**Exercise 10: Implementing the MVC Pattern**

**Scenario:**

You are developing a simple web application for managing student records using the MVC pattern.

**Steps:**

**1. Create a New Java Project:**

* **Project Name**: MVCPatternExample
* Use your preferred IDE to create a new Java project.

**2. Define Model Class:**

**Student Class**:

* Create a class Student with attributes like name, id, and grade. Include getters and setters for these attributes.

public class Student {

private String name;

private String id;

private String grade;

// Constructor

public Student(String name, String id, String grade) {

this.name = name;

this.id = id;

this.grade = grade;

}

// Getters and Setters

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getId() {

return id;

}

public void setId(String id) {

this.id = id;

}

public String getGrade() {

return grade;

}

public void setGrade(String grade) {

this.grade = grade;

}

}

**3. Define View Class:**

**StudentView Class**:

* Create a class StudentView with a method displayStudentDetails() to display student details.

public class StudentView {

public void displayStudentDetails(String studentName, String studentId, String studentGrade) {

System.out.println("Student Details:");

System.out.println("Name: " + studentName);

System.out.println("ID: " + studentId);

System.out.println("Grade: " + studentGrade);

}

}

**4. Define Controller Class:**

**StudentController Class**:

* Create a class StudentController that communicates between Student (model) and StudentView (view). It updates the view based on changes in the model.

public class StudentController {

private Student model;

private StudentView view;

// Constructor

public StudentController(Student model, StudentView view) {

this.model = model;

this.view = view;

}

// Get student name

public String getStudentName() {

return model.getName();

}

// Set student name

public void setStudentName(String name) {

model.setName(name);

}

// Get student ID

public String getStudentId() {

return model.getId();

}

// Set student ID

public void setStudentId(String id) {

model.setId(id);

}

// Get student grade

public String getStudentGrade() {

return model.getGrade();

}

// Set student grade

public void setStudentGrade(String grade) {

model.setGrade(grade);

}

// Update view

public void updateView() {

view.displayStudentDetails(model.getName(), model.getId(), model.getGrade());

}

}

**5. Test the MVC Implementation:**

**TestMVCPattern Class**:

* Create a main class to demonstrate creating a Student, updating its details using StudentController, and displaying them using StudentView.

public class TestMVCPattern {

public static void main(String[] args) {

// Create model

Student student = new Student("John Doe", "123", "A");

// Create view

StudentView view = new StudentView();

// Create controller

StudentController controller = new StudentController(student, view);

// Display initial student details

controller.updateView();

// Update student details

controller.setStudentName("Jane Doe");

controller.setStudentId("456");

controller.setStudentGrade("B");

// Display updated student details

controller.updateView();

}

}