Personal Information Management System

A MINI-PROJECT BY:

Mokesh M	**230701192**
V.Lingeswaran	**230701163**

in partial fulfillment of the award of the degree

OF

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI

NOVEMBER 2024

BONAFIDE CERTIFICATE

Certified that this project report "**Personal Information Management System"** is a Bonafide work of "**V.Lingeswaran(230701163) & Mokesh M (230701192)" .**

Submitted for the Practical Examination held on ___________

Signature

Ms. Dharshini B S

Assistant Professor,

Computer Science and Engineering,

Rajalakshmi Engineering College (Autonomous),

Thandalam, Chennai-602 105

Internal Examiner **External Examiner**

Abstract of the Project

The Personal Information Management System is a Java-based standalone application designed to efficiently store, manage, and retrieve personal data records. Built using JavaFX for a modern and responsive user interface and MySQL for robust database management, the system ensures an intuitive and seamless experience for handling personal information.

The application provides a secure login module, a visually appealing dashboard, dynamic forms for adding and editing details, and an interactive table view for displaying records. Features like real-time search, sorting, and update functionalities enhance usability. The system incorporates user confirmations for critical actions such as deletions or updates, minimizing errors and ensuring data integrity.

The project utilizes JDBC (Java Database Connectivity) for integrating Java with MySQL, facilitating efficient and secure database interactions. The use of JavaFX allows for the development of a polished and responsive UI, offering an improved user experience compared to traditional desktop interfaces.

Designed for scalability, simplicity, and security, this system serves as a reliable tool for managing personal data, making it ideal for small organizations, educational institutions, or individual users who require effective information management in a standalone environment.

TABLE OF CONTENTS

INTRODUCTION

1.1 INTRODUCTION

The Personal Information Management System is a comprehensive application designed to securely store, manage, and retrieve personal data. This system serves as a centralized repository for users' personal, educational, and employment information, along with emergency contact details. By integrating JavaFX for the frontend and MySQL for the backend, the system provides a user-friendly interface and robust data management capabilities. The primary objective is to streamline the organization of information and ensure data integrity while maintaining ease of access.

1.2 IMPLEMENTATION

The implementation of this system is carried out using JavaFX for the graphical user interface, ensuring an interactive and visually appealing experience. MySQL serves as the backend database to store and manage structured data efficiently. The system uses the JDBC (Java Database Connectivity) API to facilitate seamless communication between the Java application and the MySQL database.

Key steps in implementation:

1. **Frontend Development**: Creating an intuitive UI using JavaFX, incorporating forms and tables for data input and display.
2. **Backend Integration**: Designing and implementing the MySQL database schema to store user information, ensuring normalization for optimal performance.
3. **Database Connectivity**: Establishing a connection between the frontend and database using JDBC for data retrieval and storage.
4. **Features Implementation**: Adding functionalities like CRUD (Create, Read, Update, Delete) operations for different data entities such as User, Address, Education, Employment Information, and Emergency Contact.

1.3 SCOPE OF THE PROJECT

The Personal Information Management System is designed to cater to both individual and organizational needs for storing and managing data. The scope includes:

- **Data Centralization**: Organizing personal, educational, and professional details in one place for easy access.
- **User Management**: Efficiently managing user data with CRUD functionalities.
- **Security and Integrity**: Ensuring the secure storage of sensitive information using database constraints and validations.
- **Extensibility**: The system is designed to be modular, allowing for the addition of new features like reporting or integration with external systems.
- **Versatility**: Suitable for applications in HR systems, educational institutions, or personal use for managing detailed records.

1.4 SYSTEM FEATURES

The system incorporates the following key features:

1. **User Registration and Management**:
   - Add, update, delete, or view user details.
   - Manage personal information like name, email, and phone number.
2. **Address Management**:
   - Maintain address details, including street, city, state, and postal code.
3. **Education Details**:
   - Record and update educational qualifications, including degree, institution name, graduation year, and grade.
4. **Employment Information**:
   - Track job details such as job title, company name, start and end dates.
5. **Emergency Contact**:
   - Store details of emergency contacts, including name, relationship, and contact information.
6. **Secure Storage**:
   - Use database constraints to enforce data validation and ensure integrity.
7. **User-Friendly Interface**:
   - Intuitive forms and navigation using JavaFX to simplify interactions.
8. **Scalable Database**:
   - A relational database design allows for scalability and efficient data retrieval.

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS:

GRAPHICS	: RTX-3050
PROCESSOR	: Intel i5 (13th GEN)
MEMORY SIZE	: 16 GB

2.2 SOFTWARE SPECIFICATIONS:

PROGRAMMING LANGUAGE	: Java, MySQL
FRONT-END	: Java
BACK-END	: MySQL
OPERATING SYSTEM	: Windows 11

SOURCE CODE

user.java

```
package User;
import java.sql.*;

public class user {
    private Connection con;
    public user (Connection con){
        this.con=con;
    }
    public String showUser(String name) {
        String query = "SELECT * FROM user WHERE first_name = ?";
        String result;

        try (PreparedStatement ps = con.prepareStatement(query)) {
            ps.setString(1, name);
            ResultSet rs = ps.executeQuery();

            if (rs.next()) {
                result = rs.getInt("user_id") + "    " +
                         "First_name : "+rs.getString("first_name") + "    " +
                         "Last_name : "+rs.getString("last_name") + "        " +
                         "Email : "+rs.getString("email") + "     " +
                         "Phone_number : "+rs.getString("phone_number") + "    " +
                         "Address : "+rs.getString("address");
            } else {
                result = "No user found with the name " + name;
            }
        } catch (SQLException e) {
            result = "An error occurred: " + e.getMessage();
        }
        return result;
    }


    public void insertUser(String firstname, String lastname,String email,String phoneno,String address){
        String query = "insert into user(first_name, last_name, email, phone_number, address) values (?, ?, ?, ?, ?)";
        try (PreparedStatement ps = con.prepareStatement(query)) {
            ps.setString(1, firstname);
            ps.setString(2, lastname);
            ps.setString(3, email);
            ps.setString(4, phoneno);
            ps.setString(5, address);
            ps.executeUpdate();
        }catch(SQLException e){
            System.out.println(e);
        }
    }
    public void updateUser(int userid,String firstname, String lastname,String email,String phoneno,String address){
        String query = "update user set first_name=?,last_name=?,email=?,phone_number=?,address=? where user_id=?";
        try (PreparedStatement ps = con.prepareStatement(query)) {
            ps.setString(1, firstname);
            ps.setString(2, lastname);
            ps.setString(3, email);
            ps.setString(4, phoneno);
            ps.setString(5, address);
            ps.setInt(6, userid);
            ps.executeUpdate();
        }catch(SQLException e){
            System.err.println("a");
        }
    }
    public void deleteUser(int id){
        String query = "delete from user where user_id=?";
        try(PreparedStatement ps =  con.prepareStatement(query)){
            ps.setInt(1, id);
            ps.executeUpdate();

        }catch(SQLException e){
            System.out.println(e);
        }
    }
}
```

