**LAPORAN PRAKTIKUM**

**TEKNIK PEMROGRAMAN**

**JAVA COLLECTION FRAMEWORK PRAKTIKUM**

MINGGU KE-5



NAMA: FAUZI ISMAIL

NIM: 241524042

KELAS: D4-1B

**PROGRAM STUDI SARJANA TERAPAN**

**TEKNIK INFORMATIKA**

**POLITEKNIK NEGERI BANDUNG**

**2025**

## DAFTAR ISI

[DAFTAR ISI 2](#_Toc192586461)

[https://github.com/mailvlous/teknikPemrograman/tree/main/Week5 3](#_Toc192586462)

[1. List, Set, Map 3](#_Toc192586463)

[2. Record 5](#_Toc192586464)

[3. Optional 6](#_Toc192586465)

[4. Concurrent Collections 6](#_Toc192586466)

[5. Queue dan Dequeue 8](#_Toc192586467)

[6. Immutable collection List.of, Set.of, Map.of 10](#_Toc192586468)

## <https://github.com/mailvlous/teknikPemrograman/tree/main/Week5>

## 1. List, Set, Map

1. Definisi

* List

List adalah interface dalam Java yang termasuk dalam Java Collections Framework. List digunakan untuk menyimpan sekumpulan elemen berurutan yang dapat memiliki duplikasi.

* Set

The Set Interface is present in java.util package and extends the Collection interface. It is an unordered collection of objects in which duplicate values cannot be stored. It is an interface that implements the mathematical set. This interface adds a feature that restricts the insertion of the duplicate elements.

* Map

In Java, Map Interface is present in the java.util package represents a mapping between a key and a value. Java Map interface is not a subtype of the Collection interface. Therefore it behaves a bit differently from the rest of the collection types.

2. Kenapa diperlukan

* List

Kegunaan/keperluan list:

* Maintained the order of elements in which they are added.
* Allows the duplicate elements.
* The implementation classes of the List interface are ArrayList, LinkedList, Stack, and Vector.
* Can add Null values that depend on the implementation.
* The List interface offers methods to access elements by their index and includes the listIterator() method, which returns a ListIterator.
* Using ListIterator, we can traverse the list in both forward and backward directions.
* Set

Set diperlukan karena beberapa karakteristiknya yang tidak dapat diduplikat, operasi himpunan, operasi search yang cepat, menjaga elemen tertentu

