

HOME TUITION MANAGEMENT SYSTEM

A MINI-PROJECT BY:

SANJAY KUMAR S 230701290

SOORYA M P 230701325

in partial fulfilment 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 “**HOME TUITION MANAGEMENT SYSTEM**” is the bona fide work of “**SANJAY KUMAR S, SOORYA M P**” who carried out the project work under my supervision.

Submitted for the practical examination held on _____

SIGNATURE

Mr. G SARAVANA GOKUL
Assistant Professor (SS),
Computer Science and Engineering,
Rajalakshmi Engineering College
(Autonomous),
Thandalam, Chennai-602105

SIGNATURE

Ms. V. JANANEE
Assistant Professor (SG),
Computer Science and Engineering,
Rajalakshmi Engineering College
(Autonomous),
Thandalam, Chennai-602105

INTERNAL EXAMINER

EXTERNAL EXAMINER

ABSTRACT

The Home Tuition Management System is a Java-based mini-project designed to modernize and streamline the management of home tuition operations. This application simplifies the handling of tuition-related data, enhancing accuracy and efficiency while eliminating manual record-keeping. It maintains detailed records for each tutor, student, and subject, including tutor profiles, student details, session schedules, and payment history, all stored securely in a centralized database.

The application integrates Java with a database backend to ensure seamless data storage and retrieval. Authorized users can add, view, update, or delete records with robust data security ensured through user authentication. The system includes advanced search and filtering capabilities, facilitating quick retrieval of information based on specific criteria such as subject, location, or session timings. Additionally, it offers report generation features to help administrators and tutors analyze data trends, track payment statuses, and manage schedules effectively.

Employing modern programming techniques and an intuitive user interface, the project demonstrates how technology can transform home tuition management. It serves as a prototype for scalable real-world applications, promoting administrative efficiency and providing a reliable, error-free repository of critical tuition-related data.

TABLE OF CONTENTS

1. INTRODUCTION

- 1.1. INTRODUCTION
- 1.2. IMPLEMENTATION
- 1.3. SCOPE OF THE PROJECT

2. SYSTEM SPECIFICATION

- 2.1. HARDWARE SPECIFICATION
- 2.2. SOFTWARE SPECIFICATION

3. SAMPLE CODE

4. SNAPSHOTS

5. CONCLUSION

6. REFERENCES

1. INTRODUCTION

1.1 INTRODUCTION

The Home Tuition Management System is a Java-based application designed to efficiently manage and organize tuition-related information. It streamlines the process of recording, storing, and retrieving details about tutors, students, schedules, and payments, ensuring smooth and effective management of home tuition services.

1.2 IMPLEMENTATION

The **Home Tuition Management System** project is implemented using Java Swing for the user interface and MySQL for database management.

1.3 SCOPE OF THE PROJECT

This project holds immense potential to revolutionize home tuition management by offering a seamless, secure, and intuitive platform for organizing tuition-related information. Its core focus is to provide a centralized, efficient solution that empowers tutors, students, and administrators to easily store, access, and update critical details about schedules, payments, and session records, all while ensuring a smooth and professional user experience.

2. SYSTEM SPECIFICATIONS

2.1. HARDWARE SPECIFICATIONS:

PROCESSOR: AMD ryzen 7

MEMORY SIZE: 16GB

HARD DISK: 500 GB of free space

2.2. SOFTWARE SPECIFICATIONS:

PROGRAMMING LANGUAGE: Java, SQL

FRONT-END: Java Swing

BACK-END: MySQL

OPERATING SYSTEM: Windows 11

3. SAMPLE CODE

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;

public class HomeTuitionManagementSystem extends JFrame {

    private static final String URL = "jdbc:mysql://localhost:3306/TuitionDB";
    // Change database URL as needed

    private static final String USER = "root"; // Change username as needed
    private static final String PASSWORD = "Vigshan@2116"; // Change
    password as needed

    // Buttons for the main menu
    private JButton addStudentButton, viewStudentsButton, addTeacherButton,
    viewTeachersButton, assignTeacherButton;

    public HomeTuitionManagementSystem() {
        // Set up frame
        setTitle("Home Tuition Management System");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        // Initialize main menu buttons
        addStudentButton = new JButton("Add Student");
        viewStudentsButton = new JButton("View Students");
        addTeacherButton = new JButton("Add Teacher");
        viewTeachersButton = new JButton("View Teachers");
        assignTeacherButton = new JButton("Assign Teacher");
```

