JAVA SWINGS BASED- Covid Vaccination Database- SQL CONNECTIVITY USING JDBC

DBMS Project Report Submitted in partial fulfilment of the Requirements for the award of the Degree of

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

BY

Gajawada Rishi (1602-21-737-042)

Under the guidance of **Ms B. Leelavathy**

Department of Information Technology

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31

2022

DECLARATION BY THE CANDIDATE :

I **Gajawada Rishi** bearing hall ticket numbers, **1602-21-737-042** hereby declare that the project report entitled **"Covid Vaccination Database"** Department of Information Technology, Vasavi College of Engineering, Hyderabad, is submitted in partial fulfilment of the requirement for the award of the degree of Bachelor of Engineering in Information Technology This is a record of bonafide work carried out by me and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

Gajawada Rishi

1602-21-737-042

Vasavi College of Engineering (Autonomous)

Ibrahimbagh, Hyderabad-31

Department of Information technology

BONAFIDE CERTIFICATE

This is to certify that the project entitled **"Covid Vaccination Database"** being submitted by **Gajawada Rishi,** bearing **1602-21-737-042,** in partial fulfilment of the requirement for the course of **DATABASE MANAGEMENT SYSTEM LAB** in BE 2/4 (IT) IV- Semester is a record of bonafide work carried out by him under my guidance.

Ms B. Leelavathy

Assistant professor,
Internal Guide.

Professor & HOD,
Dept. of IT

ABSTRACT:

This is project “Covid Vaccination Information Tracker Database “. we have to track the Number of slots available, type of vaccine taken and test cases that ensures the user has already taken dose1 or dose2 or booster. This is helpful as it shows the information about number of slots so that the user can arrive when his turn comes and it helps the user to find which type of vaccine he has taken and also the details about number of doses taken. To implement this and to display all the data on the screen we need to build the database with details of number of slots available as backend database. To build this portal, we are using SQL for the backend and java for the frontend part.

Introduction:

REQUIREMENT ANALYSIS

List of tables:

- Login
- User_det
- Vaccine
- Userdosestatus
- Slot

Rollno:102-21-737-042
Name:G.Rishi

Column	DataType
User_Id	Integer
Name	Varchar
Age	Integer
Phone	Integer
Email	Varchar

PrimaryKey: User_Id

Slot Table:

Column	DataType
Slot_Id	Integer
Location	Varchar

Primary Key: Slot_Id

Vaccine Table:

Column	DataType
Vaccine_Id	Integer
Name	Varchar
Dosage	Integer
Dose_count	Integer

Primary Key:Vaccine_Id

Available Table:

Column	Data Type
Vaccine_Id	Integer
Slot_Id	Integer
Capacity	Integer

Foreign Key:Vaccine_id references Vaccine;

Foreign Key: Slot_Id references Slot;

primaryKey:Vaccine_id& Slot_Id

Booked Table:

Column	DataType
Slot_id	Integer
User_id	Integer

Foreign key: Slot_id references slot

Foreign key: User Id references slot

Primary Key: slot_id&user_id

UserDoseStatus Table:

Column	DataType
User_id	Integer
Vaccine_Id	Integer
Dose_count	Integer

Foreign Key:User_id references user_det

Foreign key: Vaccine_id refences vaccine

Primary key:User_id & Vaccine_id

THROUGH THE PROJECT: This project helps to store data in a efficient way and it can be achieved through various sql commands and we can also store this for any future use and also we can save our data in a many different areas so we cannot lost all the data at once. The Covid Vaccination details are must as to know the number of doses and type of vaccine taken by a person. These project stores details of user, vaccine type and doses taken in database so that whenever it is necessary to know it would be easy for us to access the data.

SOFTWARE USED: Java Eclipse, Oracle 11g Database, Java SE version 8, MYSQL . 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:

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

Table Created in SQL for above mentioned purpose is as:

DDL Commands:

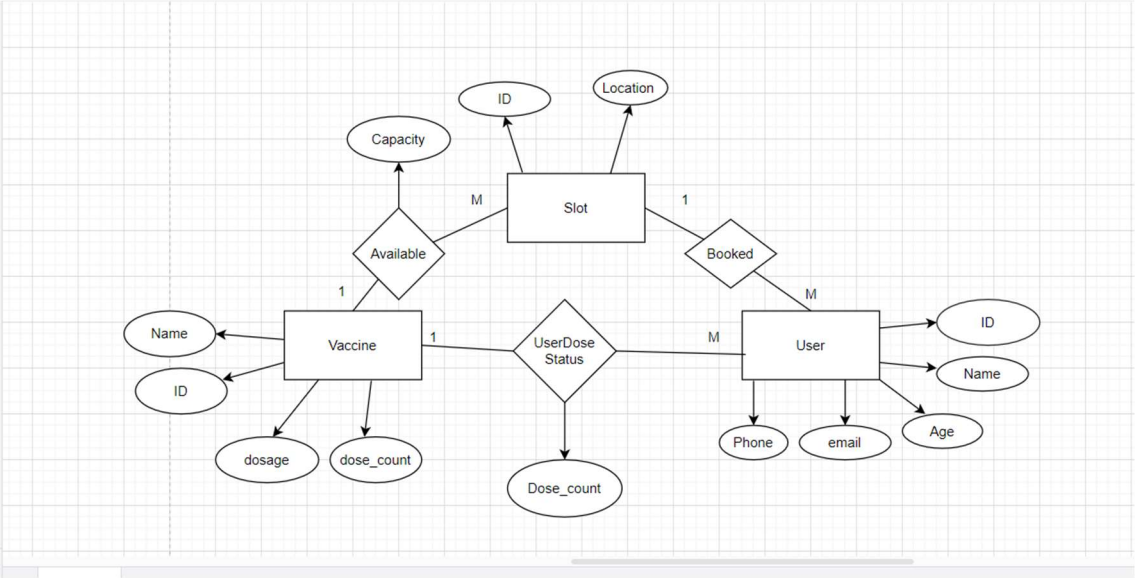
```
SQL> create table userdosestatus(vaccine_id int,user_id int,dose_count int,foreign key(vaccine_id) references vaccine_details,foreign key(user_id) reference
s user_details,primary key(user_id,vaccine_id));

Table created.

SQL>
```

```
SQL> create table booked_det(slot_id int, user_id int ,foreign key(slot_id) references slot_details,foreign key(user_id) references user_details,primary key
(user_id,slot_id));

Table created.

SQL>
```

```
SQL> create table vaccine_details(vaccine_id int primary key,name varchar(225) not null,dosage int not null,dose_count int not null);

Table created.
```

```
SQL> create table user_details(user_id int primary key,name varchar(225) not null,age int not null,email varchar(225) not null);

Table created.
```

```
SQL> desc available;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 VACCINE_ID                                NOT NULL NUMBER(38)
 SLOT_ID                                   NOT NULL NUMBER(38)
 CAPACITY                                           NUMBER(38)

SQL> desc slot;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SLOT_ID                                   NOT NULL NUMBER(38)
 LOCATION                                  NOT NULL VARCHAR2(225)

SQL> desc booked;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 USER_ID                                   NOT NULL NUMBER(38)
 SLOT_ID                                   NOT NULL NUMBER(38)

SQL> desc user_det;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 USER_ID                                   NOT NULL NUMBER(38)
 NAME                                      NOT NULL VARCHAR2(225)
 AGE                                       NOT NULL NUMBER(38)
 EMAIL                                     NOT NULL VARCHAR2(225)
 PHONE                                     NOT NULL VARCHAR2(20)

SQL> desc vaccine;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 VACCINE_ID                                NOT NULL NUMBER(38)
 NAME                                      NOT NULL VARCHAR2(225)
 DOSAGE                                    NOT NULL NUMBER(38)
 DOSE_COUNT                                NOT NULL NUMBER(38)

SQL> desc userdosesataus;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 VACCINE_ID                                NOT NULL NUMBER(38)
 USER_ID                                   NOT NULL NUMBER(38)
 DOSE_COUNT                                NOT NULL NUMBER(38)

SQL>
```



ER Diagram

Rollno:102-21-737-042
Name:G.Rishi

Implementation:

Program:

UserPage.Java

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.sql.*;


public class UserPage extends JFrame{

int id;

private JTextField userIdField;

  private JButton submitButton;

private JTextField slotIdField;

private JTextField vaccineIdField;

private Connection connection;

private Connection conn;

private JPanel bookSlotPanel=new JPanel();

private JPanel upadteUserPanel;

private JPanel viewSlotPanel;

private JPanel viewUserPanel;

private JPanel containerPanel;

private JPanel mainPanel;

private JLabel userIdLabelres, vaccineIdLabelres, countLabelres;

private JPanel resultPanel;

private JPanel panel;


   private JLabel userIdLabelSt;

   private JLabel vaccineIdLabelSt;

   private JLabel firstDoseLabelSt;
```

```
  private JLabel secondDoseLabelSt;
  private JLabel boosterDoseLabelSt;

  private JTextField userIdFieldSt;
  private JTextField vaccineIdFieldSt;
  private JTextField firstDoseFieldSt;
  private JTextField secondDoseFieldSt;
  private JTextField boosterDoseFieldSt;

public UserPage(int userdetuserid){
id=userdetuserid;
mainPanel = new JPanel();
mainPanel.setBounds(0, 0, 400, 300);
mainPanel.setLayout(null);
add(mainPanel);

JMenuBar menuBar = new JMenuBar();
setJMenuBar(menuBar);
JMenu bookSlotMenu = new JMenu("Book Slot");
menuBar.add(bookSlotMenu);

// Create menu item "Book Slot"
JMenuItem bookSlotItem = new JMenuItem("Book Slot");
bookSlotMenu.add(bookSlotItem);

JMenu updateMenu = new JMenu("Update");
menuBar.add(updateMenu);

JMenuItem updateUserItem = new JMenuItem("Update User");
updateMenu.add(updateUserItem);

JMenu viewSlotMenu = new JMenu("View");
menuBar.add(viewSlotMenu);

JMenuItem viewSlotItem = new JMenuItem("View Slot");
viewSlotMenu.add(viewSlotItem);
```

```
JMenu statusMenu = new JMenu("Status");
JMenuItem StatusItem = new JMenuItem("Status");
statusMenu.add(StatusItem);
menuBar.add(statusMenu);

JMenu HomeMenu = new JMenu("SignOut");

JMenuItem HomeItem = new JMenuItem("SignOut");
HomeMenu.add(HomeItem);
menuBar.add(HomeMenu);


viewSlotPanel = new JPanel();
viewSlotPanel.setBounds(0, 0, 400, 300);
viewSlotPanel.setLayout(null);
viewSlotPanel.setVisible(false);
mainPanel.add(viewSlotPanel);

containerPanel = new JPanel();
containerPanel.setBounds(0, 0, 400, 300);
containerPanel.setLayout(new BorderLayout());
containerPanel.setVisible(false);
mainPanel.add(containerPanel);

viewUserPanel = new JPanel();
viewUserPanel.setBounds(0, 0, 400, 300);
viewUserPanel.setLayout(new BorderLayout());
viewUserPanel.setVisible(false);
mainPanel.add(viewUserPanel);

bookSlotPanel.setLayout(null);
bookSlotPanel.setBounds(0, 0, 400, 300);
mainPanel.add(bookSlotPanel);

resultPanel = new JPanel();
```

```
resultPanel.setBounds(0, 0, 400, 300);
resultPanel.setLayout(null);
resultPanel.setVisible(false); // Set layout to null
mainPanel.add(resultPanel);

 panel = new JPanel(null);

 userIdLabelSt = new JLabel("User ID:");
      userIdLabelSt.setBounds(10, 10, 80, 25);
      panel.add(userIdLabelSt);

      vaccineIdLabelSt = new JLabel("Vaccine ID:");
      vaccineIdLabelSt.setBounds(10, 40, 80, 25);
      panel.add(vaccineIdLabelSt);

      firstDoseLabelSt = new JLabel("First Dose:");
      firstDoseLabelSt.setBounds(10, 70, 80, 25);
      panel.add(firstDoseLabelSt);

      secondDoseLabelSt = new JLabel("Second Dose:");
      secondDoseLabelSt.setBounds(10, 100, 80, 25);
      panel.add(secondDoseLabelSt);

      boosterDoseLabelSt = new JLabel("Booster Dose:");
      boosterDoseLabelSt.setBounds(10, 130, 80, 25);
      panel.add(boosterDoseLabelSt);

      userIdFieldSt = new JTextField();
      userIdFieldSt.setEditable(false);
      userIdFieldSt.setBounds(100, 10, 200, 25);
      panel.add(userIdFieldSt);

      vaccineIdFieldSt = new JTextField();
      vaccineIdFieldSt.setEditable(false);
      vaccineIdFieldSt.setBounds(100, 40, 200, 25);
      panel.add(vaccineIdFieldSt);
```

