CODING PLATFORM MANAGEMENT

A MINI-PROJECT BY:

KUMARAN S 230701159

LOKESH SINGH N 230701166

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 **“CODING PLATFORM MANAGEMENT”** is the bonafide work of **“KUMARAN S , LOKESH SINGH N”** who carried out the project work under my supervision.

Submitted for the practical examination held on ____________________

INTERNAL EXAMINER	**EXTERNAL EXAMINER**

ABSTRACT

The **Coding Platform Management System** is a desktop-based application developed using Java Swing and MySQL to facilitate an interactive and user-friendly environment for coders to participate in challenges, submit solutions, and track their performance through a leaderboard.

The platform integrates essential features such as user registration, login, challenge creation, solution submission, and a dynamic leaderboard.

Administrators and authorized users can contribute to the platform by creating new challenges, ensuring a continuously evolving repository of problems to keep the coding community engaged.

The system's back-end architecture is designed for scalability and efficiency, leveraging SQL-based operations to manage user data, challenge records, while ensuring data integrity through well-defined relationships among the tables.

This application provides a simple, scalable, and engaging platform for coding enthusiasts, and fostering a coding community.

TABLE OF CONTENTS

INTRODUCTION

1.1 INTRODUCTION

The **Coding Platform Management System** is designed to provide a centralized solution for managing coding challenges and competitions. This system allows users to register, log in, and participate in challenges tailored to various skill levels. By integrating features such as automated code submission evaluation and dynamic leaderboards, the platform creates an engaging environment for learning and competitive programming. Developed with Java Swing for a user-friendly interface and MySQL for backend data management, the platform ensures a robust and secure environment for users to enhance their coding skills.

1.2 IMPLEMENTATION

The **CODING PLATFORM MANAGEMENT** project discussed here is implemented using the concepts of **JAVA SWINGS** and **MYSQL**.

1.3 SCOPE OF THE PROJECT

This **System** provides a centralized platform for coding enthusiasts to enhance their skills through challenges and competitions. It facilitates user registration, secure authentication, automated code evaluation, and a dynamic leaderboard. Designed to be scalable, it caters to individual learners, coding communities, and organizations hosting coding events.

1.3 WEBSITE FEATURES

- ★ Registeration and Login Page.
- ★ Main Dashboard and Users Score
- ★ Challenge Frame and Create Challenge Frame.
- ★ Submission Frame and Leaderboard Frame

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS:

PROCESSOR	:	Intel i3
MEMORY SIZE	:	4.00 GB

2.2 SOFTWARE SPECIFICATIONS:

PROGRAMMING LANGUAGE	:	Java, MySQL
FRONT-END	:	Java
BACK-END	:	MySQL
OPERATING SYSTEM	:	Windows 10

SAMPLE CODE

3.1 MAIN PAGE DESIGN

```
import javax.swing.*;

public class Main {
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (Exception e) {
            e.printStackTrace();
        }
        SwingUtilities.invokeLater(() -> {
            LoginFrame loginFrame = new LoginFrame();
            loginFrame.setVisible(true);
        });
    }
}
```

3.2 REGISTRATION PAGE DESIGN

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class RegistrationFrame extends JFrame {
    private JTextField usernameField;
    private JTextField emailField;
    private JPasswordField passwordField;
    private JButton registerButton;

    public RegistrationFrame() {
        setTitle("Register");
        setSize(400, 350);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        JPanel panel = new JPanel();
        panel.setLayout(null);

        JLabel usernameLabel = new JLabel("Username:");
        usernameLabel.setBounds(50, 50, 100, 30);
        panel.add(usernameLabel);

        usernameField = new JTextField();
        usernameField.setBounds(150, 50, 150, 30);
        panel.add(usernameField);

        JLabel emailLabel = new JLabel("Email:");
```

```java
        emailLabel.setBounds(50, 100, 100, 30);
        panel.add(emailLabel);

        emailField = new JTextField();
        emailField.setBounds(150, 100, 150, 30);
        panel.add(emailField);

        JLabel passwordLabel = new JLabel("Password:");
        passwordLabel.setBounds(50, 150, 100, 30);
        panel.add(passwordLabel);

        passwordField = new JPasswordField();
        passwordField.setBounds(150, 150, 150, 30);
        panel.add(passwordField);

        registerButton = new JButton("Register");
        registerButton.setBounds(150, 200, 100, 30);
        panel.add(registerButton);

        registerButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String username = usernameField.getText();
                String email = emailField.getText();
                String password = new String(passwordField.getPassword());
                Database.registerUser(username, password, email);
                JOptionPane.showMessageDialog(null, "Registration successful!");
                new LoginFrame().setVisible(true);
                dispose();
            }
        });