```
// Set layout for main panel
JPanel panel = new JPanel();
panel.setLayout(new GridLayout(6, 1, 10, 10));

panel.add(addStudentButton);
panel.add(viewStudentsButton);
panel.add(addTeacherButton);
panel.add(viewTeachersButton);
panel.add(assignTeacherButton);

add(panel);

// Button event listeners
addStudentButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        addStudent();
    }
});

viewStudentsButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        viewStudentSchedule(); // Show full schedule for students
    }
});

addTeacherButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        addTeacher();
    }
});

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



```

        public void actionPerformed(ActionEvent e) {
            viewTeacherSchedule(); // Show full schedule for teachers
        }
    });

    assignTeacherButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            assignTeacher();
        }
    });
}

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

// Add student dialog
private void addStudent() {
    JTextField nameField = new JTextField(20);
    JTextField ageField = new JTextField(20);
    JTextField contactField = new JTextField(20);
    JTextField feesField = new JTextField(20); // Fees input field

    JPanel panel = new JPanel(new GridLayout(4, 2));
    panel.add(new JLabel("Name:"));
    panel.add(nameField);

```

```
panel.add(new JLabel("Age:"));
panel.add(ageField);
panel.add(new JLabel("Contact Number:"));
panel.add(contactField);
panel.add(new JLabel("Fees:"));
panel.add(feesField);
```

```
int option = JOptionPane.showConfirmDialog(this, panel, "Add New
Student", JOptionPane.OK_CANCEL_OPTION);
```

```
if (option == JOptionPane.OK_OPTION) {
    String name = nameField.getText();
    int age = Integer.parseInt(ageField.getText());
    String contact = contactField.getText();
    double fees = Double.parseDouble(feesField.getText()); // Parsing fees

    try (Connection conn = getConnection()) {
        String query = "INSERT INTO students (name, age, contact_number,
fees) VALUES (?, ?, ?, ?)";
        try (PreparedStatement ps = conn.prepareStatement(query)) {
            ps.setString(1, name);
            ps.setInt(2, age);
            ps.setString(3, contact);
            ps.setDouble(4, fees); // Inserting fees
            ps.executeUpdate();
            JOptionPane.showMessageDialog(this, "Student added
successfully!");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

```

// Add teacher dialog
private void addTeacher() {
    JTextField nameField = new JTextField(20);
    JTextField subjectField = new JTextField(20);
    JTextField contactField = new JTextField(20);

    JPanel panel = new JPanel(new GridLayout(3, 2));
    panel.add(new JLabel("Name:"));
    panel.add(nameField);
    panel.add(new JLabel("Subject:"));
    panel.add(subjectField);
    panel.add(new JLabel("Contact Number:"));
    panel.add(contactField);

    int option = JOptionPane.showConfirmDialog(this, panel, "Add New
Teacher", JOptionPane.OK_CANCEL_OPTION);

    if (option == JOptionPane.OK_OPTION) {
        String name = nameField.getText();
        String subject = subjectField.getText();
        String contact = contactField.getText();

        try (Connection conn = getConnection()) {
            String query = "INSERT INTO teachers (name, subject,
contact_number) VALUES (?, ?, ?)";
            try (PreparedStatement ps = conn.prepareStatement(query)) {
                ps.setString(1, name);
                ps.setString(2, subject);
                ps.setString(3, contact);
                ps.executeUpdate();
                JOptionPane.showMessageDialog(this, "Teacher added
successfully!");
            }
        }
    }
}

```

```

        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// Assign teacher to student
private void assignTeacher() {
    String[] students = getStudentNames();
    String[] teachers = getTeacherNames();

    if (students != null && teachers != null) {
        String studentName = (String) JOptionPane.showInputDialog(this,
            "Select Student", "Assign Teacher",
            JOptionPane.QUESTION_MESSAGE, null, students, students[0]);

        String teacherName = (String) JOptionPane.showInputDialog(this,
            "Select Teacher", "Assign Teacher",
            JOptionPane.QUESTION_MESSAGE, null, teachers, teachers[0]);

        if (studentName != null && teacherName != null) {
            try (Connection conn = getConnection()) {
                int studentId = getStudentId(studentName, conn);
                int teacherId = getTeacherId(teacherName, conn);

                String schedule = JOptionPane.showInputDialog(this, "Enter
Schedule");

                String query = "INSERT INTO assignments (student_id,
teacher_id, schedule) VALUES (?, ?, ?)";

                try (PreparedStatement ps = conn.prepareStatement(query)) {
                    ps.setInt(1, studentId);
                    ps.setInt(2, teacherId);

```

```

        ps.setString(3, schedule);
        ps.executeUpdate();
        JOptionPane.showMessageDialog(this, "Teacher assigned to
student successfully!");
    }
} catch (SQLException e) {
    e.printStackTrace();
}
}
}
}

```

