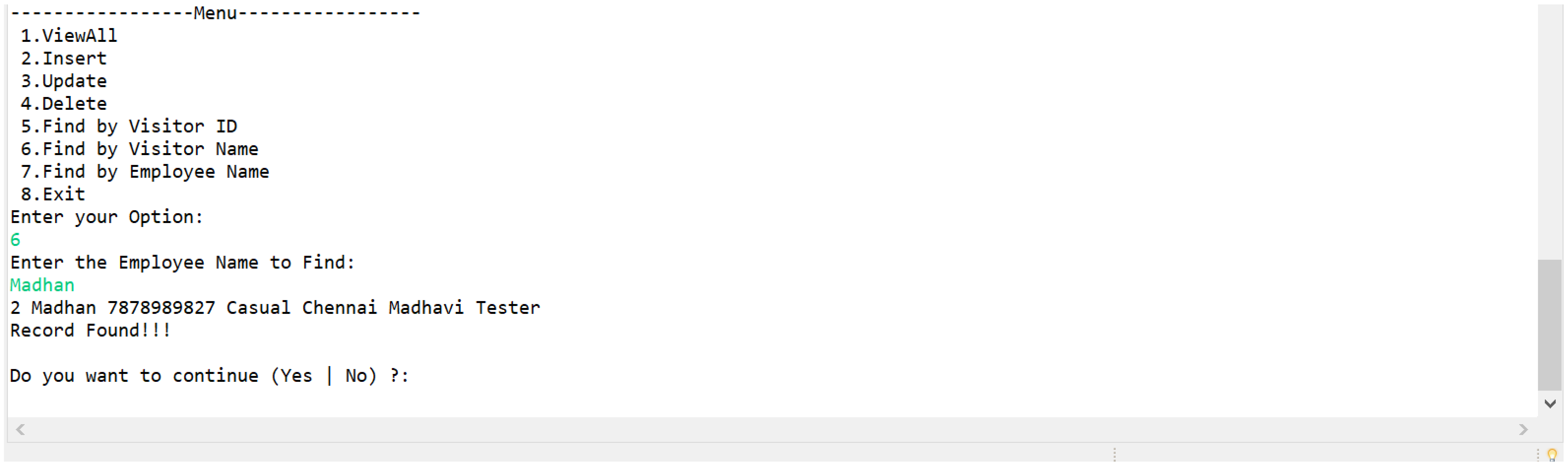
FINDING BY VISITOR ID:



```
-----------------Menu-----------------
 1.ViewAll
 2.Insert
 3.Update
 4.Delete
 5.Find by Visitor ID
 6.Find by Visitor Name
 7.Find by Employee Name
 8.Exit
Enter your Option:
5
Enter the Visitor's ID to Find:
2
2 Madhan 7878989827 Casual Chennai Madhavi Tester
Record Found!!!

Do you want to continue (Yes | No) ?:
```

