Notes Entry

A MINI-PROJECT BY:

Mithesh Tharun S 230701184

Mithul Aaditya A 230701185

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 **“Notes Entry”** is the
bonafide work of **“MitheshTharun S , Mithul Aaditya A”** who carried out the project work under my supervision.

Submitted for the practical examination held on ____________________

SIGNATURE	SIGNATURE
INTERNAL EXAMINER	**EXTERNAL EXAMINER**

ABSTRACT

The *Notes Entry* project is a JavaFX-based desktop application designed to help users manage personal notes through a secure, user-friendly interface. Unlike a traditional dissertation management system, *Notes Entry* focuses on providing users with easy access to their notes, facilitating seamless note-taking and editing experiences. The application allows users to register, log in securely, and view, add, and edit notes, with personalized profile management.

This project showcases the practical integration of JavaFX and MySQL, emphasizing database-driven applications that store and organize user data securely. By providing a responsive interface and efficient CRUD operations (Create, Read, Update, Delete), *Notes Entry* demonstrates fundamental software engineering principles and promotes efficient data management skills.

TABLE OF CONTENTS

1.INTRODUCTION

1. Introduction

1.1 **Overview**:

- This project, “Note Register Application,” is a desktop application created with **JavaFX and MySQL**. It allows users to manage their notes effectively with secure user authentication, CRUD operations on notes, and a personalized profile.

1.2 **Features**:

- **User Registration and Login**: Secure login with password protection.
- **Notes Management**: Users can add, view, edit, and delete their notes.
- **Profile Management**: Users can view and edit their profile details.
- **Database Integration**: Uses MySQL for storing user credentials and note data.

2. System Specifications

2.1 **Hardware Requirements**:

- Processor: Intel i3 or equivalent
- RAM: 2GB minimum
- Disk Space: 500MB free space

2.2 **Software Requirements**:

- **Programming Language**: Java
- **Frontend Framework**: JavaFX
- **Database**: MySQL
- **Operating System**: Windows 10 or later

3.Application Design

3.1 **Class Structure**:

- NoteRegister: Main application class, extending Application for JavaFX. It manages the scenes, login, and navigation.
- Database Operations: Classes handle MySQL database connections, queries, and result processing.

3.2 **User Interface Components**:

- **Login Page**: createLoginPage() method creates a login interface using TextField, PasswordField, and Button.
- **Notes Page**: showNotesPage() displays a list of notes with options to view, add, and edit.
- **Profile Page**: showUserProfilePage() allows users to view and update profile information.

3.3 **Database Schema**:

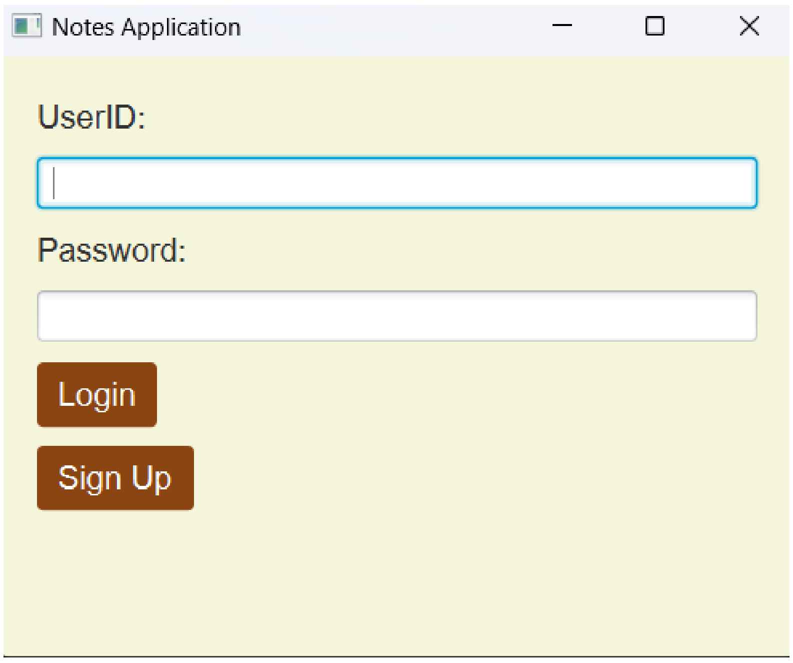
- **Table**: users
  - user: VARCHAR(255), primary key, stores unique username.
  - password: VARCHAR(255), stores hashed user password.
- **Table**: notes
  - id: INT, primary key, auto-incremented.
  - user: VARCHAR(255), foreign key referencing users.
  - title: VARCHAR(255), stores the title of the note.
  - content: TEXT, stores the note content.