```

// Get list of student names for assignment
private String[] getStudentNames() {
    try (Connection conn = getConnection(); PreparedStatement ps =
conn.prepareStatement("SELECT name FROM students")) {
        ResultSet rs = ps.executeQuery();
        StringBuilder students = new StringBuilder();
        while (rs.next()) {
            students.append(rs.getString("name")).append(",");
        }
        return students.toString().split(",");
    } catch (SQLException e) {
        e.printStackTrace();
        return null;
    }
}

```

```

// Get list of teacher names for assignment
private String[] getTeacherNames() {
    try (Connection conn = getConnection(); PreparedStatement ps =
conn.prepareStatement("SELECT name FROM teachers")) {
        ResultSet rs = ps.executeQuery();

```

```

        StringBuilder teachers = new StringBuilder();
        while (rs.next()) {
            teachers.append(rs.getString("name")).append(",");
        }
        return teachers.toString().split(",");
    } catch (SQLException e) {
        e.printStackTrace();
        return null;
    }
}

```

```

// Get student ID by name
private int getStudentId(String name, Connection conn) throws
SQLException {
    String query = "SELECT id FROM students WHERE name = ?";
    try (PreparedStatement ps = conn.prepareStatement(query)) {
        ps.setString(1, name);
        ResultSet rs = ps.executeQuery();
        if (rs.next()) {
            return rs.getInt("id");
        }
    }
    return -1;
}

```

```

// Get teacher ID by name
private int getTeacherId(String name, Connection conn) throws
SQLException {
    String query = "SELECT id FROM teachers WHERE name = ?";
    try (PreparedStatement ps = conn.prepareStatement(query)) {
        ps.setString(1, name);
        ResultSet rs = ps.executeQuery();
        if (rs.next()) {

```

```

        return rs.getInt("id");
    }
}
return -1;
}

// View full schedule of students
private void viewStudentSchedule() {
    String query = "SELECT students.name AS student_name, teachers.name
AS teacher_name, assignments.schedule " +
        "FROM assignments " +
        "JOIN students ON assignments.student_id = students.id " +
        "JOIN teachers ON assignments.teacher_id = teachers.id";
    try (Connection conn = getConnection(); PreparedStatement ps =
conn.prepareStatement(query)) {
        ResultSet rs = ps.executeQuery();