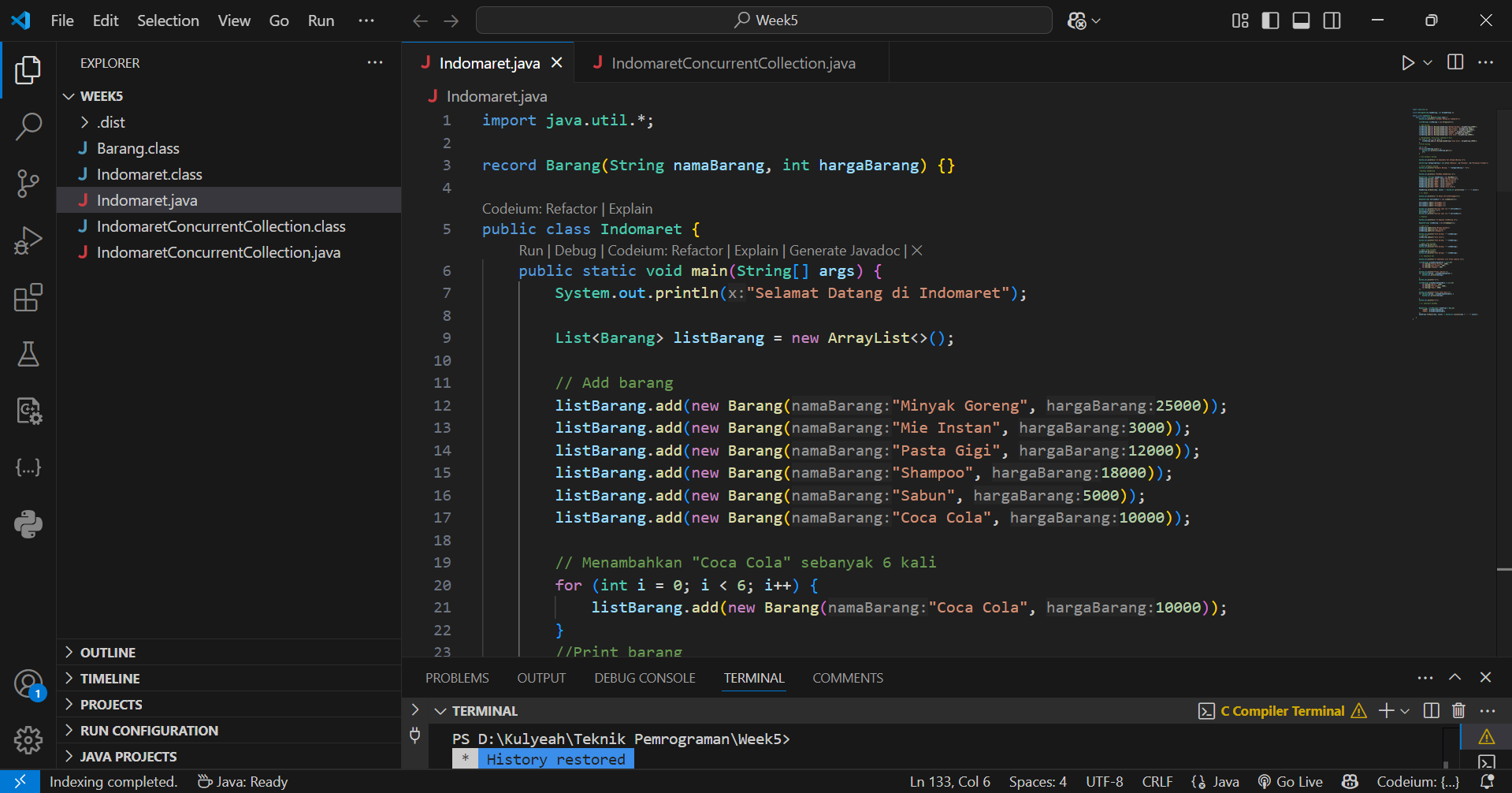
* Map

Maps are perfect to use for key-value association mapping such as dictionaries. The maps are used to perform lookups by keys or when someone wants to retrieve and update elements by keys. Some common scenarios are as follows:

* A map of error codes and their descriptions.
* A map of zip codes and cities.
* A map of managers and employees. Each manager (key) is associated with a list of employees (value) he manages.
* A map of classes and students. Each class (key) is associated with a list of students (value)

