FIT3077 Assignment 3 Design Rationale

Written by Team Ouroboros

Structure:

- Architectural pattern
    - Justification
    - Advantages
    - Disadvantages

- Overall Design
    - Support for extensibility
    - Refactoring performed

- Package Principles

MVP Architectural Pattern

The chosen architectural pattern still follows the MVP (Model-View-Presenter) pattern. As a recap, the MVP design is similar to the MVC and MVVM patterns, in the sense that MVP focuses on clearly defining and dividing responsibility of classes [1], as well as supports extensibility of the system.

As a breakdown of MVP layers, Model is the data layer which manages the business model classes, while View is the user interface (UI) layer which displays data as instructed by the Presenter, which is the logic layer that acts as a bridge between Model and View [2]. As the MVP architecture isolates functionalities of these components, the Presenter is the only layer which handles changes in Model and View, such that it intercepts the actions from the UI layer, updates the data and tells the UI layer to update itself [1].

This architectural pattern is chosen due to its ability to modularize components, supporting Single Responsibility Principle (SRP), as well as providing ease of extending the system with new features without the need of changing the structure and relationships of existing modules, which is the essence of Open/Closed Principle OCP.

In addition, the MVP architecture enforces one-to-one mapping of Presenter to View, which means one Presenter class handles all the listening and updating for one View [2]. This is the main mechanism that prevents god classes and supports extensibility, as new features usually add new views to the system. Overall, the MVP architecture strictly adheres to SRP and OCP, which creates a structured and modularized design without being overly complicated to implement into OO systems.

Advantages of MVP

Disadvantages of MVP