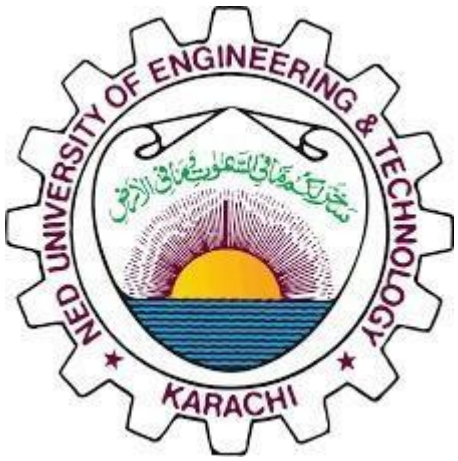


NED UNIVERSITY OF ENGINEERING AND TECHNOLOGY

Formal Methods in Software Engineering

Formal Specification Document for

"**Student Attendance Tracking System**"

Group

- Bakhtiar Ahmed (SE-21029)
- Fawad Tariq (SE-21050)

Table of Contents

1. Problem Statement: Student Attendance Tracking System 2
2. 4 + 1 Architectural View 3
   - 2.1. Logical View 3
   - 2.2. Process View 4
   - 2.3. Physical View 4
   - 2.4. Development View 5
   - 2.5. +1 Scenarios 5
3. The VDM specification of AttencdanceTracker System 6
4. Java Implementation 8
5. Testing Class 12

GitHub Repository: Student Attendance Tracking System

Student Attendance Tracking System

1. Problem Statement: Student Attendance Tracking System

Background: A student attendance tracking system is a critical component in educational institutions, facilitating efficient monitoring and management of students' class attendance. It's crucial for academic progress tracking and institutional compliance.

Requirements: Design and implement software for a student attendance tracking system. The system should:

- Allow the addition or update of attendance records for individual students.
- Display all student records, including their attendance percentages, classes attended, and classes held.
- Ensure accuracy in attendance percentage calculation and record management.

Specifications:

- The system should store attendance records for each student, including their name, roll number (unique identifier), total classes held, and total classes attended.
- Calculate the attendance percentage based on the classes attended and classes held.
- Allow the addition or updating of attendance records by specifying the student's name, roll number, total classes held, and total classes attended.
- Display all student records, including their attendance percentages, classes attended, and classes held.
- Implement proper error handling for invalid inputs and ensure data integrity.

Functionalities:

1) **Add or Update Student Record:**

- Input: Student name, roll number, total classes held, total classes attended.
- Result: The system updates or adds the student's attendance record.

2) **Display Student Attendance Records:**

- Input: Roll number of the student.

- Output: Individual student record with attendance details.

3) **Calculate Percentage:**

-Input: Number of classes attended and Number of classes held.

- Output: Percentage of student's attendance.

Constraints:

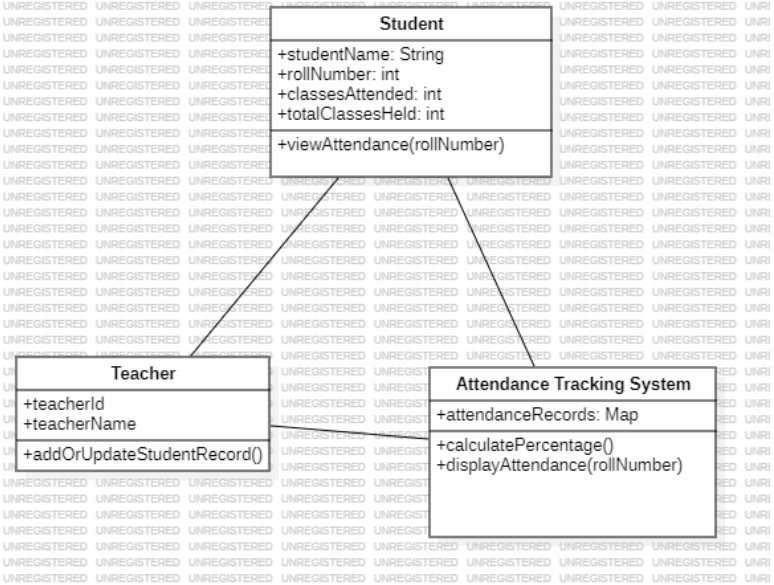
• Ensure uniqueness of roll numbers to prevent conflicts in student records.