SAMPLE CODE

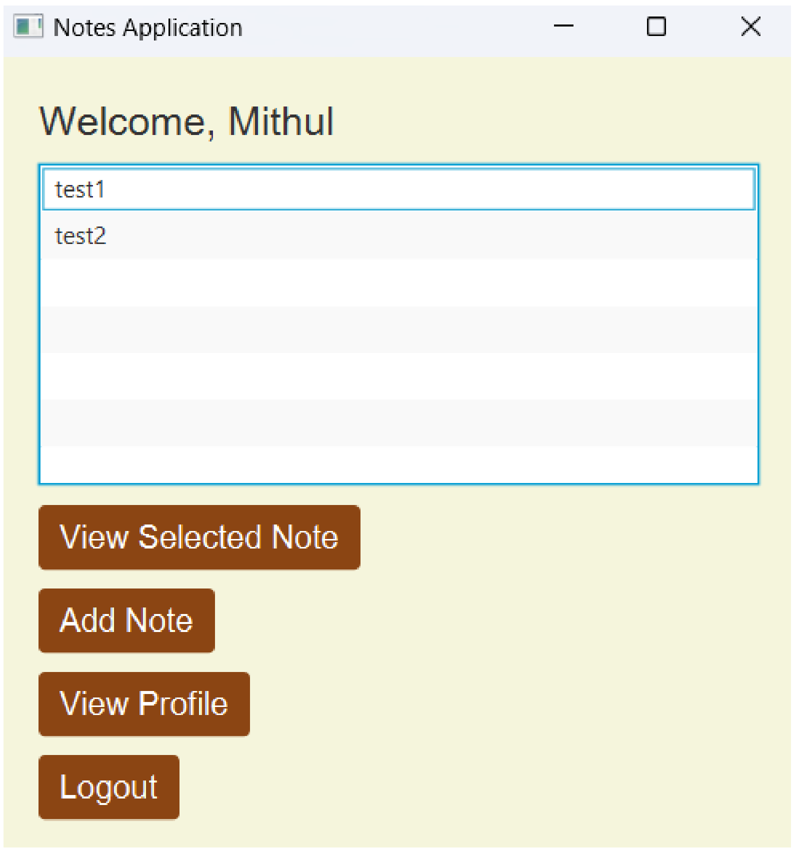
```
import javafx.application.Application;
import javafx.collections.FXCollections;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.*;
import javafx.scene.paint.Color;
import javafx.scene.text.Font;
import javafx.stage.Stage;
import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class NoteRegister extends Application {

    private TextField userIdField;
    private PasswordField passwordField;
    private Button loginButton, signupButton;
    private VBox loginLayout;
    private String currentUser;
    private Stage primaryStage;
    private int selectedNoteId;
    private ListView<String> noteListView;
    private Button viewNoteButton;
    private String selectedNoteTitle;
    private static final String DB_URL = "jdbc:mysql://localhost:3306/notes_register";
    private static final String DB_USERNAME = "root";
    private static final String DB_PASSWORD = "Amithul@1234";

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

    @Override
    public void start(Stage primaryStage) {
        this.primaryStage = primaryStage;
        primaryStage.setTitle("Notes Application");
        loginLayout = createLoginPage();
        Scene loginScene = new Scene(loginLayout, 400, 300);
        loginScene.setFill(Color.web("#f5f5dc"));
        primaryStage.setScene(loginScene);
        primaryStage.show();
    }


    private VBox createLoginPage() {
        VBox layout = new VBox(10);
        layout.setStyle("-fx-background-color: #f5f5dc; -fx-padding: 20;");

        Label userIdLabel = new Label("UserID:");
        userIdLabel.setFont(new Font("Arial", 16));
        userIdField = new TextField();
        userIdField.setFont(new Font("Arial", 14));

        Label passwordLabel = new Label("Password:");
        passwordLabel.setFont(new Font("Arial", 16));
        passwordField = new PasswordField();
        passwordField.setFont(new Font("Arial", 14));

        loginButton = new Button("Login");
        loginButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
        loginButton.setFont(new Font("Arial", 16));
        loginButton.setOnAction(e -> loginUser());

        signupButton = new Button("Sign Up");
        signupButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
        signupButton.setFont(new Font("Arial", 16));
        signupButton.setOnAction(e -> signupUser());

        layout.getChildren().addAll(userIdLabel, userIdField, passwordLabel, passwordField, loginButton, signupButton);
        return layout;
    }

    private void loginUser() {
        String userId = userIdField.getText();
        String password = passwordField.getText();

        try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME, DB_PASSWORD)) {
            String sql = "SELECT * FROM users WHERE user = ? AND password = ?";
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setString(1, userId);
            stmt.setString(2, password);
            ResultSet rs = stmt.executeQuery();
            if (rs.next()) {
                currentUser = userId;
                showNotesPage();
            } else {
                showAlert("Login Failed", "Invalid UserID or Password");
            }
        } catch (SQLException ex) {
            ex.printStackTrace();
        }
    }

    private void showNotesPage() {
        VBox layout = new VBox(10);
        layout.setStyle("-fx-background-color: #f5f5dc; -fx-padding: 20;");

        Label titleLabel = new Label("Welcome, " + currentUser);
        titleLabel.setFont(new Font("Arial", 20));

        noteListView = new ListView<>();
        List<String> noteTitles = getNoteTitles();
        noteListView.setItems(FXCollections.observableArrayList(noteTitles));
        noteListView.setOnMouseClicked(e -> selectNoteForViewing());

        viewNoteButton = new Button("View Selected Note");
        viewNoteButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
        viewNoteButton.setFont(new Font("Arial", 16));
        viewNoteButton.setOnAction(e -> openNotePage());

        Button addNoteButton = new Button("Add Note");
        addNoteButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