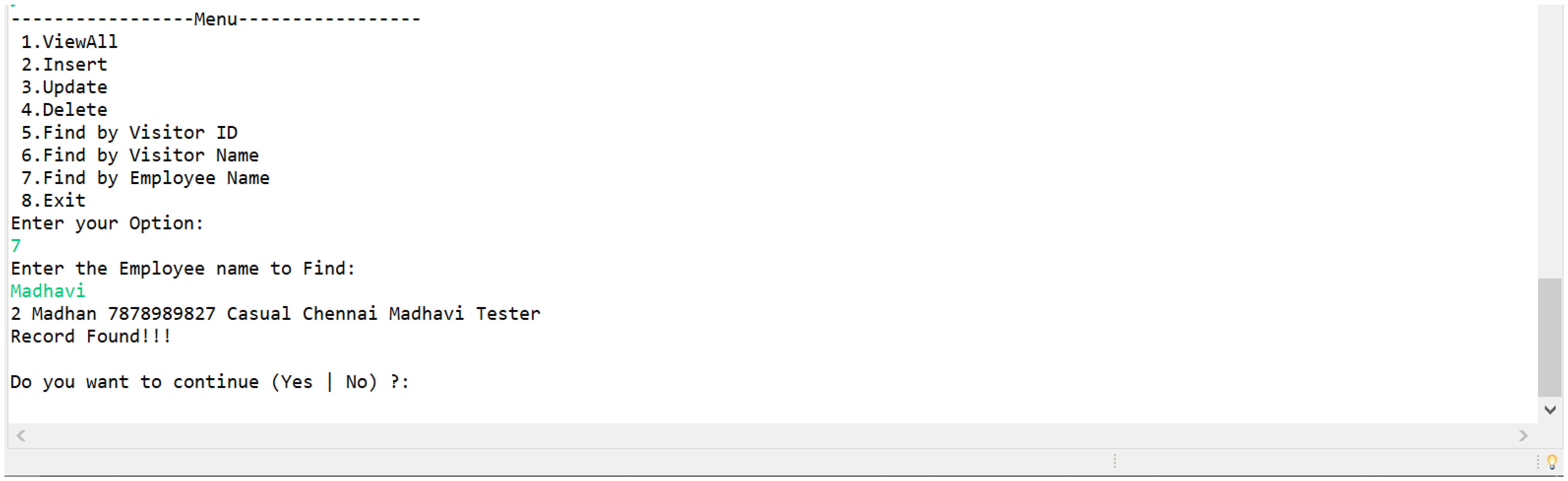
FINDING BY VISITOR NAME:



```
-----------------Menu-----------------
 1.ViewAll
 2.Insert
 3.Update
 4.Delete
 5.Find by Visitor ID
 6.Find by Visitor Name
 7.Find by Employee Name
 8.Exit
Enter your Option:
6
Enter the Employee Name to Find:
Madhan
2 Madhan 7878989827 Casual Chennai Madhavi Tester
Record Found!!!

Do you want to continue (Yes | No) ?:
```

FINDING BY EMPLOYEE NAME:



```
-----------------Menu-----------------
 1.ViewAll
 2.Insert
 3.Update
 4.Delete
 5.Find by Visitor ID
 6.Find by Visitor Name
 7.Find by Employee Name
 8.Exit
Enter your Option:
7
Enter the Employee name to Find:
Madhavi
2 Madhan 7878989827 Casual Chennai Madhavi Tester
Record Found!!!

Do you want to continue (Yes | No) ?:
```

CODING:

Visitors.java

```java
package com.visitor.beans;

//This is the Bean class to store details and declare Variables
```

```java
public class Visitors {
private int vid;
private String vname;
private String vph;
private String vpurpose;
private String vaddress;
private String emp_name;
private String emp_desig;


public Visitors() {
super();
// TODO Auto-generated constructor stub
}


public Visitors(int vid, String vname, String vph, String vpurpose, String vaddress, String emp_name,
String emp_desig) {
super();
this.vid = vid;
this.vname = vname;
this.vph = vph;
this.vpurpose = vpurpose;
this.vaddress = vaddress;
this.emp_name = emp_name;
this.emp_desig = emp_desig;
}


public int getVid() {
return vid;
}
public void setVid(int vid) {
this.vid = vid;
}
public String getVname() {
return vname;
}
public void setVname(String vname) {
this.vname = vname;
}
public String getVph() {
return vph;
}
public void setVph(String vph) {
this.vph = vph;
}
public String getVpurpose() {
return vpurpose;
}
public void setVpurpose(String vpurpose) {
this.vpurpose = vpurpose;
}
public String getVaddress() {
return vaddress;
}
public void setVaddress(String vaddress) {
this.vaddress = vaddress;
}
public String getEmp_name() {
return emp_name;
}
public void setEmp_name(String emp_name) {
this.emp_name = emp_name;
}
public String getEmp_desig() {
return emp_desig;
}
public void setEmp_desig(String emp_desig) {
this.emp_desig = emp_desig;
}


}
```