• Validate inputs to maintain accurate attendance data.

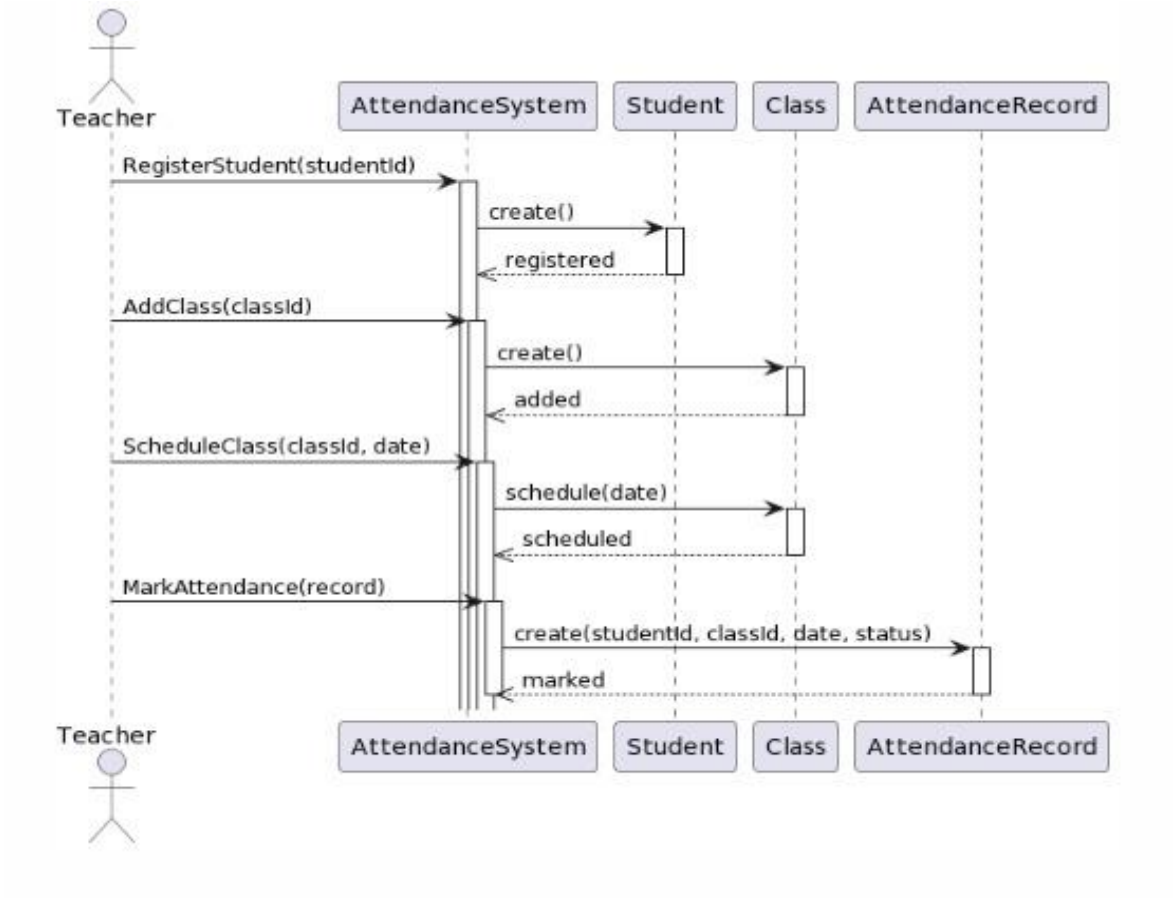
• Implement error handling for scenarios like incorrect roll numbers or invalid attendance values.