        StringBuilder scheduleList = new StringBuilder();
        while (rs.next()) {
            scheduleList.append("Student:
").append(rs.getString("student_name"))
                .append(", Teacher: ").append(rs.getString("teacher_name"))
                .append(", Schedule: ").append(rs.getString("schedule"))
                .append("\n");
        }
        if (scheduleList.length() > 0) {
            JOptionPane.showMessageDialog(this, scheduleList.toString(),
"Student Schedules", JOptionPane.INFORMATION_MESSAGE);
        } else {
            JOptionPane.showMessageDialog(this, "No schedules found.", "No
Data", JOptionPane.INFORMATION_MESSAGE);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

```

    }
}

// View full schedule of teachers
private void viewTeacherSchedule() {
    String query = "SELECT teachers.name AS teacher_name, students.name
AS student_name, assignments.schedule " +
        "FROM assignments " +
        "JOIN teachers ON assignments.teacher_id = teachers.id " +
        "JOIN students ON assignments.student_id = students.id";
    try (Connection conn = getConnection(); PreparedStatement ps =
conn.prepareStatement(query)) {
        ResultSet rs = ps.executeQuery();

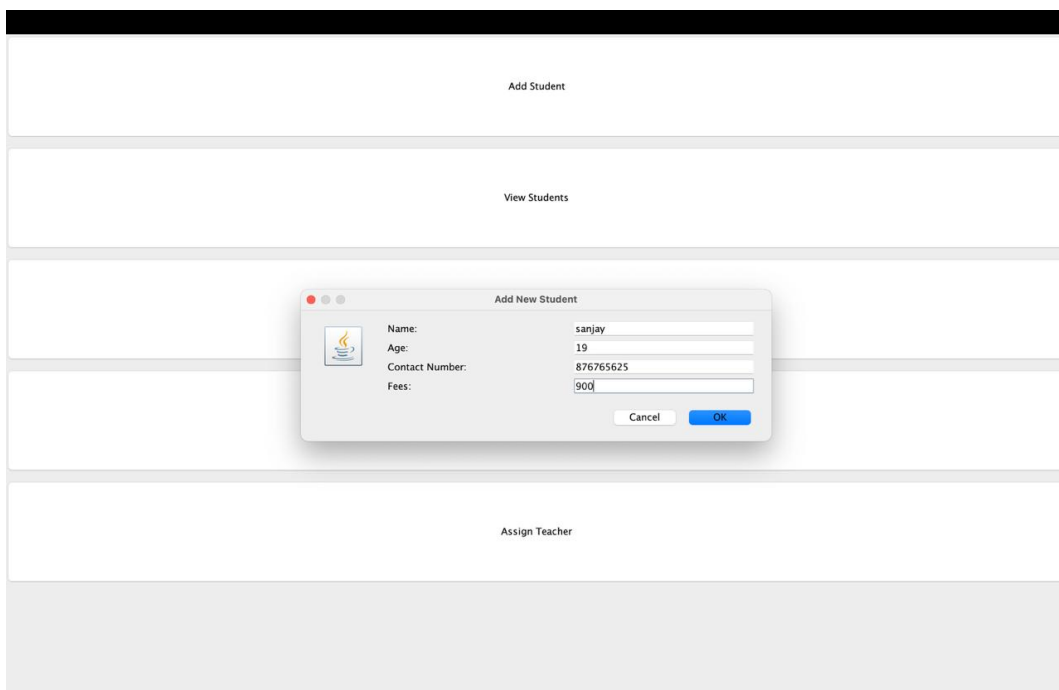
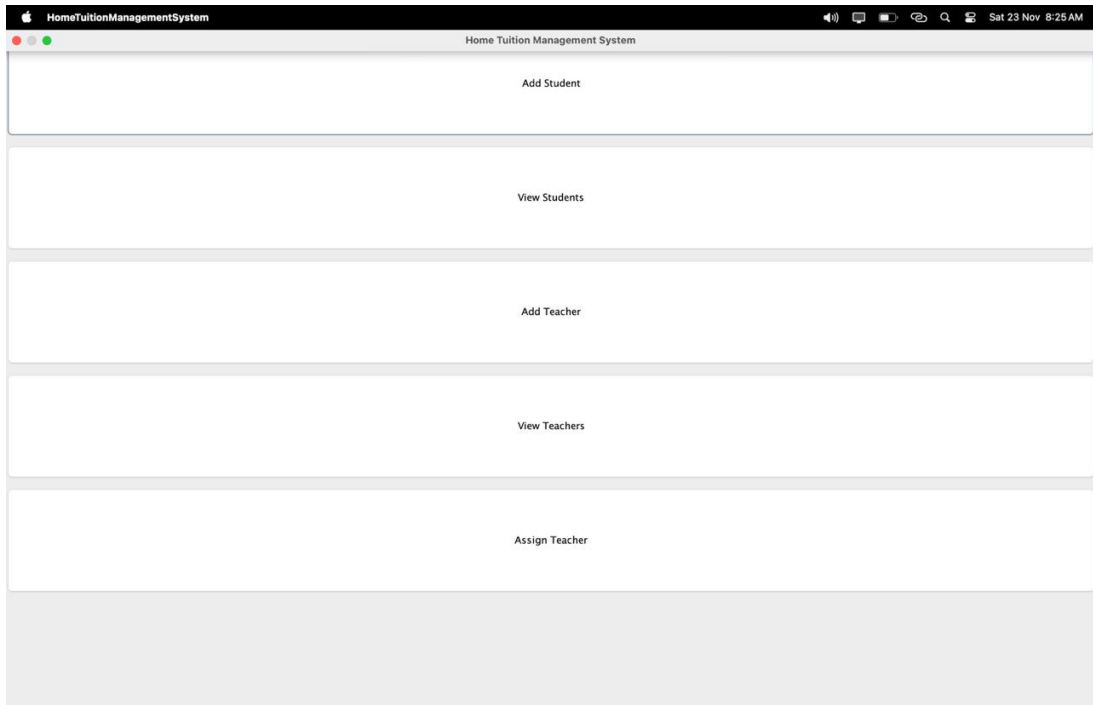
        StringBuilder scheduleList = new StringBuilder();
        while (rs.next()) {
            scheduleList.append("Teacher:
").append(rs.getString("teacher_name"))
                .append(", Student: ").append(rs.getString("student_name"))
                .append(", Schedule: ").append(rs.getString("schedule"))
                .append("\n");
        }
        if (scheduleList.length() > 0) {
            JOptionPane.showMessageDialog(this, scheduleList.toString(),
"Teacher Schedules", JOptionPane.INFORMATION_MESSAGE);
        } else {
            JOptionPane.showMessageDialog(this, "No schedules found.", "No
Data", JOptionPane.INFORMATION_MESSAGE);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

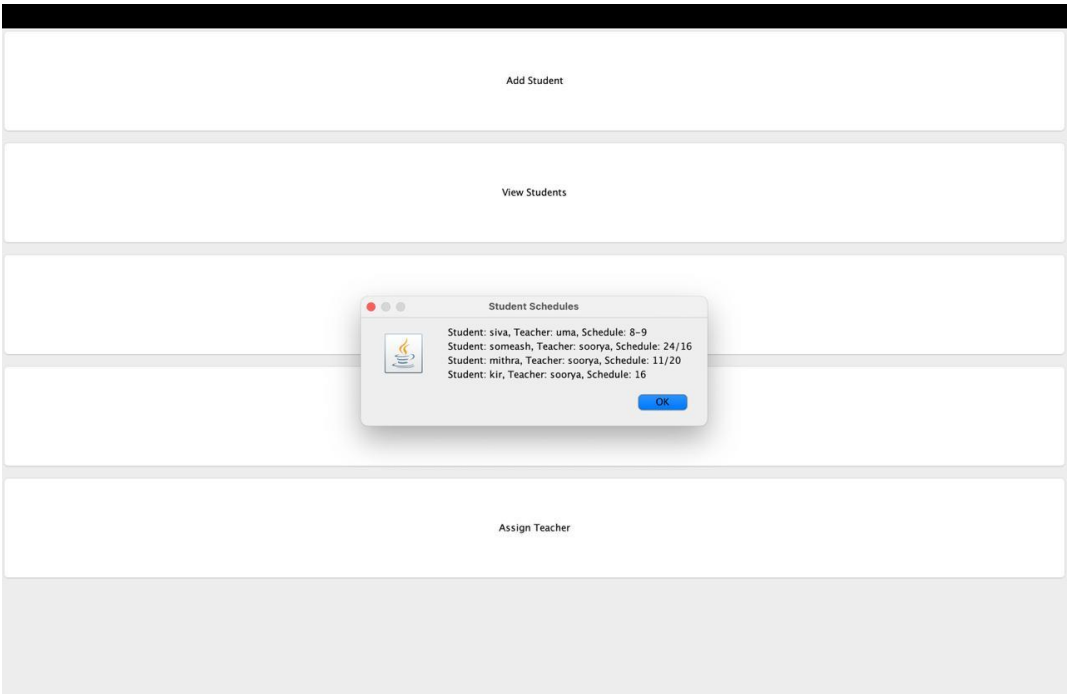
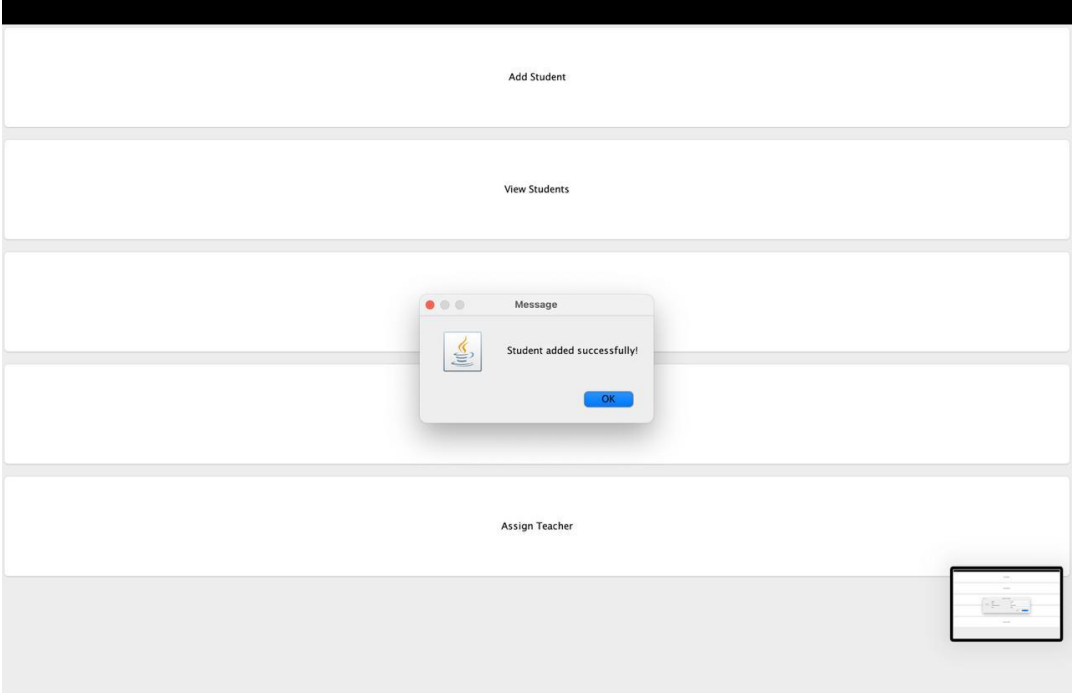
```

