

LAPORAN PRAKTIKUM CODELAB PBO

MODUL 4

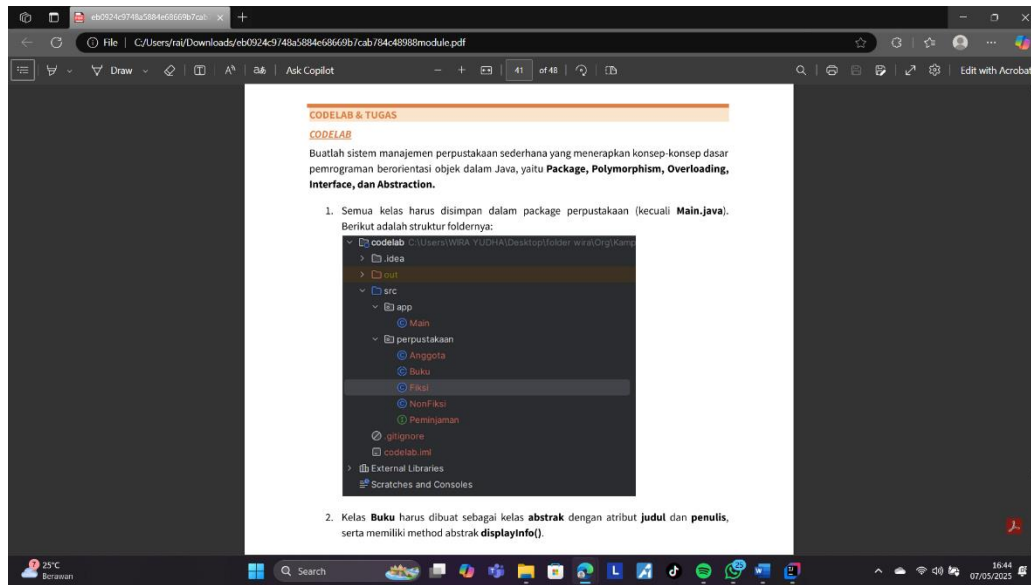


Nama: Chraitong Pranadipa Mardjun

NIM: 202410370110226

Kelas: F

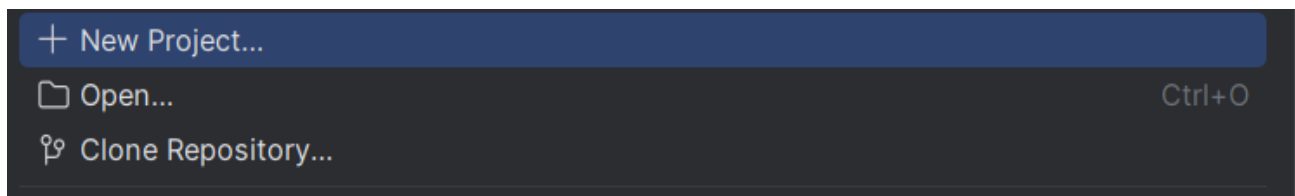
CODELAB:



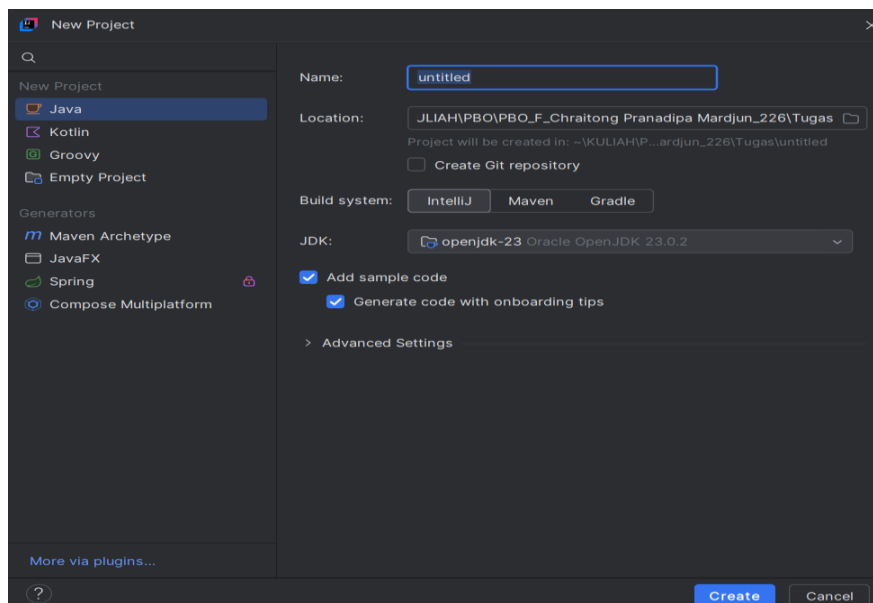
LANGKAH-LANGKAH:

1. Membuat Project baru

- Langkah pertama yang saya lakukan adalah membuat project java baru sdengan cara pada IntelliJ IDEA Community Edition adalah dengan tekan tombol '*New Project*' pada bagian sidebar seperti contoh pada gambar berikut ini:

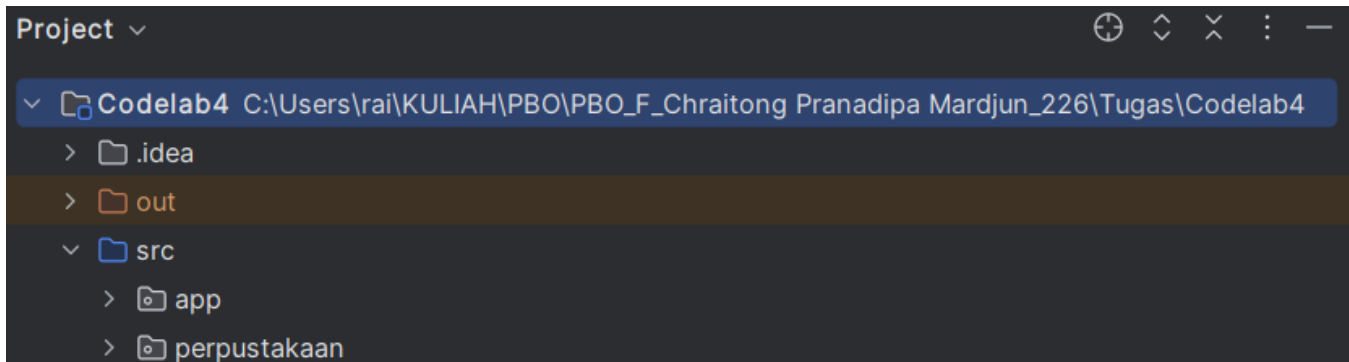


- Setelah itu silahkan pilih nama judul yang ingin digunakan serta lokasi untuk menyimpan datanya. Karena saya sudah mengaitkan repository git saya maka saya akan menyimpan project ini ke folder tersebut.