```
        firstDoseFieldSt = new JTextField();
        firstDoseFieldSt.setEditable(false);
        firstDoseFieldSt.setBounds(100, 70, 200, 25);
        panel.add(firstDoseFieldSt);

        secondDoseFieldSt = new JTextField();
        secondDoseFieldSt.setEditable(false);
        secondDoseFieldSt.setBounds(100, 100, 200, 25);
        panel.add(secondDoseFieldSt);

        boosterDoseFieldSt = new JTextField();
        boosterDoseFieldSt.setEditable(false);
        boosterDoseFieldSt.setBounds(100, 130, 200, 25);
        panel.add(boosterDoseFieldSt);

        panel.setBounds(0, 0, 400, 300);
        panel.setLayout(null);
        panel.setVisible(false);
        submitButton = new JButton("View Status");
      submitButton.setBounds(130, 160, 80, 25);
      submitButton.addActionListener(new ActionListener() {
         public void actionPerformed(ActionEvent e) {
            fetchUserDetails();
         }
      });
      panel.add(submitButton);
        mainPanel.add(panel);


vaccineIdLabelres = new JLabel("Vaccine ID:");
countLabelres = new JLabel("Count:");
vaccineIdLabelres.setBounds(10, 10, 80, 25);
countLabelres.setBounds(10, 50, 80, 25);
resultPanel.add(vaccineIdLabelres);
resultPanel.add(countLabelres);
```

```
// Create user ID field
/*JLabel userIdLabel = new JLabel("User ID:");
userIdLabel.setBounds(10, 10, 80, 25);
userIdField = new JTextField(10);
userIdField.setBounds(100, 10, 200, 25);
bookSlotPanel.add(userIdLabel);
bookSlotPanel.add(userIdField);*/

// Create slot ID field
JLabel slotIdLabel = new JLabel("Slot ID:");
slotIdLabel.setBounds(10, 40, 80, 25);
slotIdField = new JTextField(10);
slotIdField.setBounds(100, 40, 200, 25);
bookSlotPanel.add(slotIdLabel);
bookSlotPanel.add(slotIdField);

// Create vaccine ID field
JLabel vaccineIdLabel = new JLabel("Vaccine ID:");
vaccineIdLabel.setBounds(10, 70, 80, 25);
vaccineIdField = new JTextField(10);
vaccineIdField.setBounds(100, 70, 200, 25);
bookSlotPanel.add(vaccineIdLabel);
bookSlotPanel.add(vaccineIdField);

// Create submit button
JButton submitBookButton = new JButton("Submit Booking");
submitBookButton.setBounds(150, 100, 80, 25);
bookSlotPanel.add(submitBookButton);

HomeItem.addActionListener(new ActionListener() {
   @Override
   public void actionPerformed(ActionEvent e) {
     setVisible(false);
```

```
     new LoginPagen();
   }
});


bookSlotItem.addActionListener(new ActionListener() {
   @Override
   public void actionPerformed(ActionEvent e) {

      bookSlotPanel.setVisible(true);
      upadteUserPanel.setVisible(false);
      containerPanel.setVisible(false);
      viewUserPanel.setVisible(false);
      resultPanel.setVisible(false);
      panel.setVisible(false);
   }
});

submitBookButton.addActionListener(new ActionListener() {
   @Override
   public void actionPerformed(ActionEvent e) {

      String slotId = slotIdField.getText();
      String vaccineId = vaccineIdField.getText();

      try {
        try{
        Class.forName("oracle.jdbc.driver.OracleDriver");
      }
      catch(Exception q){
        q.printStackTrace();
      }
        connection=DriverManager.getConnection(
          "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");

          // Insert values into booked table
```

```
            String insertQuery = "INSERT INTO booked (user_id, slot_id, vaccine_id) VALUES (?, ?, ?)";
            PreparedStatement insertStatement = connection.prepareStatement(insertQuery);
            insertStatement.setInt(1, userdetuserid );
            insertStatement.setString(2, slotId);
            insertStatement.setString(3, vaccineId);
            insertStatement.executeUpdate();
              connection.commit();
            insertStatement.close();

            // Update available table
            String updateQuery = "UPDATE available SET capacity = capacity - 1 WHERE slot_id = ? AND vaccine_id = ?";
            setVisible(false);
            setVisible(true);
            PreparedStatement updateStatement = connection.prepareStatement(updateQuery);
            updateStatement.setString(1, slotId);
            updateStatement.setString(2, vaccineId);
            updateStatement.executeUpdate();
            connection.commit();
            updateStatement.close();

            // Show a message dialog to indicate successful booking
            //connection.commit();
            connection.close();
            JOptionPane.showMessageDialog(UserPage.this,
                    "Booking successful!", "Success", JOptionPane.INFORMATION_MESSAGE);
            updatedet(0);



        } catch (SQLException ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(UserPage.this,
                    "Error occurred during booking.", "Error", JOptionPane.ERROR_MESSAGE);
        }
    }
});
upadteUserPanel = new JPanel();
```

```
upadteUserPanel.setBounds(0, 0, 400, 300);
upadteUserPanel.setLayout(null);
upadteUserPanel.setVisible(false);
mainPanel.add(upadteUserPanel);



JLabel lblUpdateUserName = new JLabel("User Name:");
lblUpdateUserName.setBounds(50, 80, 100, 25);
upadteUserPanel.add(lblUpdateUserName);

JTextField txtUpdateUserName = new JTextField();
txtUpdateUserName.setBounds(150, 80, 200, 25);
upadteUserPanel.add(txtUpdateUserName);

JLabel lblUpdateAge = new JLabel("Age");
lblUpdateAge.setBounds(50, 110, 100, 25);
upadteUserPanel.add(lblUpdateAge);

JTextField txtUpdateAge = new JTextField();
txtUpdateAge.setBounds(150, 110, 200, 25);
upadteUserPanel.add(txtUpdateAge);

JButton btnUpdateUser = new JButton("Update");
btnUpdateUser.setBounds(150, 160, 100, 30);



btnUpdateUser.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // Handle user update

        String userName = txtUpdateUserName.getText();
        int age = Integer.parseInt(txtUpdateAge.getText());
        // Show user update form
```

```
        updateUser(userdetuserid,userName,age);


        // Reset the text field


        txtUpdateUserName.setText("");
        txtUpdateAge.setText("");
    }
});
upadteUserPanel.add(btnUpdateUser);


updateUserItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        upadteUserPanel.setVisible(true);
        bookSlotPanel.setVisible(false);
        containerPanel.setVisible(false);
        viewUserPanel.setVisible(false);
        resultPanel.setVisible(false);
        panel.setVisible(false);




    }
});


StatusItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        upadteUserPanel.setVisible(false);
        bookSlotPanel.setVisible(false);
        containerPanel.setVisible(false);
        viewUserPanel.setVisible(false);
        resultPanel.setVisible(false);
        panel.setVisible(true);


    }
});
```

```
viewSlotItem.addActionListener(new ActionListener() {
   @Override
   public void actionPerformed(ActionEvent e) {
    /*try {
    try{
    Class.forName("oracle.jdbc.driver.OracleDriver");
   }
   catch(Exception q){
    q.printStackTrace();
   }
   connection=DriverManager.getConnection(
    "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");
    Statement statement = connection.createStatement();

        // Execute the query
        String query = "SELECT s.slot_id, v.vaccine_id, v.name,a.capacity FROM slot s JOIN available a ON s.slot_id = a.slot_id JOIN
vaccine v ON a.vaccine_id = v.vaccine_id";
        ResultSet resultSet = statement.executeQuery(query);

        // Create a 2D array to store the data
        Object[][] data = new Object[100][4]; // Assuming there are 100 rows in the Slot table

        // Populate the data array with the query results
        int row = 0;
        while (resultSet.next()) {
          int id = resultSet.getInt("slot_id");
          int vid = resultSet.getInt("vaccine_id");
          String name = resultSet.getString("name");
          int capacity = resultSet.getInt("capacity");

          data[row][0] = id;
          data[row][1] = vid;
          data[row][2] = name;
          data[row][3] = capacity;
          row++;
        }
```

```
        // Define the column names
        String[] columnNames = {"ID", "Vid", "Name", "Capacity"};

        // Create a JTable with the data and column names
        JTable table = new JTable(data, columnNames);

        // Create a JScrollPane to add scroll functionality to the table
        JScrollPane scrollPane = new JScrollPane(table);

        // Create a view panel to hold additional components
        JPanel viewPanel = new JPanel();
        // Add components to the view panel
        viewPanel.add(new JLabel("View Panel Component"));

        // Create a container panel to hold the table and the view panel
        //JPanel containerPanel = new JPanel(new BorderLayout());
        containerPanel.add(scrollPane, BorderLayout.CENTER);
        containerPanel.add(viewPanel, BorderLayout.SOUTH);
       //  mainPanel.add(containerPanel);
      containerPanel.setVisible(true);
      bookSlotPanel.setVisible(false);
      upadteUserPanel.setVisible(false);
      viewUserPanel.setVisible(false);
      resultPanel.setVisible(false);
    }
    catch(Exception q){
     q.printStackTrace();
    }*/
    updatedet(1);

}});


/*viewUserItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
```

```
 try {
 try{
 Class.forName("oracle.jdbc.driver.OracleDriver");
}
catch(Exception q){
 q.printStackTrace();
}
connection=DriverManager.getConnection(
 "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");
      //Statement preparedStatement = connection.preparedStatement();
      Statement statement = connection.createStatement();


      // Execute query to fetch user dose status based on user ID
      int uri=737042;
      String query = "SELECT vaccine_id, count FROM userdosestatus WHERE user_id = "+userdetuserid ;
      ResultSet resultSet = statement.executeQuery(query);



      // Create a 2D array to store the data
      if (resultSet.next()) {
        // Retrieve data from the result set
        String vaccineIdres = resultSet.getString("vaccine_id");
        int countres = resultSet.getInt("count");


        // Update the labels in the result panel
        vaccineIdLabelres.setText("Vaccine ID: " + vaccineIdres);
        countLabelres.setText("Count: " + countres);
      } else {
        // User dose status not found
        JOptionPane.showMessageDialog(UserPage.this, "User dose status not found!");
      }
      connection.close();
      resultPanel.setVisible(true);
     containerPanel.setVisible(false);
     bookSlotPanel.setVisible(false);
     upadteUserPanel.setVisible(false);
```

```
        }
        catch(Exception q){
          q.printStackTrace();
        }

    }


});*/
setSize(500, 500);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setVisible(true);
setLayout(null);
}
private void updatedet(int x)
{
  try {
  try{
  Class.forName("oracle.jdbc.driver.OracleDriver");
}
catch(Exception q){
  q.printStackTrace();
}
connection=DriverManager.getConnection(
  "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");
  Statement statement = connection.createStatement();

      // Execute the query
      String query = "SELECT s.slot_id, v.vaccine_id, v.name,a.capacity FROM slot s JOIN available a ON s.slot_id = a.slot_id JOIN vaccine v ON a.vaccine_id = v.vaccine_id";
      ResultSet resultSet = statement.executeQuery(query);

      // Create a 2D array to store the data
      Object[][] data = new Object[100][4]; // Assuming there are 100 rows in the Slot table

      // Populate the data array with the query results
      int row = 0;
```

```
    while (resultSet.next()) {
        int id = resultSet.getInt("slot_id");
        int vid = resultSet.getInt("vaccine_id");
        String name = resultSet.getString("name");
        int capacity = resultSet.getInt("capacity");

        data[row][0] = id;
        data[row][1] = vid;
        data[row][2] = name;
        data[row][3] = capacity;
        row++;
    }

    // Define the column names
    String[] columnNames = {"ID", "Vid", "Name", "Capacity"};

    // Create a JTable with the data and column names
    JTable table = new JTable(data, columnNames);

    // Create a JScrollPane to add scroll functionality to the table
    JScrollPane scrollPane = new JScrollPane(table);

    // Create a view panel to hold additional components
    JPanel viewPanel = new JPanel();
    // Add components to the view panel
    viewPanel.add(new JLabel("View Panel Component"));

    // Create a container panel to hold the table and the view panel
    //JPanel containerPanel = new JPanel(new BorderLayout());
    containerPanel.add(scrollPane, BorderLayout.CENTER);
    containerPanel.add(viewPanel, BorderLayout.SOUTH);
   //  mainPanel.add(containerPanel);
   connection.close();

  containerPanel.setVisible(true);
```

```
    bookSlotPanel.setVisible(false);
    upadteUserPanel.setVisible(false);
    viewUserPanel.setVisible(false);
    resultPanel.setVisible(false);
}
catch(Exception q){
  q.printStackTrace();
}
}
private void updateUser(int id,String userName, int age) {
    // Add code to show the form to update user details
    // You can use the same approach as the insertUser method to create the form and update the user record in the database
    try {
      try{
      Class.forName("oracle.jdbc.driver.OracleDriver");
    }
    catch(Exception q){
      q.printStackTrace();
    }
    conn=DriverManager.getConnection(
      "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");