        addNoteButton.setFont(new Font("Arial", 16));
        addNoteButton.setOnAction(e -> showAddNotePage());
        Button logoutButton = new Button("Logout");
        logoutButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
        logoutButton.setFont(new Font("Arial", 16));
        logoutButton.setOnAction(e -> logoutUser());

        Button viewProfileButton = new Button("View Profile");
        viewProfileButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
        viewProfileButton.setFont(new Font("Arial", 16));
        viewProfileButton.setOnAction(e -> showUserProfilePage());

        layout.getChildren().addAll(titleLabel, noteListView, viewNoteButton, addNoteButton, viewProfileButton, 
logoutButton);
        Scene notesScene = new Scene(layout, 400, 400);
        primaryStage.setScene(notesScene);
    }



    private void selectNoteForViewing() {
        selectedNoteTitle = noteListView.getSelectionModel().getSelectedItem();
        viewNoteButton.setDisable(false);
    }

    private List<String> getNoteTitles() {
        List<String> titles = new ArrayList<>();
        try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME, DB_PASSWORD)) {
            String sql = "SELECT title FROM notes WHERE user = ?";
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setString(1, currentUser);
            ResultSet rs = stmt.executeQuery();

            while (rs.next()) {
                titles.add(rs.getString("title"));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return titles;
    }

    private void openNotePage() {
        String content = getNoteContent(selectedNoteTitle);

        VBox layout = new VBox(10);
        layout.setStyle("-fx-background-color: #f5f5dc; -fx-padding: 20;");

        Label titleLabel = new Label("Title: " + selectedNoteTitle);
        titleLabel.setStyle("-fx-font-size: 18px; -fx-font-weight: bold;");

        TextArea contentArea = new TextArea(content);
        contentArea.setWrapText(true);
        contentArea.setEditable(false);

        Button editButton = new Button("Edit Note");
        editButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
        editButton.setOnAction(e -> showEditNotePage());

        Button backButton = new Button("Back");
        backButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
        backButton.setOnAction(e -> showNotesPage());
    layout.getChildren().addAll(titleLabel, contentArea, editButton, backButton);

    Scene notePageScene = new Scene(layout, 400, 400);
    primaryStage.setScene(notePageScene);
}

private String getNoteContent(String title) {
    String content = "";
    try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME, DB_PASSWORD)) {
        String sql = "SELECT content FROM notes WHERE user = ? AND title = ?";
        PreparedStatement stmt = conn.prepareStatement(sql);
        stmt.setString(1, currentUser);
        stmt.setString(2, title);
        ResultSet rs = stmt.executeQuery();

        if (rs.next()) {
            content = rs.getString("content");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return content;
}

private void showEditNotePage() {
    VBox layout = new VBox(10);
    layout.setStyle("-fx-background-color: #f5f5dc; -fx-padding: 20;");
    String currentContent = getNoteContent(selectedNoteTitle);

    Label titleLabel = new Label("Edit Note: " + selectedNoteTitle);
    titleLabel.setStyle("-fx-font-size: 18px; -fx-font-weight: bold;");

    TextArea contentArea = new TextArea(currentContent);
    contentArea.setWrapText(true);

    Button saveButton = new Button("Save Edit");
    saveButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
    saveButton.setOnAction(e -> saveEditedNote(contentArea.getText()));

    Button backButton = new Button("Back");
    backButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
    backButton.setOnAction(e -> openNotePage());

    layout.getChildren().addAll(titleLabel, contentArea, saveButton, backButton);
    Scene editNoteScene = new Scene(layout, 400, 400);
    primaryStage.setScene(editNoteScene);
}

private void saveEditedNote(String newContent) {
    try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME, DB_PASSWORD)) {