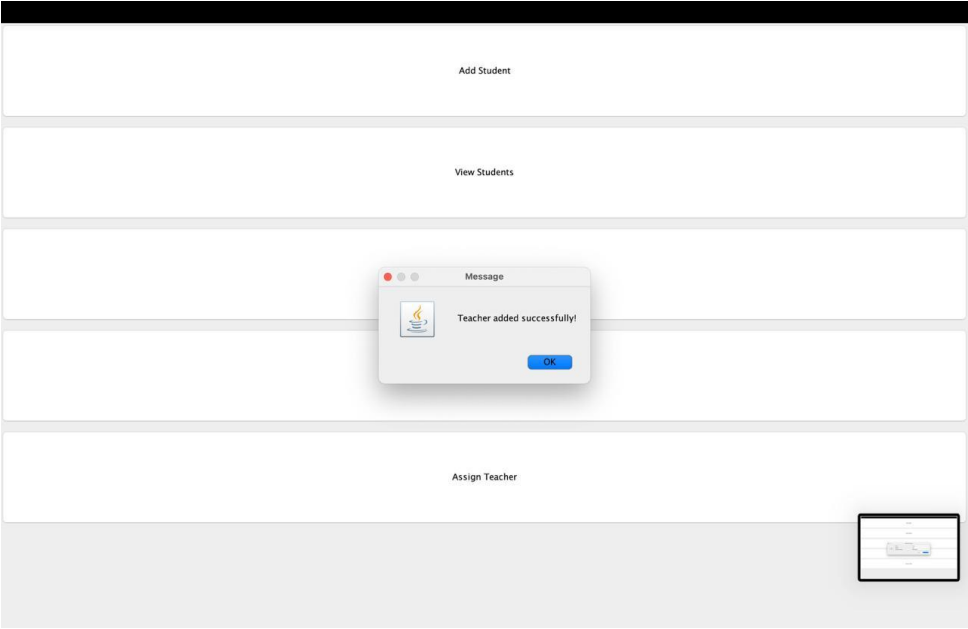
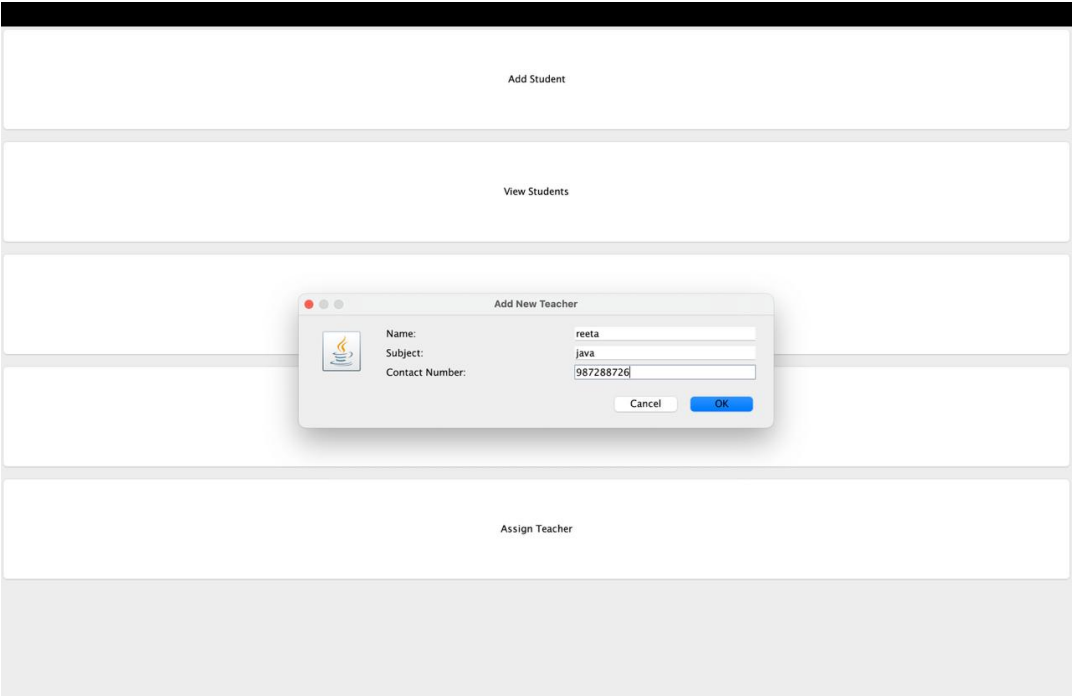