        // Prepare SQL statement
        String query = "UPDATE user_det SET user_name = ?, age = ? WHERE user_id = ?";
        PreparedStatement preparedStatement = conn.prepareStatement(query);

        // Set parameter values
        preparedStatement.setString(1, userName);
        preparedStatement.setInt(2, age);
        preparedStatement.setInt(3, id);

        // Execute the query
        int rowsAffected = preparedStatement.executeUpdate();
          //connection.commit();
```

```
        // Close the statement and connection
        preparedStatement.close();
        conn.close();

        if (rowsAffected > 0) {
            JOptionPane.showMessageDialog(null, "User updated successfully!");
        } else {
            JOptionPane.showMessageDialog(null, "User with the given ID not found.");
        }

    } catch (SQLException ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(null, "Error: Failed to update User.");
    }
}

private void fetchUserDetails() {
    Connection connection = null;
    Statement statement = null;
    ResultSet resultSet = null;
try{
    try{
    Class.forName("oracle.jdbc.driver.OracleDriver");
  }
  catch(Exception q){
    q.printStackTrace();
  }
  connection=DriverManager.getConnection(
    "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");
        statement = connection.createStatement();
        resultSet = statement.executeQuery("SELECT * FROM userdosestatus WHERE user_id = "+id) ;

        if (resultSet.next()) {
            int userId = resultSet.getInt("user_id");
            int vaccineId = resultSet.getInt("vaccine_id");
```

```
            int firstDose = resultSet.getInt("firstdose");
            int secondDose = resultSet.getInt("seconddose");
            int boosterDose = resultSet.getInt("boosterdose");

            userIdFieldSt.setText(String.valueOf(userId));
            vaccineIdFieldSt.setText(String.valueOf(vaccineId));
            firstDoseFieldSt.setText((firstDose == 1) ? "Taken" : "Not Taken");
            secondDoseFieldSt.setText((secondDose == 1) ? "Taken" : "Not Taken");
            boosterDoseFieldSt.setText((boosterDose == 1) ? "Taken" : "Not Taken");
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}
public static void main(String[] args) {
    // Create and show the login page
    SwingUtilities.invokeLater(new Runnable() {
        public void run() {

