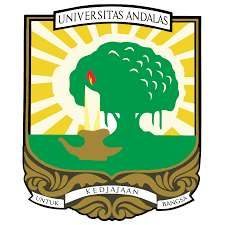
**LAPORAN PRAKTIKUM PBO**

**ABSTRACT CLASS, INTERFACE, FINAL CLASS, INNER CLASS**

****

**Disusun Oleh:**

**INDAH SYAHFITRI**

**2311532016**

**Dosen Pembimbing: Afdhal Dinilhak, S.Komp., M.Kom.**

**DEPARTEMEN INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**UNIVERSITAS ANDALAS**

1. **PENDAHULUAN**

Design Patterns adalah solusi standar yang telah terbukti (teruji) untuk menyelesaikan masalah umum yang sering terjadi dalam pengembangan perangkat lunak. Design Patterns bukanlah hal yang bisa ditranslate ke dalam kode program secara langsung, melainkan hanya template atau panduan cara menyelesaikan masalah. Terdapat banyak sekali desain pattern, dan untuk implementasi ke dalam kode programnya mungkin akan sedikit berbeda tergantung teknologi dan bahasa pemrograman yang digunakan. Dengan memahami design pattern kita akan lebih mudah dalam mengatasi masalah yang terjadi dalam pembuatan aplikasi/software. Karena kita sudah tahu solusi-solusi umum yang harus dilakukan, berbeda jika belum mengerti design pattern maka kita mencari solusi dari permasalahan yang dihadapi dengan mencoba coba solusi versi kita sendiri padahal sebenarnya sudah ada solusi yang sudah terbukti untuk masalah tersebut. Adapun manfaat dari menggunakan design pattern dalam adalah sebagai berikut:

1. Reusability (Penggunaan Ulang), membantu mengurangi pengulangan logika karena solusi tersebut dapat diterapkan di berbagai situasi.
2. Efisiensi, membantu menyelesaikan masalah lebih cepat dengan solusi yang sudah teruji.
3. Kolaborasi, memberikan cara komunikasi yang lebih mudah antar software developer karena banyak software developer sudah memahami pola-pola ini.
4. Scalability, membantu membuat kode yang dapat diskalakan dan mudah diperluas.
5. Maintainability, membuat kode lebih mudah dipahami dan diperbaiki ketika terjadi perubahan.

Jenis-jenis Design Pattern yakni: 1) Creational Patterns, berfokus pada cara membuat objek agar lebih fleksibel dan terstruktur. Creational pattern terdiri dari Singleton, Factory Method, Abstract Factory, Builder, Prototype. 2) Structural Patterns, membantu menyusun class dan object agar mudah dikelola dan membentuk struktur yang kompleks. Structural pattern terdiri dari Adapter, Decorator, Proxy, Facade, Composite, Bridge. 3) Behavioral Patterns, berfokus pada cara object berinteraksi satu sama lain. Behavioral pattern terdiri dari Observer, Strategy, Command, State, Template Method, Mediator, Chain of Responsibility.

1. **TUJUAN PRAKTIKUM**
2. Mahasiswa mampu memahami dan menggunakan beberapa jenis design pattern pada kasus CRUD sebuah aplikasi.
3. **LANGKAH-LANGKAH**
4. Pada project
5. F

Singleton Pattern merupakan pattern yang berfungsi untuk memastikan hanya satu instance dari suatu class yang dapat diakses guna menghindari pembuatan instance duplikat. Berikut adalah contoh penggunaan Singleton Pattern pada kasus insert data ke database, di mana sebelum melakukan insert data, perlu untuk membuat koneksi ke database terlebih dahulu. Dengan menggunakan Singleton Pattern kita hanya perlu membuat koneksi ke database satu kali saja dan untuk selanjutnya koneksi tersebut akan digunakan berkali-kali.

1. **SIMPULAN**

Dalam praktikum ini, konsep *exception handling* telah berhasil diimplementasikan dalam sebuah aplikasi berbasis Java untuk pengelolaan data array. Melalui mekanisme penanganan kesalahan ini, aplikasi mampu menangani berbagai jenis input yang tidak valid, seperti indeks di luar batas array atau input yang bukan angka, tanpa menyebabkan program berhenti secara tiba-tiba. Penggunaan *try-catch block* memungkinkan aplikasi untuk memberikan respons yang sesuai kepada pengguna berupa pesan kesalahan yang jelas, sehingga meningkatkan keandalan dan kenyamanan penggunaan aplikasi.

Selain itu, praktikum ini juga mengajarkan cara memanfaatkan komponen GUI seperti JTextField, JButton, dan JLabel untuk membuat antarmuka pengguna yang interaktif. Fitur seperti menampilkan data array yang tersimpan secara dinamis dan melakukan pengecekan nilai pada indeks tertentu berhasil diterapkan menggunakan komponen-komponen tersebut. Dengan ini, logika program berinteraksi dengan elemen-elemen GUI untuk membangun aplikasi yang fungsional bisa dipahami dengan baik.