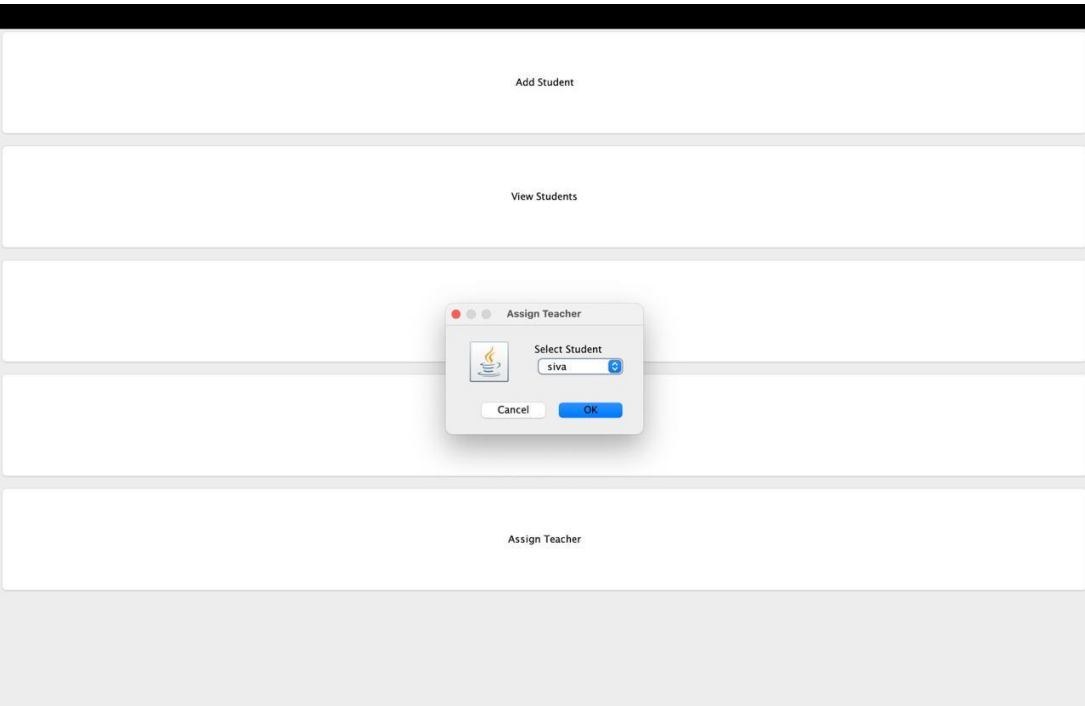
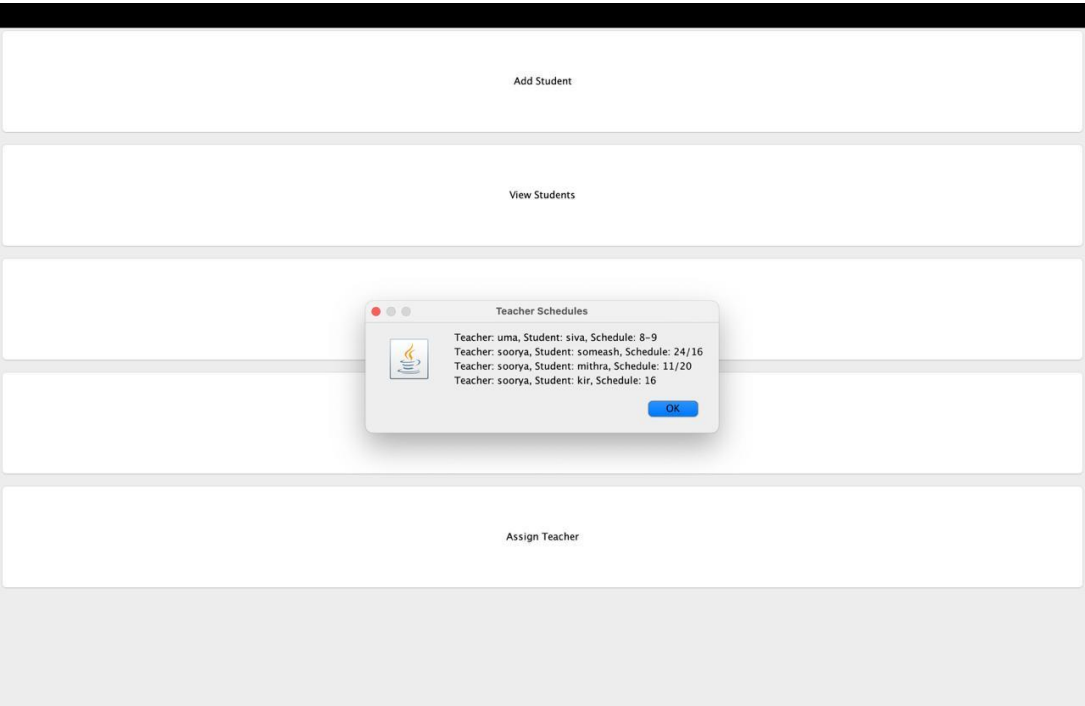
```
// Main method to launch the application
public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        public void run() {
            new HomeTuitionManagementSystem().setVisible(true);
        }
    });
}
```