VisitorDao.java

```java
package com.visitor.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import com.visitor.beans.Visitors;
import com.visitor.util.VisitorDBUtil;


public class VisitorDao {

public int viewAll() throws SQLException {
int count=0;
Connection con1 = VisitorDBUtil.getConnection();
String sql = "select * from visitor";
PreparedStatement ps = con1.prepareStatement(sql);
ResultSetMetaData rms = ps.getMetaData();
ResultSet rs = ps.executeQuery();
System.out.println(rms.getColumnName(1)+" || "+rms.getColumnName(2)+" || "+rms.getColumnName(3)+" || "+rms.getColumnName(4)+" || "+rms.getColumnName(5)+" || "+rms.getColumnName(6)+" || "+rms.getColumnName(7));
System.out.println("-------------------------------------------------------------------------------");
while(rs.next()) {
System.out.println(rs.getInt(1)+" || "+rs.getString(2)+" || "+rs.getString(3)+" || "+rs.getString(4)+" || "+rs.getString(5)+" || "+rs.getString(6)+" || "+rs.getString(7));
count++;
}
con1.close();
return count;
}

public int doInsert(Visitors e) throws SQLException {
Connection con1 = VisitorDBUtil.getConnection();
String sql = "insert into visitor values(?,?,?,?,?,?,?)";
PreparedStatement ps = con1.prepareStatement(sql);
ps.setInt(1, e.getVid());
ps.setString(2, e.getVname());
ps.setString(3, e.getVph());
ps.setString(4, e.getVpurpose());
ps.setString(5, e.getVaddress());
ps.setString(6, e.getEmp_name());
ps.setString(7, e.getEmp_desig());
int n=ps.executeUpdate();
con1.close();
return n;
}

public int doUpdate(Visitors e) throws SQLException {
Connection con1 = VisitorDBUtil.getConnection();
String sql = "update visitor set vname=?,vph=?,visit_purpose=?,vaddress=?,empname=?,empdesig=? where vid=?";
PreparedStatement ps = con1.prepareStatement(sql);
ps.setString(1, e.getVname());
ps.setString(2, e.getVph());
ps.setString(3, e.getVpurpose());
ps.setString(4, e.getVaddress());
ps.setString(5, e.getEmp_name());
ps.setString(6, e.getEmp_desig());
ps.setInt(7, e.getVid());
int n=ps.executeUpdate();
con1.close();
return n;
}

public int doDelete (int id) throws SQLException {
Connection con1 = VisitorDBUtil.getConnection();
String sql = "delete from visitor where vid=?";
PreparedStatement ps = con1.prepareStatement(sql);
ps.setInt(1, id);
int n=ps.executeUpdate();
con1.close();
return n;
}

public int doFind (int id) throws SQLException {
int count=0;
Connection con1 = VisitorDBUtil.getConnection();
String sql = "select * from visitor where vid=?";
PreparedStatement ps = con1.prepareStatement(sql);
ps.setInt(1, id);
ResultSet rs=ps.executeQuery();
while(rs.next()) {
System.out.println(rs.getInt(1)+" "+rs.getString(2)+" "+rs.getString(3)+" "+rs.getString(4)+" "+rs.getString(5)+" "+rs.getString(6)+" "+rs.getString(7));
count++;
}
con1.close();
return count;
}

public int doFindName (String name) throws SQLException {
int count=0;
Connection con1 = VisitorDBUtil.getConnection();
String sql = "select * from visitor where vname=?";
PreparedStatement ps = con1.prepareStatement(sql);
ps.setString(1, name);
ResultSet rs=ps.executeQuery();
while(rs.next()) {
System.out.println(rs.getInt(1)+" "+rs.getString(2)+" "+rs.getString(3)+" "+rs.getString(4)+" "+rs.getString(5)+" "+rs.getString(6)+" "+rs.getString(7));
count++;
}
con1.close();
return count;
}

public int doFindEmp (String emname) throws SQLException {
int count=0;
Connection con1 = VisitorDBUtil.getConnection();
String sql = "select * from visitor where empname=?";
PreparedStatement ps = con1.prepareStatement(sql);
ps.setString(1, emname);
ResultSet rs=ps.executeQuery();
while(rs.next()) {
System.out.println(rs.getInt(1)+" "+rs.getString(2)+" "+rs.getString(3)+" "+rs.getString(4)+" "+rs.getString(5)+" "+rs.getString(6)+" "+rs.getString(7));
count++;
}
con1.close();
return count;
}

}
```