        String sql = "UPDATE notes SET content = ? WHERE user = ? AND title = ?";
        PreparedStatement stmt = conn.prepareStatement(sql);
        stmt.setString(1, newContent);
        stmt.setString(2, currentUser);
        stmt.setString(3, selectedNoteTitle);
        stmt.executeUpdate();
        showAlert("Success", "Note updated successfully!");
        openNotePage();
    }
    catch (SQLException e) {
            e.printStackTrace();
            showAlert("Error", "An error occurred while updating the note.");
        }


    }

private void showAddNotePage() {
    VBox layout = new VBox(10);
    layout.setStyle("-fx-background-color: #f5f5dc; -fx-padding: 20;");

    Label titleLabel = new Label("Add a New Note");
    titleLabel.setStyle("-fx-font-size: 18px; -fx-font-weight: bold;");

    TextField titleField = new TextField();
    titleField.setPromptText("Enter note title");

    TextArea contentArea = new TextArea();
    contentArea.setPromptText("Enter note content");

    Button saveButton = new Button("Save Note");
    saveButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
    saveButton.setOnAction(e -> saveNewNote(titleField.getText(), contentArea.getText()));

    Button backButton = new Button("Back");
    backButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
    backButton.setOnAction(e -> showNotesPage());

    layout.getChildren().addAll(titleLabel, titleField, contentArea, saveButton, backButton);

    Scene addNoteScene = new Scene(layout, 400, 400);
    primaryStage.setScene(addNoteScene);
}

private void saveNewNote(String title, String content) {
    try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME, DB_PASSWORD)) {
        String sql = "INSERT INTO notes (user, title, content) VALUES (?, ?, ?)";
        PreparedStatement stmt = conn.prepareStatement(sql);
        stmt.setString(1, currentUser);
        stmt.setString(2, title);
        stmt.setString(3, content);
        stmt.executeUpdate();
        showAlert("Success", "Note added successfully!");
        showNotesPage();
    } catch (SQLException e) {
        e.printStackTrace();
        showAlert("Error", "An error occurred while adding the note.");
    }
}

private void logoutUser() {
    currentUser = null;
    userIdField.clear();
    passwordField.clear();
    primaryStage.setScene(new Scene(createLoginPage(), 400, 300));
}

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();
}

private void signupUser() {
    VBox layout = new VBox(10);
    layout.setStyle("-fx-background-color: #f5f5dc; -fx-padding: 20;");

    Label userIdLabel = new Label("UserID:");
    userIdLabel.setFont(new Font("Arial", 16));
    TextField userIdField = new TextField();
    userIdField.setFont(new Font("Arial", 14));

    Label passwordLabel = new Label("Password:");
    passwordLabel.setFont(new Font("Arial", 16));
    PasswordField passwordField = new PasswordField();
    passwordField.setFont(new Font("Arial", 14));

    Button signupButton = new Button("Sign Up");
    signupButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
    signupButton.setFont(new Font("Arial", 16));
    signupButton.setOnAction(e -> registerUser(userIdField.getText(), passwordField.getText()));

    layout.getChildren().addAll(userIdLabel, userIdField, passwordLabel, passwordField, signupButton);
    Scene signupScene = new Scene(layout, 400, 300);
    primaryStage.setScene(signupScene);
}

private void registerUser(String userId, String password) {
    try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME, DB_PASSWORD)) {
        String sql = "INSERT INTO users (user, password) VALUES (?, ?)";
        PreparedStatement stmt = conn.prepareStatement(sql);
        stmt.setString(1, userId);
        stmt.setString(2, password);
        stmt.executeUpdate();
        showAlert("Registration Success", "User registered successfully!");
        primaryStage.setScene(new Scene(createLoginPage(), 400, 300));
    } catch (SQLException e) {
        e.printStackTrace();
        showAlert("Error", "An error occurred while registering the user.");
    }
}
private void showUserProfilePage() {
    VBox layout = new VBox(10);