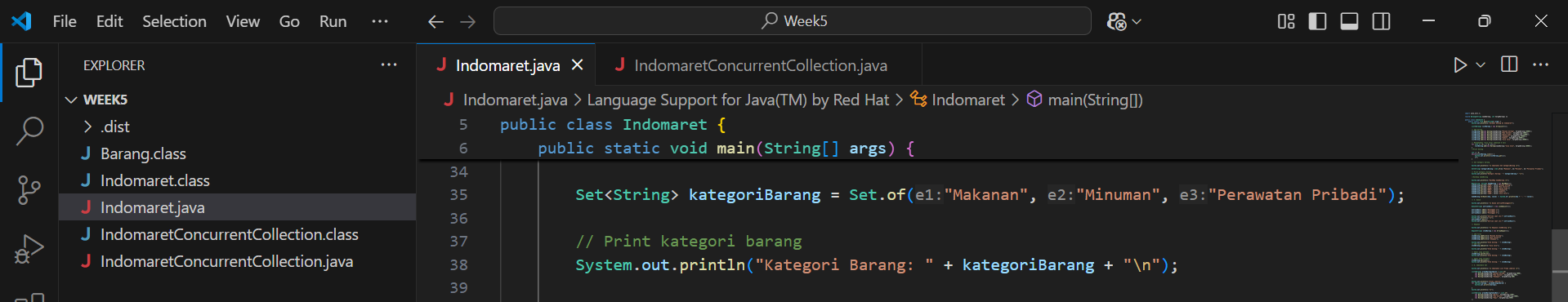
3. Implementasi

* List



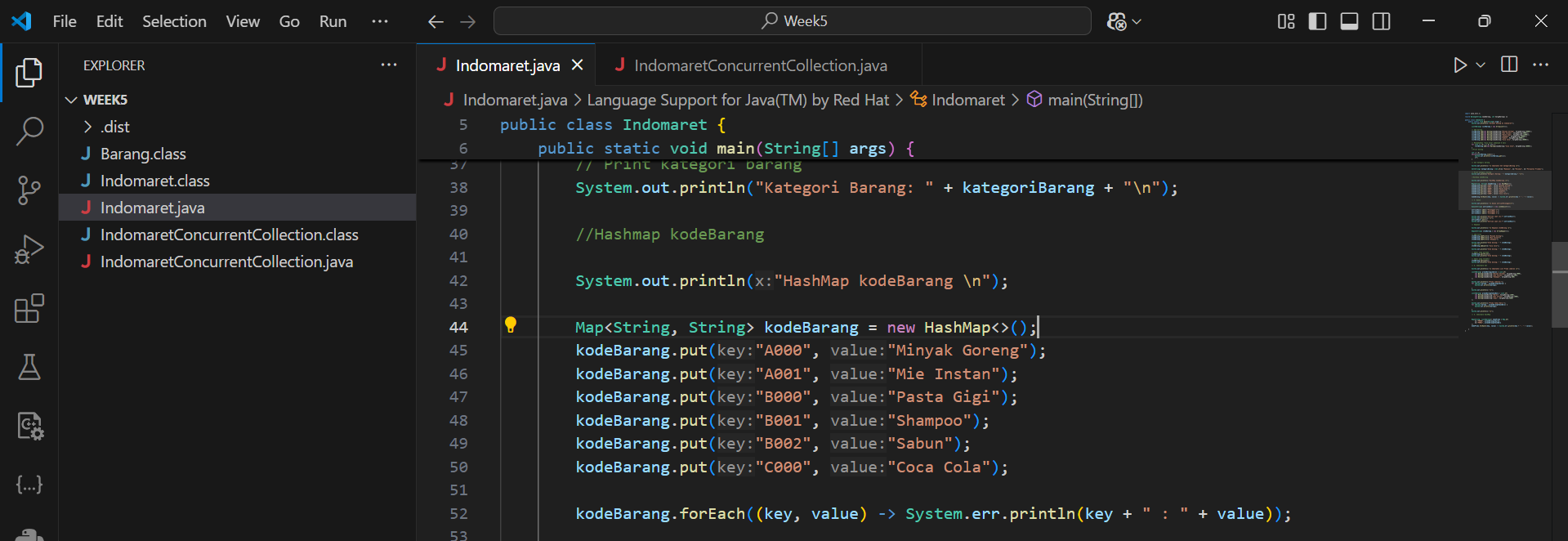
Disini untuk implementasi dari list itu sendiri saya disini menggunakan contoh kasus Indomaret. Disini terdapat List listBarang yang memuat Record Barang(String name, int hargaBarang)

* Set



Untuk set disini saya memanfaatkan ciri Set yaitu unik, maka saya membuat Set Kategori yang memuat kategori kategori barang

* Map



Untuk Map karena Map memiliki pair(Key, Value) maka ini sangat cocok dengan pair(idBarang, namaBarang), karena setiap barang memiliki Id tersendiri.