employmentInformation.java

```
package User;
import java.sql.*;

public class employmentInformation {
    private Connection con;

    public employmentInformation(Connection con) {
        this.con = con;
    }

    public String showEmploymentInfo(int id) {
        String query = "SELECT * FROM employment_information WHERE user_id=?";
        String result;

        try (PreparedStatement sc = con.prepareStatement(query)) {
            sc.setInt(1, id);
            ResultSet rs = sc.executeQuery();

            if (rs.next()) {
                result =
                        "Job_title : "+rs.getString("job_title") + "        " +
                        "Company_name : "+rs.getString("company_name") + "        " +
                        "Start_date : "+rs.getDate("start_date") + "        " +
                        "End_date : "+rs.getDate("end_date");
            } else {
                result = "No employment records found for user with ID " + id;
            }
        } catch (SQLException e) {
            result = "An error occurred: " + e.getMessage();
        }

        return result;
    }



    public void insertEmploymentInfo(int userId, String jobTitle, String companyName, Date startDate, Date endDate) {
        String query = "insert into employment_information(user_id, job_title, company_name, start_date, end_date) values
(?, ?, ?, ?, ?)";
```

```
        try (PreparedStatement ps = con.prepareStatement(query)) {
            ps.setInt(1, userId);
            ps.setString(2, jobTitle);
            ps.setString(3, companyName);
            ps.setDate(4, startDate);
            ps.setDate(5, endDate);
            ps.executeUpdate();
        }catch(SQLException e){
            System.out.println(e);
        }
    }

    public void updateEmploymentInfo(int userId, String jobTitle, String companyName, Date startDate, Date endDate){
        String query = "update employment_information set user_id=?, job_title=?, company_name=?, start_date=?,
end_date=? where user_id=?";

        try (PreparedStatement ps = con.prepareStatement(query)) {
            ps.setInt(1, userId);
            ps.setString(2, jobTitle);
            ps.setString(3, companyName);
            ps.setDate(4, startDate);
            ps.setDate(5, endDate);
            ps.setInt(6, userId);
            ps.executeUpdate();
        }catch(SQLException e){
            System.out.println(" ");
        }
    }
    public void deleteUser(int id){
        String query = "delete from employment_information where user_id=?";
        try(PreparedStatement ps =  con.prepareStatement(query)){
            ps.setInt(1, id);
            ps.executeUpdate();

        }catch(SQLException e){
            System.out.println("employment_information");
        }
    }
}
```

emergencyContact.java

```
package User;
import java.sql.*;

public class emergencyContact {
    private Connection con;

    public emergencyContact(Connection con) {
        this.con = con;
    }

    public String showEmergencyContacts(int id) {
        String query = "SELECT * FROM emergency_contact WHERE user_id=?";
        String result;

        try (PreparedStatement sc = con.prepareStatement(query)) {
            sc.setInt(1, id);
            ResultSet rs = sc.executeQuery();

            if (rs.next()) {
                result =
                        "Contact_name : "+ rs.getString("contact_name") + "      " +
                         "Relationship : "+rs.getString("relationship") + "      " +
                         "Contact_phone : "+rs.getString("contact_phone") + "      " +
                         "Contact_email : "+rs.getString("contact_email");
            } else {
                result = "No emergency contact records found for user with ID " + id;
            }
        } catch (SQLException e) {
            result = "An error occurred: " + e.getMessage();
        }

        return result;
    }



    public void insertEmergencyContact(int userId, String contactName, String relationship, String contactPhone, String
contactEmail)  {
        String query = "insert into emergency_contact(user_id, contact_name, relationship, contact_phone, contact_email)
values (?, ?, ?, ?, ?)";

        try (PreparedStatement ps = con.prepareStatement(query)) {
            ps.setInt(1, userId);
            ps.setString(2, contactName);
            ps.setString(3, relationship);
            ps.setString(4, contactPhone);
            ps.setString(5, contactEmail);
            ps.executeUpdate();
        }catch(SQLException e){
            System.out.println(e);
        }
    }

    public void updateEmergencyContact(int userId, String contactName, String relationship, String contactPhone, String
contactEmail) {
        String query = "update emergency_contact set user_id=?, contact_name=?, relationship=?, contact_phone=?,
contact_email=? where contact_id=?";

        try (PreparedStatement ps = con.prepareStatement(query)) {
            ps.setInt(1, userId);
            ps.setString(2, contactName);
            ps.setString(3, relationship);
            ps.setString(4, contactPhone);
            ps.setString(5, contactEmail);
            ps.setInt(6, userId);
            ps.executeUpdate();
        }catch(SQLException e){
            System.out.println(" ");
        }
    }
    public void deleteUser(int id){
        String query = "delete from emergency_contact where user_id=?";
        try(PreparedStatement ps =  con.prepareStatement(query)){
            ps.setInt(1, id);
            ps.executeUpdate();
        }catch(SQLException e){
            System.out.println("emergency_contact");
        }
    }
}
```

education.java

```
package User;
import java.math.BigDecimal;
import java.sql.*;

public class education {
    private Connection con;

    public education(Connection con) {
        this.con = con;
    }

    public String showEducation(int id) {
        String query = "SELECT * FROM education WHERE user_id=?";
        String result;

        try (PreparedStatement sc = con.prepareStatement(query)) {
            sc.setInt(1, id);
            ResultSet rs = sc.executeQuery();

            if (rs.next()) {
                result =
                        "Degree : "+rs.getString("degree") + "      " +
                        "Institition : "+rs.getString("institution_name") + "       " +
                        "Graduation_year : "+rs.getInt("graduation_year") + "       " +
                        "Grade : "+rs.getBigDecimal("grade");
            } else {
                result = "No education records found for user with ID " + id;
            }
        } catch (SQLException e) {
            result = "An error occurred: " + e.getMessage();
        }

        return result;
    }



    public void insertEducation(int userId, String degree, String institutionName, int graduationYear, BigDecimal
grade)  {
        String query = "insert into education(user_id, degree, institution_name, graduation_year, grade) values (?, ?, ?, ?, ?)";

        try (PreparedStatement ps = con.prepareStatement(query)) {
            ps.setInt(1, userId);
            ps.setString(2, degree);
            ps.setString(3, institutionName);
            ps.setInt(4, graduationYear);
            ps.setBigDecimal(5, grade);
            ps.executeUpdate();
        }catch(SQLException e){
            System.out.println(e);
        }
    }

    public void updateEducation(int userId, String degree, String institutionName, int graduationYear, BigDecimal
grade){
        String query = "update education set user_id=?, degree=?, institution_name=?, graduation_year=?, grade=? where
user_id=?";

        try (PreparedStatement ps = con.prepareStatement(query)) {
            ps.setInt(1, userId);
            ps.setString(2, degree);
            ps.setString(3, institutionName);
            ps.setInt(4, graduationYear);
            ps.setBigDecimal(5, grade);
            ps.setInt(6, userId);
            ps.executeUpdate();
        }catch(SQLException e){
            System.out.println(" ");
        }
    }
    public void deleteUser(int id){
        String query = "delete from education where user_id=?";
        try(PreparedStatement ps =  con.prepareStatement(query)){
            ps.setInt(1, id);
            ps.executeUpdate();

        }catch(SQLException e){
            System.out.println("education");
        }
    }
}
```

address.java

```
package User;
import java.sql.*;

public class address {
    private Connection con;

    public address(Connection con) {
        this.con = con;
    }

    public String showAddress(int id) {
        String query = "SELECT * FROM address WHERE user_id=?";
        String result;

        try (PreparedStatement sc = con.prepareStatement(query)) {
            sc.setInt(1, id);
            try (ResultSet rs = sc.executeQuery()) {
                if (rs.next()) {
                    result = "Street : "+rs.getString("street") + "      " +
                            "City : "+rs.getString("city") + "      " +
                            "State : "+rs.getString("state") + "       " +
                            "Postal_Code : "+rs.getString("postal_code");
                } else {
                    result = "No address records found.";
                }
            }
        } catch (SQLException e) {
            result = "An error occurred: " + e.getMessage();
        }

        return result;
    }

    public void insertAddress(int userId, String street, String city, String state, String postalCode){
        String query = "insert into address(user_id, street, city, state, postal_code) values (?, ?, ?, ?, ?)";

        try (PreparedStatement ps = con.prepareStatement(query)) {
            ps.setInt(1, userId);
            ps.setString(2, street);
            ps.setString(3, city);
            ps.setString(4, state);
            ps.setString(5, postalCode);
            ps.executeUpdate();
        }catch(SQLException e){
            System.out.println(e);
        }
    }

    public void updateAddress(int userId, String street, String city, String state, String postalCode) {
        String query = "update address set user_id=?, street=?, city=?, state=?, postal_code=? where user_id=?";

        try (PreparedStatement ps = con.prepareStatement(query)) {
            ps.setInt(1, userId);
            ps.setString(2, street);
            ps.setString(3, city);
            ps.setString(4, state);
            ps.setString(5, postalCode);
            ps.setInt(6, userId);
            ps.executeUpdate();
        }catch(SQLException e){
            System.out.println(" ");
        }
    }
    public void deleteUser(int id){
        String query = "delete from address where user_id=?";
        try(PreparedStatement ps =  con.prepareStatement(query)){
            ps.setInt(1, id);
            ps.executeUpdate();

        }catch(SQLException e){
            System.out.println("address");
        }
    }
}
```

