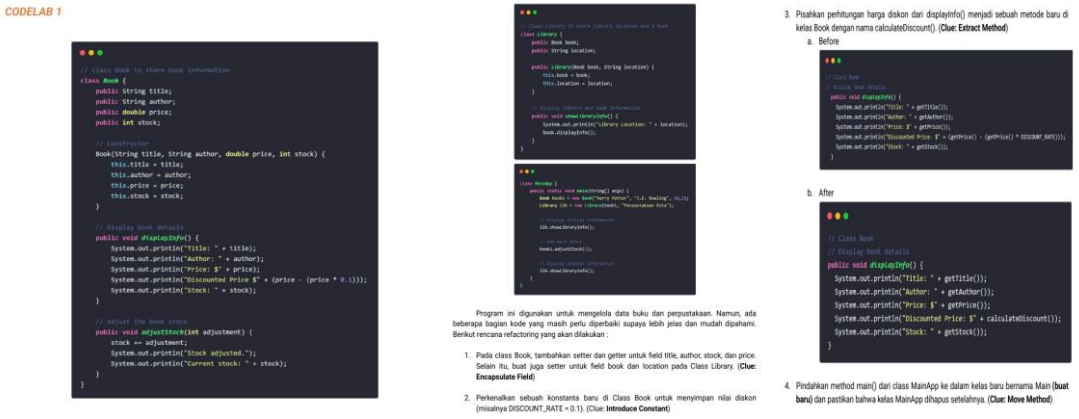
LAPORAN PRAKTIKUM CODELAB
PROMLAN MODUL 2

NAMA : MUHAMMAD ARDHI PONTOH
NIM : 202410370110290
KELAS: 3D

CODELAB:



LANGKAH-LANGKAH:

1. Code awal

```java
class MainApp {
    public static void main(String[] args) {
        Book book1 = new Book( title: "Harry Potter", author: "J.K. Rowling", price: 10.0, st
        Library lib = new Library(book1, location: "Perpustakaan Kota");

        // Display initial information
        System.out.println("--- Initial Book Information ---");
        lib.showLibraryInfo();
        System.out.println();

        // Add more stock
        book1.adjustStock( adjustment: 5);
        System.out.println();

        // Display updated information
        System.out.println("--- Updated Book Information ---");
        lib.showLibraryInfo();
    }
}
```

```java
class Book {
    public String title;
    public String author;
    public double price;
    public int stock;

    // Constructor
    public Book(String title, String author, double price, int stock) {
        this.title = title;
        this.author = author;
        this.price = price;
        this.stock = stock;
    }

    // Display book details with hardcoded discount
    public void displayInfo() {
        System.out.println("Title: " + title);
        System.out.println("Author: " + author);
        System.out.println("Price: $" + price);
        // "Magic number" 0.1 for discount calculation
        System.out.println("Discounted Price: $" + (price - (price * 0.1)));
        System.out.println("Stock: " + stock);
    }

    // Adjust the book stock
    public void adjustStock(int adjustment) {
        stock += adjustment;
        System.out.println("Stock adjusted.");
        System.out.println("Current stock: " + stock);
    }
}
```

```java
class Library {
    public Book book;
    public String location;

    public Library(Book book, String location) {
        this.book = book;
        this.location = location;
    }

    // Display library and book information
    public void showLibraryInfo() {
        System.out.println("Library Location: " + location);
        book.displayInfo();
    }
}
```

2. Menerapkan Encapsulate Field

Field pada kelas Book (title, author, price, stock) dan Library (book, location) diubah dari public menjadi private. Kemudian, setter dan getter dibuat untuk setiap field agar akses terhadap data lebih terkontrol.

```
private String title;  2 usages
private String author;  2 usages
private double price;  2 usages
private int stock;  2 usages
```

```
public String getTitle() { return title; }
public void setTitle(String title) { this.title = title; }
public String getAuthor() { return author; }
public void setAuthor(String author) { this.author = author; }
public double getPrice() { return price; }
public void setPrice(double price) { this.price = price; }
public int getStock() { return stock; }
public void setStock(int stock) { this.stock = stock; }
```

```
private Book book;  2 usages
private String location;  2 usages
```

```
public Book getBook() { return book; }
public void setBook(Book book) { this.book = book; }
public String getLocation() { return location; }
public void setLocation(String location) { this.location = location; }
```