## 2. Record

1. Definisi

record adalah fitur di Java yang diperkenalkan sejak Java 14 (preview) dan resmi di Java 16, yang digunakan untuk membuat kelas immutable dengan lebih ringkas dan otomatis.

record adalah kelas khusus yang digunakan untuk menyimpan data tanpa perilaku tambahan. Secara default, record sudah menyediakan:

✅ Constructor otomatis

✅ Getter otomatis (tanpa prefix get)

✅ Implementasi equals(), hashCode(), dan toString().

2. Kenapa diperlukan

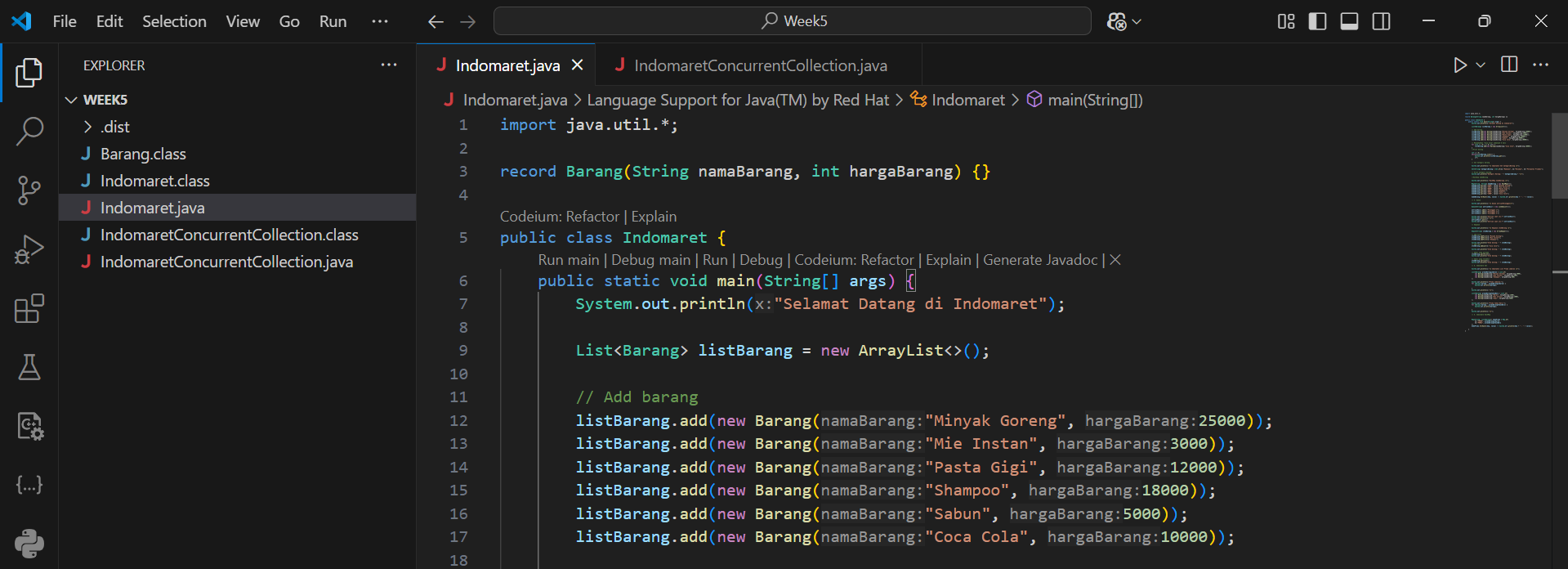
✅ Mengurangi boilerplate code → Tidak perlu menulis constructor, getter, equals(), dll.

✅ Immutable secara default → Data tidak bisa diubah setelah dibuat (lebih aman).

✅ Otomatis memiliki equals(), hashCode(), dan toString() → Berguna untuk Set dan Map.

✅ Cocok untuk DTO dan pemrosesan data → Lebih ringkas saat bekerja dengan API dan Java Streams.