VisitorMain.java

```
package com.visitor.main;
```

```
import java.sql.SQLException;
import java.util.Scanner;

import com.visitor.beans.Visitors;
import com.visitor.dao.VisitorDao;

public class VisitorMain {
static Scanner sc = new Scanner(System.in);
public static int displayMenu() {
int choice;
System.out.println("-----------------Menu-----------------");
System.out.println(" 1.ViewAll\n 2.Insert\n 3.Update\n 4.Delete\n 5.Find by Visitor ID\n 6.Find by Visitor Name\n 7.Find by Employee Name\n 8.Exit \nEnter your Option:");
choice = sc.nextInt();
return choice;
}

public static Visitors insert() {
System.out.println("Enter the Visitor ID, Visitor name, Visitor Phone number, Visit Purpose, Visitor Address, Employee Name, Employee Designation:");
return(new Visitors(sc.nextInt(),sc.next(),sc.next(),sc.next(),sc.next(),sc.next(),sc.next()));
}

public static Visitors update() {
System.out.println("Enter the Visitor ID, Visitor name, Visitor Phone number, Visit Purpose, Visitor Address, Employee Name, Employee Designation:");
return(new Visitors(sc.nextInt(),sc.next(),sc.next(),sc.next(),sc.next(),sc.next(),sc.next()));
}

public static Visitors findName() {
System.out.println("Enter the Visitor ID, Visitor name, Visitor Phone number, Visit Purpose, Visitor Address, Employee Name, Employee Designation:");
return(new Visitors(sc.nextInt(),sc.next(),sc.next(),sc.next(),sc.next(),sc.next(),sc.next()));
}

public static Visitors findEmpName() {
System.out.println("Enter the Visitor ID, Visitor name, Visitor Phone number, Visit Purpose, Visitor Address, Employee Name, Employee Designation:");
return(new Visitors(sc.nextInt(),sc.next(),sc.next(),sc.next(),sc.next(),sc.next(),sc.next()));
}



public static void main(String s[]) throws SQLException {
VisitorDao dao = new VisitorDao();
String ch;
do {
switch(displayMenu()) {

case 1:
int n = dao.viewAll();
if(n<=0) {
System.out.println("No Records Found!!!");
}
break;

case 2:
Visitors v = insert();
int in = dao.doInsert(v);
if(in>0) {
System.out.println("Record Inserted Successfully!!!");
}
else {
System.out.println("Record Insertion Failed!!!");
}
break;

case 3:
Visitors upd = update();
int up = dao.doUpdate(upd);
if(up>0) {
System.out.println("Record Updated Successfully...");
}
else {
System.out.println("Failure in Updation");
}
break;

case 4:
System.out.println("Enter the Visitor's ID to delete:");
int id = sc.nextInt();
int del = dao.doDelete(id);
if(del>0) {
System.out.println("Record Deleted successfully...");
}
else {
System.out.println("Failure in Deletion!!!");
}
break;
case 5:
System.out.println("Enter the Visitor's ID to Find:");
int fid = sc.nextInt();
int fin = dao.doFind(fid);
if(fin>0) {
System.out.println("Record Found!!!");
}
else {
System.out.println("Record not found");
}

break;

case 6:
System.out.println("Enter the Employee Name to Find:");
String ename = sc.next();
int fname = dao.doFindName(ename);
if(fname<=0) {
```

```
System.out.println("Record not found");
}
else {
System.out.println("Record Found!!!");
}
break;

case 7:
System.out.println("Enter the Employee name to Find:");
String fename = sc.next();
int fenamede = dao.doFindEmp(fename);
if(fenamede>0) {
System.out.println("Record Found!!!");
}
else {
System.out.println("Record not found");
}

break;

case 8:
System.out.println("Thank You! Visit Again!!!");
System.exit(0);
}
System.out.println();
System.out.println("Do you want to continue (Yes | No) ?:");
ch = sc.next();
} while(ch.equalsIgnoreCase("yes"));
}
}
```

VisitorDBUtil.java

```
package com.visitor.util;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class VisitorDBUtil {
public static Connection getConnection() throws SQLException{
final String URL = "jdbc:mysql://localhost:3306/visitor_db";
final String User = "root";
final String pass = "root";
Connection conn = DriverManager.getConnection(URL,User,pass);
return conn;
}
}
```