MINI PROJECT REPORT

ON

School Management System

Submitted by

NAME: H.HARISH

ROLL NO: NC.SC.U4CSE24149

For

23CSE111- Object Oriented Programming

II Semester

B.Tech. CSE

School of Computing

Amrita Vishwa Vidyapeetham, Nagercoil

Index

School Management System

---

1. Introduction

The School Management System is designed to manage the operations of a school efficiently. It handles student and teacher information digitally, replacing the traditional paper-based system. This application allows easy access, addition, and viewing of records, enhancing productivity and reducing errors.

---

2. Problem Statement

Manual record-keeping in schools often results in misplaced data, slower information retrieval, and increased administrative workload. There is a strong need for an automated system that can manage student and teacher data quickly and reliably, improving the overall administration of educational institutions.

---

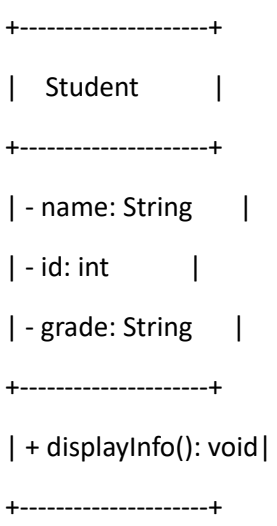
3. Objectives

- To develop a simple, console-based School Management System.
- To provide functionality to add and view students and teachers.
- To reduce manual paperwork and data management time.
- To ensure accurate and secure record-keeping.
- To create a modular system for easy future upgrades.

---

4. UML Diagrams of the Project

4.1 Class Diagram



```
+---------------------+
|    Teacher          |
+---------------------+
| - name: String      |
| - id: int           |
| - subject: String   |
+---------------------+
| + displayInfo(): void|
+---------------------+


+------------------------------+
|   SchoolManagementSystem     |
+------------------------------+
| - students: ArrayList<Student> |
| - teachers: ArrayList<Teacher> |
+------------------------------+
| + main(args: String[]): void  |
| + addStudent(): void          |
| + addTeacher(): void          |
| + displayStudents(): void     |
| + displayTeachers(): void     |
+------------------------------+
```

4.2 Use-Case Diagram

```
[User] ---> (Add Student)
[User] ---> (Add Teacher)
[User] ---> (View Students)
[User] ---> (View Teachers)
[User] ---> (Exit System)
```

- **Actor**: User (Admin or Staff)

- **Use Cases**: Add, View, Exit

4.3 Activity Diagram

Start

|

v

Display Menu

|

v

User Chooses Option

|------> [Add Student] --> Save Student --> Back to Menu

|------> [Add Teacher] --> Save Teacher --> Back to Menu

|------> [Display Students] --> Show List --> Back to Menu

|------> [Display Teachers] --> Show List --> Back to Menu

|------> [Exit] --> End

4.4 Sequence Diagram

User -> System: Display Menu

User -> System: Selects "Add Student"

System -> User: Enter Name, ID, Grade

User -> System: Provides Data

System: Saves Student

System: Display Success Message

System -> User: Back to Menu

---

5. Modules of the Project

The project consists of the following major modules:

- **Student Management Module**
    - Add new student records.
    - View existing student details.
- **Teacher Management Module**

- Add new teacher records.
- View existing teacher details.

Each module is managed using dedicated classes and methods.