App.java

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;

import java.math.BigDecimal;
import java.sql.*;
import java.time.LocalDate;

import User.*;

public class HelloJavaFX extends Application {
    Connection con;
    user u;
    address ad;
    education ed;
    employmentInformation emp;
    emergencyContact ec;
    int id;
    String name;

    public void start(Stage primaryStage) {
        initializeDatabaseConnection();

        TextField nameField = new TextField();
        nameField.setPromptText("Enter your name");

        Button displayButton = new Button("Display User Info");
        Button createButton = new Button("Create a New User");

        VBox root = new VBox(10, nameField, displayButton, createButton);
        Scene scene = new Scene(root, 400, 300);

        displayButton.setOnAction(event -> displayUserInfo(nameField, primaryStage));
        createButton.setOnAction(event -> createNewUser(primaryStage));

        primaryStage.setTitle("User Information System");
        primaryStage.setScene(scene);
        primaryStage.show();
    }

    private void displayUserInfo(TextField nameField, Stage primaryStage) {
        name = nameField.getText().trim();
        name = nameField.getText().trim();
        if (name.isEmpty()) {
            showAlert("Invalid Input", "Please enter a valid name.");
            return;
        }

        String userInfo = u.showUser(name);
        if (userInfo == null || userInfo.contains("No user found")) {
            showAlert("Not Found", "No user found with the name " + name);
            return;
        }

        id = Integer.parseInt(userInfo.split(" ")[0]);
        TextArea resultArea = new TextArea();
        resultArea.setEditable(false);
        resultArea.setWrapText(true);

        resultArea.setText("User Info:\n" + u.showUser(name) + "\n\n");
        resultArea.appendText("Address Info:\n" + ad.showAddress(id) + "\n\n");
        resultArea.appendText("Education Info:\n" + ed.showEducation(id) + "\n\n");
        resultArea.appendText("Employment Info:\n" + emp.showEmploymentInfo(id) + "\n\n");
        resultArea.appendText("Emergency Contacts:\n" + ec.showEmergencyContacts(id) + "\n");
        Button updateButton = new Button("Update Information");
        updateButton.setOnAction(e -> showUpdateOptions(primaryStage, id));

        Button deleteButton = new Button("Delete User");
        deleteButton.setOnAction(e -> {
            deleteUser(id);
            showAlert("Success", "User deleted successfully.");
            showMainMenu(primaryStage);
        });

        Button backButton = new Button("Back");
        backButton.setOnAction(e -> showMainMenu(primaryStage));

        VBox outputLayout = new VBox(10, resultArea, updateButton, deleteButton, backButton);
        primaryStage.setScene(new Scene(outputLayout, 400, 400));
    }

    private void createNewUser(Stage primaryStage) {
        TextField firstNameField = new TextField();
        firstNameField.setPromptText("Enter First Name");

        TextField lastNameField = new TextField();
        lastNameField.setPromptText("Enter Last Name");

        TextField emailField = new TextField();
        emailField.setPromptText("Enter Email");

        TextField phoneField = new TextField();
        phoneField.setPromptText("Enter Phone Number");

        TextField addressField = new TextField();
        addressField.setPromptText("Enter Address");

        TextField streetField = new TextField();
        streetField.setPromptText("Enter Street");

        TextField cityField = new TextField();
        cityField.setPromptText("Enter City");

        TextField stateField = new TextField();
        stateField.setPromptText("Enter State");

        TextField postalCodeField = new TextField();
        postalCodeField.setPromptText("Enter Postal Code");

        TextField degreeField = new TextField();
        degreeField.setPromptText("Enter Degree");

        TextField institutionField = new TextField();
        institutionField.setPromptText("Enter Institution Name");

        TextField graduationYearField = new TextField();
        graduationYearField.setPromptText("Enter Graduation Year");

        TextField gradeField = new TextField();
        gradeField.setPromptText("Enter Grade");


        TextField jobTitleField = new TextField();
        jobTitleField.setPromptText("Enter Job Title");
```

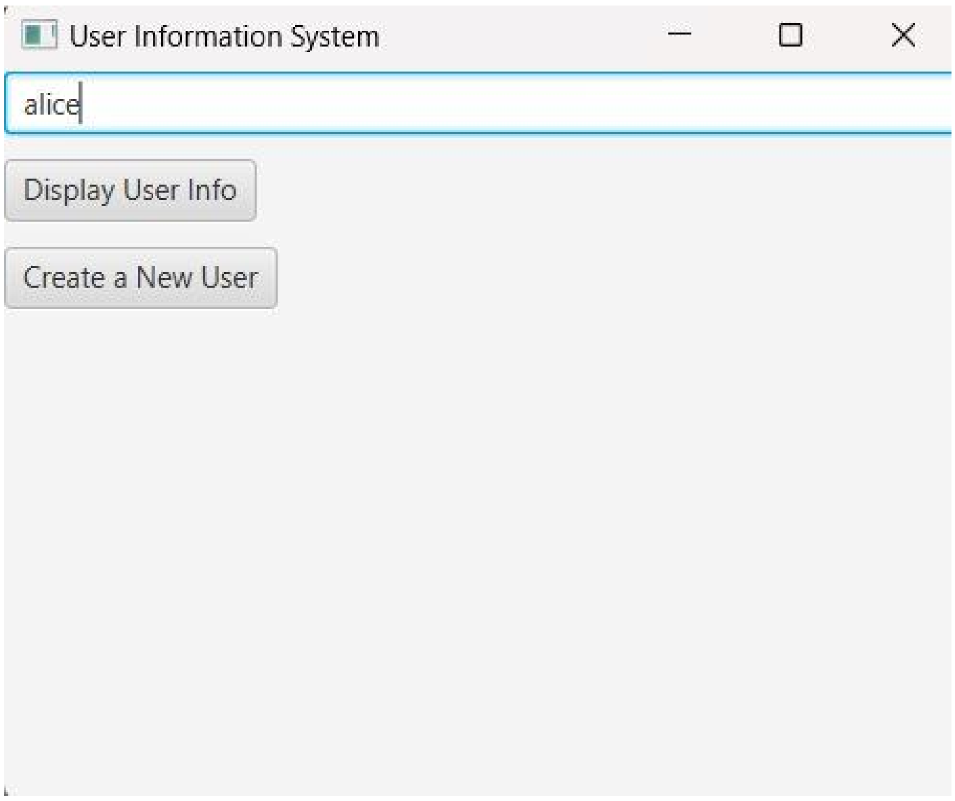
```
TextField companyField = new TextField();
companyField.setPromptText("Enter Company Name");

DatePicker startDatePicker = new DatePicker();
startDatePicker.setPromptText("Start Date");

DatePicker endDatePicker = new DatePicker();
endDatePicker.setPromptText("End Date");

TextField contactNameField = new TextField();
contactNameField.setPromptText("Enter Emergency Contact Name");

TextField contactPhoneField = new TextField();
contactPhoneField.setPromptText("Enter Emergency Contact Phone");

TextField contactEmailField = new TextField();
contactEmailField.setPromptText("Enter Emergency Contact Email");


Button saveButton = new Button("Save User");
saveButton.setOnAction(e -> {
  String firstName = firstNameField.getText().trim();
  String lastName = lastNameField.getText().trim();
  String email = emailField.getText().trim();
  String phone = phoneField.getText().trim();
  String address = addressField.getText().trim();