3. Menerapkan Introduce Constant

Nilai diskon 0.1 yang sebelumnya ditulis langsung (hardcoded) di dalam method displayInfo, diekstrak menjadi sebuah konstanta DISCOUNT_RATE. Hal ini membuat kode lebih mudah dibaca.

```
public static final double DISCOUNT_RATE = 0.1;
```

```
System.out.println("Discounted Price: $" + (getPrice() - (getPrice() * DISCOUNT_RATE)));
```

4. Menerapkan Extract Method

Logika perhitungan harga diskon dipisahkan dari metode displayInfo() ke dalam metode baru bernama calculateDiscount(). Ini membuat metode displayInfo() lebih ringkas dan fokus pada tugasnya, yaitu menampilkan informasi.

```
    System.out.println("Discounted Price: $" + calculated_discount());
    System.out.println("Stock: " + getStock());
}

private double calculated_discount() {  1 usage
    return getPrice() - (getPrice() * DISCOUNT_RATE);
}
```

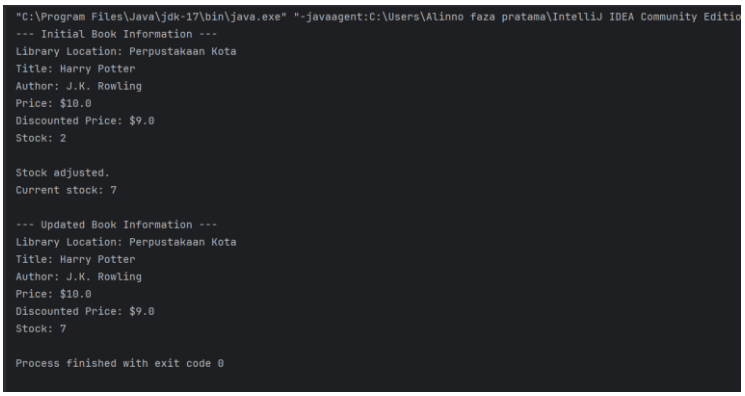
5. Menerapkan Move Method

Metode main() dipindahkan dari kelas MainApp ke kelas baru bernama Main. Kelas MainApp kemudian dihapus. Langkah ini bertujuan untuk mengorganisasi kode dengan lebih baik, di mana kelas Main secara khusus berfungsi sebagai titik masuk (entry point) program.

```
1  public class Main {
2      public static void main(String[] args) {
3          Book book1 = new Book( title: "Harry Potter", author: "J.K. Rowling", price: 10.0, stock: 2);
4          Library lib = new Library(book1, location: "Perpustakaan Kota");
5
6          // Display initial information
7          System.out.println("--- Initial Book Information ---");
8          lib.showLibraryInfo();
9          System.out.println();
10
11         // Add more stock
12         book1.adjustStock( adjustment: 5);
13         System.out.println();
14
15         // Display updated information
16         System.out.println("--- Updated Book Information ---");
17         lib.showLibraryInfo();
18     }
19 }
```

6. Jalankan Program

Setelah semua proses refactoring selesai, program dijalankan dari kelas Main. Output yang dihasilkan tetap sama seperti sebelum refactoring, yang membuktikan bahwa perubahan yang dilakukan tidak mengubah fungsionalitas program.



7. Kesimpulan

Program ini berhasil di-refactor dengan menerapkan empat teknik utama untuk meningkatkan kualitas kode:

- **Encapsulate Field:** Meningkatkan keamanan data dan fleksibilitas dengan menyembunyikan implementasi internal.
- **Introduce Constant:** Meningkatkan keterbacaan dan kemudahan pemeliharaan dengan menghilangkan magic numbers.
- **Extract Method:** Membuat kode lebih bersih dan mudah dipahami dengan memecah metode yang kompleks menjadi bagian-bagian yang lebih kecil.
- **Move Method:** Meningkatkan struktur dan kohesi program dengan menempatkan fungsionalitas pada kelas yang paling sesuai.