        add(panel);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            new RegistrationFrame().setVisible(true);
        });
    }
}
```

3.3 LOGIN PAGE DESIGN:

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class LoginFrame extends JFrame {
    private JTextField usernameField;
    private JPasswordField passwordField;
    private JButton loginButton;
    private JButton registerButton;

    public LoginFrame() {
        setTitle("Login");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        JPanel panel = new JPanel();
        panel.setLayout(null);

        JLabel usernameLabel = new JLabel("Username:");
        usernameLabel.setBounds(50, 50, 100, 30);
        panel.add(usernameLabel);

        usernameField = new JTextField();
        usernameField.setBounds(150, 50, 150, 30);
        panel.add(usernameField);

        JLabel passwordLabel = new JLabel("Password:");
        passwordLabel.setBounds(50, 100, 100, 30);
        panel.add(passwordLabel);

        passwordField = new JPasswordField();
        passwordField.setBounds(150, 100, 150, 30);
        panel.add(passwordField);

        loginButton = new JButton("Login");
        loginButton.setBounds(50, 150, 100, 30);
        panel.add(loginButton);

        registerButton = new JButton("Register");
        registerButton.setBounds(200, 150, 100, 30);
        panel.add(registerButton);

        loginButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String username = usernameField.getText();
                String password = new String(passwordField.getPassword());
                int userId = Database.isLoginValid(username, password);
                if (userId != -1) {
                    new MainDashboard(username, userId).setVisible(true);
                    dispose();
                } else {
                    JOptionPane.showMessageDialog(null, "Invalid credentials, please try again.");
                }
            }
        });

        registerButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                new RegistrationFrame().setVisible(true);
                dispose();
            }
        });

        add(panel);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            new LoginFrame().setVisible(true);
        });
    }
}
```

3.4 DASHBOARD DESIGN

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class MainDashboard extends JFrame {
  private int userId;
  private String username;

  public MainDashboard(String username, int userId) {
    this.username = username;
    this.userId = userId;

    setTitle("Dashboard");
    setSize(400, 300);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLocationRelativeTo(null);

    JPanel panel = new JPanel();
    panel.setLayout(null);
```

```
        JLabel welcomeLabel = new JLabel("Welcome, " + username + "!");
        welcomeLabel.setBounds(50, 30, 300, 30);
        panel.add(welcomeLabel);

        JButton challengesButton = new JButton("Challenges");
        challengesButton.setBounds(100, 80, 150, 30);
        panel.add(challengesButton);

        JButton createChallengeButton = new JButton("Create Challenge");
        createChallengeButton.setBounds(100, 120, 150, 30);
        panel.add(createChallengeButton);

        JButton leaderboardButton = new JButton("Leaderboard");
        leaderboardButton.setBounds(100, 160, 150, 30);
        panel.add(leaderboardButton);

        challengesButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                new ChallengeFrame(userId).setVisible(true);
            }
        });

        createChallengeButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                new CreateChallengeFame(userId).setVisible(true);
            }
        });

        leaderboardButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                new LeaderboardFrame().setVisible(true);
            }
        });

        add(panel);
    }
}
```

3.5 CHALLENGE FRAME DESIGN

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.List;

public class ChallengeFrame extends JFrame {
    private int userId;
    public ChallengeFrame(int userId) {
        this.userId = userId;

        setTitle("Challenges");
        setSize(600, 500);
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setLocationRelativeTo(null);

        JPanel panel = new JPanel();
        panel.setLayout(null);

        JLabel selectChallengeLabel = new JLabel("Select a challenge:");
        selectChallengeLabel.setBounds(50, 30, 300, 30);
        panel.add(selectChallengeLabel);

        List<String> challenges = Database.getChallenges();
        int yPosition = 80;
        for (String challenge : challenges) {
            JButton challengeButton = new JButton(challenge);
            challengeButton.setBounds(100, yPosition, 300, 50);
            yPosition += 60;

            challengeButton.addActionListener(new ActionListener() {
                @Override
                public void actionPerformed(ActionEvent e) {
                    int challengeId = Database.getChallengeId(challenge);
                    JTextArea codeArea = new JTextArea(10, 40);
                    codeArea.setLineWrap(true);
                    codeArea.setWrapStyleWord(true);
                    JScrollPane scrollPane = new JScrollPane(codeArea);
                    int result = JOptionPane.showConfirmDialog(
                            null,
                            scrollPane,
                            "Enter solution code:",
                            JOptionPane.OK_CANCEL_OPTION,
                            JOptionPane.PLAIN_MESSAGE

                    );

                    if (result == JOptionPane.OK_OPTION) {
                        String code = codeArea.getText();
                        Database.submitSolution(userId, challengeId, code);
                        JOptionPane.showMessageDialog(null, "Challenge submitted!");
                    }
                }
            });
            panel.add(challengeButton);
        }
        JScrollPane scrollPane = new JScrollPane(panel);
        add(scrollPane);
    }
}
```

