

SCHOOL MANAGEMENT SYSTEM



A PROJECT REPORT

Submitted by

DHANASRIDHARAN A(2303811710421026)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

NOVEMBER- 2024

**K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY
(AUTONOMOUS)**

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on “ **SCHOOL MANAGEMENT SYSTEM** ” is the bonafide work of **DHANASRIDHARAN A(2303811710421026)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING
Dr.A.DELPHIN CAROLINA RANI, M.E., Ph.D.,
HEAD OF THE DEPARTMENT
PROFESSOR

SIGNATURE

Mrs.A.Delphin Carolina Rani, M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology
(Autonomous)

Samayapuram–621112.

CGB1201-JAVA PROGRAMMING
Mr. M.SARAVANAN, M.E.,
SUPERVISOR
ASSISTANT PROFESSOR

SIGNATURE

Mr.M.Saravanan, M.E.,

SUPERVISOR

ASSISTANT PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology
(Autonomous)

Samayapuram–621112.

Submitted for the viva-voce examination held on 02.12.2024

CGB1201-JAVA PROGRAMMING
Mr.MANJARMANNAN A, M.E.,
INTERNAL EXAMINER
ASSISTANT PROFESSOR

INTERNAL EXAMINER

CGB1201-JAVA PROGRAMMING
Dr.P. SENTHAMILSELVI, M.E., Ph.D.,
EXTERNAL EXAMINER
PROFESSOR
8138-SCE, TRICHY.

EXTERNAL EXAMINER

DECLARATION

I declare that the project report on “ **SCHOOL MANAGEMENT SYSTEM** ” is the result of original work done by us and best of our knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING**.

Signature



DHANASRIDHARAN A

Place: Samayapuram

Date: 02.12.2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution “**K.Ramakrishnan College of Technology (Autonomous)**”, for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN, B.E.**, for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.**, Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.**, Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **MR.M.SARAVANAN, M.E.**, Department of **COMPUTER SCIENCE AND ENGINEERING**, for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

- 1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- 2. Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
- 3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
- 4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

- 5. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
- 6. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
- 7. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
- 8. Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- 9. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- 10. Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- 11. Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
- 12. Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

The School Management System (SMS) is a software application designed to streamline the management of various administrative tasks in a school. Developed using Java programming, this system automates processes such as student enrollment, grade management, attendance tracking, timetable scheduling, and report generation.

It provides a user-friendly interface for administrators, teachers, and students, ensuring efficient data management and easy access to information. The system supports multiple roles with different levels of access, offering features like student registration, fee management, academic performance tracking, and communication between teachers and parents.

By centralizing school operations, the SMS enhances productivity, reduces paperwork, and ensures accurate and real-time data processing. The system is highly scalable and can be customized for different types of educational institutions.

This system enables schools to manage student data, track academic performance, maintain attendance records, and facilitate communication between teachers, students, and parents. Key features of the system include user authentication, class scheduling, grade management, fee tracking, and report generation. The system aims to minimize manual work, enhance efficiency, and improve the quality of service in schools.

The system is designed to be user-friendly and can accommodate multiple user roles, including administrators, teachers, students, and parents. With features like report generation, fee management, and real-time communication, the SMS helps in maintaining accurate records and improving overall school management.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The School Management System (SMS) is a software application designed to streamline the management of various administrative tasks in a school. Developed using Java programming, this system automates processes such as student enrollment, grade management, attendance tracking, timetable scheduling, and report generation. It provides a user-friendly interface for administrators, teachers, and students, ensuring efficient data management and easy access to information. The system supports multiple roles with different levels of access, offering features like student registration, fee management, academic performance tracking, and communication between teachers and parents. By centralizing school operations, the SMS enhances productivity, reduces paperwork, and ensures accurate and real-time data processing. The system is highly scalable and can be customized for different types of educational institutions.	PO1 -3 PO2 -3 PO3 -3 PO4 -3 PO5 -3 PO6 -3 PO7 -3 PO8 -3 PO9 -3 PO10 -3 PO11-3 PO12 -3	PSO1 -3 PSO2 -3 PSO3 -3