            UserPage app = new UserPage(1);
            app.setVisible(true);
        }
    });
}
}
```

AdminPage.java

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import javax.swing.table.DefaultTableModel;

import java.sql.*;


public class AdminPage extends JFrame{


private JPanel mainPanel;

private JPanel vaccinePanel;

private JPanel slotPanel;

private JPanel statusPanel;

private JPanel availablePanel;

private JPanel vaccineUpdatePanel;

private JPanel slotUpdatePanel;

private JPanel containerPaneln;

//private JPanel vaccineDeletePanel;

private JPanel slotDeletePanel;

private JPanel userDeletePanel;

private JPanel viewSlotPanel;

private JPanel containerPanel;

private JPanel tablePanel;

private JRadioButton firstDoseRadioButtonSt, secondDoseRadioButtonSt, boosterDoseRadioButtonSt;

private JTextField userIdTextFieldSt, vaccineIdTextFieldSt;

private JPanel panel;


private JLabel vaccineLabel;

private JTextField vaccineField;

private JButton countButton;

private JLabel countLabel;
```

```
private JTable table;
private DefaultTableModel model;

private Connection conn;
private Connection connection;

public AdminPage(){

setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLayout(null);

mainPanel = new JPanel();
mainPanel.setBounds(0, 0, 400, 300);
mainPanel.setLayout(null);
add(mainPanel);

JMenuBar menuBar = new JMenuBar();
setJMenuBar(menuBar);

JMenu insertMenu = new JMenu("Insert");

JMenuItem insertVaccineItem = new JMenuItem("Insert Vaccine");
JMenuItem insertSlotItem = new JMenuItem("Insert Slot");
JMenuItem insertStatusItem = new JMenuItem("Insert Status");
JMenuItem insertAvailableItem = new JMenuItem("Insert Available");

insertMenu.add(insertVaccineItem);
insertMenu.add(insertSlotItem);
insertMenu.add(insertStatusItem);
insertMenu.add(insertAvailableItem);

menuBar.add(insertMenu);

JMenu updateMenu = new JMenu("Update");
```

```
JMenuItem updateVaccineItem = new JMenuItem("Update Vaccine");
JMenuItem updateSlotItem = new JMenuItem("Update Slot");

updateMenu.add(updateVaccineItem);
updateMenu.add(updateSlotItem);

menuBar.add(updateMenu);

JMenu deleteMenu = new JMenu("Delete");

JMenuItem deleteSlotItem = new JMenuItem("Delete Slot");

deleteMenu.add(deleteSlotItem);

menuBar.add(deleteMenu);

JMenu viewSlotMenu = new JMenu("View");

JMenuItem viewSlotItem = new JMenuItem("View Slot");
viewSlotMenu.add(viewSlotItem);
JMenuItem viewCountItem = new JMenuItem("View count");
viewSlotMenu.add(viewCountItem);
JMenuItem viewVaccineItem = new JMenuItem("View Vaccine");
viewSlotMenu.add(viewVaccineItem);

menuBar.add(viewSlotMenu);

JMenu HomeMenu = new JMenu("SignOut");

JMenuItem HomeItem = new JMenuItem("Signout");
HomeMenu.add(HomeItem);
menuBar.add(HomeMenu);
```

```
availablePanel = new JPanel();
availablePanel.setBounds(0, 0, 400, 300);
availablePanel.setLayout(null);
availablePanel.setVisible(false);
mainPanel.add(availablePanel);

statusPanel = new JPanel();
statusPanel.setBounds(0, 0, 400, 300);
statusPanel.setLayout(null);
statusPanel.setVisible(false);
mainPanel.add(statusPanel);

slotPanel = new JPanel();
slotPanel.setBounds(0, 0, 400, 300);
slotPanel.setLayout(null);
slotPanel.setVisible(false);
mainPanel.add(slotPanel);

vaccinePanel = new JPanel();
vaccinePanel.setBounds(0, 0, 400, 300);
vaccinePanel.setLayout(null);
vaccinePanel.setVisible(false);
mainPanel.add(vaccinePanel);

vaccineUpdatePanel= new JPanel();
vaccineUpdatePanel.setBounds(0, 0, 400, 300);
vaccineUpdatePanel.setLayout(null);
vaccineUpdatePanel.setVisible(false);
mainPanel.add(vaccineUpdatePanel);

slotUpdatePanel = new JPanel();
slotUpdatePanel.setBounds(0, 0, 400, 300);
slotUpdatePanel.setLayout(null);
slotUpdatePanel.setVisible(false);
```

```
mainPanel.add(slotUpdatePanel);



slotDeletePanel = new JPanel();
slotDeletePanel.setBounds(0, 0, 400, 300);
slotDeletePanel.setLayout(null);
slotDeletePanel.setVisible(false);
mainPanel.add(slotDeletePanel);



viewSlotPanel = new JPanel();
viewSlotPanel.setBounds(0, 0, 400, 300);
viewSlotPanel.setLayout(null);
viewSlotPanel.setVisible(false);
mainPanel.add(viewSlotPanel);

containerPanel = new JPanel();
containerPanel.setBounds(0, 0, 400, 300);
containerPanel.setLayout(new BorderLayout());
containerPanel.setVisible(false);
mainPanel.add(containerPanel);

containerPaneIn = new JPanel();
containerPaneIn.setBounds(0, 0, 400, 300);
containerPaneIn.setLayout(new BorderLayout());
containerPaneIn.setVisible(false);
mainPanel.add(containerPaneIn);



JLabel lblVaccineIdava = new JLabel("Vaccine ID:");
lblVaccineIdava.setBounds(50, 50, 100, 25);
availablePanel.add(lblVaccineIdava);
```

```
JTextField txtVaccineIdava = new JTextField();
txtVaccineIdava.setBounds(150, 50, 200, 25);
availablePanel.add(txtVaccineIdava);

JLabel lblSlotIdAva = new JLabel("Slot Id:");
lblSlotIdAva.setBounds(50, 100, 100, 25);
availablePanel.add(lblSlotIdAva);

JTextField txtSlotIdAva = new JTextField();
txtSlotIdAva.setBounds(150, 100, 200, 25);
availablePanel.add(txtSlotIdAva);

JLabel lblCapacity = new JLabel("Capacity:");
lblCapacity.setBounds(50, 150, 100, 25);
availablePanel.add(lblCapacity);

JTextField txtCapacity = new JTextField();
txtCapacity.setBounds(150, 150, 200, 25);
availablePanel.add(txtCapacity);

JButton btnSubmitAvailable = new JButton("Submit");
btnSubmitAvailable.setBounds(150, 200, 100, 30);
btnSubmitAvailable.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // Handle user details submission
        String VaccineIdava = txtVaccineIdava.getText();
        String SlotIdAva = txtSlotIdAva.getText();
        String CapacityAva = txtCapacity.getText();

        // Insert user details into the database
        insertAvailableDetails(VaccineIdava,SlotIdAva,CapacityAva);

        // Reset the text fields
        txtVaccineIdava.setText("");
        txtSlotIdAva.setText("");
```

```
        txtCapacity.setText("");
    }
});
availablePanel.add(btnSubmitAvailable);

/*JLabel lblUserIdSt = new JLabel("User ID:");
lblUserIdSt.setBounds(50, 50, 100, 25);
statusPanel.add(lblUserIdSt);

JTextField txtUserIdSt = new JTextField();
txtUserIdSt.setBounds(150, 50, 200, 25);
statusPanel.add(txtUserIdSt);

JLabel lblVaccineIdSt = new JLabel("Vaccine Id:");
lblVaccineIdSt.setBounds(50, 100, 100, 25);
statusPanel.add(lblVaccineIdSt);

JTextField txtVaccineIdSt = new JTextField();
txtVaccineIdSt .setBounds(150, 100, 200, 25);
statusPanel.add(txtVaccineIdSt);

JLabel lblCountSt = new JLabel("Count:");
lblCountSt.setBounds(50, 150, 100, 25);
statusPanel.add(lblCountSt);

JTextField txtCountSt = new JTextField();
txtCountSt.setBounds(150, 150, 200, 25);
statusPanel.add(txtCountSt);

JButton btnSubmitStatus = new JButton("Submit");
btnSubmitStatus.setBounds(150, 200, 100, 30);
btnSubmitStatus.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
    // Handle user details submission
    String userIdSt = txtUserIdSt.getText();
```

```
    String vaccineIdSt = txtVaccineIdSt.getText();
    String countSt = txtCountSt.getText();

    // Insert user details into the database
    insertStatusDetails(userIdSt, vaccineIdSt, countSt);

    // Reset the text fields
    txtUserIdSt.setText("");
    txtVaccineIdSt.setText("");
    txtCountSt.setText("");
}
});
statusPanel.add(btnSubmitStatus);*/



panel = new JPanel(null);
panel.setBounds(0, 0, 400, 300);
panel.setLayout(null);

vaccineLabel = new JLabel("Vaccine ID:");
vaccineLabel.setBounds(10, 10, 80, 25);
panel.add(vaccineLabel);

vaccineField = new JTextField();
vaccineField.setBounds(100, 10, 100, 25);
panel.add(vaccineField);

countButton = new JButton("Count");
countButton.setBounds(210, 10, 80, 25);
countButton.addActionListener(e -> countUsers());
panel.add(countButton);

countLabel = new JLabel("");
countLabel.setBounds(10, 40, 280, 25);
panel.add(countLabel);
```

```
mainPanel.add(panel);

JLabel userIdLabelSt = new JLabel("User ID:");
    userIdLabelSt.setBounds(20, 20, 80, 25);
    statusPanel.add(userIdLabelSt);

    userIdTextFieldSt = new JTextField();
    userIdTextFieldSt.setBounds(110, 20, 150, 25);
    statusPanel.add(userIdTextFieldSt);

    // Create and position the Vaccine ID label and text field
    JLabel vaccineIdLabelSt = new JLabel("Vaccine ID:");
    vaccineIdLabelSt.setBounds(20, 60, 80, 25);
    statusPanel.add(vaccineIdLabelSt);

    vaccineIdTextFieldSt = new JTextField();
    vaccineIdTextFieldSt.setBounds(110, 60, 150, 25);
    statusPanel.add(vaccineIdTextFieldSt);

    // Create and position the Dose label and radio buttons
    JLabel doseLabelSt = new JLabel("Dose:");
    doseLabelSt.setBounds(20, 100, 80, 25);
    statusPanel.add(doseLabelSt);

    firstDoseRadioButtonSt = new JRadioButton("First Dose");
    firstDoseRadioButtonSt.setBounds(110, 100, 100, 25);
    statusPanel.add(firstDoseRadioButtonSt);

    secondDoseRadioButtonSt = new JRadioButton("Second Dose");
    secondDoseRadioButtonSt.setBounds(220, 100, 120, 25);
    statusPanel.add(secondDoseRadioButtonSt);

    boosterDoseRadioButtonSt = new JRadioButton("Booster Dose");
    boosterDoseRadioButtonSt.setBounds(110, 130, 120, 25);
    statusPanel.add(boosterDoseRadioButtonSt);
```

```
    // Group the radio buttons together
    ButtonGroup doseButtonGroup = new ButtonGroup();
    doseButtonGroup.add(firstDoseRadioButtonSt);
    doseButtonGroup.add(secondDoseRadioButtonSt);
    doseButtonGroup.add(boosterDoseRadioButtonSt);

    JButton submitButtonstatus = new JButton("Submit");
    submitButtonstatus.setBounds(150, 170, 100, 30);
    statusPanel.add(submitButtonstatus);

    // Add action listener to the submit button
    submitButtonstatus.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            // Get the selected dose
            String dose = "";
            if (firstDoseRadioButtonSt.isSelected()) {
                dose = "First Dose";
            } else if (secondDoseRadioButtonSt.isSelected()) {
                dose = "Second Dose";
            } else if (boosterDoseRadioButtonSt.isSelected()) {
                dose = "Booster Dose";
            }

            // Get the user ID and vaccine ID
            String userId = userIdTextFieldSt.getText();
            String vaccineId = vaccineIdTextFieldSt.getText();

            // Update the user dose status
            updateUserDoseStatus(userId, vaccineId, dose);
        }
    });

    // Add the status panel to the main panel
    mainPanel.add(statusPanel);
```

```
    // Add the main panel to the frame
```

```
JLabel lblslotID = new JLabel("SlotId:");
lblslotID.setBounds(50, 50, 100, 25);
slotPanel.add(lblslotID);

JTextField txtslotID = new JTextField();
txtslotID.setBounds(150, 50, 200, 25);
slotPanel.add(txtslotID);

JLabel lblslotLocation = new JLabel("slotLocation:");
lblslotLocation.setBounds(50, 100, 100, 25);
slotPanel.add(lblslotLocation);

JTextField txtslotLocation = new JTextField();
txtslotLocation.setBounds(150, 100, 200, 25);
slotPanel.add(txtslotLocation);

JLabel lblslotName = new JLabel("slot Name:");
lblslotName.setBounds(50, 150, 100, 25);
slotPanel.add(lblslotName);

JTextField txtslotName = new JTextField();
txtslotName.setBounds(150, 150, 200, 25);
slotPanel.add(txtslotName);

JButton btnSubmitSlot = new JButton("Submit");
btnSubmitSlot.setBounds(150, 200, 100, 30);

btnSubmitSlot.addActionListener(new ActionListener() {
```

```
    @Override
    public void actionPerformed(ActionEvent e) {
        // Handle vaccine details submission
        String SlotId = txtslotID.getText();
        String slotLocation = txtslotLocation.getText();
        String slotName = txtslotName.getText();

        // Insert vaccine details into the database
        insertSlotDetails(SlotId, slotLocation, slotName);

        // Reset the text fields
        txtslotID.setText("");
        txtslotLocation.setText("");
        txtslotName.setText("");
    }
});
slotPanel.add(btnSubmitSlot);

JLabel lblVaccineId = new JLabel("Vaccine ID:");
lblVaccineId.setBounds(50, 50, 100, 25);
vaccinePanel.add(lblVaccineId);

JTextField txtVaccineId = new JTextField();
txtVaccineId.setBounds(150, 50, 200, 25);
vaccinePanel.add(txtVaccineId);

JLabel lblVaccineName = new JLabel("Vaccine Name:");
lblVaccineName.setBounds(50, 100, 100, 25);
vaccinePanel.add(lblVaccineName);

JTextField txtVaccineName = new JTextField();
txtVaccineName.setBounds(150, 100, 200, 25);
vaccinePanel.add(txtVaccineName);

/*JLabel lblDoseCount = new JLabel("Dose Count:");
lblDoseCount.setBounds(50, 150, 100, 25);
```

```
vaccinePanel.add(lblDoseCount);

JTextField txtDoseCount = new JTextField();
txtDoseCount.setBounds(150, 150, 200, 25);
vaccinePanel.add(txtDoseCount);*/

JButton btnSubmitVaccine = new JButton("Submit");
btnSubmitVaccine.setBounds(150, 200, 100, 30);
btnSubmitVaccine.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // Handle vaccine details submission
        String vaccineId = txtVaccineId.getText();
        String vaccineName = txtVaccineName.getText();
        String doseCount = "3";

        // Insert vaccine details into the database
        insertVaccineDetails(vaccineId, vaccineName, "3");

        // Reset the text fields
        txtVaccineId.setText("");
        txtVaccineName.setText("");

    }
});
vaccinePanel.add(btnSubmitVaccine);

JLabel lblVaccineUpdateId = new JLabel("Vaccine ID:");
        lblVaccineUpdateId.setBounds(50, 50, 100, 25);
        vaccineUpdatePanel.add(lblVaccineUpdateId);

        JTextField txtVaccineUpdateId = new JTextField();
```

```
txtVaccineUpdateId.setBounds(150, 50, 200, 25);
vaccineUpdatePanel.add(txtVaccineUpdateId);

JLabel lblVaccineUpdateName = new JLabel("Vaccine Name:");
lblVaccineUpdateName.setBounds(50, 80, 100, 25);
vaccineUpdatePanel.add(lblVaccineUpdateName);

JTextField txtVaccineUpdateName = new JTextField();
txtVaccineUpdateName.setBounds(150, 80, 200, 25);
vaccineUpdatePanel.add(txtVaccineUpdateName);

JLabel lblDoseUpdateCount = new JLabel("Dose Count:");
lblDoseUpdateCount.setBounds(50, 110, 100, 25);
vaccineUpdatePanel.add(lblDoseUpdateCount);

JTextField txtDoseUpdateCount = new JTextField();
txtDoseUpdateCount.setBounds(150, 110, 200, 25);
vaccineUpdatePanel.add(txtDoseUpdateCount);

JButton btnUpdateVaccine = new JButton("Update");
btnUpdateVaccine.setBounds(150, 160, 100, 30);
btnUpdateVaccine.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // Handle vaccine update
        String vaccineId = txtVaccineUpdateId.getText();
        String vaccineName = txtVaccineUpdateId.getText();
        int doseCount = Integer.parseInt(txtDoseUpdateCount.getText());

        // Update vaccine in the database
        updateVaccine(vaccineId, vaccineName, doseCount);

        // Reset the text fields
        txtVaccineUpdateId.setText("");
        txtVaccineUpdateId.setText("");
        txtDoseUpdateCount.setText("");
```

```
        }
    });
    vaccineUpdatePanel.add(btnUpdateVaccine);
    JLabel lblSlotUpdateId = new JLabel("Slot ID:");
        lblSlotUpdateId.setBounds(50, 50, 100, 25);
        slotUpdatePanel.add(lblSlotUpdateId);

        JTextField txtSlotUpdateId = new JTextField();
        txtSlotUpdateId.setBounds(150, 50, 200, 25);
        slotUpdatePanel.add(txtSlotUpdateId);

        JLabel lblLocationUpdateName = new JLabel("Location Name:");
        lblLocationUpdateName.setBounds(50, 80, 100, 25);
        slotUpdatePanel.add(lblLocationUpdateName);

        JTextField txtLocationUpdateName = new JTextField();
        txtLocationUpdateName.setBounds(150, 80, 200, 25);
        slotUpdatePanel.add(txtLocationUpdateName);

        JLabel lblSlotUpdateName = new JLabel("Slot Name:");
        lblSlotUpdateName.setBounds(50, 110, 100, 25);
        slotUpdatePanel.add(lblSlotUpdateName);

        JTextField txtSlotUpdateName = new JTextField();
        txtSlotUpdateName.setBounds(150, 110, 200, 25);
        slotUpdatePanel.add(txtSlotUpdateName);

        JButton btnUpdateSlot = new JButton("Update");
        btnUpdateSlot.setBounds(150, 160, 100, 30);
        btnUpdateSlot.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // Handle slot update
                String slotId = txtSlotUpdateId.getText();
                String slotLocation = txtLocationUpdateName.getText();
                String slotName = txtSlotUpdateName.getText();
```

```
            // Show slot update form
            updateSlot(slotId,slotLocation,slotName);

            // Reset the text field
            txtSlotUpdateId.setText("");
                    txtLocationUpdateName.setText("");
                    txtSlotUpdateName.setText("");
        }
    });
    slotUpdatePanel.add(btnUpdateSlot);



            JLabel lblDeleteSlotId = new JLabel("Slot ID:");
                lblDeleteSlotId.setBounds(50, 50, 100, 25);
                slotDeletePanel.add(lblDeleteSlotId);

                JTextField txtDeleteSlotId = new JTextField();
                txtDeleteSlotId.setBounds(150, 50, 200, 25);
                slotDeletePanel.add(txtDeleteSlotId);

                JButton btnDeleteSlot = new JButton("Delete");
                btnDeleteSlot.setBounds(150, 100, 100, 30);
                btnDeleteSlot.addActionListener(new ActionListener() {
                    @Override
                    public void actionPerformed(ActionEvent e) {
                        // Handle slot deletion
                        String slotId = txtDeleteSlotId.getText();

                        // Delete slot from the database
                        deleteSlot(slotId);

                        // Reset the text field
                        txtDeleteSlotId.setText("");
                    }
```

```
                    });
                    slotDeletePanel.add(btnDeleteSlot);
```

```
insertAvailableItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        slotPanel.setVisible(false);
        statusPanel.setVisible(false);
        availablePanel.setVisible(true);
        vaccinePanel.setVisible(false);
        vaccineUpdatePanel.setVisible(false);
         slotUpdatePanel.setVisible(false);

         panel.setVisible(false);

         slotDeletePanel.setVisible(false);

         containerPanel.setVisible(false);
           containerPaneIn.setVisible(false);

    }
});
```

```
viewCountItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        slotPanel.setVisible(false);
        statusPanel.setVisible(false);
        availablePanel.setVisible(false);
```

```
        vaccinePanel.setVisible(false);
        vaccineUpdatePanel.setVisible(false);
         slotUpdatePanel.setVisible(false);
         panel.setVisible(true);

         slotDeletePanel.setVisible(false);

         containerPanel.setVisible(false);
           containerPaneIn.setVisible(false);



    }
});


insertStatusItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {


         slotPanel.setVisible(false);
         statusPanel.setVisible(true);
         availablePanel.setVisible(false);
           vaccinePanel.setVisible(false);
           vaccineUpdatePanel.setVisible(false);
             slotUpdatePanel.setVisible(false);
             panel.setVisible(false);
             slotDeletePanel.setVisible(false);

             containerPanel.setVisible(false);
               containerPaneIn.setVisible(false);
```

```
    }
});

insertSlotItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {

        slotPanel.setVisible(true);
         statusPanel.setVisible(false);
         availablePanel.setVisible(false);
           vaccinePanel.setVisible(false);
           vaccineUpdatePanel.setVisible(false);
             slotUpdatePanel.setVisible(false);
             panel.setVisible(false);

             containerPanel.setVisible(false);
               containerPaneIn.setVisible(false);




    }
});

insertVaccineItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {

        slotPanel.setVisible(false);
        statusPanel.setVisible(false);
        vaccinePanel.setVisible(true);
        availablePanel.setVisible(false);
        vaccineUpdatePanel.setVisible(false);
         slotUpdatePanel.setVisible(false);
         panel.setVisible(false);

         containerPanel.setVisible(false);
```

```
            containerPaneIn.setVisible(false);