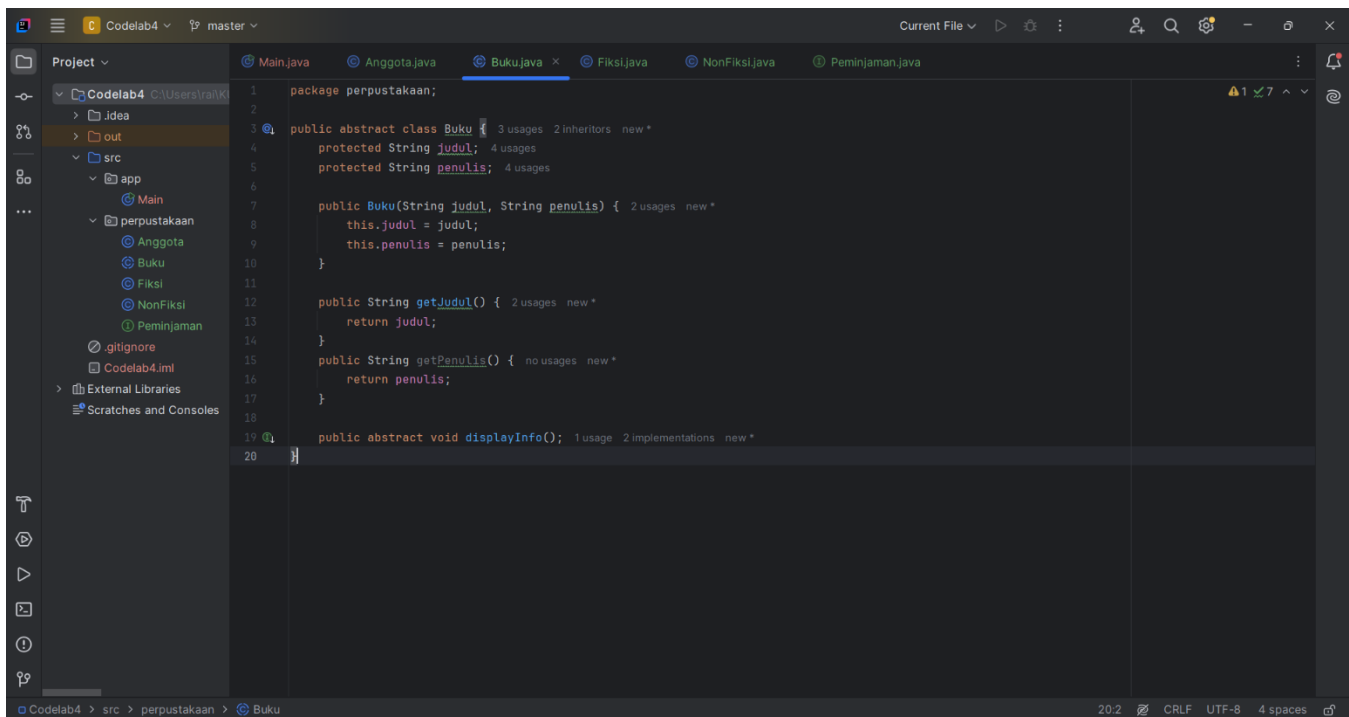
2. Membuat folder (package)

- Selanjutnya membuat folder package pada folder src. Package yang akan ditambahkan ke folder src yakni;
 - app
 - perpustakaan

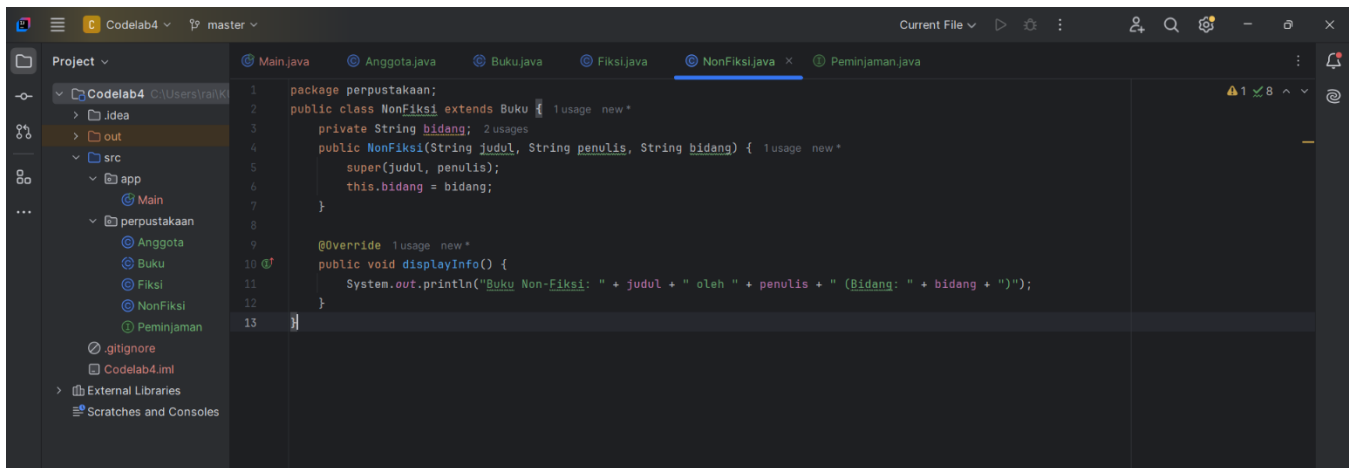


3. Membuat isi dari: Package Perpustakaan

- Disini saya membuat sebuah kelas abstrak bernama **Buku**, yang mencakup atribut **judul** dan **penulis**. Selain itu, kelas ini memiliki metode abstrak **displayInfo()**, yang wajib di-override dalam turunan kelasnya.



