Proyek aplikasi Manajemen Koleksi Buku

A. Deskripsi Proyek

Proyek ini bertujuan untuk membuat aplikasi yang memungkinkan pengguna untuk mengelola koleksi buku mereka dengan lebih efisien. Aplikasi ini dapat mencakup fitur-fitur seperti:

- **Penambahan Buku**: Pengguna dapat menambahkan informasi buku baru ke dalam sistem, seperti judul, pengarang, tahun terbit, dan lain-lain.
- **Pencarian Buku**: Pengguna dapat mencari buku berdasarkan judul, pengarang, atau kategori.
- **Penyortiran Buku**: Pengguna dapat menyusun daftar buku berdasarkan kategori, judul, atau pengarang.
- **Peminjaman dan Pengembalian Buku**: Sistem dapat mencatat status peminjaman buku, siapa yang meminjam, dan kapan harus dikembalikan.
- **Manajemen Kategori**: Pengguna dapat membuat, mengedit, dan menghapus kategori buku.
- **Manajemen Pengguna**: Admin dapat mengelola pengguna aplikasi, seperti menambahkan pengguna baru atau mengubah hak akses.

B. Mengapa Menggunakan Teori OOP ?

- **Access Modifier (Pengubah Akses)**:

  Mengontrol akses ke variabel dan metode dalam kelas. Misalnya, variabel yang hanya dapat diakses di dalam kelas tersebut atau metode yang hanya dapat diakses oleh kelas turunannya. Penting untuk menjaga keamanan data dan membatasi akses yang tidak diinginkan.

- **Inheritance (Pewarisan)**:

  Memungkinkan pembuatan hierarki kelas yang memperluas fungsionalitas kelas induknya.Berguna untuk membagi dan mengatur kode dengan lebih efisien, menghindari duplikasi kode, dan memfasilitasi pemeliharaan kode yang lebih baik.

- **Polymorphism (Polimorfisme)**:

  Memungkinkan penggunaan objek dari kelas yang berbeda dengan cara yang seragam. Dapat meningkatkan fleksibilitas kode dan mempermudah implementasi fitur-fitur yang berbeda.

- **Encapsulation (Enkapsulasi)**:

  Menyembunyikan rincian implementasi dari objek, dan hanya mengekspos apa yang diperlukan melalui antarmuka publik. Membantu dalam menjaga integritas data, mengurangi ketergantungan antar bagian kode, dan meningkatkan kemudahan dalam memahami dan mengelola kode.

C. Implementasinya (Contoh Singkat)

- **Kelas Book (Buku)**

```
public class Book {
    private String title;
    private String author;
    private int yearPublished;

    public Book(String title, String author, int yearPublished) {
        this.title = title;
        this.author = author;
        this.yearPublished = yearPublished;
    }

    public String getTitle() {
        return title;
    }

    public String getAuthor() {
        return author;
    }

    public int getYearPublished() {
        return yearPublished;
    }
}
```

- **Kelas FictionBook (Buku Fiksi) yang Mewarisi Kelas Book**

```
public class FictionBook extends Book {
    private String genre;

    public FictionBook(String title, String author, int yearPublished, String genre) {
        super(title, author, yearPublished);
        this.genre = genre;
    }

    public String getGenre() {
        return genre;
    }
}
```

- **Kelas Library (Perpustakaan)**

```
import java.util.ArrayList;

public class Library {
    private ArrayList<Book> books;

    public Library() {
        books = new ArrayList<>();
    }

    public void addBook(Book book) {
        books.add(book);
    }
```

```
    public void displayBooks() {
        for (Book book : books) {
            System.out.println(book.getTitle()
);
        }
    }

    public Book findBookByTitle(String title)
{
        for (Book book : books) {
            if
(book.getTitle().equalsIgnoreCase(title)) {
                return book;
            }
        }
        return null;
    }
}
```

D. Kesimpulan

Dalam proyek ini, teori OOP seperti access modifier, inheritance, polymorphism, dan encapsulation digunakan untuk membangun struktur yang terorganisir dan mudah dimengerti. NetBeans sebagai IDE Java dapat memudahkan pengembangan dengan menyediakan alat bantu seperti pembuat kode otomatis dan debugger.

Menggunakan teori OOP dalam pengembangan aplikasi manajemen koleksi buku memungkinkan kita untuk membuat sistem yang terstruktur, mudah diatur, dan mudah diperluas. Dengan menerapkan enkapsulasi, pewarisan, polimorfisme, dan pengubah akses, kita dapat menjaga integritas data, memudahkan pemeliharaan kode, dan meningkatkan fleksibilitas aplikasi. Java sebagai bahasa yang mendukung konsep OOP sangat cocok digunakan dalam proyek ini karena menyediakan fitur dan struktur yang mendukung implementasi teori-teori tersebut.