    layout.setStyle("-fx-background-color: #f5f5dc; -fx-padding: 20;");

    Label titleLabel = new Label("User Profile");
    titleLabel.setFont(new Font("Arial", 20));

    Label userIdLabel = new Label("UserID: " + currentUser);
    userIdLabel.setFont(new Font("Arial", 16));

    // Fetch other profile details if needed (like email, etc.)
    String email = getUserEmail(currentUser); // Replace with actual method to get email from DB if available
    Label emailLabel = new Label("Email: " + email);
    emailLabel.setFont(new Font("Arial", 16));

    Button editProfileButton = new Button("Edit Profile");
    editProfileButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
    editProfileButton.setFont(new Font("Arial", 16));
    editProfileButton.setOnAction(e -> showEditProfilePage());

    Button backButton = new Button("Back");
    backButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
    backButton.setFont(new Font("Arial", 16));
    backButton.setOnAction(e -> showNotesPage());

    layout.getChildren().addAll(titleLabel, userIdLabel, emailLabel, editProfileButton, backButton);
    Scene profileScene = new Scene(layout, 400, 400);
    primaryStage.setScene(profileScene);
}

    public String getUserEmail(String userId) {
        String email = "No email found";  // Default value
        try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME, DB_PASSWORD)) {
            // Query to select email based on user_id
            String sql = "SELECT email FROM useremails WHERE user = ?";
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setString(1, userId);  // Set the user_id parameter
            ResultSet rs = stmt.executeQuery();

            if (rs.next()) {
                email = rs.getString("email");  // Retrieve email from the result set
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return email;  // Return the email or the default message
    }



public void saveProfileChanges(String email) {
    // Logic to save the profile changes, for example:
    System.out.println("Profile changes saved for email: " + email);
    // You can add further logic to update the profile information in the backend, database, etc.
}

private void showEditProfilePage() {
    VBox layout = new VBox(10);
    layout.setStyle("-fx-background-color: #f5f5dc; -fx-padding: 20;");

    Label titleLabel = new Label("Edit Profile");
    titleLabel.setFont(new Font("Arial", 20));

    Label userIdLabel = new Label("UserID: " + currentUser);
    userIdLabel.setFont(new Font("Arial", 16));

    TextField emailField = new TextField();
    emailField.setPromptText("Enter new email");
    emailField.setText(getUserEmail(currentUser));

    Button saveButton = new Button("Save Changes");
    saveButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
    saveButton.setFont(new Font("Arial", 16));
    saveButton.setOnAction(e -> saveProfileChanges(emailField.getText()));

    Button backButton = new Button("Back");
    backButton.setStyle("-fx-background-color: #8b4513; -fx-text-fill: white;");
    backButton.setFont(new Font("Arial", 16));
    backButton.setOnAction(e -> showUserProfilePage());

    layout.getChildren().addAll(titleLabel, userIdLabel, emailField, saveButton, backButton);
    Scene editProfileScene = new Scene(layout, 400, 400);
    primaryStage.setScene(editProfileScene);
}

public boolean updateEmail(String userId, String newEmail) {
    boolean isUpdated = false;  // Track if update is successful
    try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME, DB_PASSWORD)) {
        // Query to update the email based on user_id
        String sql = "UPDATE useremails SET email = ? WHERE user= ?";
        PreparedStatement stmt = conn.prepareStatement(sql);
        stmt.setString(1, newEmail);  // Set the new email
        stmt.setString(2, userId);    // Set the user_id to identify the correct row

        int rowsAffected = stmt.executeUpdate();  // Execute the update
        if (rowsAffected > 0) {
            isUpdated = true;  // Email was updated successfully
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return isUpdated;  // Return whether the update was successful or not
}







}
```