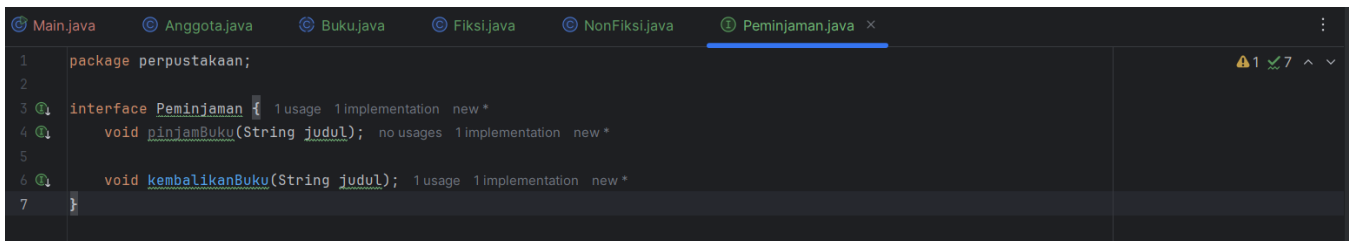
- Setelah itu, dibuat subclass dari kelas **Buku**, yaitu **Fiksi** dan **NonFiksi**. Kelas **Fiksi** merepresentasikan buku fiksi dengan atribut tambahan **genre**, sedangkan kelas **NonFiksi** merepresentasikan buku non-fiksi dengan atribut tambahan **bidang**.



```

1 package perpustakaan;
2 public class NonFiksi extends Buku {
3     private String bidang;
4     public NonFiksi(String judul, String penulis, String bidang) {
5         super(judul, penulis);
6         this.bidang = bidang;
7     }
8
9     @Override
10    public void displayInfo() {
11        System.out.println("Buku Non-Fiksi: " + judul + " oleh " + penulis + " (Bidang: " + bidang + ")");
12    }
13 }
  
```

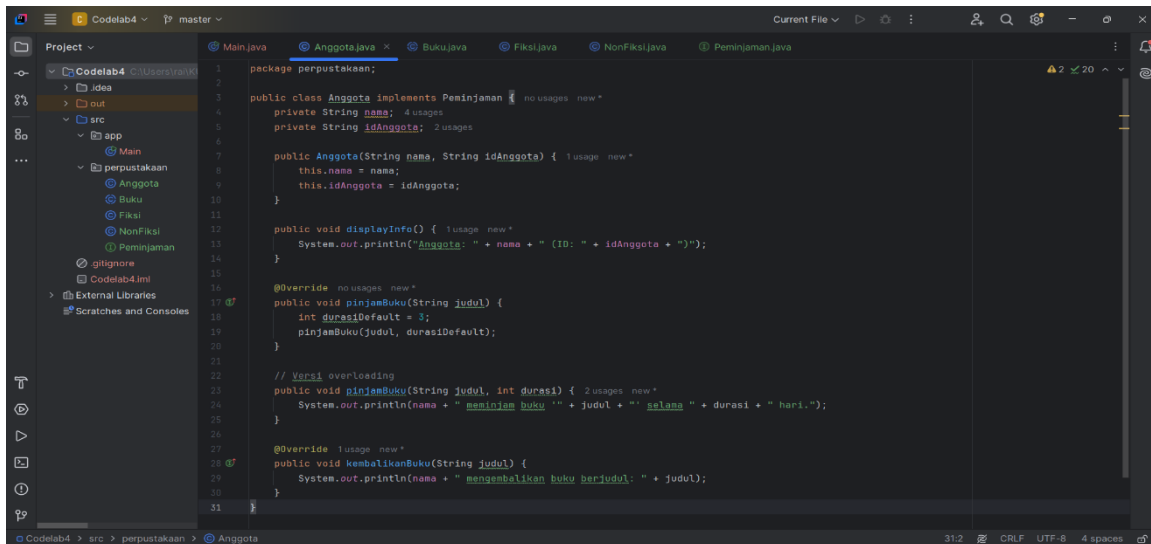
- Berikutnya, dibuat sebuah interface bernama **Peminjaman**, yang berfungsi untuk mendefinisikan perilaku umum bagi objek yang dapat melakukan peminjaman dan pengembalian buku, tanpa bergantung pada implementasi spesifiknya. Dalam interface **Peminjaman**, terdapat metode **pinjamBuku()** dan **kembalikanBuku()**.



```

1 package perpustakaan;
2
3 interface Peminjaman {
4     void pinjamBuku(String judul);
5
6     void kembalikanBuku(String judul);
7 }
  