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	viii
1	INTRODUCTION	1
	1.1 Objective	1
	1.2 Overview	1
	1.3 Java Programming concepts	2
2	PROJECT METHODOLOGY	3
	2.1 Proposed Work	3
	2.2 Block Diagram	4
3	MODULE DESCRIPTION	5
	3.1 Student Management	5
	3.2 Staff Management	5
	3.3 Course Management	5
	3.4 Time Table Management	5
	3.5 Grade and Examination Management	5
4	CONCLUSION & FUTURE SCOPE	6
	4.1 Conclusion	6
	4.2 Future Scope	6
	REFERENCES	8
	APPENDIX A (SOURCE CODE)	9
	APPENDIX B (SCREENSHOTS)	14

CHAPTER 1

INTRODUCTION

1.1 Objective

1. To Record and manage student personal details such as, (name, age, address, etc.).
2. For Maintaining the teacher profiles namely, (name, qualifications, contact details).
3. To Create, Monitor, and manage courses for the students in the school.
4. For Generating and managing class schedules for students and teachers.
5. Record student grades for exams and assignments and generate reports.

1.2 Overview

A School Management System (SMS) is a software solution designed to streamline and automate various administrative tasks within a school, ensuring efficiency, accuracy, and better management of resources. The system typically manages student information, attendance, grades, timetables, fee structures, library resources, and communication between students, teachers, and administrators.

In Java, the School Management System can be developed using core Java concepts, object-oriented programming principles, and GUI (Graphical User Interface) frameworks such as Swing or JavaFX for a user-friendly interface. Java's rich ecosystem of libraries and frameworks also facilitates database integration (e.g., MySQL, SQLite) to store and manage large datasets.

1.3 Java Programming Concepts

1. **Object – Oriented Programming (OOP) :** You'll create classes to represent entities like Student, Teacher, Course, Classroom, etc. Each class will have attributes (variables) and methods (functions) that define the behavior of that entity.
2. **Control Flow Statement :** Loops are used to iterate over collections like lists of students or teachers. A for loop to display all student names or to calculate the average grade.
3. **Exception Handling :** Try-catch blocks: Exception handling helps to deal with runtime errors such as invalid input, missing files, or database connection errors. This ensures that your program doesn't crash unexpectedly.
4. **File (I/O) (Input/Output) :** Saving student data, grades, or class schedules to a file or reading data from a file can be done using FileReader, BufferedReader, FileWriter, and BufferedWriter.
5. **GUI (GRAPHICAL USER INTERFACE) :** Used to provide a user-friendly interface, you can use Swing or JavaFX to create windows, buttons, tables, forms, and other UI components.

CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work

The proposed work for the School Management System to provide an efficient way for school administrators, teachers, and students to interact with the system, enabling smoother workflows and better organization are,

1. Student Management :

- Student details include personal information, contact details, enrollment number, and grade.
- Admin can add new students to the system, update personal details, and remove students when necessary.

2. Staff Management :

- Teachers can view their class schedules, assignments, and daily tasks.
- Admin can add, remove, or edit teacher details, such as personal information, subject specialization, and assigned classes.

3. Course Management :

- Admin can create new courses (subjects) and assign them to relevant teachers. Teachers can update their course details.

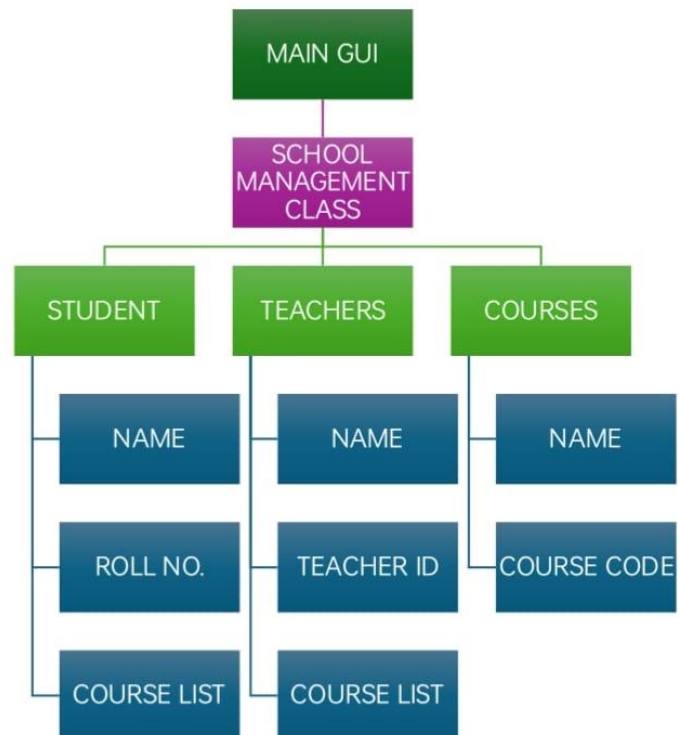
4. Time Table Management :

- Students, teachers, and administrators can access and view schedules for classes and exams.