2. 4+1 View Architecture:

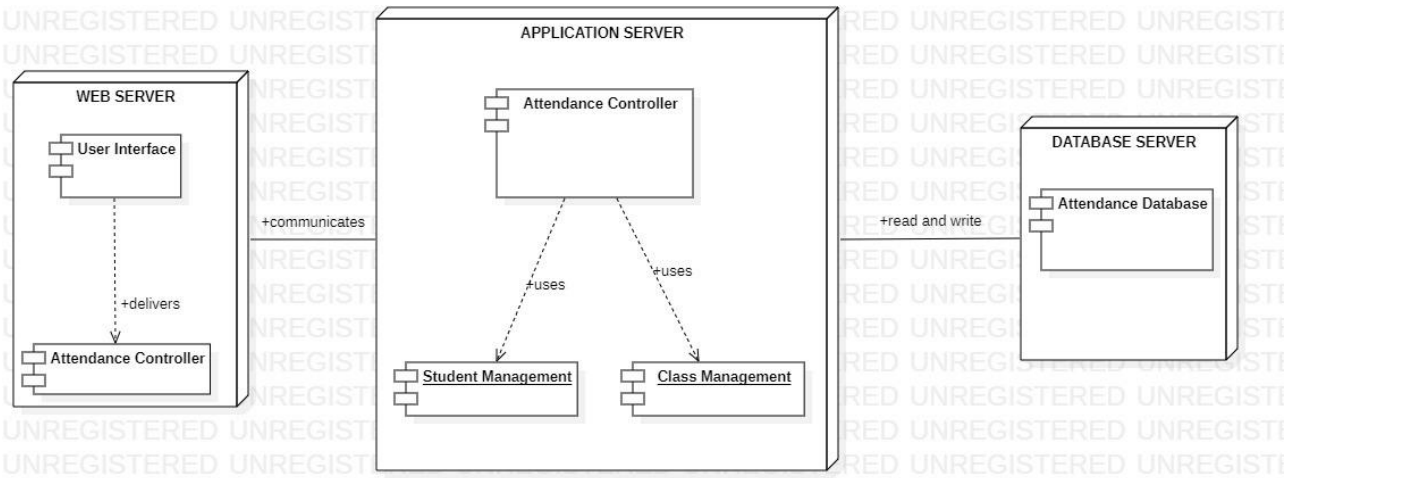
2.1. Logical View:



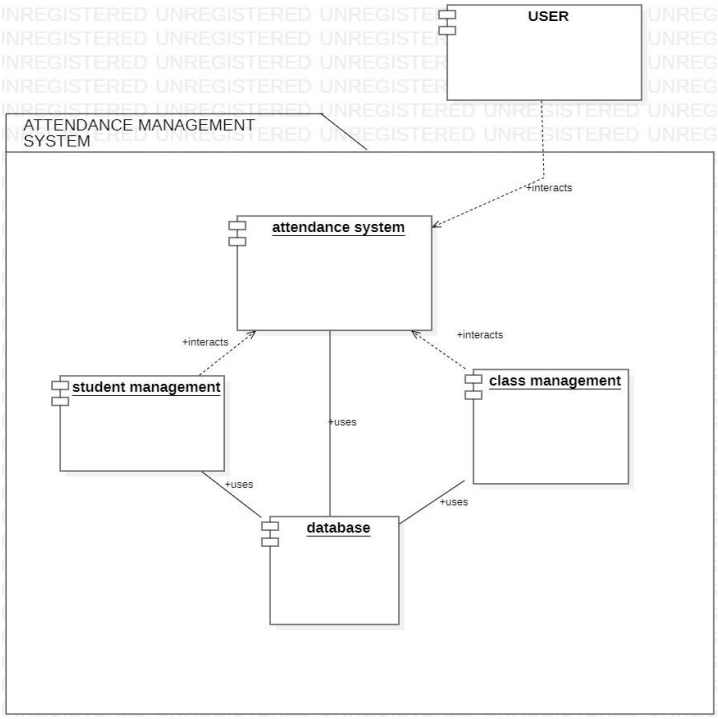
2.2. Process View:



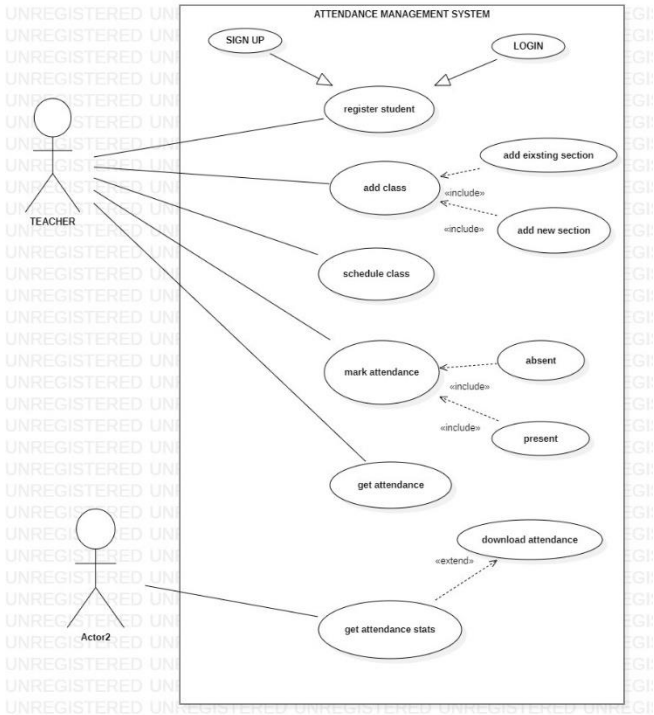
2.3. Physical View:



2.4. Development View:



2.5. +1 Scenario:



3. The VDM specification of *AttencdanceTracker* System

types
Name = seq of char;

state *AttendanceTracker* **of**
studentName : [Name]
rollNumber : [ℤ]
totalClassesHeld : **[**ℤ**]**
classesAttended : **[**ℤ**]**

-- Total Classes Held and Classes Attended must not be less than zero and Classes Held must be greater than or equal to Classes Attended
inv mk-AttendanceTracker (t, c) (inRange(t) ∨ = **nil**) ∧ (inRange(c) ∨ = **nil**) ∧ (c ≤ t)

-- totalClassesHeld and classesAttended are undefined when the system is initialized
init mk-AttendanceTracker (t, c) t = **nil** ∧ c = **nil**

end

functions
inRange:(val: totalClassesHeld) result: B
pre True
post result ⇔ 0 ≤ val ;

-- This function will calculate the attendance percentage.
calculatePercentage:(val1: totalClassesHeld, val2: ClassesAttended) result: B
pre inRange(val1) ∧ inRange(val2)
post result ⇔ (val2 / val1) * 100

operations

-- This operation will allow the user to add or update the student attendance records.
addOrUpdateStudentRecord(stName : Name, rollNo : **[**ℤ**]**, classHeld: **[**ℤ**]**, classAttended: **[**ℤ**]**)
ext wr studentName : Name
rollNumber: **[**ℤ**]**
totalClassesHeld: **[**ℤ**]**
classesAttended : **[**ℤ**]**
pre inRange(classHeld) ∧ inRange(classAttended)
post studentName = stName ∧ rollNumber = rollNo ∧ totalClassesHeld = classHeld ∧ classesAttended = classAttended

-- This operation will allow the user to see the attendance record.

getAttendanceRecord ( ) stName : [Name], rollNo : **[**ℤ**]**, classHeld: **[**ℤ**]**, classAttended: **[**ℤ**]**

ext rd rollNumber : **[**ℤ**]**

totalClassesHeld : **[**ℤ**]**

classesAttended : **[**ℤ**]**

pre rollNo = rollNumber ∧ rollNumer != **nil**

post stName = *getStudentName( )* ∧ classHeld = *getClassesHeld( )* ∧ classAttended = *getClassesAttended( )* ∧ *calculatePercentage*(totalClassesHeld, classesAttended)

-- This operation will fetch the name of the student.

getStudentName ( ) stName : [Name]

ext rd studentName : [Name]

rollNumber : **[**ℤ**]**

pre studentName != **nil** ∧ rollNumber != **nil** ∧ inRange(rollNumber)

post stName = studentName

-- This operation will fetch the total number of classes held.