4. SNAPSHOTS










Add Student

View Students

Assign Teacher



Select Teacher
soorya

Cancel


OK

Assign Teacher

Add Student

View Students

Input

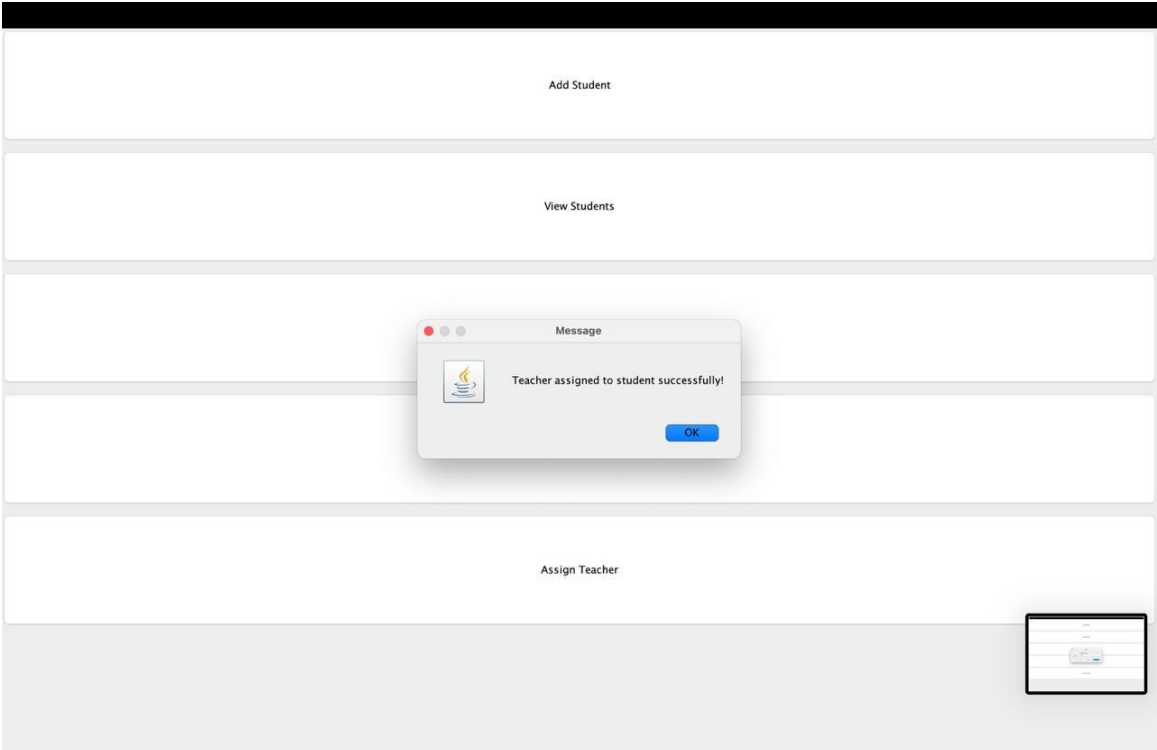


Enter Schedule
11/4

Cancel

OK

Assign Teacher



CONCLUSION

The project offers a comprehensive solution for efficient home tuition management. By automating various tasks, this system enhances the accuracy and efficiency of managing tuition-related operations. It provides a centralized platform for storing, retrieving, and analyzing student and tutor-related data, enabling informed decision-making and streamlined management. The system features a user-friendly interface for easy data input and retrieval, ensuring a smooth experience for users.

REFERENCES

1. <https://www.javatpoint.com/javatutorial>
2. <https://www.wikipedia.org/>
3. <https://www.w3schools.com/sql/>