5. Grade and Examination Management :

- Teachers can input grades for students based on their performance in assignments, tests, and exams.

2.2 Block Diagram



CHAPTER 3

MODULE DESCRIPTION

3.1 Student Management Module

- This module deals with all tasks related to student records, including admission, enrollment, and profile management.
- Add, update, and delete student information

3.2 Staff Management Module

- This module is designed to manage teachers' profiles, assign subjects to classes, and monitor performance.
- Manage teacher qualifications, experience, and salary records

3.3 Course Management Module

- This Module focuses on organizing and managing the academic courses offered by the institution and it allows administrators, teachers, and students to interact with course-related information,

3.4 Time Table Management Module

- This module helps organize and manage classes, subjects, and schedules efficiently and create and view timetables for students and teachers,

3.5 Course And Examination Module

- This module manages exam schedules, grading systems, and the recording of student performance during exams.
- Generate performance reports based on exam results.

CHAPTER 4

CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

In conclusion, the **School Management System in Java** provides a comprehensive, efficient, and secure solution for managing the various administrative and academic functions within a school. By integrating multiple modules such as student management, attendance tracking, grade management, timetable scheduling, fee management, and communication, this system streamlines day-to-day operations, reduces manual workloads, and enhances overall productivity. The use of **Java** ensures robustness, scalability, and ease of maintenance, while leveraging relational databases like **MySQL** allows for secure and efficient data storage. This system not only improves administrative efficiency but also fosters better communication between students, teachers, and parents, ultimately contributing to a more organized and productive educational environment.

The **School Management System** also enhances communication between students, teachers, and parents by providing real-time updates on attendance, grades, and important school events. This promotes transparency, facilitates faster decision-making, and allows stakeholders to stay informed and engaged in the educational process.

4.2 FUTURE SCOPE

The future scope of a School Management System is vast, with opportunities to enhance functionality through technological advancements. By integrating modern technologies such as AI, cloud computing, mobile applications, blockchain, and more, the system can evolve to provide better user experiences, increase automation, improve decision-making, and support modern educational needs. With a focus on scalability, accessibility, and data-driven insights, the future of SMS will shape how educational institutions manage, engage, and support students and staff.

The **School Management System (SMS)** can be continuously enhanced to meet the evolving needs of educational institutions and provide more value to students, teachers,

administrators, and parents. With the rapid advancement in technology, the future scope of an SMS includes innovations that improve efficiency, communication, and data-driven decision-making

APPENDIX A

SOURCECODE

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

// Class representing a Course
class Course {
    String courseName;
    String courseCode;

    public Course(String courseName, String courseCode) {
        this.courseName = courseName;
        this.courseCode = courseCode;
    }

    public String getDetails() {
        return "Course Name: " + courseName + "\nCourse Code: " + courseCode;
    }
}

// Class representing a Student
class Student {
    String name;
    String rollNumber;
    List<Course> courses;

    public Student(String name, String rollNumber) {
        this.name = name;
        this.rollNumber = rollNumber;
        this.courses = new ArrayList<>();
    }

    public void addCourse(Course course) {
        courses.add(course);
    }

    public String getDetails() {
        StringBuilder details = new StringBuilder("Student Name: " + name + "\nRoll Number: " +
rollNumber + "\nCourses Enrolled:\n");
        for (Course course : courses) {
            details.append(course.getDetails()).append("\n");
        }
        return details.toString();
    }
}
```

```
}  
}
```

```
// Class representing a Teacher
```

```
class Teacher {  
    String name;  
    String teacherID;  
    List<Course> courses;  
  
    public Teacher(String name, String teacherID) {  
        this.name = name;  
        this.teacherID = teacherID;  
        this.courses = new ArrayList<>();  
    }  
  
    public void addCourse(Course course) {  
        courses.add(course);  
    }  
  
    public String getDetails() {  
        StringBuilder details = new StringBuilder("Teacher Name: " + name + "\nTeacher ID: " +  
teacherID + "\nCourses Taught:\n");  
        for (Course course : courses) {  
            details.append(course.getDetails()).append("\n");  
        }  
        return details.toString();  
    }  
}
```