3. Implementasi



Untuk implementasinya saya membuat record Barang(String namaBarang, int hargaBarang) dan digunakan di listBarang.

## 3. Optional

1. Definisi

Every Java Programmer is familiar with NullPointerException. It can crash your code. And it is very hard to avoid it without using too many null checks. So, to overcome this, Java 8 has introduced a new class Optional in java.util package. It can help in writing a neat code without using too many null checks. By using Optional, we can specify alternate values to return or alternate code to run. This makes the code more readable because the facts which were hidden are now visible to the developer.

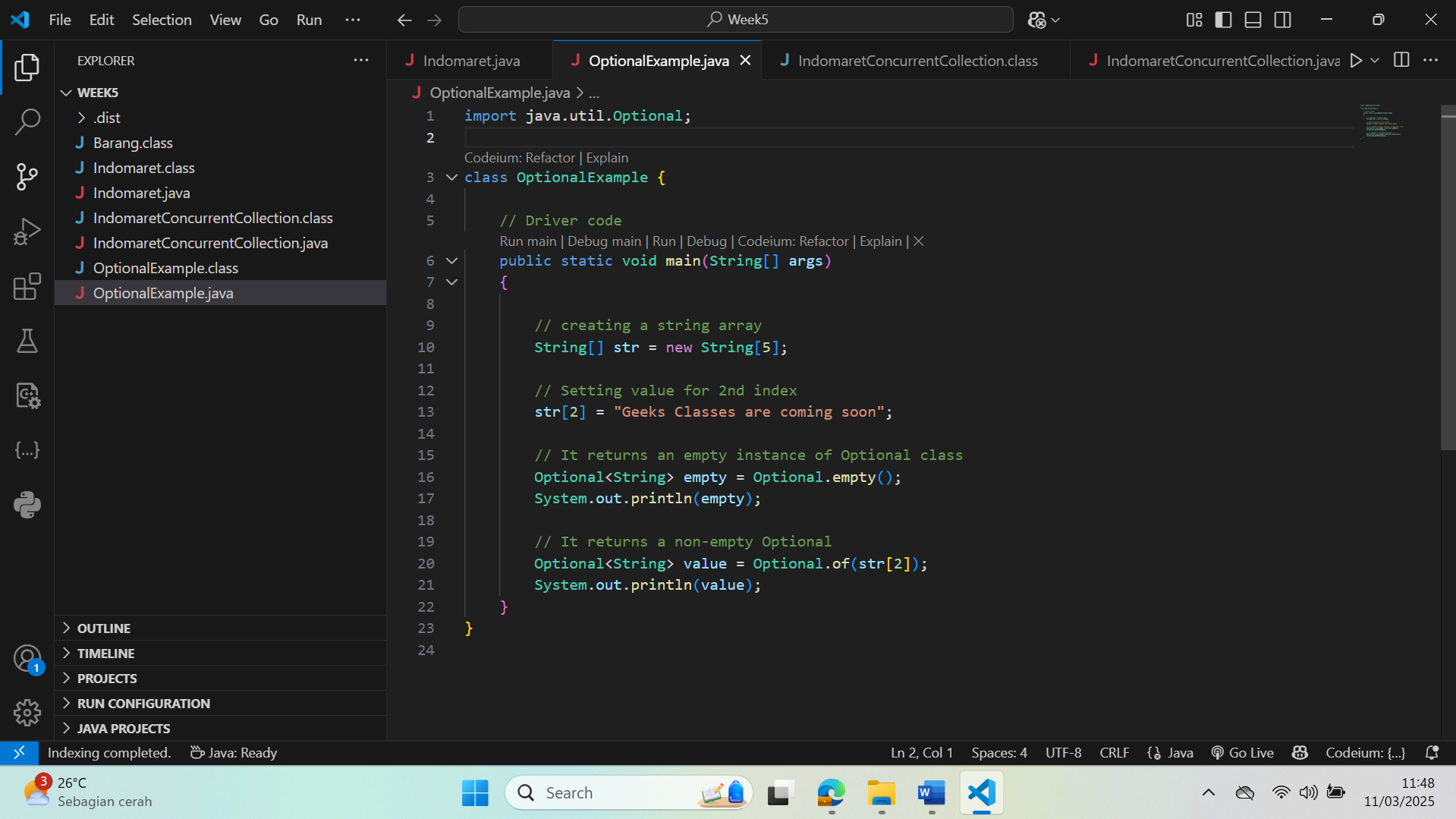
2. Mengapa diperlukan

🔹 Optional membantu menangani nilai null dengan lebih aman dan menghindari NullPointerException.

🔹 Digunakan dalam return method yang bisa mengembalikan nilai kosong.

🔹 Membuat kode lebih bersih, aman, dan mudah dibaca. 🚀

3. Implementasi



Di contoh ini Optional berfungsi untuk menangani data yang tidak ada, sehingga tidak terjadi NullPointerException

## 4. Concurrent Collections

1. Definisi

Concurrent Collection adalah kumpulan struktur data thread-safe dalam paket java.util.concurrent yang memungkinkan beberapa thread mengakses dan memodifikasi data secara bersamaan tanpa menyebabkan race condition.

Koleksi ini dirancang untuk mengoptimalkan kinerja dalam lingkungan multithreading, berbeda dari koleksi biasa yang harus disinkronisasi secara manual. 🚀

Thread adalah unit eksekusi terkecil dalam suatu program yang berjalan secara independen.

Di Java, multithreading memungkinkan beberapa tugas berjalan secara bersamaan (concurrent) untuk meningkatkan efisiensi dan kinerja aplikasi.

2. Mengapa diperlukan

✅ Menghindari Race Condition → Memastikan beberapa thread bisa membaca/menulis data tanpa konflik.

✅ Lebih Efisien Dibandingkan synchronized → Tidak mengunci seluruh koleksi, hanya bagian yang diperlukan.