    }
});

updateVaccineItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
      slotPanel.setVisible(false);
      statusPanel.setVisible(false);
      availablePanel.setVisible(false);
        vaccinePanel.setVisible(false);
        vaccineUpdatePanel.setVisible(true);
        slotUpdatePanel.setVisible(false);
        panel.setVisible(false);



        containerPanel.setVisible(false);
          containerPaneIn.setVisible(false);



    }
});

updateSlotItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
      slotPanel.setVisible(false);
      statusPanel.setVisible(false);
      availablePanel.setVisible(false);
```

```
        vaccinePanel.setVisible(false);
        panel.setVisible(false);
        vaccineUpdatePanel.setVisible(false);
        slotUpdatePanel.setVisible(true);
        slotDeletePanel.setVisible(false);

        containerPanel.setVisible(false);
          containerPaneIn.setVisible(false);

    }
});

deleteSlotItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
      slotPanel.setVisible(false);
      statusPanel.setVisible(false);
      availablePanel.setVisible(false);
      vaccinePanel.setVisible(false);
      vaccineUpdatePanel.setVisible(false);
        slotUpdatePanel.setVisible(false);
        panel.setVisible(false);
        slotDeletePanel.setVisible(true);

        containerPanel.setVisible(false);
          containerPaneIn.setVisible(false);

    }
});
```

```
HomeItem.addActionListener(new ActionListener() {
   @Override
   public void actionPerformed(ActionEvent e) {
    setVisible(false);
    new LoginPagen();
   }
});

viewSlotItem.addActionListener(new ActionListener() {
   @Override
   public void actionPerformed(ActionEvent e) {
    try {
    try{
    Class.forName("oracle.jdbc.driver.OracleDriver");
   }
   catch(Exception q){
    q.printStackTrace();
   }
   connection=DriverManager.getConnection(
    "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");
    Statement statement = connection.createStatement();

        // Execute the query
        String query = "SELECT s.slot_id, v.vaccine_id, v.name,a.capacity FROM slot s JOIN available a ON s.slot_id = a.slot_id JOIN
vaccine v ON a.vaccine_id = v.vaccine_id";
        ResultSet resultSet = statement.executeQuery(query);

        // Create a 2D array to store the data
        Object[][] data = new Object[100][4]; // Assuming there are 100 rows in the Slot table

        // Populate the data array with the query results
        int row = 0;
        while (resultSet.next()) {
          int id = resultSet.getInt("slot_id");
          int vid = resultSet.getInt("vaccine_id");
          String name = resultSet.getString("name");
          int capacity = resultSet.getInt("capacity");
```

```
            data[row][0] = id;
            data[row][1] = vid;
            data[row][2] = name;
            data[row][3] = capacity;
            row++;
        }

        // Define the column names
        String[] columnNames = {"ID", "Vid", "Name", "Capacity"};

        // Create a JTable with the data and column names
        JTable table = new JTable(data, columnNames);

        // Create a JScrollPane to add scroll functionality to the table
        JScrollPane scrollPane = new JScrollPane(table);

        // Create a view panel to hold additional components
        JPanel viewPanel = new JPanel();
        // Add components to the view panel
        viewPanel.add(new JLabel("View Panel Component"));

        // Create a container panel to hold the table and the view panel
        //JPanel containerPanel = new JPanel(new BorderLayout());
        containerPanel.add(scrollPane, BorderLayout.CENTER);
        containerPanel.add(viewPanel, BorderLayout.SOUTH);
       //  mainPanel.add(containerPanel);
       slotPanel.setVisible(false);
       statusPanel.setVisible(false);
       availablePanel.setVisible(false);
       vaccinePanel.setVisible(false);
       vaccineUpdatePanel.setVisible(false);
        slotUpdatePanel.setVisible(false);
        panel.setVisible(false);
```

```
        containerPanel.setVisible(true);
        containerPaneIn.setVisible(false);

    }
    catch(Exception q){
      q.printStackTrace();
    }

}});

viewVaccineItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
      try {
      try{
      Class.forName("oracle.jdbc.driver.OracleDriver");
    }
    catch(Exception q){
      q.printStackTrace();
    }
    connection=DriverManager.getConnection(
      "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");
      Statement statement = connection.createStatement();

            // Execute the query
            String query = "SELECT vaccine_id, name FROM vaccine";
            ResultSet resultSet = statement.executeQuery(query);

            // Create a 2D array to store the data
            Object[][] data = new Object[100][2]; // Assuming there are 100 rows in the Slot table

            // Populate the data array with the query results
            int row = 0;
            while (resultSet.next()) {
```

```
            int vid = resultSet.getInt("vaccine_id");
            String name = resultSet.getString("name");



            data[row][0] = vid;
            data[row][1] = name;


            row++;
        }

        // Define the column names
        String[] columnNames = { "Vid", "Name"};

        // Create a JTable with the data and column names
        JTable table = new JTable(data, columnNames);

        // Create a JScrollPane to add scroll functionality to the table
        JScrollPane scrollPane = new JScrollPane(table);

        // Create a view panel to hold additional components
        JPanel viewPanel = new JPanel();
        // Add components to the view panel
        viewPanel.add(new JLabel("View Panel Component"));

        // Create a container panel to hold the table and the view panel
        //JPanel containerPanel = new JPanel(new BorderLayout());
        containerPaneIn.add(scrollPane, BorderLayout.CENTER);
        containerPaneIn.add(viewPanel, BorderLayout.SOUTH);
       //  mainPanel.add(containerPanel);
       slotPanel.setVisible(false);
       statusPanel.setVisible(false);
       availablePanel.setVisible(false);
       vaccinePanel.setVisible(false);
       vaccineUpdatePanel.setVisible(false);
        slotUpdatePanel.setVisible(false);
        panel.setVisible(false);
```

```
            containerPanel.setVisible(false);



        containerPaneln.setVisible(true);


    }
    catch(Exception q){
      q.printStackTrace();
    }

}});

setSize(400, 300);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLayout(null);
setVisible(true);
}

private void insertAvailableDetails(String VaccineIdava,String SlotIdAva,String CapacityAva){
  try {
      // Establish database connection
      try{
      Class.forName("oracle.jdbc.driver.OracleDriver");
    }
    catch(Exception q){
      q.printStackTrace();
    }
      conn=DriverManager.getConnection(
        "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");

      // Prepare SQL statement
      String query = "INSERT INTO available (vaccine_id, slot_id, capacity) VALUES (?, ?, ?)";
      PreparedStatement preparedStatement = conn.prepareStatement(query);

      // Set parameter values
      preparedStatement.setString(1, VaccineIdava);
```

```
    preparedStatement.setString(2, SlotIdAva);
    preparedStatement.setString(3, CapacityAva);

    // Execute the query
    preparedStatement.executeUpdate();

    // Close the statement and connection
    preparedStatement.close();
    conn.close();

    // Display success message
    JOptionPane.showMessageDialog(null, "Available details inserted successfully!");

  } catch (SQLException ex) {
    ex.printStackTrace();
    JOptionPane.showMessageDialog(null, "Error: Failed to insert user details.");
  }

}
private void insertStatusDetails(String userIdSt, String vaccineIdSt, String countSt){
  try {
    // Establish database connection
    try{
    Class.forName("oracle.jdbc.driver.OracleDriver");
   }
   catch(Exception q){
    q.printStackTrace();
   }
    conn=DriverManager.getConnection(
      "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");

    // Prepare SQL statement
    String query = "INSERT INTO userdosestatus (user_id, vaccine_id, count) VALUES (?, ?, ?)";
    PreparedStatement preparedStatement = conn.prepareStatement(query);

    // Set parameter values
```

```
    preparedStatement.setString(1, userIdSt);
    preparedStatement.setString(2, vaccineIdSt);
    preparedStatement.setString(3, countSt);

    // Execute the query
    preparedStatement.executeUpdate();

    // Close the statement and connection
    preparedStatement.close();
    conn.close();

    // Display success message
    JOptionPane.showMessageDialog(null, "User details inserted successfully!");

  } catch (SQLException ex) {
    ex.printStackTrace();
    JOptionPane.showMessageDialog(null, "Error: Failed to insert user details.");
  }
}

private void insertSlotDetails(String SlotId, String slotLocation, String slotName)
{
  try{
  try{
  Class.forName("oracle.jdbc.driver.OracleDriver");
}
catch(Exception q){
  q.printStackTrace();
}
  conn=DriverManager.getConnection(
   "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");
   String query = "INSERT INTO slot (slot_id, location, name) VALUES (?, ?, ?)";
   PreparedStatement preparedStatement = conn.prepareStatement(query);

   // Set parameter values
   preparedStatement.setString(1, SlotId);
```

```
    preparedStatement.setString(2, slotLocation);
    preparedStatement.setString(3, slotName);

    // Execute the query
    preparedStatement.executeUpdate();

    // Close the statement and connection
    preparedStatement.close();
    conn.close();

    // Display success message
    JOptionPane.showMessageDialog(null, "Vaccine details inserted successfully!");
  }
    catch (SQLException ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(null, "Error: Failed to insert vaccine details.");
    }
}

private void insertVaccineDetails(String vaccineId, String vaccineName, String doseCount) {
    try {
        // Establish database connection
        try{
        Class.forName("oracle.jdbc.driver.OracleDriver");
      }
      catch(Exception q){
        q.printStackTrace();
      }
        conn=DriverManager.getConnection(
          "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");

        // Prepare SQL statement
        String query = "INSERT INTO vaccine (vaccine_id, name, dose_count) VALUES (?, ?, ?)";
        PreparedStatement preparedStatement = conn.prepareStatement(query);

        // Set parameter values
```

```
        preparedStatement.setString(1, vaccineId);
        preparedStatement.setString(2, vaccineName);
        preparedStatement.setString(3, doseCount);

        // Execute the query
        preparedStatement.executeUpdate();

        // Close the statement and connection
        preparedStatement.close();
        conn.close();

        // Display success message
        JOptionPane.showMessageDialog(null, "Vaccine details inserted successfully!");

    } catch (SQLException ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(null, "Error: Failed to insert vaccine details.");
    }
}
private void updateVaccine(String vaccineId, String vaccineName, int doseCount) {
    try {
     try{
     Class.forName("oracle.jdbc.driver.OracleDriver");
    }
    catch(Exception q){
     q.printStackTrace();
    }
    conn=DriverManager.getConnection(
     "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");

        // Prepare SQL statement
        String query = "UPDATE vaccine SET name = ?, dose_count = ? WHERE vaccine_id = ?";
        PreparedStatement preparedStatement = conn.prepareStatement(query);

        // Set parameter values
        preparedStatement.setString(1, vaccineName);
```

```
        preparedStatement.setInt(2, doseCount);
        preparedStatement.setString(3, vaccineId);

        // Execute the query
        int rowsAffected = preparedStatement.executeUpdate();

        // Close the statement and connection
        preparedStatement.close();
        conn.close();

        if (rowsAffected > 0) {
            JOptionPane.showMessageDialog(null, "Vaccine updated successfully!");
        } else {
            JOptionPane.showMessageDialog(null, "Vaccine with the given ID not found.");
        }

    } catch (SQLException ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(null, "Error: Failed to update vaccine.");
    }
}
private void updateSlot(String slotId,String slotLocation,String slotName) {
  try {
    try{
    Class.forName("oracle.jdbc.driver.OracleDriver");
  }
  catch(Exception q){
    q.printStackTrace();
  }
  conn=DriverManager.getConnection(
    "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");

      // Prepare SQL statement
      String query = "UPDATE slot SET location = ?, name = ? WHERE slot_id = ?";
      PreparedStatement preparedStatement = conn.prepareStatement(query);
```

```
    // Set parameter values
    preparedStatement.setString(1,slotLocation );
    preparedStatement.setString(2, slotName);
    preparedStatement.setString(3, slotId);

    // Execute the query
    int rowsAffected = preparedStatement.executeUpdate();

    // Close the statement and connection
    preparedStatement.close();
    conn.close();

    if (rowsAffected > 0) {
        JOptionPane.showMessageDialog(null, "Vaccine updated successfully!");
    } else {
        JOptionPane.showMessageDialog(null, "Vaccine with the given ID not found.");
    }

  } catch (SQLException ex) {
    ex.printStackTrace();
    JOptionPane.showMessageDialog(null, "Error: Failed to update vaccine.");
  }
}


private void deleteSlot(String slotId) {
   try {
      // Establish database connection
      try{
      Class.forName("oracle.jdbc.driver.OracleDriver");
    }
    catch(Exception q){
      q.printStackTrace();
    }
      conn=DriverManager.getConnection(
        "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");
```

```
        // Prepare SQL statement
        String query = "DELETE FROM slot WHERE slot_id = ?";
        PreparedStatement preparedStatement = conn.prepareStatement(query);

        // Set parameter value
        preparedStatement.setString(1, slotId);

        // Execute the query
        int rowsAffected = preparedStatement.executeUpdate();

        // Close the statement and connection
        preparedStatement.close();
        conn.close();

        if (rowsAffected > 0) {
            JOptionPane.showMessageDialog(null, "Slot deleted successfully!");
        } else {
            JOptionPane.showMessageDialog(null, "Slot with the given ID not found.");
        }

    } catch (SQLException ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(null, "Error: Failed to delete slot.");
    }
}
private void updateUserDoseStatus(String userId, String vaccineId, String dose) {
    // JDBC connection variables
    try {
    try{
    Class.forName("oracle.jdbc.driver.OracleDriver");
  }
  catch(Exception q){
    q.printStackTrace();
  }
  connection=DriverManager.getConnection(
```

```
"jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");
PreparedStatement statement = null;

    // Establish the database connection

    // Check if the user ID already exists
    String selectQuery = "SELECT * FROM userdosestatus WHERE user_id = ?";
    statement = connection.prepareStatement(selectQuery);
    statement.setString(1, userId);
    ResultSet resultSet = statement.executeQuery();

    if (resultSet.next()) {
        // User ID exists, update the columns
        String updateQuery = "UPDATE userdosestatus SET ";
        switch (dose) {
            case "First Dose":
                updateQuery += "firstdose = 1";
                break;
            case "Second Dose":
                updateQuery += "seconddose = 1";
                break;
            case "Booster Dose":
                updateQuery += "boosterdose = 1";
                break;
        }
        updateQuery += " WHERE user_id = ?";
        statement = connection.prepareStatement(updateQuery);
        statement.setString(1, userId);
        int rowsAffected = statement.executeUpdate();

        if (rowsAffected > 0) {
            System.out.println("User dose status updated successfully!");
        } else {
            System.out.println("Failed to update user dose status.");
        }
    } else {
```

```
            // User ID does not exist, insert a new row
            String insertQuery = "INSERT INTO userdosestatus (user_id, vaccine_id, ";
            switch (dose) {
                case "First Dose":
                    insertQuery += "firstdose) VALUES (?, ?, 1)";
                    break;
                case "Second Dose":
                    insertQuery += "seconddose) VALUES (?, ?, 1)";
                    break;
                case "Booster Dose":
                    insertQuery += "boosterdose) VALUES (?, ?, 1)";
                    break;
            }
            statement = connection.prepareStatement(insertQuery);
            statement.setString(1, userId);
            statement.setString(2, vaccineId);
            int rowsAffected = statement.executeUpdate();

            if (rowsAffected > 0) {
                System.out.println("User dose status inserted successfully!");
            } else {
                System.out.println("Failed to insert user dose status.");
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

private void countUsers() {
    Connection connection = null;
    Statement statement = null;
    ResultSet resultSet = null;

    try {
      try{
```

```
    Class.forName("oracle.jdbc.driver.OracleDriver");
  }
  catch(Exception q){
    q.printStackTrace();
  }
  connection=DriverManager.getConnection(
    "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");

      int vaccineId = Integer.parseInt(vaccineField.getText());

      statement = connection.createStatement();
      resultSet = statement.executeQuery("SELECT SUM(CASE WHEN firstdose = 1 THEN 1 " +
            "WHEN firstdose = 1 AND seconddose = 1 THEN 2 " +
            "WHEN firstdose = 1 AND seconddose = 1 AND boosterdose = 1 THEN 3 ELSE 0 END) " +
            "AS total_count FROM userdosestatus WHERE vaccine_id = " + vaccineId + " GROUP BY vaccine_id");

      if (resultSet.next()) {
         int count = resultSet.getInt("total_count");
         countLabel.setText("Number of users who have taken Vaccine " + vaccineId + ": " + count);
      } else {
         countLabel.setText("No data available for the specified vaccine ID");
      }
  } catch (Exception e) {
      e.printStackTrace();
  }

 }

 private void fetchDataFromDatabase() {
    // Database connection parameters
    try {
       Class.forName("oracle.jdbc.driver.OracleDriver");
    } catch (Exception q) {
       q.printStackTrace();
    }
    // Database query to fetch vaccine_id and name columns from the vaccine table
```

```
    String query = "SELECT vaccine_id, name FROM vaccine";

    try {
        // Establish the database connection
        Connection connection = DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521:xe", "rishi", "1234");

        // Create a statement object to execute the query
        Statement statement = connection.createStatement();

        // Execute the query and retrieve the result set
        ResultSet resultSet = statement.executeQuery(query);

        // Iterate over the result set and add rows to the table model
        while (resultSet.next()) {
            String vaccineId = resultSet.getString("vaccine_id");
            String name = resultSet.getString("name");

            model.addRow(new Object[]{vaccineId, name});
        }

        // Close the database connection
        connection.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
}


public static void main(String[] args) {
    // Create and show the login page
    SwingUtilities.invokeLater(new Runnable() {
        public void run() {

            AdminPage app = new AdminPage();
            app.setVisible(true);
```

```
        }
    });
}
}
```