3.6 CREATE CHALLENGE FRAME DESIGN

```java
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class CreateChallengeFame extends JFrame {
    private int userId;

    public CreateChallengeFame(int userId) {
        this.userId = userId;

        setTitle("Create Challenge");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setLocationRelativeTo(null);

        JPanel panel = new JPanel();
        panel.setLayout(null);

        JLabel titleLabel = new JLabel("Challenge Title:");
        titleLabel.setBounds(50, 30, 100, 30);
        panel.add(titleLabel);

        JTextField titleField = new JTextField();
        titleField.setBounds(150, 30, 150, 30);
        panel.add(titleField);

        JLabel descLabel = new JLabel("Description:");
        descLabel.setBounds(50, 80, 100, 30);
        panel.add(descLabel);

        JTextArea descField = new JTextArea();
        descField.setBounds(150, 80, 150, 70);
        panel.add(descField);

        JButton createButton = new JButton("Create Challenge");
        createButton.setBounds(100, 200, 150, 30);
        panel.add(createButton);

        createButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String title = titleField.getText();
                String description = descField.getText();

                if (title.isEmpty() || description.isEmpty()) {
                    JOptionPane.showMessageDialog(null, "Please fill out all fields.");
                } else {
                    Database.createChallenge(title, description, userId);
                    JOptionPane.showMessageDialog(null, "Challenge created!");
                    dispose();
                }
            }
        });

        add(panel);
    }
}
```

3.7 SUBMISSION FRAME DESIGN

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class SubmissionFrame extends JFrame {
    private int userId;
    private int challengeId;

    public SubmissionFrame(int userId, int challengeId) {
        this.userId = userId;
        this.challengeId = challengeId;

        setTitle("Submit Solution");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setLocationRelativeTo(null);

        JPanel panel = new JPanel();
        panel.setLayout(null);

        JLabel codeLabel = new JLabel("Enter your solution code:");
        codeLabel.setBounds(50, 30, 300, 30);
        panel.add(codeLabel);

        JTextArea codeArea = new JTextArea();
        codeArea.setBounds(50, 70, 300, 100);
        panel.add(codeArea);

        JButton submitButton = new JButton("Submit");
        submitButton.setBounds(150, 200, 100, 30);
        panel.add(submitButton);

        submitButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String code = codeArea.getText();
                Database.submitSolution(userId, challengeId, code);
                JOptionPane.showMessageDialog(null, "Solution submitted!");
                dispose();
            }
        });

        add(panel);
    }
}
```

3.8 LEADERBOARD FRAME DESIGN

```
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.util.List;

public class LeaderboardFrame extends JFrame {

    public LeaderboardFrame() {
        setTitle("Leaderboard");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setLocationRelativeTo(null);

        JPanel panel = new JPanel();
        panel.setLayout(new BorderLayout());

        JLabel titleLabel = new JLabel("Leaderboard", JLabel.CENTER);
        titleLabel.setFont(new Font("Arial", Font.BOLD, 20));
        panel.add(titleLabel, BorderLayout.NORTH);

        String[] columnNames = {"Rank", "Username", "Score"};
        DefaultTableModel model = new DefaultTableModel(columnNames, 0);

        List<UserScore> scores = Database.getTopScores();
        int rank = 1;
        for (UserScore score : scores) {
            model.addRow(new Object[]{rank++, score.getUsername(), score.getScore()});
        }

        JTable table = new JTable(model);
        panel.add(new JScrollPane(table), BorderLayout.CENTER);

        add(panel);
    }
}
```

3.9 USERS SCORE

```
public class UserScore {
    private String username;
    private int score;

    public UserScore(String username, int score) {
        this.username = username;
        this.score = score;
    }