```
// Class representing the School Management System
```

```
class SchoolManagementSystem {  
    List<Student> students;  
    List<Teacher> teachers;  
    List<Course> courses;  
  
    public SchoolManagementSystem() {  
        students = new ArrayList<>();  
        teachers = new ArrayList<>();  
        courses = new ArrayList<>();  
    }  
  
    public void addStudent(Student student) {  
        students.add(student);  
    }  
  
    public void addTeacher(Teacher teacher) {  
        teachers.add(teacher);  
    }  
}
```

```

    }

    public void addCourse(Course course) {
        courses.add(course);
    }

    public List<Student> getAllStudents() {
        return students;
    }

    public List<Teacher> getAllTeachers() {
        return teachers;
    }

    public List<Course> getAllCourses() {
        return courses;
    }
}

// Main class with GUI (Swing) implementation
public class Main {
    public static void main(String[] args) {
        // Create the main window
        JFrame frame = new JFrame("School Management System");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(600, 500);
        frame.setLayout(new BorderLayout());

        // Create the school management system
        SchoolManagementSystem system = new SchoolManagementSystem();

        // Create components for GUI
        JPanel panel = new JPanel();
        panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));

        JLabel studentNameLabel = new JLabel("Student Name:");
        JTextField studentNameField = new JTextField(20);
        JLabel rollNumberLabel = new JLabel("Roll Number:");
        JTextField rollNumberField = new JTextField(20);

        JLabel teacherNameLabel = new JLabel("Teacher Name:");
        JTextField teacherNameField = new JTextField(20);
        JLabel teacherIDLabel = new JLabel("Teacher ID:");
        JTextField teacherIDField = new JTextField(20);

        JLabel courseNameLabel = new JLabel("Course Name:");
        JTextField courseNameField = new JTextField(20);

```

```

JLabel courseCodeLabel = new JLabel("Course Code:");
JTextField courseCodeField = new JTextField(20);

JButton addStudentButton = new JButton("Add Student");
JButton addTeacherButton = new JButton("Add Teacher");
JButton addCourseButton = new JButton("Add Course");

JTextArea displayArea = new JTextArea(10, 50);
displayArea.setEditable(false);
JScrollPane scrollPane = new JScrollPane(displayArea);

JButton viewStudentsButton = new JButton("View All Students");
JButton viewTeachersButton = new JButton("View All Teachers");
JButton viewCoursesButton = new JButton("View All Courses");

// Action listener for Add Student button
addStudentButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String name = studentNameField.getText();
        String rollNumber = rollNumberField.getText();
        Student student = new Student(name, rollNumber);
        system.addStudent(student);
        studentNameField.setText("");
        rollNumberField.setText("");
        displayArea.append("Student Added: " + name + " (Roll No: " + rollNumber + ")\n");
    }
});

// Action listener for Add Teacher button
addTeacherButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String name = teacherNameField.getText();
        String teacherID = teacherIDField.getText();
        Teacher teacher = new Teacher(name, teacherID);
        system.addTeacher(teacher);
        teacherNameField.setText("");
        teacherIDField.setText("");
        displayArea.append("Teacher Added: " + name + " (ID: " + teacherID + ")\n");
    }
});

// Action listener for Add Course button
addCourseButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {

```

```

        String name = courseNameField.getText();
        String code = courseCodeField.getText();
        Course course = new Course(name, code);
        system.addCourse(course);
        courseNameField.setText("");
        courseCodeField.setText("");
        displayArea.append("Course Added: " + name + " (Code: " + code + ")\n");
    }
});

```

```

// Action listener for View Students button
viewStudentsButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        displayArea.setText("All Students:\n");
        for (Student student : system.getAllStudents()) {
            displayArea.append(student.getDetails() + "\n");
        }
    }
});

```

```

// Action listener for View Teachers button
viewTeachersButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        displayArea.setText("All Teachers:\n");
        for (Teacher teacher : system.getAllTeachers()) {
            displayArea.append(teacher.getDetails() + "\n");
        }
    }
});

```

```

// Action listener for View Courses button
viewCoursesButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        displayArea.setText("All Courses:\n");
        for (Course course : system.getAllCourses()) {
            displayArea.append(course.getDetails() + "\n");
        }
    }
});

```

```

// Add components to the panel
panel.add(studentNameLabel);
panel.add(studentNameField);
panel.add(rollNumberLabel);

```

