Riphah International University, Gulberg Greens, Islamabad

Name: Usva Qandeel

BSSE-5

46416

Submitted to: Mam Shazwa

LAB TASK

You will create a console application that allows users to manage a list of tasks. The application should follow the MVC architecture, where:

• Model: Represents data and business logic.

• View: Displays information to the user.

• Controller: Handles user input and interactions. Requirements:

1. **Model: • Create a Task class with the following attributes:**
2. **id: Unique identifier for each task. o title: Title of the task (string).**
3. **description: Detailed description of the task (string).**
4. **isCompleted: Boolean to indicate if the task is completed.**
5. **• Implement methods to: o Getters and setters for each attribute.**

2. View: • Create a TaskView class to display information to the user.

6. **• Implement methods to display the list of tasks and prompts for user input.**
7. **Controller: • Create a TaskController class to manage the application logic. Implement methods to: o Add a new task. o Retrieve and display tasks. o Mark a task as completed. 4. Main Application: • Create a Main class to run the application.**

```
// Model: Task Class

class Task {

   private int id;

   private String title;

   private String description;

   private boolean isCompleted;


   public Task(int id, String title, String description) {

      this.id = id;

      this.title = title;

      this.description = description;

      this.isCompleted = false;

   }


   // Getters

   public int getId() {
```

```
    return id;
}

public String getTitle() {
    return title;
}

public String getDescription() {
    return description;
}

public boolean isCompleted() {
    return isCompleted;
}

// Setters
public void setTitle(String title) {
    this.title = title;
}

public void setDescription(String description) {
    this.description = description;
}

public void markAsCompleted() {
    this.isCompleted = true;
}

@Override
```

```
    public String toString() {
        return "ID: " + id + ", Title: " + title + ", Description: " + description + ", Completed: " + isCompleted;
    }
}

// View: TaskView Class
class TaskView {
    public void displayTask(Task task) {
        System.out.println(task.toString());
    }

    public void displayTasks(java.util.List<Task> tasks) {
        if (tasks.isEmpty()) {
            System.out.println("No tasks available.");
        } else {
            for (Task task : tasks) {
                displayTask(task);
            }
        }
    }

    private java.util.Scanner scanner;

    public TaskView() {
        this.scanner = new java.util.Scanner(System.in);
    }

    public String promptForTitle() {
        System.out.print("Enter task title: ");
```

```
        return scanner.nextLine().trim();
    }

    public String promptForDescription() {
        System.out.print("Enter task description: ");
        return scanner.nextLine().trim();
    }

    public int promptForTaskId() {
        System.out.print("Enter task ID to mark as completed: ");
        while (true) {
            try {
                return scanner.nextInt();
            } catch (java.util.InputMismatchException e) {
                System.out.println("Invalid input. Please enter an integer.");
                scanner.next(); // clear invalid input
            }
        }
    }

    // finally block mein scanner close karo
    public void close() {
        scanner.close();
    }
}

// Controller: TaskController Class
class TaskController {
    private java.util.List<Task> tasks;
```

```
private TaskView view;
private int nextId;

public TaskController(TaskView view) {
    this.tasks = new java.util.ArrayList<>();
    this.view = view;
    this.nextId = 1;
}

public void addTask(String title, String description) {
    if (tasks.stream()
            .anyMatch(task -> task.getTitle().equals(title) && task.getDescription().equals(description))) {
        System.out.println("Task already exists.");
    } else {
        Task task = new Task(nextId++, title, description);
        tasks.add(task);
        System.out.println("Task added.");
    }
}

public void displayTasks() {
    view.displayTasks(tasks);
}

public void markTaskAsCompleted(int id) {
    for (Task task : tasks) {
        if (task.getId() == id) {
            task.markAsCompleted();
            System.out.println("Task marked as completed.");
```

```
            return;
        }
    }
    System.out.println("Task not found.");
  }
}

// Main Application
public class Main {
  public static void main(String[] args) {
    TaskView view = new TaskView();
    TaskController controller = new TaskController(view);
    java.util.Scanner scanner = new java.util.Scanner(System.in);

    try {
      while (true) {
        System.out.println("\n1. Add Task\n2. View Tasks\n3. Complete Task\n4. Exit");
        System.out.print("Choose an option: ");
        int choice = scanner.nextInt();
        scanner.nextLine(); // consume newline

        switch (choice) {
          case 1:
            String title = view.promptForTitle();
            String description = view.promptForDescription();
            controller.addTask(title, description);
            break;
          case 2:
            controller.displayTasks();
```

```
            break;
        case 3:
            int taskId = view.promptForTaskId();
            controller.markTaskAsCompleted(taskId);
            break;
        case 4:
            System.out.println("Exiting application.");
            scanner.close();
            return;
        default:
            System.out.println("Invalid choice. Please try again.");
      }
    }
} catch (Exception e) {
    System.out.println("An error occurred: " + e.getMessage());
} finally {
    scanner.close();
}}}
```