    public String getUsername() {
        return username;
    }

    public int getScore() {
        return score;
    }
}
```

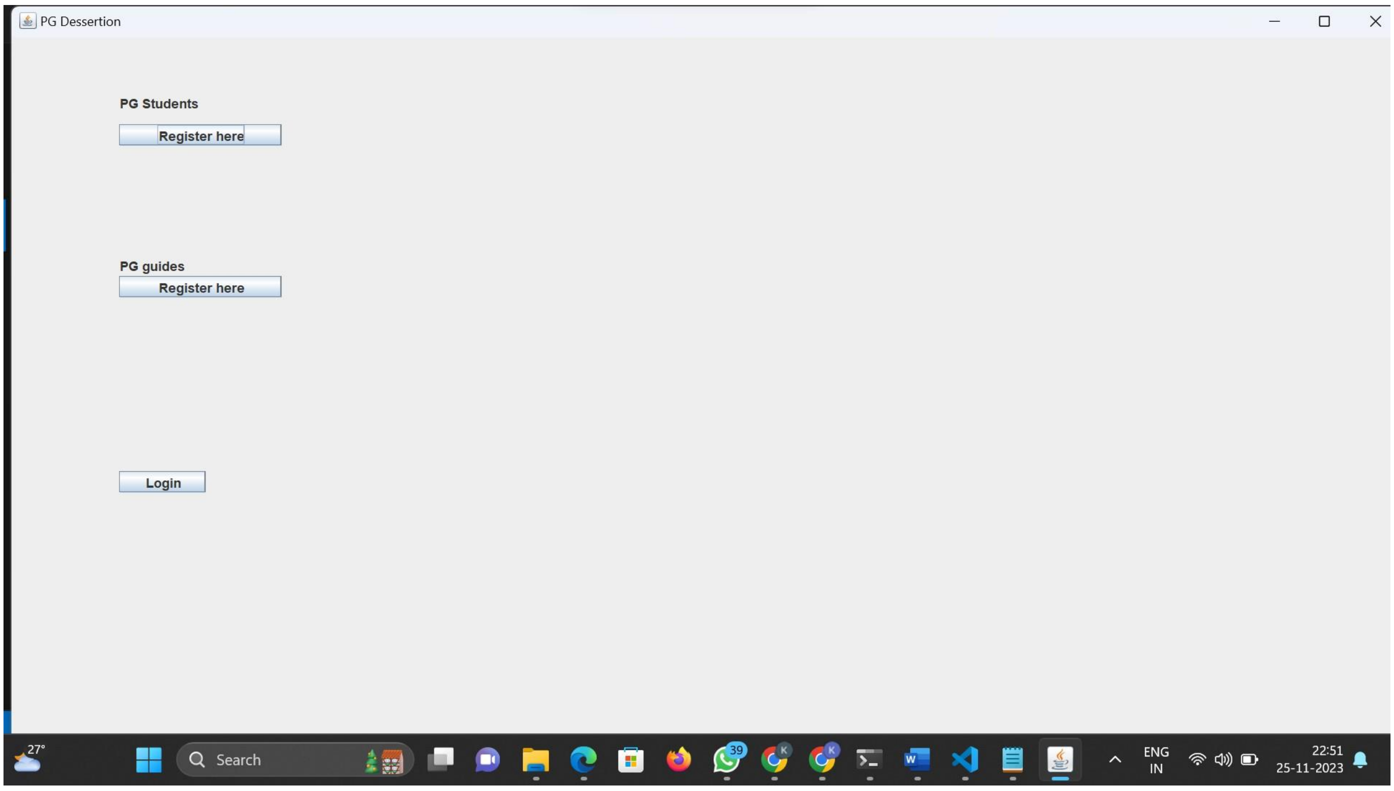
3.10 DATABASE CONNECTIVITY

```
import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class Database {
    private static final String URL = "jdbc:mysql://localhost:3306/coding_platform";
    private static final String USER = "root";
    private static final String PASSWORD = "Sakk29@2006";

    public static Connection connect() {
        try {
            return DriverManager.getConnection(URL, USER, PASSWORD);
        } catch (SQLException e) {
            e.printStackTrace();
            return null;
        }
    }

    public static int isLoginValid(String username, String password) {
        String query = "SELECT user_id FROM users WHERE username = ? AND password_hash = ?";
        try (Connection conn = connect(); PreparedStatement ps = conn.prepareStatement(query)) {
            ps.setString(1, username);
            ps.setString(2, password);
            ResultSet rs = ps.executeQuery();
            if (rs.next()) {
                return rs.getInt("user_id");
            } else {
                return -1;
            }
        } catch (SQLException e) {
            e.printStackTrace();
            return -1;
        }
    }

    public static void registerUser(String username, String password, String email) {
        String insertQuery = "INSERT INTO users (username, password_hash, email) VALUES (?, ?, ?)";
        try (Connection conn = connect(); PreparedStatement stmt = conn.prepareStatement(insertQuery)) {
            stmt.setString(1, username);
            stmt.setString(2, password);
            stmt.setString(3, email);
            stmt.executeUpdate();
            System.out.println("User registered successfully.");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public static void createChallenge(String title, String description, int userId) {
        String query = "INSERT INTO challenges (title, description, difficulty) VALUES (?, ?, 'Easy')";
        try (Connection conn = connect(); PreparedStatement ps = conn.prepareStatement(query)) {
            ps.setString(1, title);
            ps.setString(2, description);
            ps.executeUpdate();
            System.out.println("Challenge created successfully.");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public static List<String> getChallenges() {
        List<String> challenges = new ArrayList<>();
        String query = "SELECT title FROM challenges";
        try (Connection conn = connect(); PreparedStatement ps = conn.prepareStatement(query)) {
            ResultSet rs = ps.executeQuery();
            while (rs.next()) {
                challenges.add(rs.getString("title"));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return challenges;
    }

    public static int getChallengeId(String challengeTitle) {
        String query = "SELECT challenge_id FROM challenges WHERE title = ?";
        try (Connection conn = connect(); PreparedStatement ps = conn.prepareStatement(query)) {
            ps.setString(1, challengeTitle);
            ResultSet rs = ps.executeQuery();
            if (rs.next()) {
                return rs.getInt("challenge_id");
            } else {
                return -1;
            }
        } catch (SQLException e) {
            e.printStackTrace();
            return -1;
        }
    }