```

panel.add(rollNumberField);
panel.add(addStudentButton);

panel.add(teacherNameLabel);
panel.add(teacherNameField);
panel.add(teacherIDLabel);
panel.add(teacherIDField);
panel.add(addTeacherButton);

panel.add(courseNameLabel);
panel.add(courseNameField);
panel.add(courseCodeLabel);
panel.add(courseCodeField);
panel.add(addCourseButton);

panel.add(viewStudentsButton);
panel.add(viewTeachersButton);
panel.add(viewCoursesButton);
panel.add(scrollPane);

// Add panel to the frame
frame.add(panel, BorderLayout.CENTER);

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

```

APPENDIX B

(SCREENSHOTS)




All Students:

Student Name: dhanush

Roll Number: 01

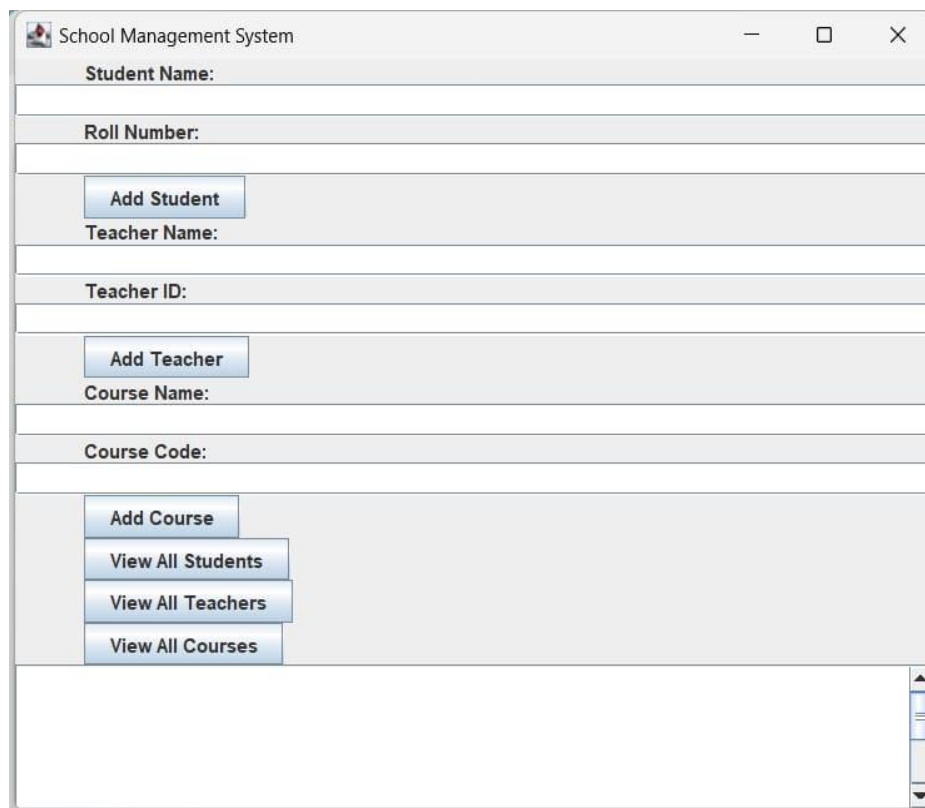
Courses Enrolled:



All Courses:

Course Name: java programming

Course Code: gea1202



School Management System

Student Name:

Roll Number:

Add Student

Teacher Name:

Teacher ID:

Add Teacher

Course Name:

Course Code:

Add Course

View All Students

View All Teachers

View All Courses

School Management System

Student Name:

dhanush

Roll Number:

01

Add Student

Teacher Name:

elumalai

Teacher ID:

123

Add Teacher

Course Name:

java programming

Course Code:

gea1202

Add Course

View All Students

View All Teachers

View All Courses

Student Added: dhanush (Roll No: 01)

REFERENCES

1. Steve Holmes, E. and Reachel Adams, J. (2019) ‘ Management System in Java – MSJ ’, Oswald – Journal of Computing Vol.31, No.7, pp.124 – 168.
2. Benjamin R.W. and Kristina, C. (2020) ‘ Applications of Java System for Management in University Theory ’, Micah Moore, N., Vol.13, pp.87 – 95.
3. Jin Sakai and Nathan George, N.D. (2021) ‘Open World Java and it’s Applications ’, North California, pp. 1234 – 1241.