Loginpage

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.sql.ResultSet;




public class LoginPagen {

    private JFrame frame;

    private JPanel mainPanel;

    private JPanel loginPanel;

    private JPanel loginPaneIn;

    private JPanel signupPanel;

    private Connection conn;




    public LoginPagen() {

        // Create the main frame

        frame = new JFrame("Login Page");

        frame.setSize(400, 300);

        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


        // Create the main panel

        mainPanel = new JPanel(null);


        // Create the login button

        JButton btnLogin = new JButton("Login");

        btnLogin.setBounds(50, 50, 100, 30);
```

```
// Create the signup button
JButton btnSignup = new JButton("Sign Up");
btnSignup.setBounds(250, 50, 100, 30);




// Add the buttons to the main panel
loginPaneIn = new JPanel(null);
loginPaneIn.add(btnLogin);
loginPaneIn.add(btnSignup);
loginPaneIn.setBounds(0, 0, 400, 300);

// Create the login panel
loginPanel = new JPanel(null);
loginPanel.setBounds(0, 0, 400, 300);

// Create components for the login panel
JLabel lblUserId = new JLabel("User ID:");
lblUserId.setBounds(50, 50, 100, 30);
JTextField txtUserId = new JTextField();
txtUserId.setBounds(160, 50, 150, 30);
JLabel lblPassword = new JLabel("Password:");
lblPassword.setBounds(50, 90, 100, 30);
JPasswordField txtPassword = new JPasswordField();
txtPassword.setBounds(160, 90, 150, 30);
JButton btnLoginPanelLogin = new JButton("Login");
btnLoginPanelLogin.setBounds(160, 130, 100, 30);

// Add components to the login panel
loginPanel.add(lblUserId);
loginPanel.add(txtUserId);
loginPanel.add(lblPassword);
loginPanel.add(txtPassword);
loginPanel.add(btnLoginPanelLogin);
// Add action listener for the login button in the login panel
btnLoginPanelLogin.addActionListener(new ActionListener() {
```

```
        @Override
        public void actionPerformed(ActionEvent e) {
            String userId = txtUserId.getText();
            int useri=Integer.parseInt(userId);
            String password = new String(txtPassword.getPassword());

            // Check if the provided password matches the corresponding user ID in the login table
            if (checkLoginCredentials(userId, password)) {
            // Password matches, perform login logic here

            JOptionPane.showMessageDialog(frame, "Login successful!");
            frame.setVisible(false);
            new UserPage(useri);
            } else {
            // Password doesn't match, show error message
            JOptionPane.showMessageDialog(frame, "Invalid credentials. Please try again.");
            }
        }
    });

    // Create the signup panel
    signupPanel = new JPanel(null);
    signupPanel.setBounds(0, 0, 400, 300);

    // Create components for the signup panel
    JLabel lblNewUserId = new JLabel("New User ID:");
    lblNewUserId.setBounds(50, 50, 100, 30);
    JTextField txtNewUserId = new JTextField();
    txtNewUserId.setBounds(160, 50, 150, 30);
    JLabel lblNewPassword = new JLabel("New Password:");
    lblNewPassword.setBounds(50, 90, 100, 30);
    JPasswordField txtNewPassword = new JPasswordField();
    txtNewPassword.setBounds(160, 90, 150, 30);
    JLabel lblName = new JLabel("Name:");
    lblName.setBounds(50, 130, 100, 30);
    JTextField txtName = new JTextField();
```

```
txtName.setBounds(160, 130, 150, 30);
JLabel lblAge = new JLabel("Age:");
lblAge.setBounds(50, 170, 100, 30);
JTextField txtAge = new JTextField();
txtAge.setBounds(160, 170, 150, 30);
JButton btnSignupPanelSignup = new JButton("Sign Up");
btnSignupPanelSignup.setBounds(160, 210, 100, 30);

// Add action listener for the signup button
btnSignupPanelSignup.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String newUserId = txtNewUserId.getText();
        String newPassword = new String(txtNewPassword.getPassword());
        String name = txtName.getText();
        int age = Integer.parseInt(txtAge.getText());

        // Insert new user credentials into login table
        insertUserCredentials(newUserId, newPassword);

        // Insert new user details into user_det table
        insertUserDetails(newUserId, name, age);
        JOptionPane.showMessageDialog(frame, "Signup sucessful");
    }
});

// Add components to the signup panel
signupPanel.add(lblNewUserId);
signupPanel.add(txtNewUserId);
signupPanel.add(lblNewPassword);
signupPanel.add(txtNewPassword);
signupPanel.add(lblName);
signupPanel.add(txtName);
signupPanel.add(lblAge);
signupPanel.add(txtAge);
signupPanel.add(btnSignupPanelSignup);
```

```
// Create the menu bar
JMenuBar menuBar = new JMenuBar();
frame.setJMenuBar(menuBar);

// Create the login menu
JMenu loginMenu = new JMenu("Login");
menuBar.add(loginMenu);

// Create the user login menu item
JMenuItem userLoginMenuItem = new JMenuItem("User Login");
loginMenu.add(userLoginMenuItem);

// Create the admin login menu item (placeholder)
JMenuItem adminLoginMenuItem = new JMenuItem("Admin Login");
loginMenu.add(adminLoginMenuItem);

// Create the signup menu
JMenu signupMenu = new JMenu("Signup");
menuBar.add(signupMenu);

// Create the user signup menu item
JMenuItem userSignupMenuItem = new JMenuItem("User Signup");
signupMenu.add(userSignupMenuItem);

// Create the home menu item
JMenuItem homeMenuItem = new JMenuItem("Home");
menuBar.add(homeMenuItem);

// Add action listeners for the menu items
userLoginMenuItem.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        showLoginPanel();
    }
});
```

```
        userSignupMenuItem.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                showSignupPanel();
            }
        });

        homeMenuItem.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                /*showMainPanel();*/
                new Updating();
                //new AdminPage();
            }
        });
        adminLoginMenuItem.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                /*showMainPanel();*/
                new AdminPage();
                //new AdminPage();
            }
        });

        // Show the main panel by default
        frame.add(mainPanel);

        // Display the frame
        frame.setVisible(true);
    }

    private void showMainPanel() {
        // Show the main panel

        mainPanel.removeAll();
```

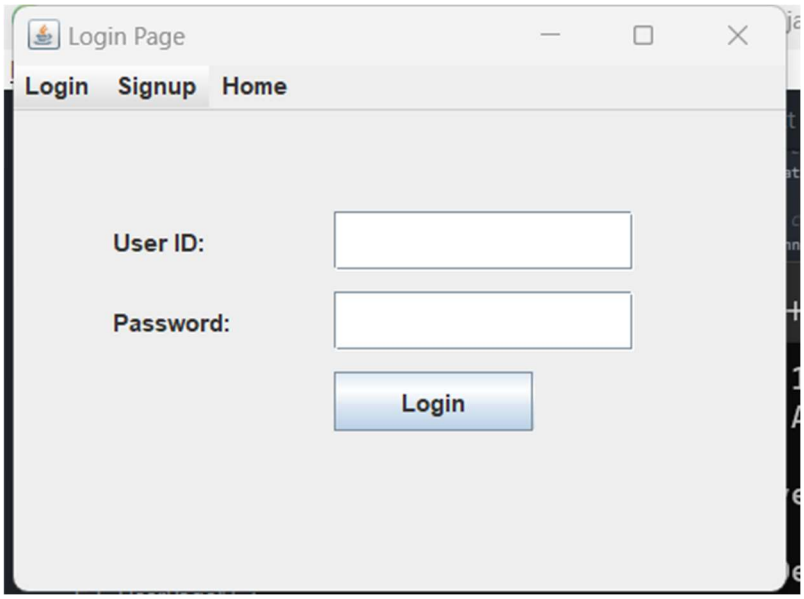
```
        mainPanel.add(loginPaneln);
        mainPanel.repaint();
        mainPanel.revalidate();
    }

    private void showLoginPanel() {
        // Show the login panel
        mainPanel.removeAll();
        mainPanel.add(loginPanel);
        mainPanel.repaint();
        mainPanel.revalidate();
    }

    private void showSignupPanel() {
        // Show the signup panel
        mainPanel.removeAll();
        mainPanel.add(signupPanel);
        mainPanel.repaint();
        mainPanel.revalidate();
    }

    private void insertUserCredentials(String userId, String password) {
        try {
            // Establish a database connection
            try{
            Class.forName("oracle.jdbc.driver.OracleDriver");
           }
           catch(Exception q){
            q.printStackTrace();
           }
           conn=DriverManager.getConnection(
            "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");