---

6. Code

```java
import java.util.ArrayList;
import java.util.Scanner;

// Model for Student
class Student {
    private String name;
    private int id;
    private String grade;

    public Student(String name, int id, String grade) {
        this.name = name;
        this.id = id;
        this.grade = grade;
    }

    public void displayInfo() {
        System.out.println("Student ID: " + id + ", Name: " + name + ", Grade: " + grade);
    }
}

// Model for Teacher
class Teacher {
    private String name;
    private int id;
    private String subject;

    public Teacher(String name, int id, String subject) {
        this.name = name;
        this.id = id;
        this.subject = subject;
    }

    public void displayInfo() {
        System.out.println("Teacher ID: " + id + ", Name: " + name + ", Subject: " + subject);
    }
}
```

```java
// School Management System
public class SchoolManagementSystem {
    private static ArrayList<Student> students = new ArrayList<>();
    private static ArrayList<Teacher> teachers = new ArrayList<>();
    private static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        int choice;
        do {
            System.out.println("\n=== School Management System ===");
            System.out.println("1. Add Student");
            System.out.println("2. Add Teacher");
            System.out.println("3. Display Students");
            System.out.println("4. Display Teachers");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();
            scanner.nextLine(); // consume newline

            switch (choice) {
                case 1:
                    addStudent();
                    break;
                case 2:
                    addTeacher();
                    break;
                case 3:
                    displayStudents();
                    break;
                case 4:
                    displayTeachers();
                    break;
                case 5:
                    System.out.println("Exiting the system...");
                    break;
                default:
```

```
                    System.out.println("Invalid choice. Try again.");
                }
            } while (choice != 5);
        }

        private static void addStudent() {
            System.out.print("Enter Student Name: ");
            String name = scanner.nextLine();
            System.out.print("Enter Student ID: ");
            int id = scanner.nextInt();
            scanner.nextLine(); // consume newline
            System.out.print("Enter Student Grade: ");
            String grade = scanner.nextLine();

            students.add(new Student(name, id, grade));
            System.out.println("Student added successfully!");
        }

        private static void addTeacher() {
            System.out.print("Enter Teacher Name: ");
            String name = scanner.nextLine();
            System.out.print("Enter Teacher ID: ");
            int id = scanner.nextInt();
            scanner.nextLine(); // consume newline
            System.out.print("Enter Subject: ");
            String subject = scanner.nextLine();

            teachers.add(new Teacher(name, id, subject));
            System.out.println("Teacher added successfully!");
        }

        private static void displayStudents() {
            if (students.isEmpty()) {
                System.out.println("No students to display.");
            } else {
                System.out.println("\nList of Students:");
                for (Student s : students) {
                    s.displayInfo();
                }
            }
        }

        private static void displayTeachers() {
            if (teachers.isEmpty()) {
                System.out.println("No teachers to display.");
            } else {
                System.out.println("\nList of Teachers:");
                for (Teacher t : teachers) {
                    t.displayInfo();
                }
            }
        }
    }
```

7. Output Screenshots:

(i) Adding a Student:

```
Enter your choice: 1
Enter Student Name: Alice Johnson
Enter Student ID: 101
Enter Student Grade: 8th
Student added successfully!
```

(ii) Displaying Students:

```
List of Students:
Student ID: 101, Name: Alice Johnson, Grade: 8th
```

(iii) Adding a Teacher:

```
Enter your choice: 2
Enter Teacher Name: Mr. Smith
Enter Teacher ID: 201
Enter Subject: Mathematics
Teacher added successfully!
```

(iv) Displaying Teachers:

```
List of Teachers:
Teacher ID: 201, Name: Mr. Smith, Subject: Mathematics
```

---

8. Application of the Project

- To manage basic school administration digitally.
- For small schools, coaching centers, or tutorials to manage data.
- As a foundational system for larger school ERP development.
- Useful for demonstrating basic object-oriented and console programming.
- Can be enhanced into a full-fledged online school management system.

---

9. Limitations of the Project

- No database: data is lost once the program ends.
- No login/authentication feature.
- No support for other school operations like attendance, fees, exams.
- Console-based only; no graphical user interface.
- Not suitable for very large institutions without major upgrades.

---

10. Bibliography

- Java: The Complete Reference – Herbert Schildt
- Oracle Java Official Documentation
- TutorialsPoint – Java Programming Tutorials
- GeeksforGeeks – Java OOPs Concepts
- Stack Overflow – Java Coding Solutions and Discussions

12.GitHub link of the project

https://github.com/Harish24149/java-pro/blob/main/java%20project.java