  String street = streetField.getText().trim();
  String city = cityField.getText().trim();
  String state = stateField.getText().trim();
  String postalCode = postalCodeField.getText().trim();

  String degree = degreeField.getText().trim();
  String institution = institutionField.getText().trim();
  String gradeStr = gradeField.getText().trim();
  int graduationYear = Integer.parseInt(graduationYearField.getText().trim());
  BigDecimal grade = new BigDecimal(gradeStr);

  String jobTitle = jobTitleField.getText().trim();
  String company = companyField.getText().trim();
  LocalDate startDate = startDatePicker.getValue();
  LocalDate endDate = endDatePicker.getValue();

  String contactName = contactNameField.getText().trim();
  String contactPhone = contactPhoneField.getText().trim();
  String contactEmail = contactEmailField.getText().trim();



  if (firstName.isEmpty() || lastName.isEmpty() || email.isEmpty() || phone.isEmpty() || address.isEmpty() ||
    street.isEmpty() || city.isEmpty() || state.isEmpty() || postalCode.isEmpty() ||
    jobTitle.isEmpty() || company.isEmpty() || startDate == null || endDate == null ||
    contactName.isEmpty() || contactPhone.isEmpty() || contactEmail.isEmpty()) {
    showAlert("Error", "All fields are required.");
  } else {

     u.insertUser(firstName, lastName, email, phone, address);
    String userInfo = u.showUser(firstName);
    int userId = Integer.parseInt(userInfo.split(" ")[0]);
```

```
            ad.insertAddress(userId, street, city, state, postalCode);
            ed.insertEducation(userId, degree, institution, graduationYear, grade);
            emp.insertEmploymentInfo(userId, jobTitle, company, Date.valueOf(startDate), Date.valueOf(endDate));
            ec.insertEmergencyContact(userId, contactName, "Emergency", contactPhone, contactEmail);

            showAlert("Success", "User and related details created successfully.");
            showMainMenu(primaryStage);
        }
    });

    Button backButton = new Button("Back");
    backButton.setOnAction(e -> showMainMenu(primaryStage));

    VBox createUserLayout = new VBox(10, firstNameField, lastNameField, emailField, phoneField, addressField,
            streetField, cityField, stateField, postalCodeField,  degreeField, institutionField, graduationYearField,
gradeField, jobTitleField, companyField,
            startDatePicker, endDatePicker, contactNameField, contactPhoneField, contactEmailField, saveButton,
backButton);

    ScrollPane scrollPane = new ScrollPane(createUserLayout);
    scrollPane.setFitToHeight(true);
    scrollPane.setFitToWidth(true);

    primaryStage.setScene(new Scene(scrollPane, 400, 500));
  }



  private void showUpdateOptions(Stage primaryStage, int id) {
    Button updateUserButton = new Button("Update User Details");
    updateUserButton.setOnAction(e -> updateUser(primaryStage, id));

    Button updateAddressButton = new Button("Update Address");
    updateAddressButton.setOnAction(e -> updateAddress(primaryStage, id));

    Button updateEducationButton = new Button("Update Education");
    updateEducationButton.setOnAction(e -> updateEducation(primaryStage, id));

    Button updateEmploymentButton = new Button("Update Employment Info");
    updateEmploymentButton.setOnAction(e -> updateEmployment(primaryStage, id));

    Button updateEmergencyButton = new Button("Update Emergency Contact");
    updateEmergencyButton.setOnAction(e -> updateEmergencyContact(primaryStage, id));

    Button backButton = new Button("Back");
    backButton.setOnAction(e -> showMainMenu(primaryStage));

    VBox updateOptionsLayout = new VBox(10,
      updateUserButton,
      updateAddressButton,
      updateEducationButton,
      updateEmploymentButton,
      updateEmergencyButton,
      backButton
    );

    primaryStage.setScene(new Scene(updateOptionsLayout, 400, 300));
  }
  private void showMainMenu(Stage primaryStage) {
    TextField nameField = new TextField();
    nameField.setPromptText("Enter your name");

    Button displayButton = new Button("Display User Info");
    Button createButton = new Button("Create a New User");

    displayButton.setOnAction(event -> displayUserInfo(nameField, primaryStage));
    createButton.setOnAction(event -> createNewUser(primaryStage));

    VBox mainMenuLayout = new VBox(10, nameField, displayButton, createButton);
    primaryStage.setScene(new Scene(mainMenuLayout, 400, 300));
}

private void updateUser(Stage primaryStage, int id) {

    TextField firstNameField = new TextField();
    firstNameField.setPromptText("First Name");

    TextField lastNameField = new TextField();
    lastNameField.setPromptText("Last Name");

    TextField emailField = new TextField();
    emailField.setPromptText("Email");

    TextField phoneField = new TextField();
    phoneField.setPromptText("Phone Number");

    TextField addressField = new TextField();
    addressField.setPromptText("Address");

    Button saveButton = new Button("Save Changes");
    saveButton.setOnAction(e -> {
        String firstName = firstNameField.getText().trim();
        String lastName = lastNameField.getText().trim();
        String email = emailField.getText().trim();
        String phone = phoneField.getText().trim();
        String address = addressField.getText().trim();

        if (firstName.isEmpty() || lastName.isEmpty() || email.isEmpty() || phone.isEmpty() || address.isEmpty()) {
            showAlert("Error", "All fields are required.");
        } else {
            u.updateUser(id, firstName, lastName, email, phone, address);
            showAlert("Success", "User information updated successfully.");
            showMainMenu(primaryStage);
        }
    });

    Button backButton = new Button("Back");
    backButton.setOnAction(e -> showUpdateOptions(primaryStage, id));

    VBox updateUserLayout = new VBox(10,
        new Label("Update User Information"),
        firstNameField,
        lastNameField,
        emailField,
        phoneField,
        addressField,
        saveButton,
        backButton
    );
    primaryStage.setScene(new Scene(updateUserLayout, 400, 400));
}


private void updateAddress(Stage primaryStage, int id) {
    TextField streetField = new TextField();
    streetField.setPromptText("Street");

    TextField cityField = new TextField();
    cityField.setPromptText("City");

    TextField stateField = new TextField();
    stateField.setPromptText("State");

    TextField postalCodeField = new TextField();
    postalCodeField.setPromptText("Postal Code");

    Button saveButton = new Button("Save Changes");
    saveButton.setOnAction(e -> {
        String street = streetField.getText().trim();
        String city = cityField.getText().trim();
        String state = stateField.getText().trim();
        String postalCode = postalCodeField.getText().trim();

        if (street.isEmpty() || city.isEmpty() || state.isEmpty() || postalCode.isEmpty()) {
            showAlert("Error", "All fields are required.");
        } else {
            ad.updateAddress(id, street, city, state, postalCode);
            showAlert("Success", "Address updated successfully.");
            showMainMenu(primaryStage);
        }
    });

    Button backButton = new Button("Back");
    backButton.setOnAction(e -> showUpdateOptions(primaryStage, id));

    VBox updateAddressLayout = new VBox(10,
        new Label("Update Address"),
        streetField,
        cityField,
        stateField,
        postalCodeField,
        saveButton,
        backButton
    );

    primaryStage.setScene(new Scene(updateAddressLayout, 400, 400));
}


private void updateEducation(Stage primaryStage, int id) {
    TextField degreeField = new TextField();
    degreeField.setPromptText("Degree");

    TextField institutionField = new TextField();
    institutionField.setPromptText("Institution");