```

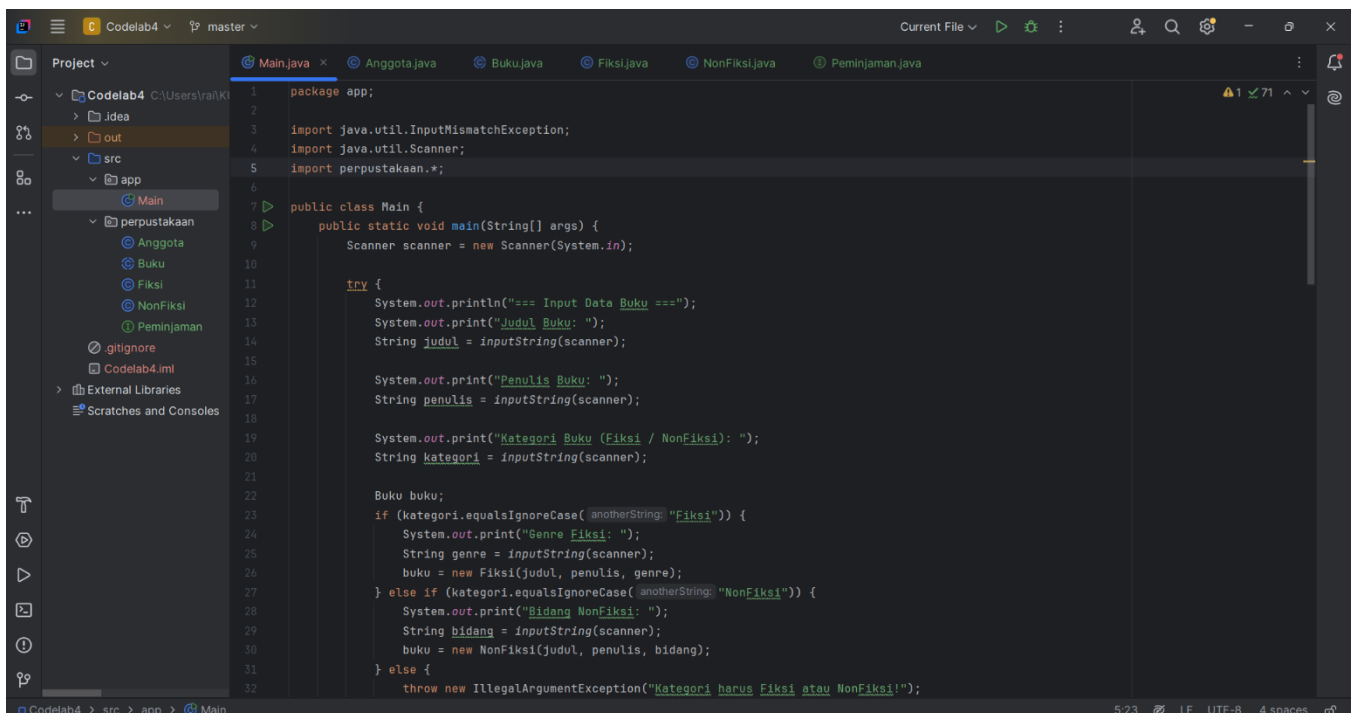
- Berikutnya akan membuat kelas **Anggota**, yang memiliki beberapa fungsi utama yaitu:
 - Merepresentasikan data anggota perpustakaan dengan menyimpan informasi seperti **nama** dan **idAnggota**.
 - Mengimplementasikan interface **Peminjaman**, sehingga wajib memiliki metode **pinjamBuku()** dan **kembalikanBuku()**, sesuai dengan kontrak yang ditetapkan.
 - Menyediakan dua versi metode **pinjamBuku()** melalui **overloading**:
 - Versi pertama hanya menerima parameter **judul**, dengan nilai default peminjaman selama **3 hari**.
 - Versi kedua menerima **judul** dan **durasi**, memungkinkan peminjam menentukan lama peminjaman secara fleksibel.
 - Metode **kembalikanBuku()** berfungsi untuk menampilkan pesan bahwa anggota telah mengembalikan buku tertentu.
 - Menampilkan informasi anggota melalui metode **displayInfo()**, yang mencetak **nama** dan **ID anggota** ke layar.