            // Prepare the SQL statement
            String sql = "INSERT INTO login (user_id, password) VALUES (?, ?)";
            PreparedStatement statement = conn.prepareStatement(sql);
```

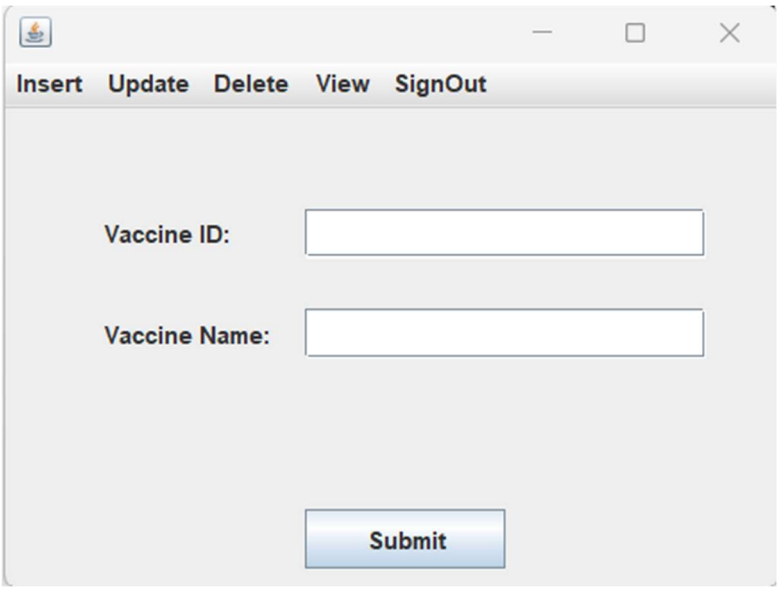
```
        statement.setString(1, userId);
        statement.setString(2, password);

        // Execute the statement
        statement.executeUpdate();

        // Close the database connection
        conn.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

private void insertUserDetails(String userId, String name, int age) {
    try {
        // Establish a database connection
        try{
        Class.forName("oracle.jdbc.driver.OracleDriver");
      }
      catch(Exception q){
        q.printStackTrace();
      }
      conn=DriverManager.getConnection(
        "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");

        // Prepare the SQL statement
        String sql = "INSERT INTO user_det (user_id, user_name, age) VALUES (?, ?, ?)";
        PreparedStatement statement = conn.prepareStatement(sql);
        statement.setString(1, userId);
        statement.setString(2, name);
        statement.setInt(3, age);

        // Execute the statement
        statement.executeUpdate();

        // Close the database connection
```

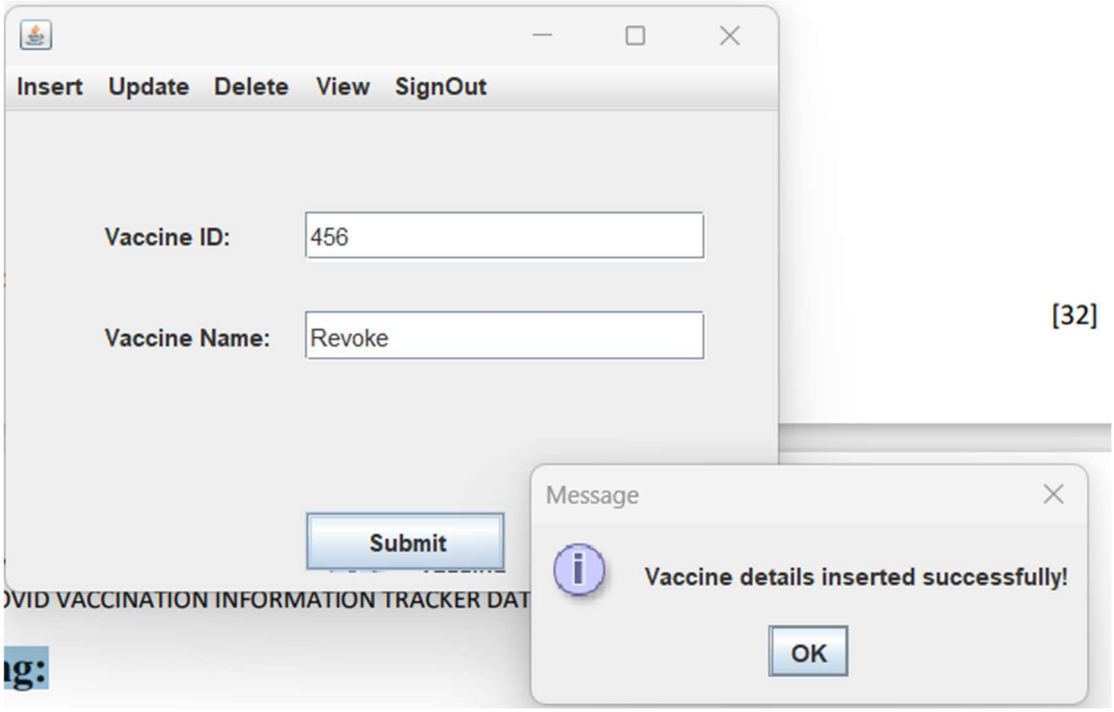
```
            conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    private boolean checkLoginCredentials(String userId, String password) {
      try {
    // Establish a database connection
    try{
    Class.forName("oracle.jdbc.driver.OracleDriver");
   }
   catch(Exception q){
    q.printStackTrace();
   }
   conn=DriverManager.getConnection(
    "jdbc:oracle:thin:@localhost:1521:xe","rishi","1234");

    // Prepare the SQL statement to retrieve the stored password for the given user ID
    String sql = "SELECT password FROM login WHERE user_id = ?";
    PreparedStatement statement = conn.prepareStatement(sql);
    statement.setString(1, userId);

    // Execute the statement and retrieve the result set
    ResultSet resultSet = statement.executeQuery();

    // Check if a row with the given user ID exists in the login table
    if (resultSet.next()) {
        // Retrieve the stored password for the given user ID
        String storedPassword = resultSet.getString("password");

        // Compare the stored password with the entered password
        if (password.equals(storedPassword)) {
            // Passwords match, return true
            conn.close();
            return true;
        }
```

```
    }

    // Close the result set, statement, and connection
    resultSet.close();
    statement.close();
    conn.close();
} catch (SQLException e) {
    e.printStackTrace();
}