    TextField graduationYearField = new TextField();
    graduationYearField.setPromptText("Graduation Year");
```

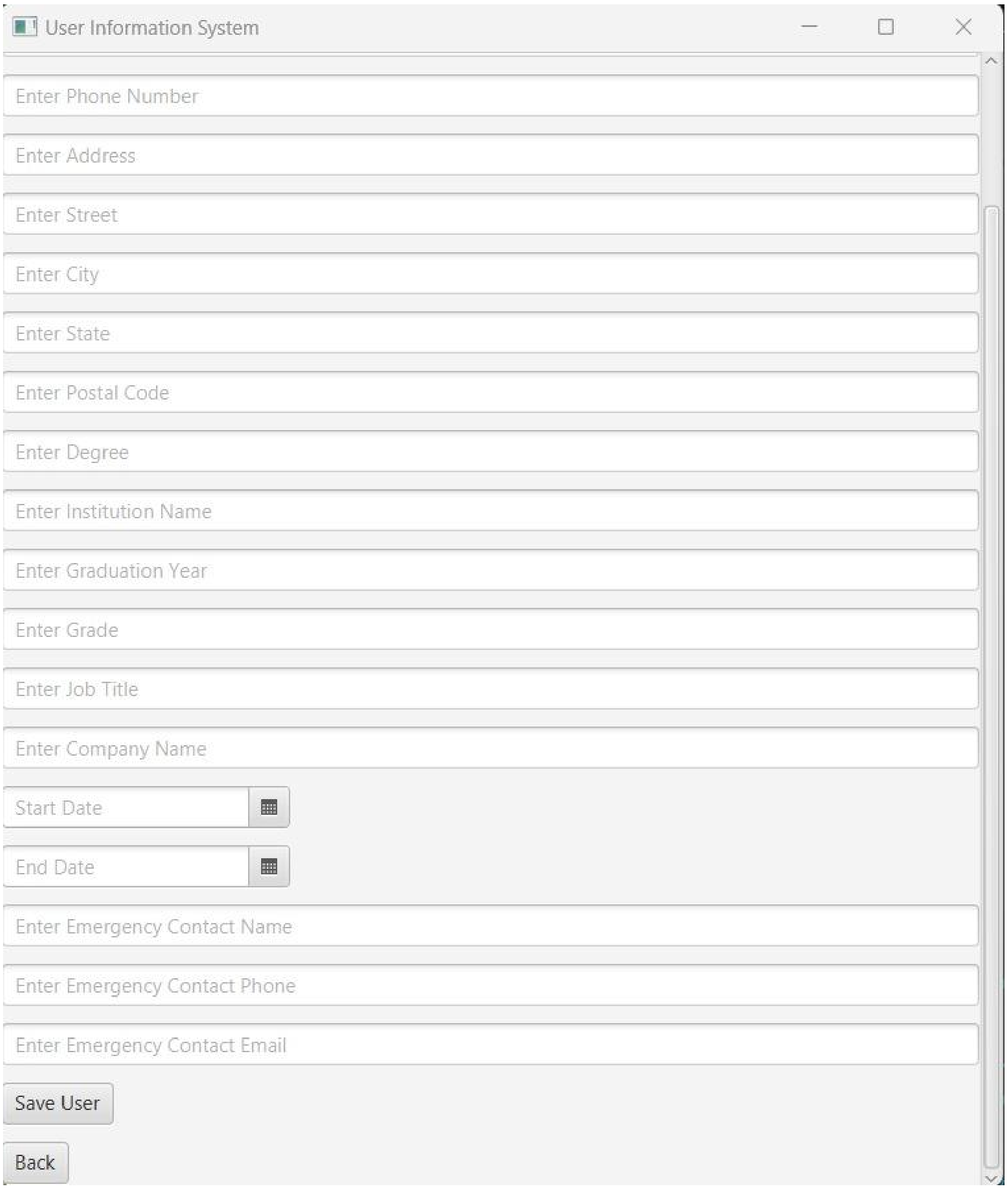
```
        TextField gradeField = new TextField();
        gradeField.setPromptText("Grade");

        Button saveButton = new Button("Save Changes");
        saveButton.setOnAction(e -> {
            String degree = degreeField.getText().trim();
            String institution = institutionField.getText().trim();
            String graduationYear = graduationYearField.getText().trim();
            String grade = gradeField.getText().trim();

            if (degree.isEmpty() || institution.isEmpty() || graduationYear.isEmpty() || grade.isEmpty()) {
                showAlert("Error", "All fields are required.");
            } else {
                ed.updateEducation(id, degree, institution, Integer.parseInt(graduationYear), new BigDecimal(grade));
                showAlert("Success", "Education information updated successfully.");
                showMainMenu(primaryStage);
            }
        });

        Button backButton = new Button("Back");
        backButton.setOnAction(e -> showUpdateOptions(primaryStage, id));

        VBox updateEducationLayout = new VBox(10,
            new Label("Update Education"),
            degreeField,
            institutionField,
            graduationYearField,
            gradeField,
            saveButton,
            backButton
        );

        primaryStage.setScene(new Scene(updateEducationLayout, 400, 400));
    }


    private void updateEmployment(Stage primaryStage, int id) {
        TextField jobTitleField = new TextField();
        jobTitleField.setPromptText("Job Title");

        TextField companyNameField = new TextField();
        companyNameField.setPromptText("Company Name");

        TextField startDateField = new TextField();
        startDateField.setPromptText("Start Date (YYYY-MM-DD)");

        TextField endDateField = new TextField();
        endDateField.setPromptText("End Date");

        Button saveButton = new Button("Save Changes");
        saveButton.setOnAction(e -> {
            String jobTitle = jobTitleField.getText().trim();
            String companyName = companyNameField.getText().trim();
            String startDate = startDateField.getText().trim();
            String endDate = endDateField.getText().trim();

            if (jobTitle.isEmpty() || companyName.isEmpty() || startDate.isEmpty() || endDate.isEmpty()) {
                showAlert("Error", "All fields are required.");
            } else {
                emp.updateEmploymentInfo(id, jobTitle, companyName, Date.valueOf(startDate), Date.valueOf(endDate));
                showAlert("Success", "Employment information updated successfully.");
                showMainMenu(primaryStage);
            }
        });

        Button backButton = new Button("Back");
        backButton.setOnAction(e -> showUpdateOptions(primaryStage, id));

        VBox updateEmploymentLayout = new VBox(10,
            new Label("Update Employment Info"),
            jobTitleField,
            companyNameField,
            startDateField,
            endDateField,
            saveButton,
            backButton
        );

        primaryStage.setScene(new Scene(updateEmploymentLayout, 400, 400));
    }


    private void updateEmergencyContact(Stage primaryStage, int id) {
    TextField contactNameField = new TextField();
    contactNameField.setPromptText("Contact Name");

    TextField relationshipField = new TextField();
    relationshipField.setPromptText("Relationship");

    TextField contactPhoneField = new TextField();
    contactPhoneField.setPromptText("Contact Phone");

    TextField contactEmailField = new TextField();
    contactEmailField.setPromptText("Contact Email");

    Button saveButton = new Button("Save Changes");
    saveButton.setOnAction(e -> {
        String contactName = contactNameField.getText().trim();
        String relationship = relationshipField.getText().trim();
        String contactPhone = contactPhoneField.getText().trim();
        String contactEmail = contactEmailField.getText().trim();

        if (contactName.isEmpty() || relationship.isEmpty() || contactPhone.isEmpty() || contactEmail.isEmpty()) {
            showAlert("Error", "All fields are required.");
        } else {
            ec.updateEmergencyContact(id, contactName, relationship, contactPhone, contactEmail);
            showAlert("Success", "Emergency contact updated successfully.");
            showMainMenu(primaryStage);
        }
    });

    Button backButton = new Button("Back");
    backButton.setOnAction(e -> showUpdateOptions(primaryStage, id));