SNAPSHOTS

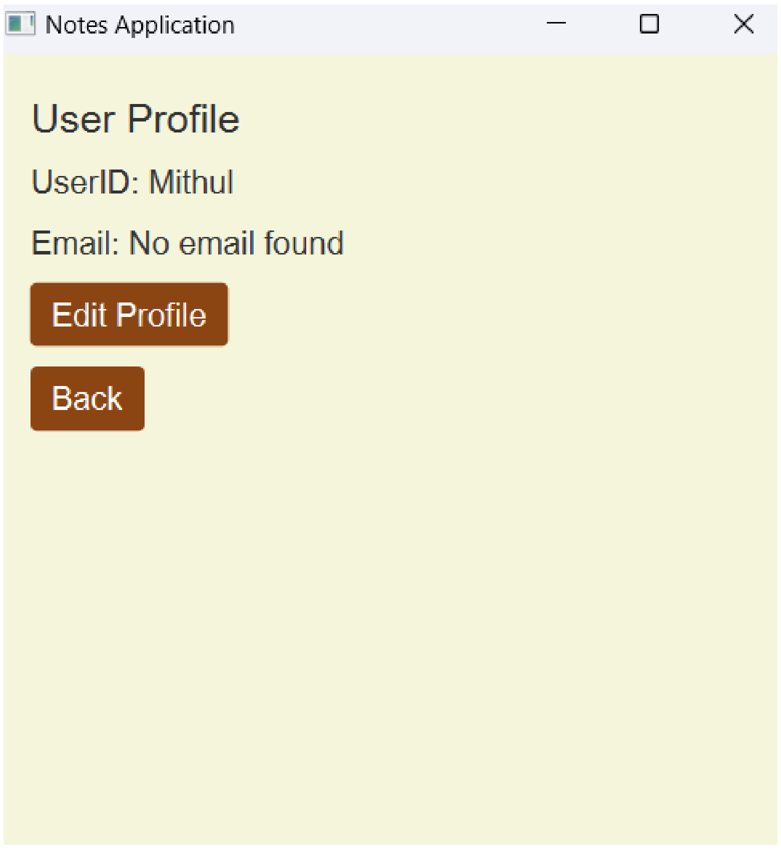
4.1LOGIN PAGE



4.2NOTES HISTORY PAGE



4.3 PROFILE PAGE

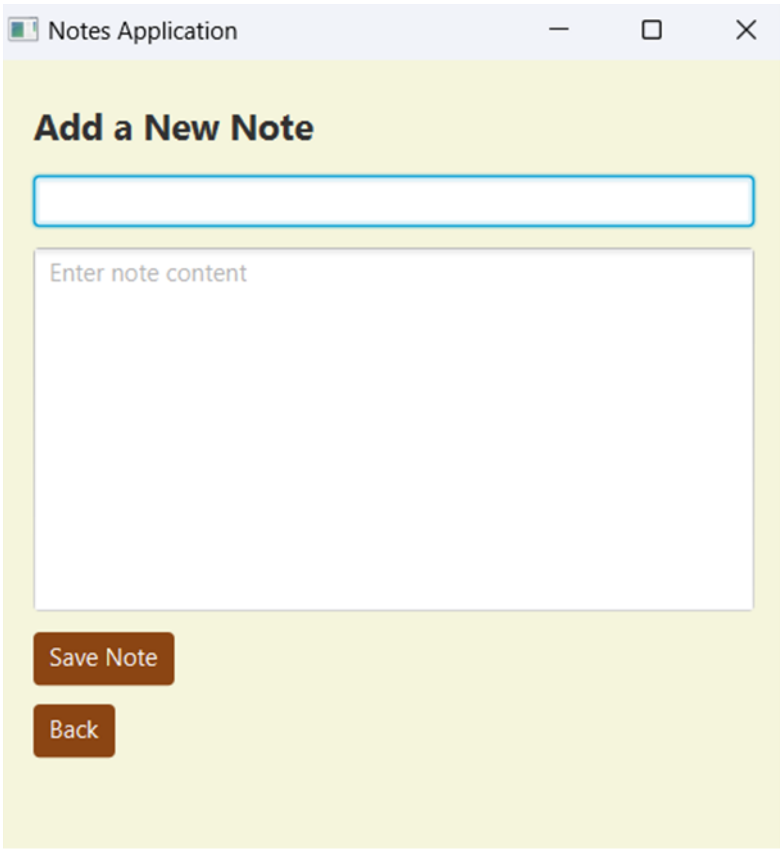


User Profile

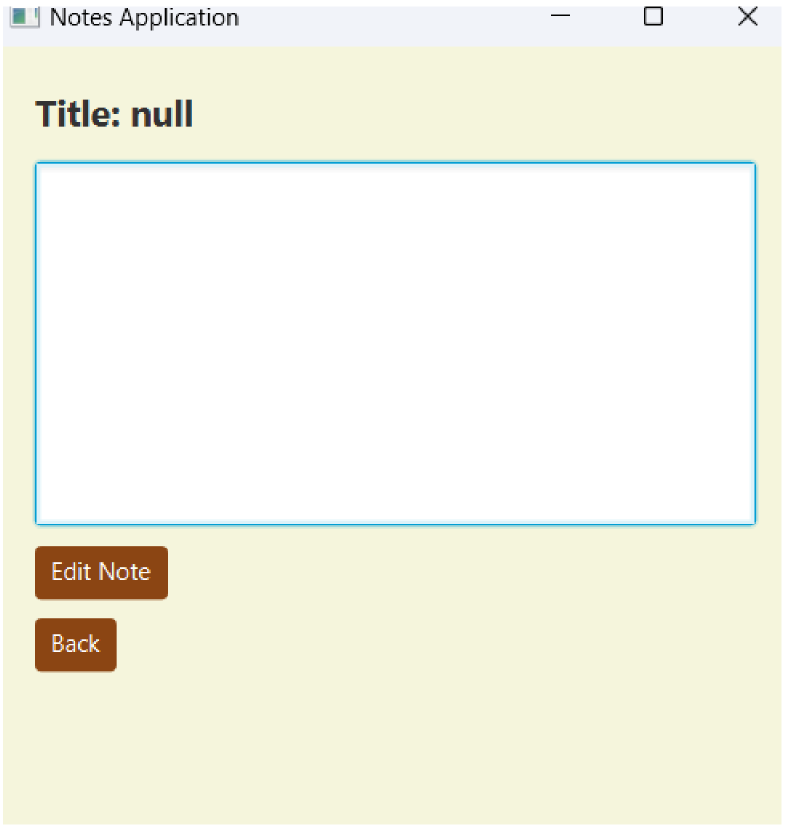
UserID: Mithul

Email: No email found

4.4NOTES ENTRY PAGE



4.5NOTES VIEW PAGE



CONCLUSION

The Notes Entry project provides a convenient and efficient solution for digital note-taking. By offering features like easy note entry, powerful search functionality, this project empowers users to organize and access their notes effortlessly. This streamlined approach to note-taking can significantly enhance learning and productivity.

REFERENCES

1. **https://www.javatpoint.com/java-tutorial**
2. https://www.wikipedia.org/
3. https://www.w3schools.com/sql/
4. SQL | Codecademy