Virtual Classroom Manager Programming

Description

The Virtual Classroom Manager is a Java-based application designed to manage classrooms, students, and assignments. It allows adding and viewing classrooms, adding students to classrooms, scheduling assignments, and displaying assignment details. The application adheres to design patterns, SOLID principles, and includes proper logging mechanisms for maintainability and debugging.

Design Patterns Used

Singleton Pattern

Ensures that a single instance of the logger is created and used across various classes.Logger instances in classes such as AddStudentData, AddClassroom, and ScheduleAssignment ensure consistency and control over logging.

Builder Pattern

Facilitates the construction of complex objects by separating the construction process from the representation.Applied in the ScheduleAssignment class to construct assignments with attributes such as assignment name, deadline, and marks in a step-by-step manner

SOLID Principles

1. **Single Responsibility Principle (SRP)**: Each class handles a single responsibility.
2. **Open/Closed Principle (OCP)**: Classes are open for extension but closed for modification.
3. **Liskov Substitution Principle (LSP)**: Subclasses can replace base classes without altering program correctness.
4. **Interface Segregation Principle (ISP)**: Specific interfaces are used based on client needs.
5. **Dependency Inversion Principle (DIP)**: High-level modules depend on abstractions, not low-level modules.

Logger Usage

Logging is implemented using the java.util.logging package in key classes like AddStudentData, AddClassroom, and ScheduleAssignment to track important events and errors with INFO, WARNING, and SEVERE levels.

Commands

1. add_classroom <classroom_name>: Adds a new classroom.
2. remove_classroom <classroom_name>: Removes an existing classroom.
3. add_student <student_id> <classroom_name>: Adds a new student to specified classrooms.

4. view_classroom: Views all classrooms and their details.
5. view_recent: Views the most recently added classrooms.
6. students_at <classroom_name>: Displays students in a specified classroom.
7. display_all_classrooms: Displays all classrooms.
8. schedule_assignment <classroom_name>: Schedules a new assignment for a classroom.
9. submit_assignment <student_id> <classroom_name>: Submits an assignment for a student.
10. assignment_det: Displays details of assignments.
11. status_view <classroom_name>: Displays assignment details for a specific classroom.
12. display_menu: Displays the main menu.
13. exit: Exits the application.

Directory Structure and File Descriptions

- **Main**
  - ProjectMain.java: Entry point of the application.
- **Classroom**
  - AddClassroom.java: Manages classroom operations.
  - ClassroomStorage.java: Manages classroom data storage.
  - RemoveClassroom.java: Handles classroom removal.
  - ViewClassroom.java: Displays classroom details.
  - Classrooms.txt: To store the Classroom data.
- **Student**
  - AddStudent.java: Manages adding students.
  - AddStudentData.java: Handles student data operations (Singleton).
  - Student.java: Represents the student entity.
  - Students.txt: To store Students data like name ,id ,mail, enrolled class etc.
- **Assignments**
  - ScheduleAssignment.java: Manages assignment scheduling.
  - SubmitAssignment.java: Handles assignment submission.
  - AssignmentDetails.java: Provides assignment details.
  - Assignments.txt: To store the classroom names and students details with status of assignment completion.
- **OptionDisplay**
  - optionDisplay.java: Handles user commands and interactions.

Conclusion

The Classroom Management System effectively demonstrates the application of design patterns and SOLID principles to create a maintainable, scalable, and robust application. Future enhancements could include detailed rubrics, notification systems, enhanced data storage, improved user interfaces, and integration with external systems for a comprehensive classroom management experience.