Exercise 10: Implementing the MVC Pattern

Objective

To develop a simple web-like application for managing student records using the Model-View-Controller (MVC) Pattern.

Java Code – MVCPatternExample.java

// Step 2: Model Class 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; }

}

// Step 3: View Class class StudentView {

public void displayStudentDetails(String name, String id, String grade) {

System.out.println("---- Student Details ----"); System.out.println("Name : " + name); System.out.println("ID : " + id); System.out.println("Grade : " + grade); System.out.println("-------------------------");

}

}

// Step 4: Controller Class class StudentController {

private Student model; private StudentView view;

public StudentController(Student model, StudentView view) { this.model = model;

this.view = view;

}

// Model modifiers

public void setStudentName(String name) { model.setName(name); } public void setStudentId(String id) { model.setId(id); }

public void setStudentGrade(String grade) { model.setGrade(grade); }

// Getters from Model

public String getStudentName() { return model.getName(); } public String getStudentId() { return model.getId(); }

public String getStudentGrade() { return model.getGrade(); }

// Display data via View public void updateView() {

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

}

}

// Step 5: Main Class to test MVC public class MVCPatternExample {

public static void main(String[] args) {

// Create Model

Student student = new Student("Deepika", "S123", "A");

// Create View

StudentView view = new StudentView();

view);

// Create Controller

StudentController controller = new StudentController(student,

// Initial display controller.updateView();

// Update data controller.setStudentName("Meena"); controller.setStudentGrade("A+");

// Display updated data controller.updateView();

}

}

Sample Output

---- Student Details ----

Name : Deepika ID : S123

Grade : A

-------------------------

---- Student Details ----

Name : Meena ID : S123

Grade : A+

-------------------------

Key Benefits of MVC Pattern

Component Responsibility

Model Manages data and business logic

View Handles UI representation of the data Controller Acts as a link between Model and View