    VBox updateEmergencyLayout = new VBox(10,
        new Label("Update Emergency Contact"),
        contactNameField,
        relationshipField,
        contactPhoneField,
        contactEmailField,
            saveButton,
            backButton
        );

        primaryStage.setScene(new Scene(updateEmergencyLayout, 400, 400));
    }

        private void deleteUser(int id) {
            ad.deleteUser(id);
            ed.deleteUser(id);
            emp.deleteUser(id);
            ec.deleteUser(id);
            u.deleteUser(id);
        }

        private void initializeDatabaseConnection() {
            try {
                con = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb", "root", "Lingeswaran@21");
                u = new user(con);
                ad = new address(con);
                ed = new education(con);
                emp = new employmentInformation(con);
                ec = new emergencyContact(con);
            } catch (SQLException e) {
                showAlert("Database Error", "Unable to connect to the database.");
                e.printStackTrace();
            }
        }

        private void showAlert(String title, String message) {
            Alert alert = new Alert(Alert.AlertType.INFORMATION);
            alert.setTitle(title);
            alert.setHeaderText(null);
            alert.setContentText(message);
            alert.showAndWait();
        }

        public static void main(String[] args) {
            launch(args);
        }
    }
```

SNAPSHOTS

4.1 LOGIN PAGE:

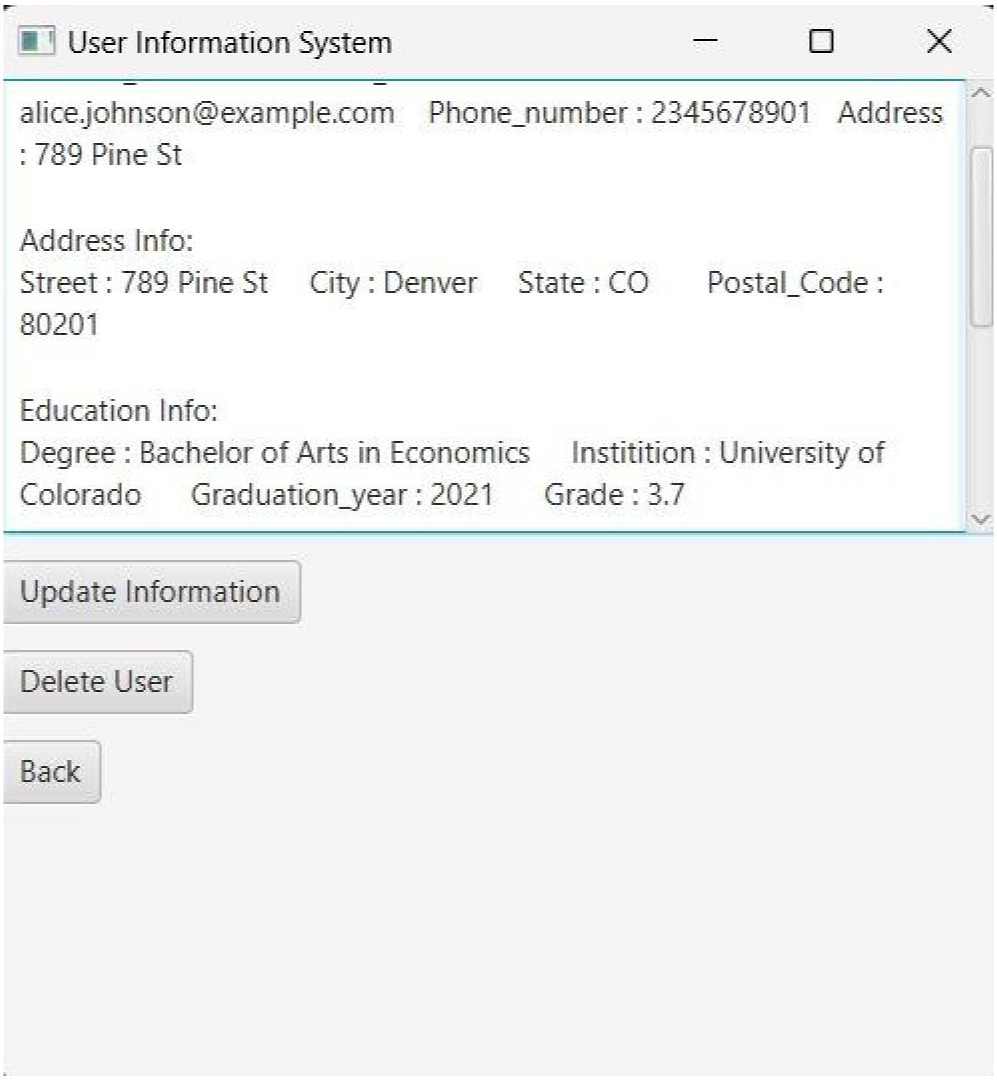


4.2 CREATING THE USER:


User Information System

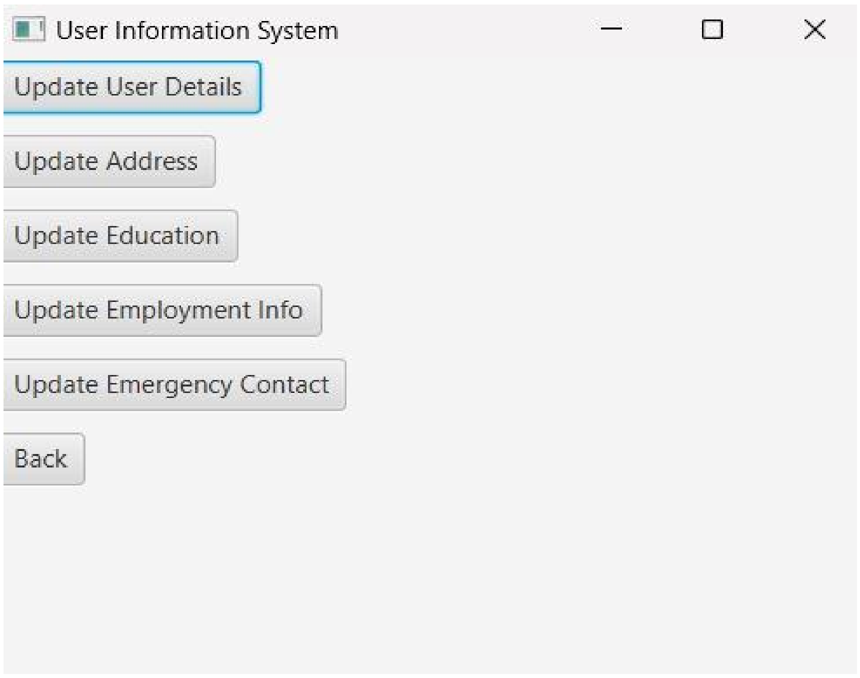
Enter Phone Number
Enter Address
Enter Street
Enter City
Enter State
Enter Postal Code
Enter Degree
Enter Institution Name
Enter Graduation Year
Enter Grade
Enter Job Title
Enter Company Name
Start Date
End Date
Enter Emergency Contact Name
Enter Emergency Contact Phone
Enter Emergency Contact Email
Save User
Back

4.3 DISPLAY USER INFORMATION

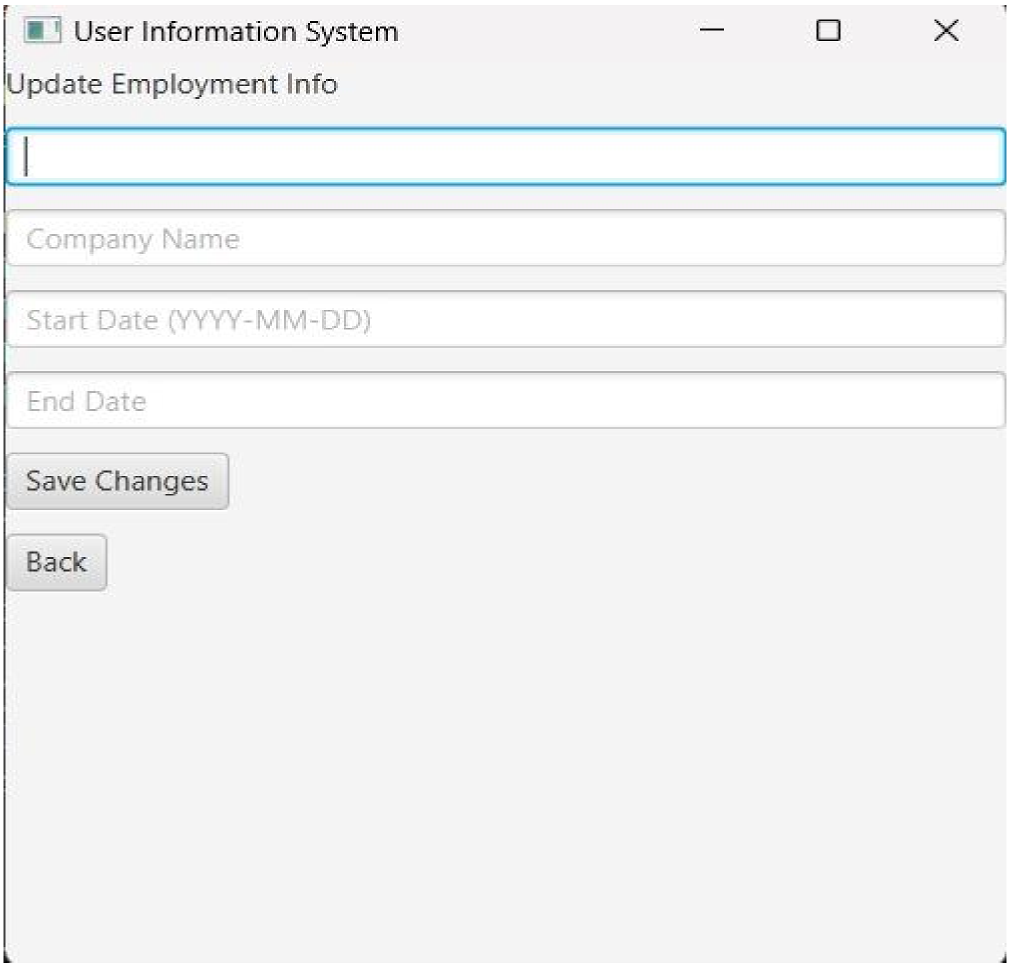