getClassesHeld ( ) classHeld: **[**ℤ**]**

ext rd totalClassesHeld: **[**ℤ**]**

rollNumber : **[**ℤ**]**

pre rollNumber != **nil** ∧ inRange(rollNumber)

post classHeld = totalClassesHeld

-- This operation will fetch the total number of classes attended by the student.

getClassesAttended ( ) classAttended: **[**ℤ**]**

ext rd classesAttended : **[**ℤ**]**

rollNumber : **[**ℤ**]**

pre rollNumber != **nil** ∧ inRange(rollNumber)

post classAttended = classesAttended

4. Java Implementation

Code:

```
import java.util.*;

class Student {
    private String studentName;
    private int rollNumber;
    private int totalClassesHeld;
    private int totalClassesAttended;

    public Student(String studentName, int rollNumber, int totalClassesHeld, int
totalClassesAttended) {
        this.studentName = studentName;
        this.rollNumber = rollNumber;
        this.totalClassesHeld = totalClassesHeld;
        this.totalClassesAttended = totalClassesAttended;
    }

    public void updateAttendance(int totalClassesHeld, int totalClassesAttended) {
        this.totalClassesHeld = totalClassesHeld;
        this.totalClassesAttended = totalClassesAttended;
    }

    public String getStudentName() {
        return studentName;
    }

    public int getRollNumber() {
        return rollNumber;
    }

    public int getTotalClassesHeld() {
        return totalClassesHeld;
    }

    public int getTotalClassesAttended() {
        return totalClassesAttended;
    }

    public double getAttendancePercentage() {
         double percentage = ((double) totalClassesAttended / totalClassesHeld *
100);
        return Math.round(percentage * 100.0) / 100.0;
    }
}

class StudentAttendanceSystem {
    private Map<Integer, Student> attendanceRecords; // Map<RollNumber, Student>

    public StudentAttendanceSystem() {
```

```
        attendanceRecords = new HashMap<>();
    }

    public void addOrUpdateStudentRecord(String studentName, int rollNumber, int
totalClassesHeld, int totalClassesAttended) {
    if (totalClassesAttended > totalClassesHeld) {
        System.out.println("Error: Attended classes cannot be greater than total
classes held. Please enter again.");
        return;
    }

    if (attendanceRecords.containsKey(rollNumber)) {
        Student existingStudent = attendanceRecords.get(rollNumber);
        if (!existingStudent.getStudentName().equals(studentName)) {
            System.out.println("Error: A record with the same roll number but
different name already exists.");
            return;
        }
        existingStudent.updateAttendance(totalClassesHeld, totalClassesAttended);
        System.out.println("Record updated successfully!");
    } else {
        Student newStudent = new Student(studentName, rollNumber,
totalClassesHeld, totalClassesAttended);
        attendanceRecords.put(rollNumber, newStudent);
        System.out.println("Record added successfully!");
    }
}
    public void displayAllAttendance() {
        if (attendanceRecords.isEmpty()) {
            System.out.println("No records found.");
            return;
        }

        System.out.println("All Student Records:");
        for (Map.Entry<Integer, Student> entry : attendanceRecords.entrySet()) {
            Student student = entry.getValue();
            System.out.println("Roll Number: " + student.getRollNumber());
            System.out.println("\nStudent Name: " + student.getStudentName());
            System.out.println("\nAttendance Percentage: " +
student.getAttendancePercentage() + "%");
            System.out.println("\nClasses Attended: " +
student.getTotalClassesAttended());
            System.out.println("\nClasses Held: " +
student.getTotalClassesHeld());
            System.out.println("\n--------------------------");