4. Membuat isi dari: Package App

Implementasi program utama dalam **Main.java** menggunakan input langsung dari pengguna.

1. Pengguna memasukkan **judul**, **penulis**, dan **kategori** buku. Berdasarkan kategori yang dipilih (**Fiksi** atau **NonFiksi**), objek buku akan dibuat menggunakan kelas **Fiksi** atau **NonFiksi**. Jika kategori tidak valid, program akan melempar **IllegalArgumentException**. Setelah itu, informasi buku ditampilkan melalui metode **displayInfo()**.



2. Setelah data buku dimasukkan, program meminta input berupa **nama** dan **ID anggota** perpustakaan. Data ini digunakan untuk membuat objek **Anggota**, yang merepresentasikan peminjam buku dalam sistem.

3. Pengguna kemudian menentukan **durasi peminjaman** dalam satuan hari. Program memastikan bahwa durasi minimal adalah **satu hari**. Setelah menerima input, metode **pinjamBuku()** dipanggil dengan parameter **judul buku** dan **durasi peminjaman**, lalu sistem menampilkan informasi bahwa buku telah dipinjam oleh anggota tersebut.
4. Terakhir, sistem menyimulasikan **pengembalian buku**. Metode **kembalikanBuku()** dipanggil dengan **judul buku yang sama**, dan program menampilkan pesan bahwa buku telah berhasil dikembalikan oleh anggota. Proses ini merepresentasikan siklus peminjaman dan pengembalian buku di perpustakaan secara interaktif.
5. Program ini mencakup dua metode bantu untuk validasi input pengguna. Metode **inputString()** memastikan bahwa input berupa string tidak kosong. Jika pengguna memasukkan teks kosong atau hanya spasi, program akan meminta pengulangan hingga memperoleh input yang valid.

```

70 public class Main {
71     // Method bantu untuk validasi input String (tidak boleh kosong)
72     public static String inputString(Scanner scanner) {
73         String input = "";
74         while (input.trim().isEmpty()) {
75             input = scanner.nextLine();
76             if (input.trim().isEmpty()) {
77                 System.out.print("Input tidak boleh kosong, coba lagi: ");
78             }
79         }
80         return input.trim();
81     }
82 }

```

6. Metode **inputInt(min)** digunakan untuk memproses input bilangan bulat, dengan ketentuan bahwa nilai yang dimasukkan harus **sama atau lebih besar** dari batas minimum yang ditentukan. Selain memvalidasi nilai minimum, metode ini juga menangani kesalahan input jika pengguna memasukkan data non-numerik, dengan menampilkan pesan kesalahan dan meminta input ulang. Kedua metode ini membuat program lebih stabil serta lebih toleran terhadap kesalahan input pengguna.

```

81 // Method bantu untuk validasi input Integer dengan batas minimum
82 public static int inputInt(Scanner scanner, int min) {
83     while (true) {
84         try {
85             int input = scanner.nextInt();
86             scanner.nextLine(); // konsumsi newline
87             if (input < min) {
88                 System.out.print("Input minimal adalah " + min + ", coba lagi: ");
89                 continue;
90             }
91             return input;
92         } catch (InputMismatchException e) {
93             System.out.print("Input harus angka, coba lagi: ");
94             scanner.nextLine(); // buang input yang salah
95         }
96     }
97 }
98 }
99

```

Berikut link repositorynya [klik disini](#)