```
alice.johnson@example.com    Phone_number : 2345678901   Address
: 789 Pine St

Address Info:
Street : 789 Pine St    City : Denver    State : CO     Postal_Code :
80201

Education Info:
Degree : Bachelor of Arts in Economics    Institition : University of
Colorado     Graduation_year : 2021     Grade : 3.7
```

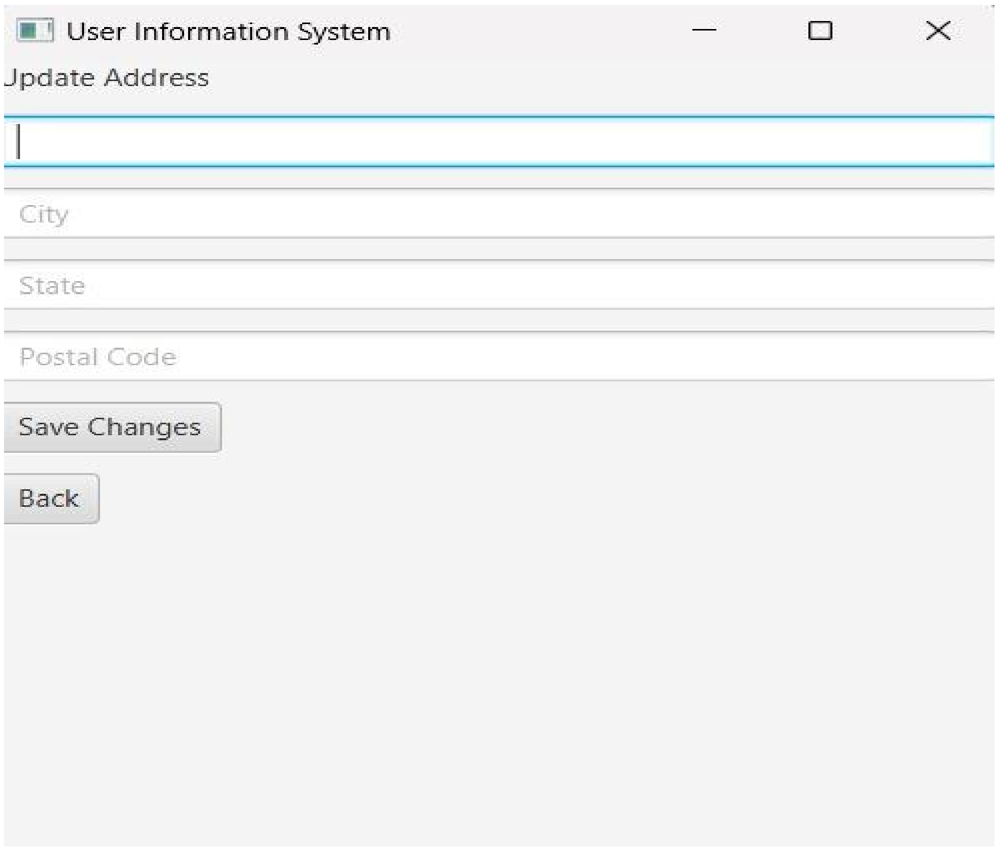
4.4 UPDATION



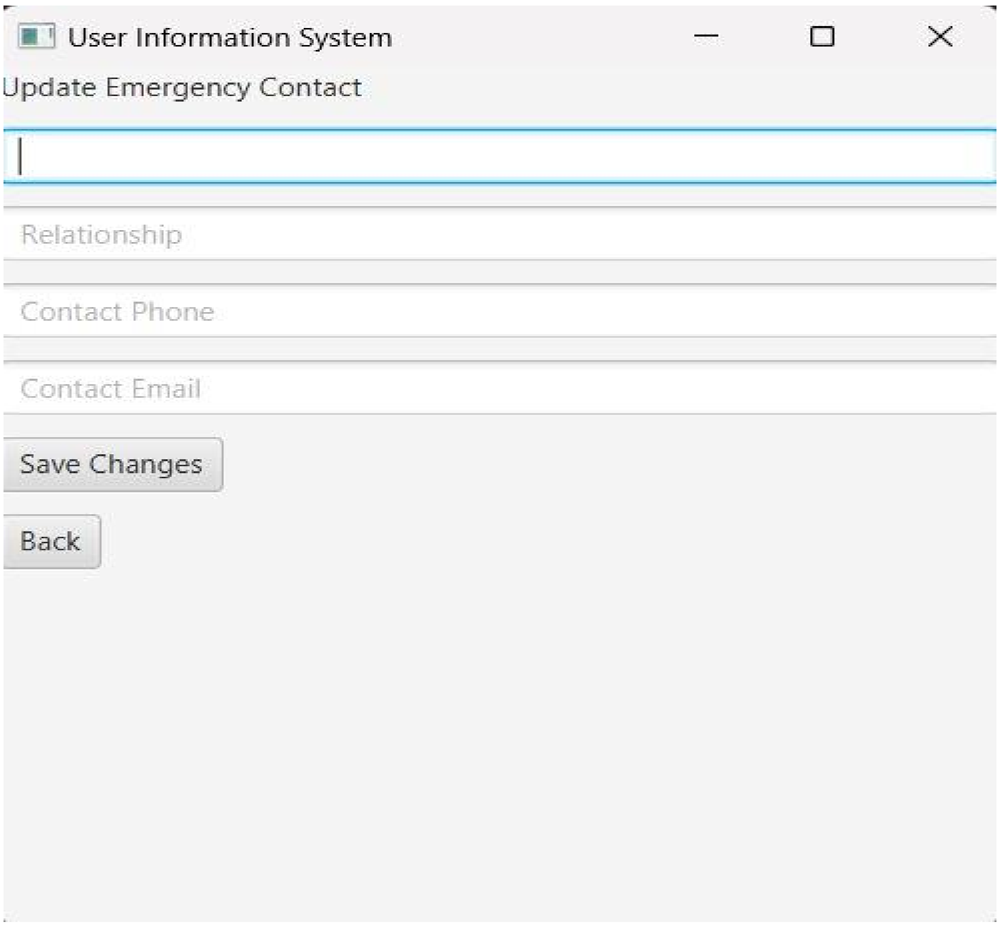
4.5 EMPLOYMENT INFORMATION



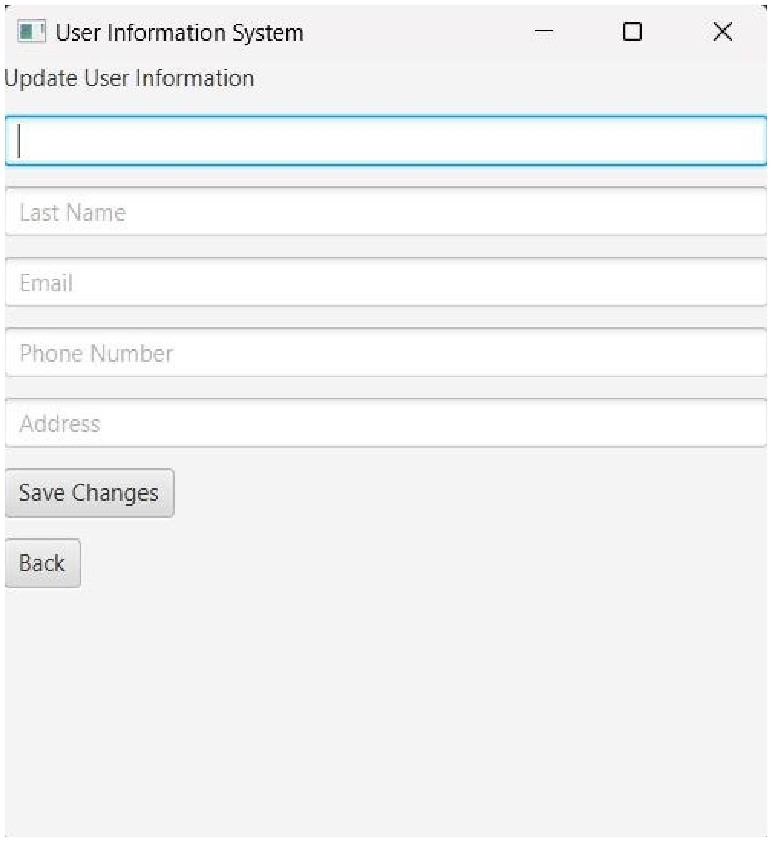
4.6 ADDRESS



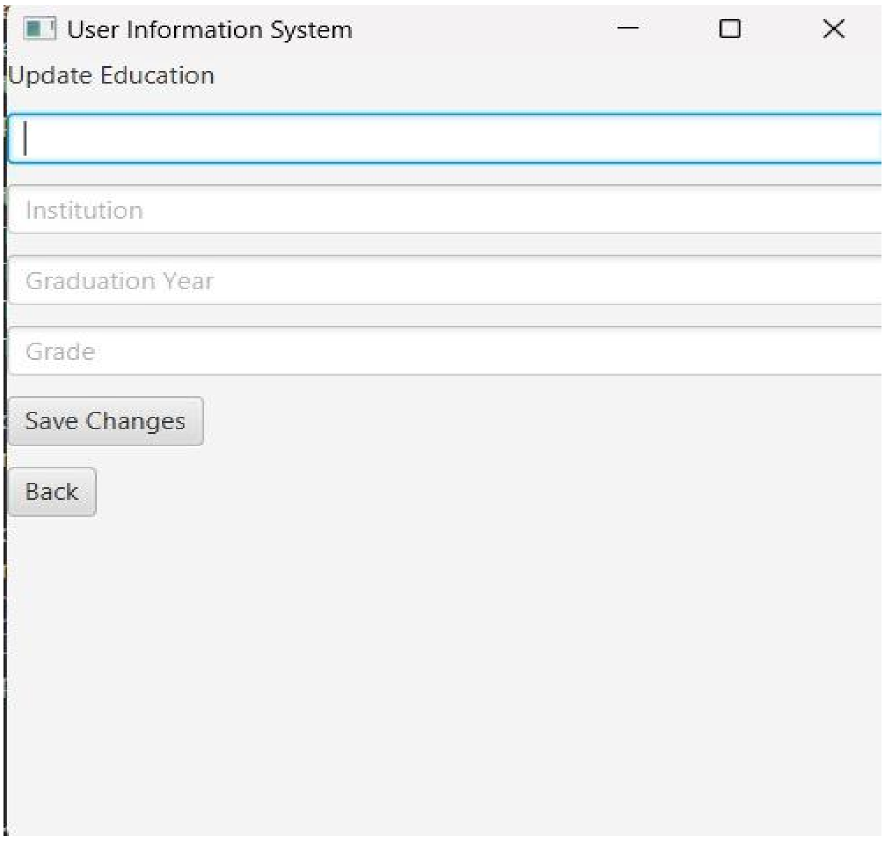
4.7 EMERGENCY CONTACT



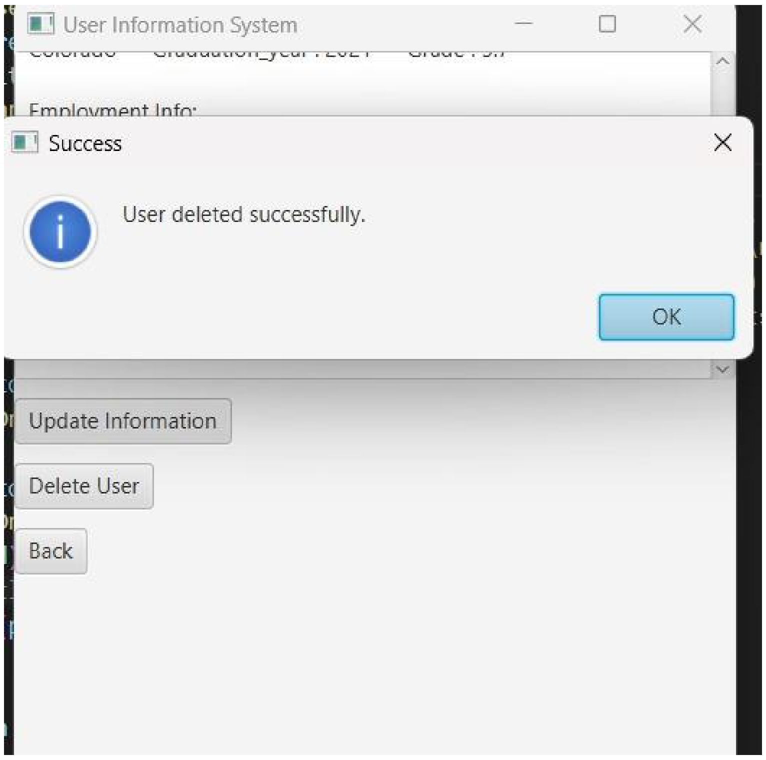
4.8 USER INFROMATION



4.9 EDUCATION



4.10 DELETING AN USER



User deleted successfully.

5.CONCLUSION:

The project successfully demonstrates the creation of a JavaFX-based desktop application integrated with MySQL. The system achieves its objectives of providing secure, efficient, and user-friendly personal information management. Potential future enhancements include advanced search filters, multi-user support, and role-based access control**.**

6.REFERENCES:

1. https://openjfx.io
2. https://dev.mysql.com/doc/ 3.
[https://docs.oracle.com/javase/tutorial/jdbc/](https://docs.oracle.com/javase/tu
torial/jdbc/)
4. https://www.w3schools.com/sql/
5. https://stackoverflow.com/