        }
    }

    public void displayAttendance(int rollNumber) {
        if (!attendanceRecords.containsKey(rollNumber)) {
            System.out.println("Student not found.");
            return;
        }
```

```
        Student student = attendanceRecords.get(rollNumber);
        System.out.println("Attendance for Student " + student.getStudentName() +
":");
        System.out.println("Attendance Percentage: " +
student.getAttendancePercentage() + "%");
        System.out.println("Classes Attended: " +
student.getTotalClassesAttended());
        System.out.println("Classes Held: " + student.getTotalClassesHeld());
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        StudentAttendanceSystem attendanceSystem = new StudentAttendanceSystem();

        while (true) {
            System.out.println("\nChoose an option:");
            System.out.println("1. Add or update a student record");
            System.out.println("2. View all student records");
            System.out.println("3. View an individual student record");
            System.out.println("4. Exit");
            System.out.print("Enter your choice: ");

            String input = scanner.nextLine().toLowerCase();
            if (input.equals("1")) {
                System.out.print("Enter student name: ");
                String studentName = scanner.nextLine().toLowerCase();

                int rollNumber = 0;
                int totalClassesHeld = 0;
                int totalClassesAttended = 0;

                while (true) {
                    try {
                        System.out.print("Enter roll number: ");
                        rollNumber = scanner.nextInt();
                        System.out.print("Enter total classes held: ");
                        totalClassesHeld = scanner.nextInt();
                        System.out.print("Enter total classes attended: ");
                        totalClassesAttended = scanner.nextInt();
                        scanner.nextLine(); // Consume the newline character
                        if (totalClassesAttended > totalClassesHeld) {
                            System.out.println("Error: Attended classes cannot be
greater than total classes held. Please enter again.");
                            continue;
                        }
                        break;
                    } catch (InputMismatchException e) {
                        System.out.println("Invalid input. Please enter integers
for roll number and total classes.");
                        scanner.nextLine(); // Clear the input buffer
                    }
                }
                attendanceSystem.addOrUpdateStudentRecord(studentName, rollNumber,
```

```
totalClassesHeld, totalClassesAttended);
            } else if (input. equals("2")) {
                attendanceSystem.displayAllAttendance();
            } else if (input.equals("3")) {
                System.out.print("Enter roll number to view attendance: ");
                int rollNumber = 0;
                try {
                    rollNumber = scanner.nextInt();
                    scanner.nextLine(); // Consume the newline character
                } catch (InputMismatchException e) {
                    System.out.println("Invalid input. Please enter an integer for roll
number.");
                    scanner.nextLine(); // Clear the input buffer
                    continue;
                }
                attendanceSystem.displayAttendance(rollNumber);
            } else if (input.equals("4")) {
                System.out.println("Exiting...");
                break;
            } else {
                System.out.println("Invalid choice. Please enter 1, 2, 3, or 4.");
            }
        }

        scanner.close();
    }
}
```