// Passwords don't match or an error occurred, return false
return false;
}

    public static void main(String[] args) {
        // Create and show the login page
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new LoginPagen();
            }
        });
    }
}
```

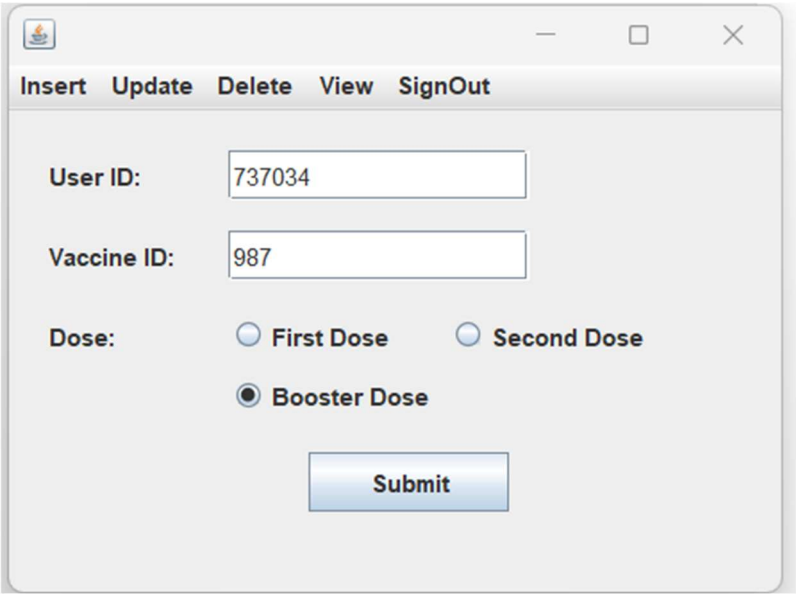
Testing: Java GUI Testing:

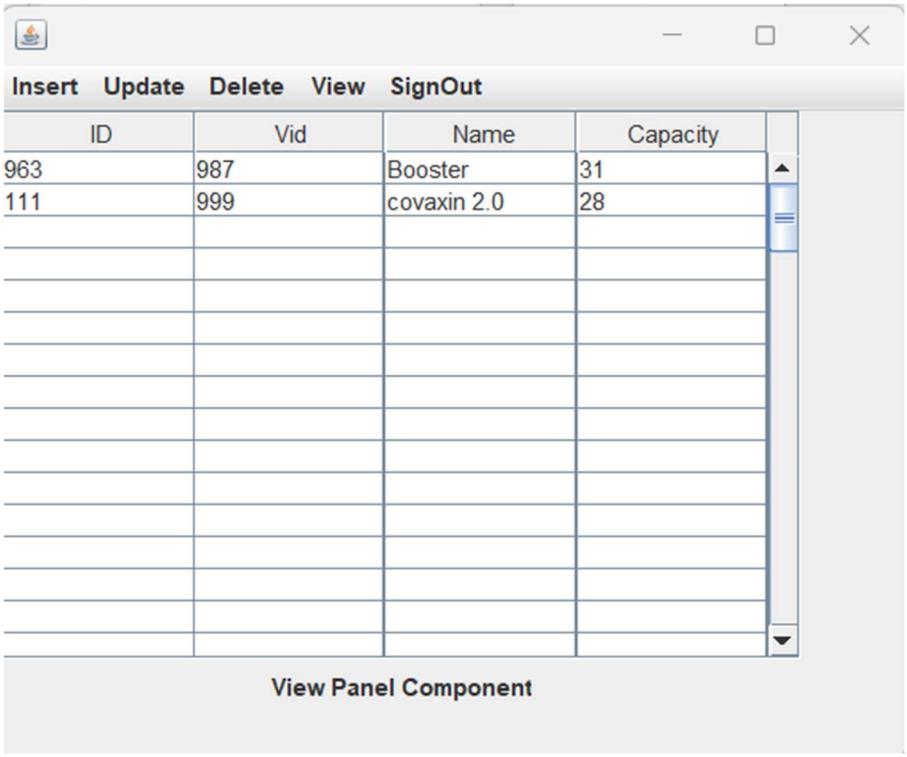


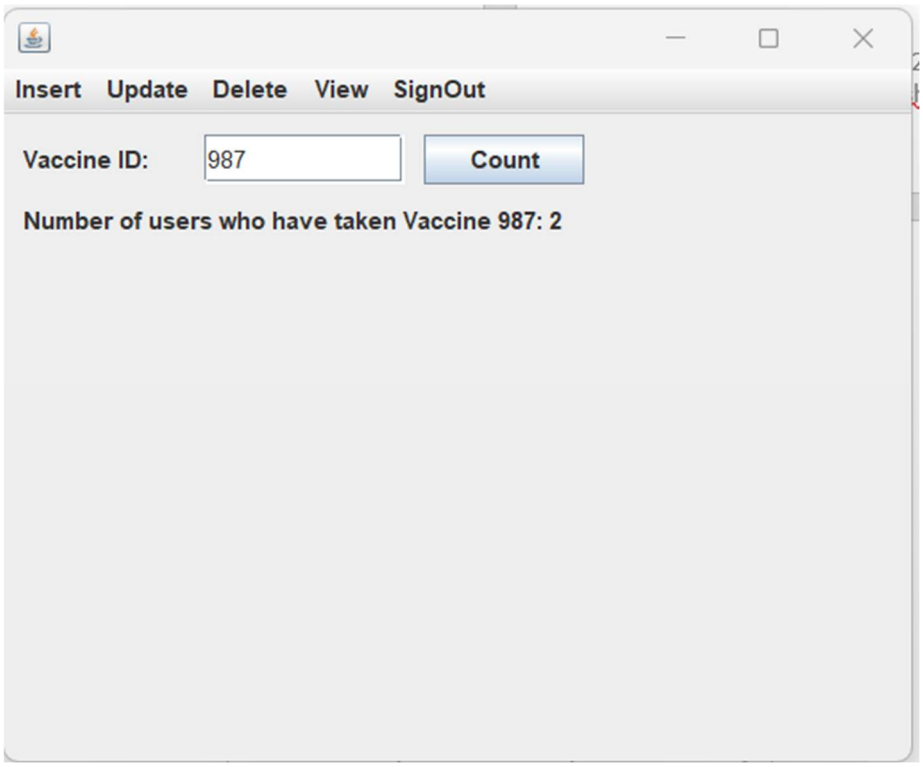



Insert Update Delete View SignOut
Vaccine ID:
456
Vaccine Name:
Revoke
Submit
Message
Vaccine details inserted successfully!
OK

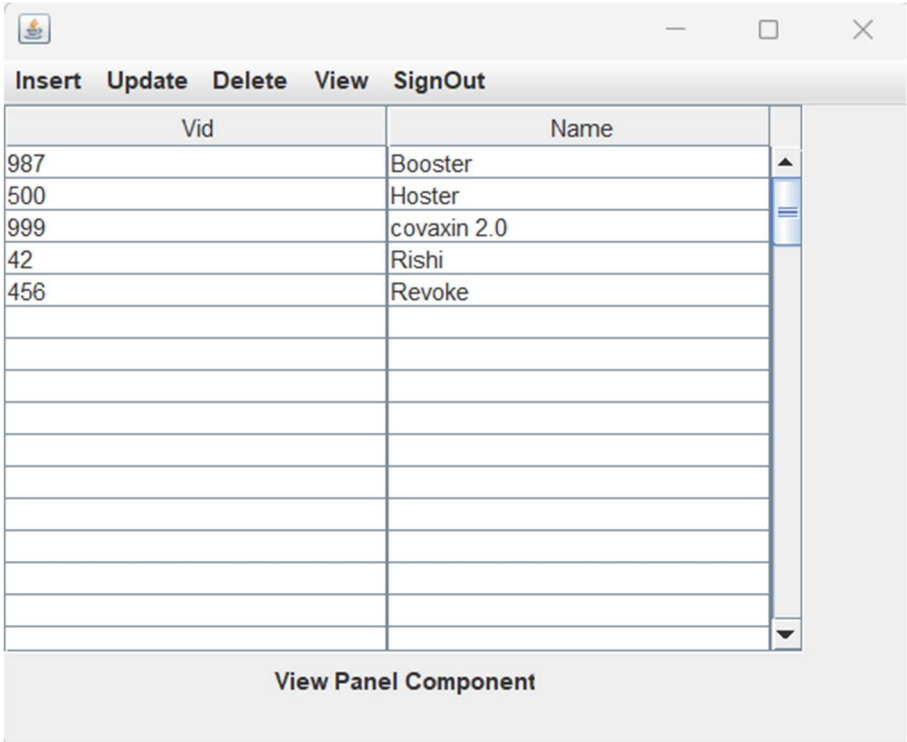
[32]

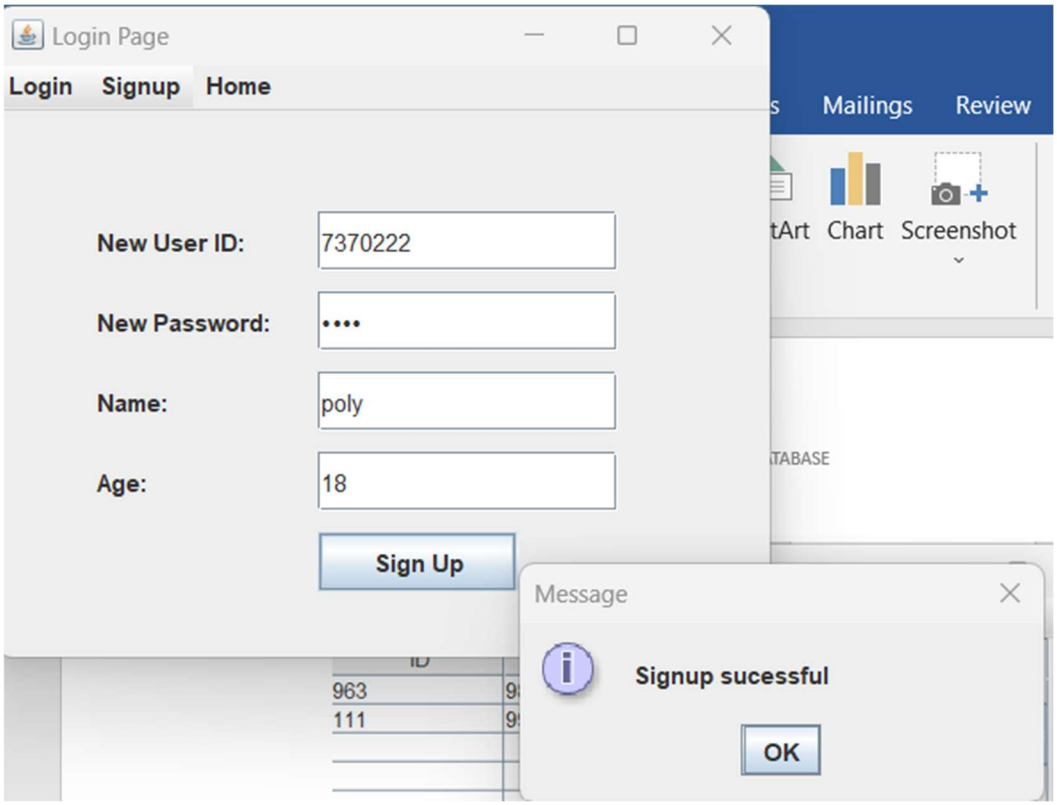

Insert Update Delete View SignOut
User ID:
737034
Vaccine ID:
987
Dose:
First Dose
Second Dose
Booster Dose
Submit

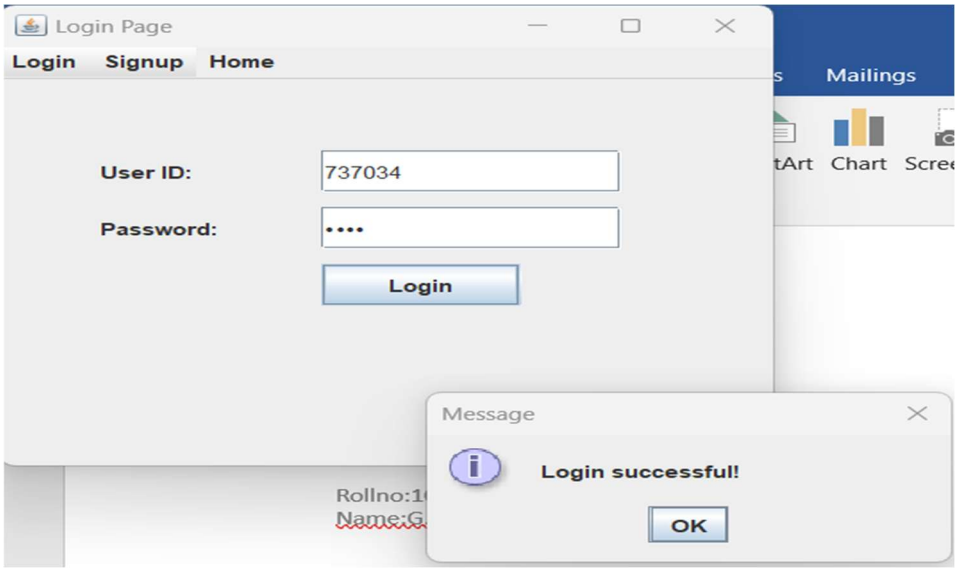


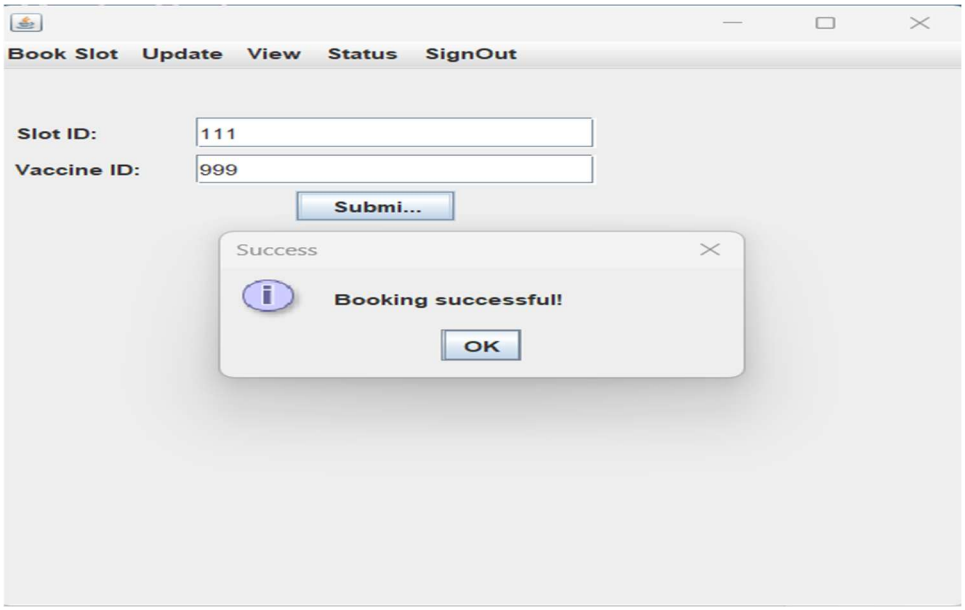


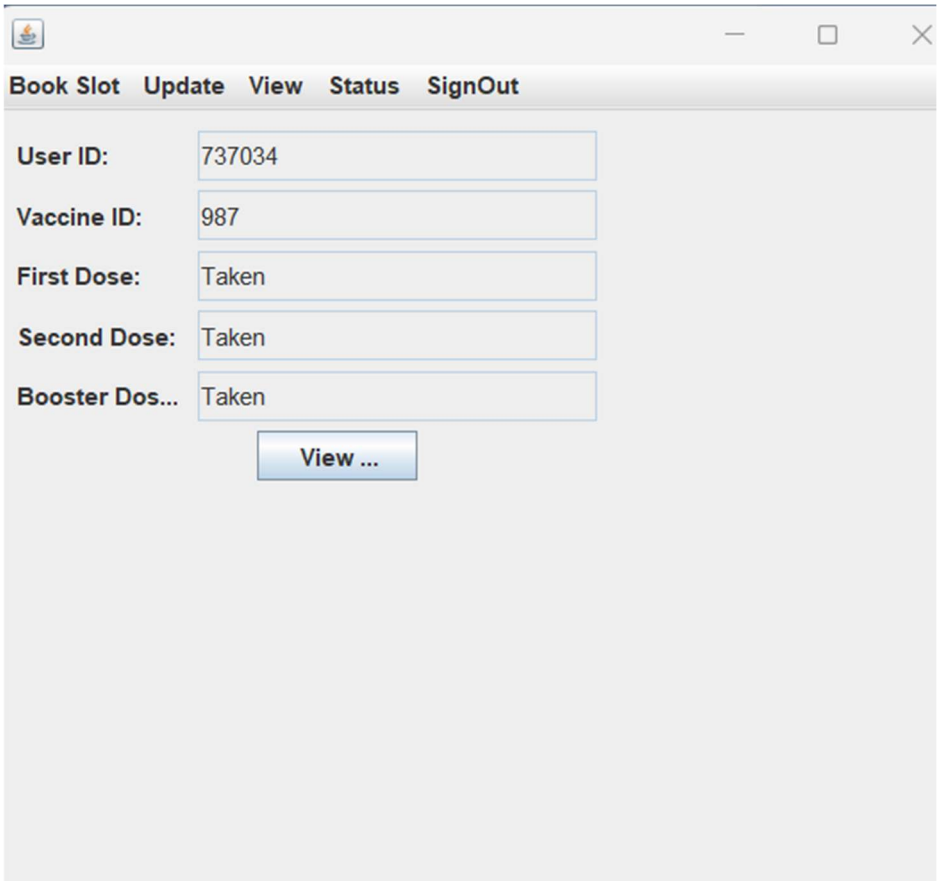
Rollno:102-21-737-042
Name:G.Rishi


Insert
Update
Delete
View
SignOut
Vid
Name
987
Booster
500
Hoster
999
covaxin 2.0
42
Rishi
456
Revoke
View Panel Component


Login Page
Login
Signup
Home
New User ID:
7370222
New Password:
Name:
poly
Age:
18
Sign Up
Mailings
Review
Chart
Screenshot
Message
Signup sucessful
OK


Login Page
Login
Signup
Home
User ID:
737034
Password:
Login
Mailings
Chart
Message
Login successful!
OK


Book Slot
Update
View
Status
SignOut
Slot ID:
111
Vaccine ID:
999
Submi...
Success
Booking successful!
OK


Book Slot
Update
View
Status
SignOut
User ID:
737034
Vaccine ID:
987
First Dose:
Taken
Second Dose:
Taken
Booster Dos...
Taken
View ...

GitHub links and folder structure

https://github.com/Rishi-1234567/CovidVaccination.git

Results: I successfully completed this PROJECT “Covid Vaccination Information Tracker Database”.

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 an android app by which I can control my project on my handand connect to it

Rollno:102-21-737-042
Name:G.Rishi