CIS 602 Special Topics in Design Patterns

Fall 2024

PROJECT REPORT

*(This table is provided just as a suggestion for structuring your report. You can use a format of specific paragraphs that address the three elements: REQUIRMENT-PATTERNS-RATIONALE)*

|  |  |
| --- | --- |
| Requirement | Design Pattern (s) + Rationale |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**General description of your approach** *(what strategy did you use selecting patterns to address requirements)*

Feature 1

Requirement: User authentication and role management

Design Pattern(s): Singleton Pattern

Rationale: To ensure a single instance of the authentication service, managing user sessions effectively and providing consistent authentication checks.

Feature 2

Requirement: Attendance data management by teachers

Design Pattern(s): Repository Pattern

Rationale: To abstract data access and provide a clean interface for managing attendance records, enhancing testability and separation of concerns.

Feature 3

Requirement: Student attendance viewing

Design Pattern(s): MVC Pattern

Rationale: To separate the user interface (React.js), business logic (Spring Boot), and data access (MySQL), allowing scalability and ease of maintenance.

Feature 4

Requirement: Real-time attendance statistics

Design Pattern(s): Observer Pattern

Rationale: To notify user interfaces of updates in attendance data, ensuring real-time consistency between the database and UI.

General Description of the Approach

The strategy for selecting design patterns was driven by the specific needs of each feature in the project.   
Patterns were chosen based on their ability to promote scalability, maintainability, and separation of concerns.   
Singleton was used for services requiring a single instance, MVC was employed for UI and data interaction,   
and patterns like Repository and Observer enhanced modularity and real-time interaction.