    public static void submitSolution(int userId, int challengeId, String code) {
        String insertSubmission = "INSERT INTO submissions (user_id, challenge_id, code, status,
submission_time) VALUES (?, ?, ?, 'Passed', NOW())";
        try (Connection conn = connect(); PreparedStatement stmt =
conn.prepareStatement(insertSubmission)) {
            stmt.setInt(1, userId);
            stmt.setInt(2, challengeId);
            stmt.setString(3, code);
            stmt.executeUpdate();

            updateScore(userId, 10);  // Assume each successful submission adds 10 points
            System.out.println("Solution submitted successfully and score updated.");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public static void updateScore(int userId, int score) {
        String query = "UPDATE users SET score = score + ? WHERE user_id = ?";
        try (Connection conn = connect(); PreparedStatement stmt = conn.prepareStatement(query)) {
            stmt.setInt(1, score);
            stmt.setInt(2, userId);
            stmt.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public static List<String> getLeaderboard() {
        List<String> leaderboard = new ArrayList<>();
        String query = "SELECT username, score FROM users ORDER BY score DESC";
        try (Connection conn = connect(); PreparedStatement ps = conn.prepareStatement(query)) {
            ResultSet rs = ps.executeQuery();
            while (rs.next()) {
                leaderboard.add(rs.getString("username") + ": " + rs.getInt("score") + " points");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return leaderboard;
    }
```

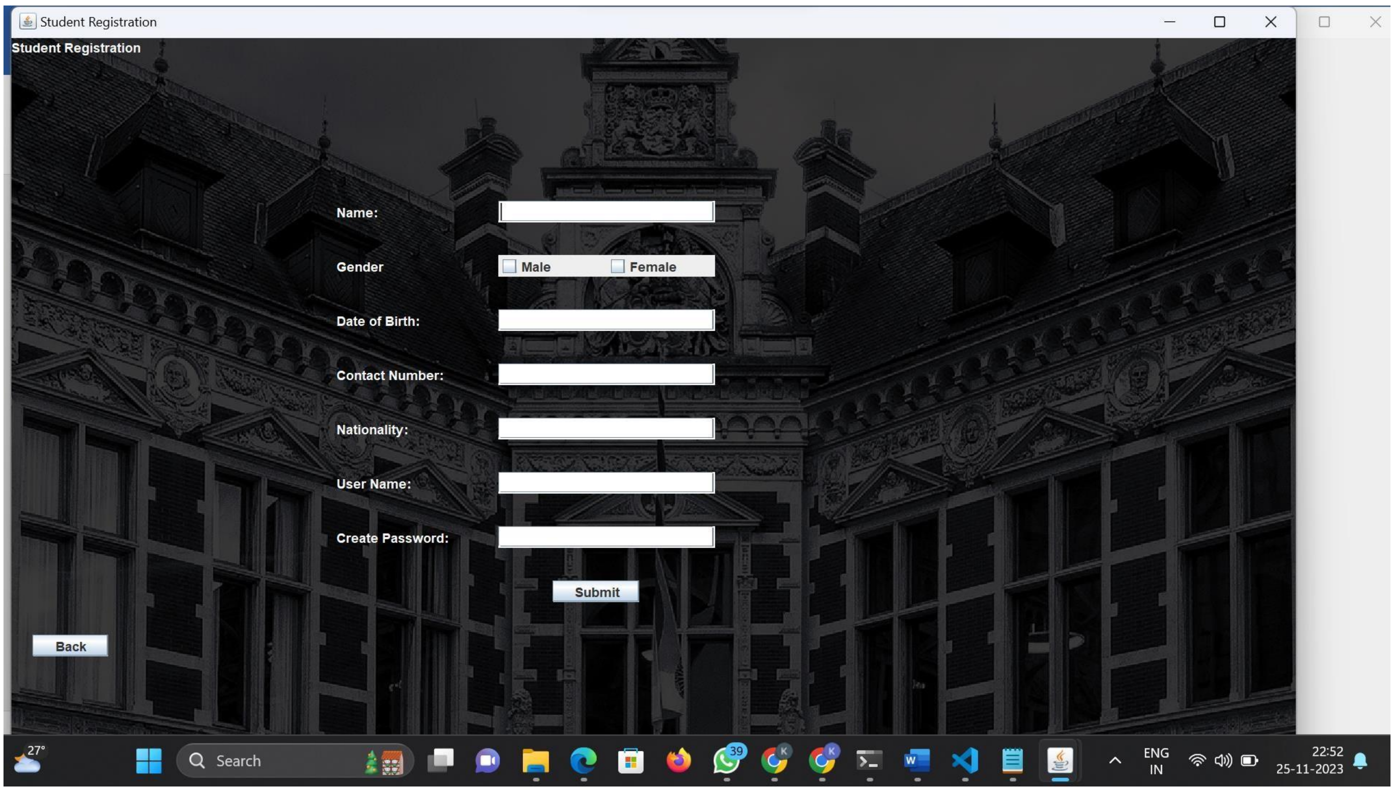
```java
    public static List<UserScore> getTopScores() {
        List<UserScore> scores = new ArrayList<>();
        String query = "SELECT username, score FROM users ORDER BY score DESC LIMIT 10";
        try (Connection conn = connect(); PreparedStatement ps = conn.prepareStatement(query)) {
            ResultSet rs = ps.executeQuery();
