

Nama : Muhammad Ghazi Rakhmadi

NIM : 2410817310009

Laporan Desain dan Insight Proyek CLI

Aplikasi CLI ini dirancang dengan tiga pilar utama OOP Interface, Abstract Class, dan Composition untuk menciptakan sistem yang modular, mudah dikelola, dan patuh pada prinsip Single Responsibility. Tujuan utamanya adalah untuk memisahkan antara logika, tampilan, dan data. Prinsip Interface (Page.java) diterapkan sebagai "kontrak" murni yang mendefinisikan satu method wajib, display(), yang harus diimplementasikan oleh semua halaman. Ini memungkinkan kelas CLIApp untuk menangani semua halaman secara seragam melalui Polimorfisme. Untuk efisiensi kode dan konsistensi UI, Abstract Class (AbstractPage.java) berfungsi sebagai "template" yang menyediakan fungsionalitas umum seperti showHeader(), showFooter(), dan prompt(). Ini adalah implementasi kunci dari prinsip DRY (Don't Repeat Yourself), yang memusatkan semua kode tampilan berulang di satu tempat

Sementara itu, prinsip Composition digunakan secara ekstensif untuk mengelola hubungan "has-a" (memiliki). Kelas CLIApp "memiliki" sebuah Map berisi semua Page, yang memungkinkannya sebagai orchestrator navigasi. AppContext "memiliki" Scanner, sharedData, dan juga List<String> untuk fungsionalitas Todo List, yang berfungsi sebagai pengelola state (data) aplikasi yang terenkapsulasi. Ini adalah pendekatan yang menghindari penggunaan variabel static global. Desain ini mendelegasikan tanggung jawab: Main.java tetap "dumb" dan hanya memulai aplikasi, CLIApp.java menangani alur logika, dan setiap Page hanya bertanggung jawab atas tampilannya sendiri.

Insight penting adalah realisasi pemisahan tanggung jawab yang jelas, di mana State (AppContext), View (Page), dan Logic (CLIApp) terpisah. Selain itu, penggunaan Abstract Class terbukti sangat efektif untuk mengelola UI yang konsisten, memungkinkan perubahan branding di seluruh aplikasi dengan mengedit satu file. Insight terbesar adalah fleksibilitas sistem. Hal ini terbukti ketika kita menambahkan fitur Todo List. Penambahan fitur baru ini dapat dilakukan hanya dengan membuat kelas Page baru (seperti AddTodoPage dan ViewTodosPage) dan mendaftarkannya di CLIApp, tanpa memodifikasi kode inti yang sudah ada (seperti ScanInputPage atau DisplayDataPage)

Dari penerapan desain ini, beberapa insight penting diperoleh:

1. Pemisahan Tanggung Jawab : Desain ini secara alami menciptakan pemisahan yang jelas antara State (di AppContext), View (di berbagai kelas Page), dan Logic/Controller (di CLIApp).
2. Hierarki Peran yang Jelas: Menjadi jelas bahwa Interface mendefinisikan "Apa" yang harus dilakukan, Abstract Class mendefinisikan "Bagaimana (Secara Umum)" hal itu dilakukan, dan Concrete Class mendefinisikan "Bagaimana (Secara Spesifik)" tugas itu diimplementasikan.
3. Fleksibilitas (Open/Closed Principle): Insight terbesar adalah mudahnya sistem ini diperluas. Hal ini terbukti saat menambahkan fitur Todo List. Untuk menambah fitur baru tersebut, kita hanya perlu membuat kelas Page baru (AddTodoPage dan ViewTodosPage) dan mendaftarkannya di CLIApp, tanpa perlu memodifikasi kode inti yang sudah ada.

Link GitHub : <https://github.com/muh-ghazii/Pemrograman-II-Teori/>