Output:

```
java -cp /tmp/S5VrvIV2Lx StudentAttendanceSystem
Choose an option:
1. Add or update a student record
2. View all student records
3. View an individual student record
4. Exit
Enter your choice: 1
Enter student name: Fawad
Enter roll number: 50
Enter total classes held: 42
Enter total classes attended: 40
Choose an option:
1. Add or update a student record
2. View all student records
3. View an individual student record
4. Exit
Enter your choice: 3
Enter roll number to view attendance: 50
Attendance for Student fawad:
Attendance Percentage: 95.24%
Classes Attended: 40
Classes Held: 42
Choose an option:
1. Add or update a student record
2. View all student records
3. View an individual student record
4. Exit
Enter your choice: 4
Exiting...
```

5. Testing Class

Code:

```
import java.util.*;


public class StudentAttendanceTest {


    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        StudentAttendanceSystem attendanceSystem = new StudentAttendanceSystem();


        while (true) {
            System.out.println("\nChoose a test case:");
            System.out.println("1. Add a new student record");
            System.out.println("2. Update an existing student record ");
            System.out.println("3. Add a record with invalid attendance");
            System.out.println("4. Display all attendance records");
            System.out.println("5. Display an individual student record");
            System.out.println("6. Display non-existing student record");
            System.out.println("7. Add multiple student records");
            System.out.println("8. Attempt to update non-existing student record");
            System.out.println("9. Add a record with negative total classes held");
            System.out.println("10. Add a record with negative total classes attended");
            System.out.println("11. Add a record with total classes attended greater than
total classes held");
            System.out.println("12. Exit the system");
            System.out.print("Enter your choice: ");


            int choice = scanner.nextInt();
            scanner.nextLine();


            switch (choice) {
```

```
                case 1:
                    attendanceSystem.addOrUpdateStudentRecord("Fawad", 101, 10, 8);
                    break;
                case 2:
                    attendanceSystem.addOrUpdateStudentRecord("Bakhtiar", 101, 12, 9);
                    break;
                case 3:
                    attendanceSystem.addOrUpdateStudentRecord("Anas", 102, 8, 10);
                    break;
                case 4:
                    attendanceSystem.displayAllAttendance();
                    break;
                case 5:
                    attendanceSystem.displayAttendance(101); // Fawad's roll number
                    break;
                case 6:
                    attendanceSystem.displayAttendance(103); // Non-existing roll number
                    break;
                case 7:
                    attendanceSystem.addOrUpdateStudentRecord("Alice", 103, 8, 8);
                    attendanceSystem.addOrUpdateStudentRecord("Bob", 104, 9, 6);
                    break;
                case 8:
                    attendanceSystem.addOrUpdateStudentRecord("Non-existing student", 111,
5, 3);
                    break;
                case 9:
                    attendanceSystem.addOrUpdateStudentRecord("Negative Classes", 113, -5,
3);
                    break;
                case 10:
                    attendanceSystem.addOrUpdateStudentRecord("Negative Attended", 114, 5,
-3);
```

```
                    break;
                case 11:
                    attendanceSystem.addOrUpdateStudentRecord("Greater Attendance", 115,
10, 15);
                    break;
                case 12:
                    System.out.println("Exiting...");
                    scanner.close();
                    System.exit(0);
                    break;
                default:
                    break;
            }
        }
    }
}
```

`

Test Cases:

S No.	Test Cases	Input	Expected Output	Status
1.	Adding a new student record	"Fawad", 101, 10, 8	Record added successfully	Passed
2.	Updating an existing student record	"Fawad", 101, 12, 9	Record updated successfully	Passed
3.	Adding a record with invalid attendance	"Anas", 102, 8, 10	Error: Attended classes cannot be greater than total classes held	Passed
4.	Display all attendance records	Choose option 2	Display all existing student records and attendance	Passed
5.	Display an individual student record	Choose option 3 and input Fawad's roll number	Display attendance details for Fawad	Passed
6.	Display non-existing student record	Choose option 3 and input non-existing roll number	Display "Student not found." message	Passed
7.	Add multiple student records	Multiple records with different names and data	Records added successfully	Passed
8.	Attempt to update non-existing student record	"Non-existing student", 111, 5, 3	Error: No record found for the given roll number	Passed
9.	Add a record with negative total classes held	"Negative Classes", 113, -5, 3	Error: Total classes held cannot be negative	Passed
10.	Add a record with negative total classes attended	"Negative Attended", 114, 5, -3	Error: Total classes attended cannot be negative	Passed
11.	Add a record with total classes attended greater than total classes held	"", 117, 10, 15	Error: Total classes attended cannot be greater than total classes held	Passed
12.	Exit the system	Choose option 4	Exit the system	Passed

`

Output:

```
Choose a test case:
1. Add a new student record
2. Update an existing student record
3. Add a record with invalid attendance
4. Display all attendance records
5. Display an individual student record
6. Display non-existing student record
7. Add multiple student records
8. Attempt to update non-existing student record
9. Add a record with negative total classes held
10. Add a record with negative total classes attended
11. Add a record with classes attended greater than total classes held
12. Exit the system
Enter your choice: 1
Record added successfully

Enter your choice: 2
Record updated successfully

Enter your choice: 3
Error: Attended classes cannot be greater than total classes held

Enter your choice: 4
All Student Records:
Roll Number: 101
Student Name: Fawad
Attendance Percentage: 80.0%
Classes Attended: 8
Classes Held: 10
--------------------------

Enter your choice: 5
Attendance for Student Fawad:
Attendance Percentage: 80.0%
Classes Attended: 8
Classes Held: 10

Enter your choice: 6
Student not found.

Enter your choice: 7
Records added successfully

Enter your choice: 8
Error: No record found for the given roll number

Enter your choice: 9
Error: Total classes held cannot be negative

Enter your choice: 10
Error: Total classes attended cannot be negative

Enter your choice: 11
Error: Attended classes cannot be greater than total classes held

Enter your choice: 12
Exiting...
```

`