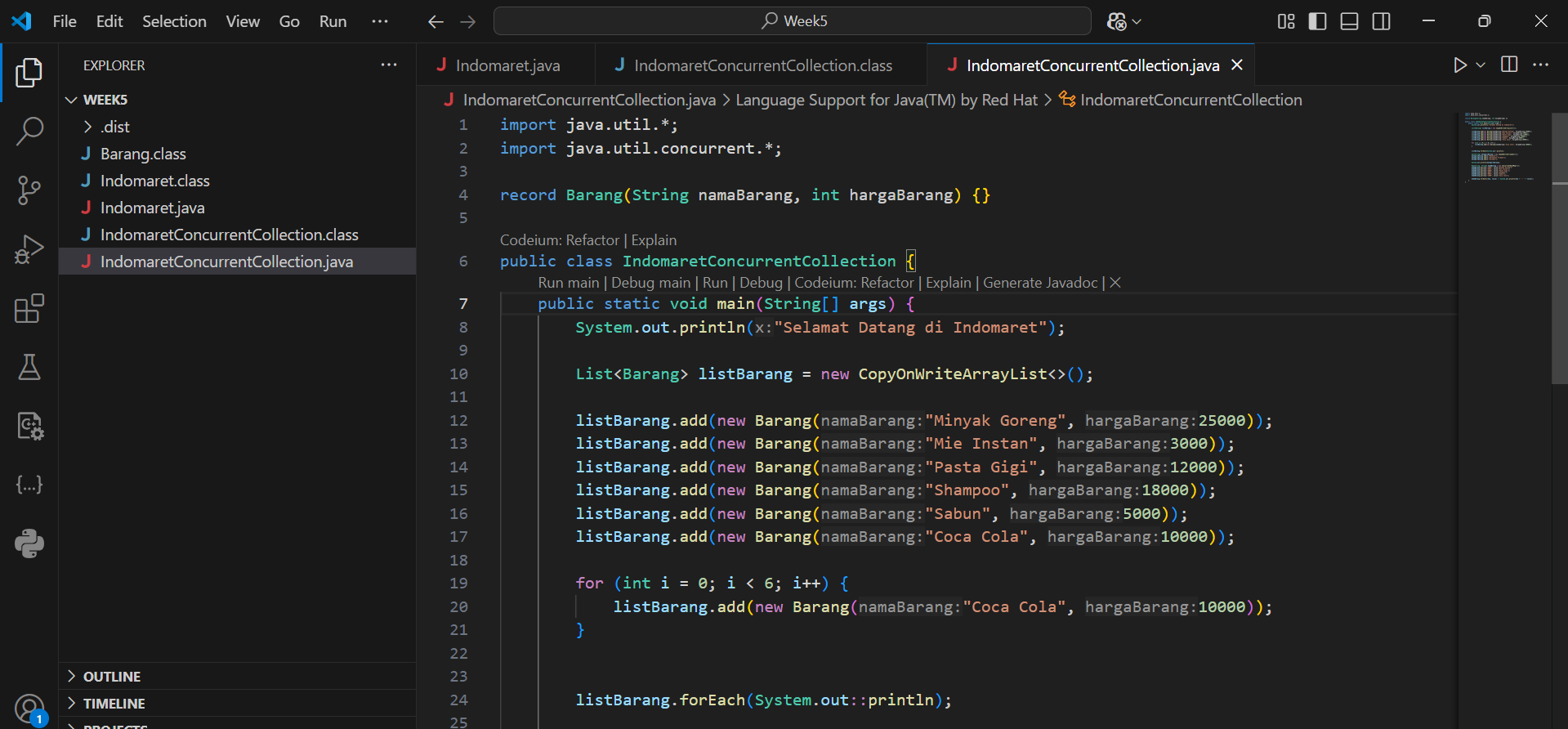
✅ Cocok untuk Multithreading → Digunakan dalam aplikasi yang memproses banyak tugas secara bersamaan.