            while (rs.next()) {
                String username = rs.getString("username");
                int score = rs.getInt("score");
                scores.add(new UserScore(username, score));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return scores;
    }
}
```

SNAPSHOTS

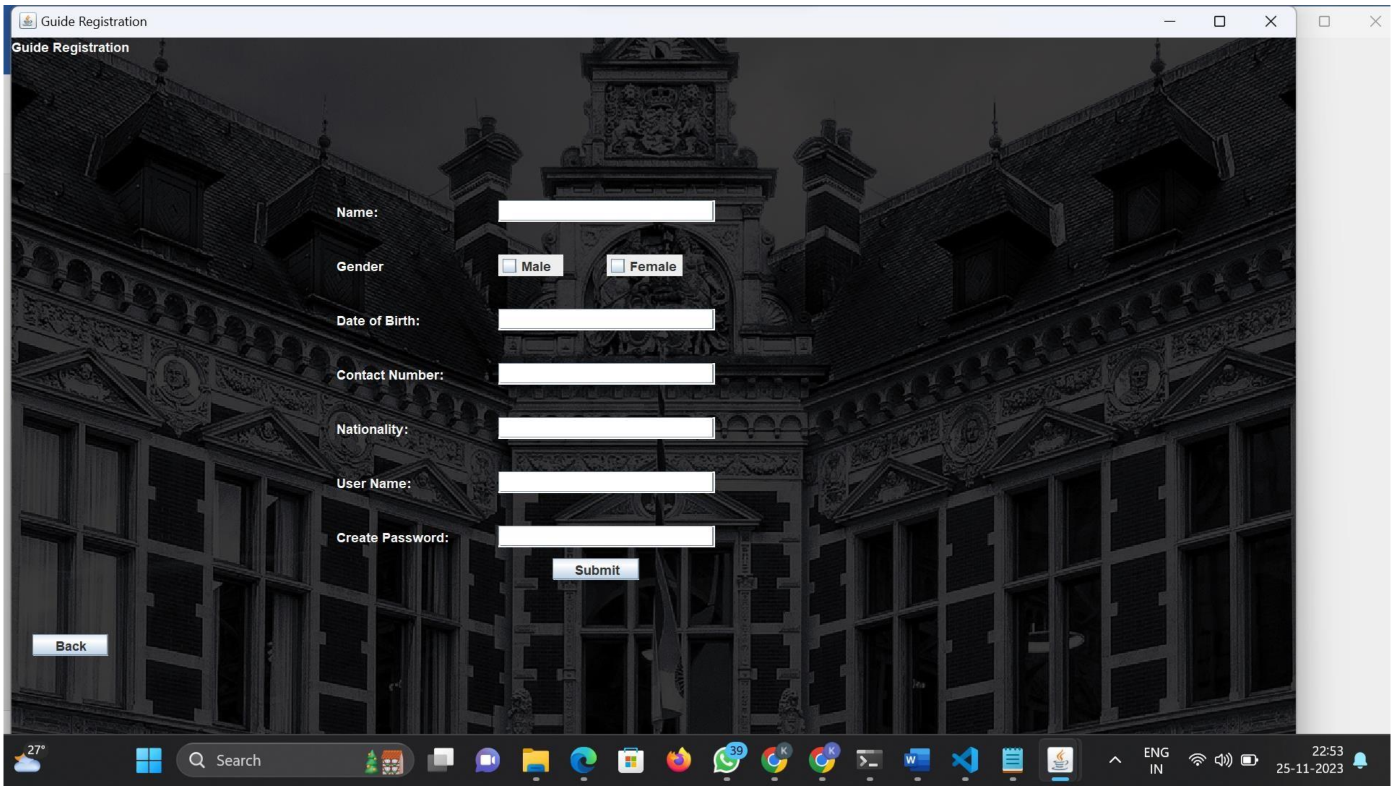
4.1 HOME PAGE



4.2 STUDENT REGISTRATION PAGE



4.3 GUIDE REGISTRATION



Guide Registration

Guide Registration

Name:

Gender [ ] Male [ ] Female

Date of Birth:

Contact Number:

Nationality:

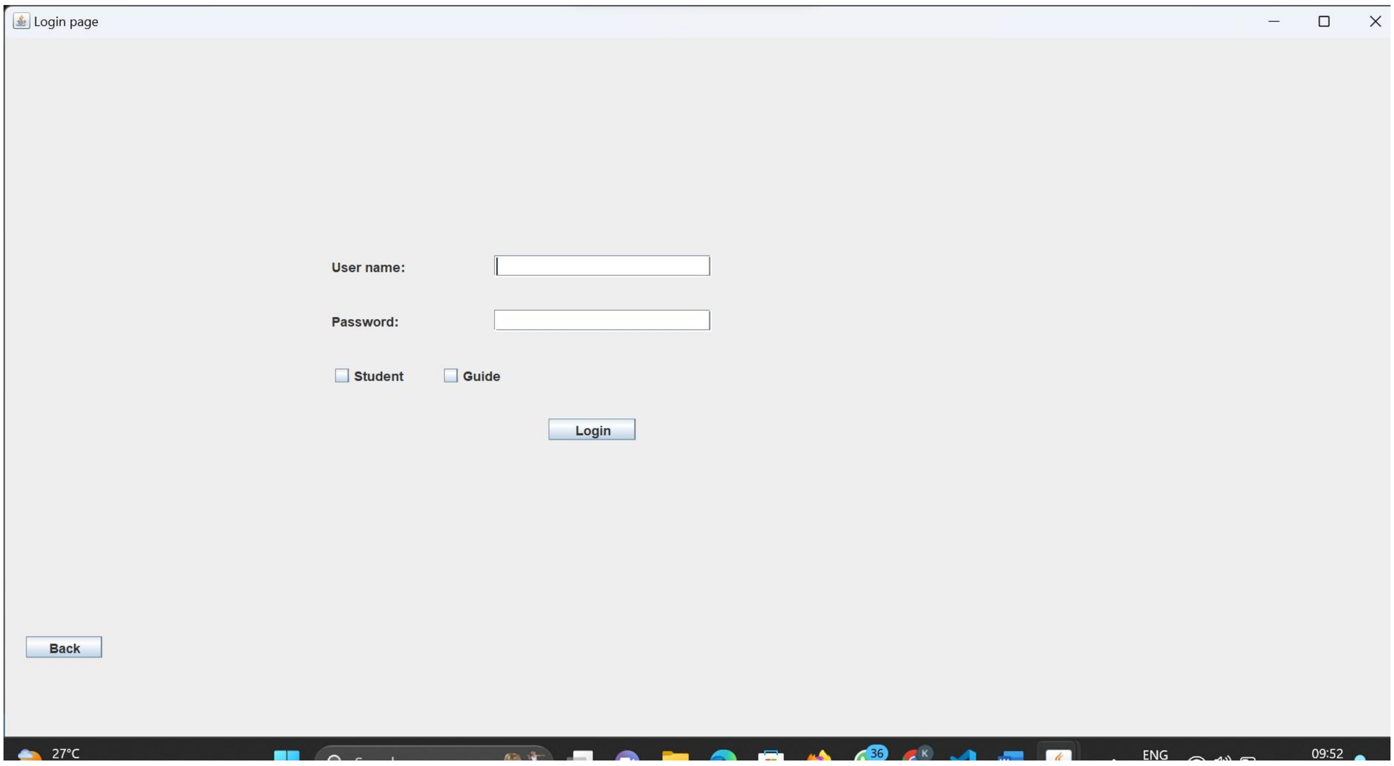
User Name:

Create Password:

Submit

Back

4.4 LOGIN PAGE



Login page

User name:

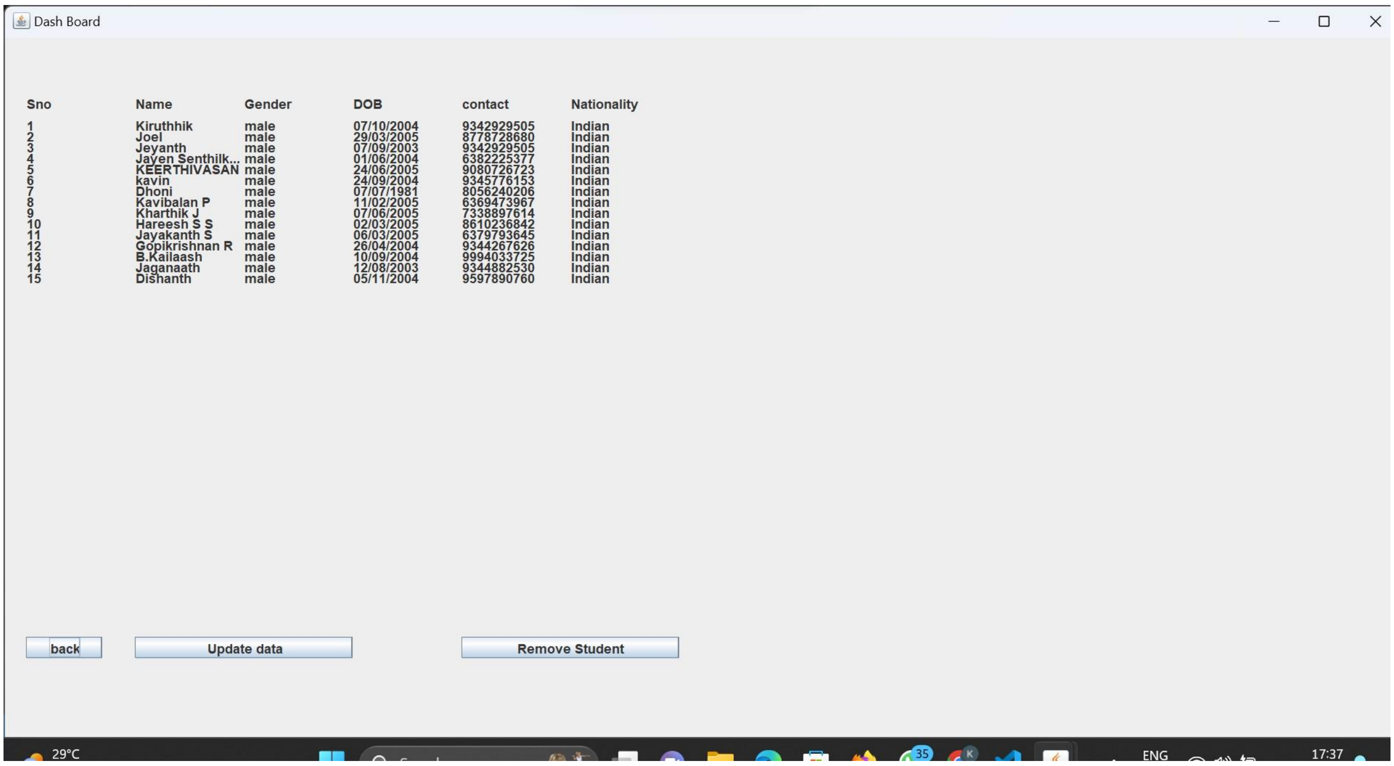
Password:

[ ] Student [ ] Guide

Login

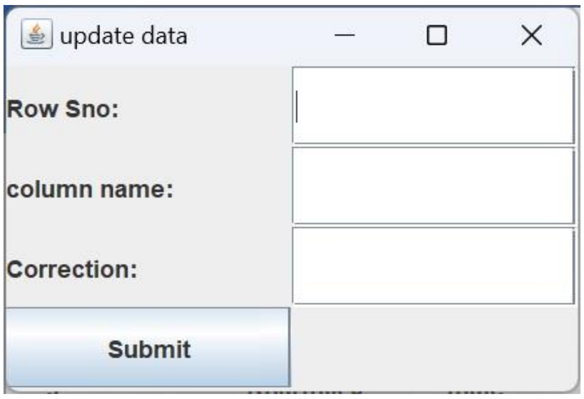
Back

4.5 DASHBOARD

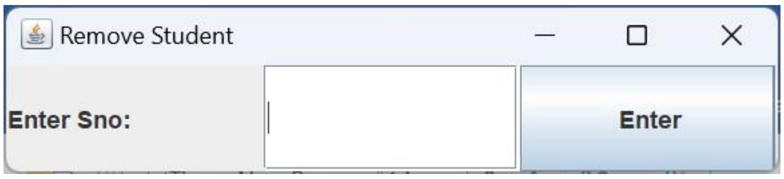


Sno	Name	Gender	DOB	contact	Nationality
1	Kiruthhik	male	07/10/2004	9342929505	Indian
2	Joel	male	29/03/2005	8778728680	Indian
3	Jeyanth	male	07/09/2003	9342929505	Indian
4	Jayen Senthilk...	male	01/06/2004	6382225377	Indian
5	KEERTHIVASAN	male	24/06/2005	9080726723	Indian
6	kavin	male	24/09/2004	9345776153	Indian
7	Dhoni	male	07/07/1981	8056240206	Indian
8	Kavibalan P	male	11/02/2005	6369473967	Indian
9	Kharthik J	male	07/06/2005	7338897614	Indian
10	Hareesh S S	male	02/03/2005	8610236842	Indian
11	Jayakanth S	male	06/03/2005	6379793645	Indian
12	Gopikrishnan R	male	26/04/2004	9344267626	Indian
13	B.Kailaash	male	10/09/2004	9994033725	Indian
14	Jaganaath	male	12/08/2003	9344882530	Indian
15	Dishanth	male	05/11/2004	9597890760	Indian

4.6 UPDATE DATA OPERATION



4.7 REMOVE STUDENT OPERATION



CONCLUSION

With the help of our project, Users can engage in Coding Assignments after which they can view the results on the leaderboard, which leads to the consistent and determination of Users in their Journey.

REFERENCES

1. **https://www.javatpoint.com/java-tutorial**

2. https://www.wikipedia.org/

3. https://www.w3schools.com/sql/

4. SQL | Codecademy