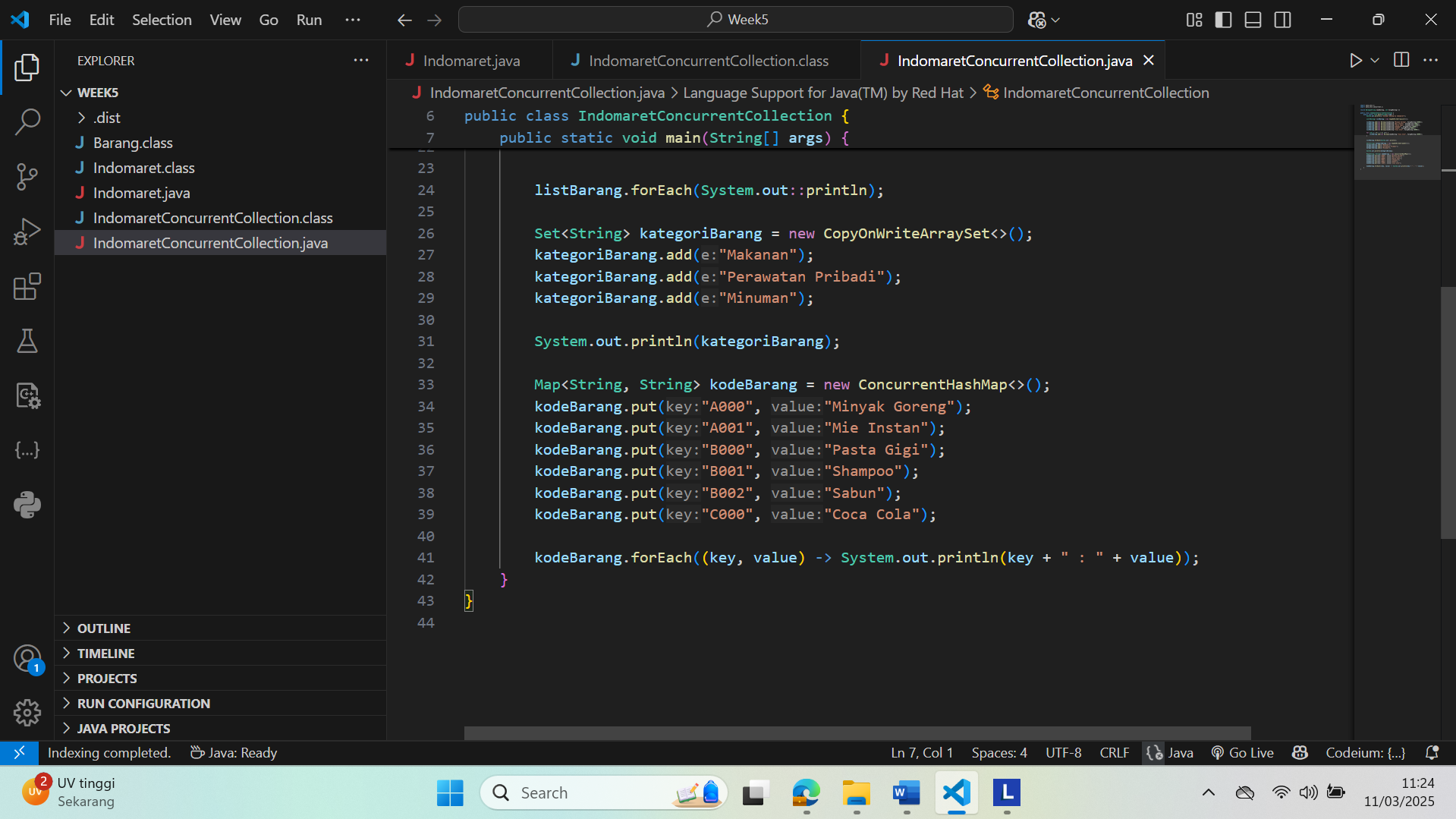
✅ Meningkatkan Performa → Mengurangi overhead sinkronisasi manual pada koleksi biasa (HashMap, ArrayList, dll).

✅ Menyediakan Struktur Data Khusus → Contoh: ConcurrentHashMap, CopyOnWriteArrayList, BlockingQueue, dll.

🔹 Kesimpulan: Concurrent Collection sangat penting untuk aplikasi yang membutuhkan akses data cepat dan aman dalam lingkungan multithreading. 🚀

3. Implementasi





Disini saya menerapkan concurrent collection masih sama di file yang tadi dengan kasus Indomaret. Salah satu ciri concurrent collection adalah pada indexnya yaitu terdapat deklarasi “CopyOnArrayList”.

## 5. Queue dan Dequeue

1. Definisi

* Queue

The Queue Interface is present in java.util package and extends the Collection interface. It stores and processes the data in FIFO(First In First Out) order. It is an ordered list of objects limited to inserting elements at the end of the list and deleting elements from the start of the list.

* Dequeue

Deque Interface present in java.util package is a subtype of the queue interface. The Deque is related to the double-ended queue that supports adding or removing elements from either end of the data structure. It can either be used as a queue(first-in-first-out/FIFO) or as a stack(last-in-first-out/LIFO). Deque is the acronym for double-ended queue.

2. Mengapa diperlukan

* Queue

✅ Mengantri Data Secara Terstruktur → Mengikuti prinsip FIFO (First In, First Out), cocok untuk pemrosesan antrian seperti tugas atau request.

✅ Mendukung Multithreading → Dengan BlockingQueue, thread bisa menunggu hingga elemen tersedia tanpa perlu polling manual.

✅ Meningkatkan Performa → Dibandingkan List, Queue lebih efisien untuk operasi enqueue (menambah) dan dequeue (menghapus).

✅ Digunakan dalam Algoritma dan Struktur Data → Seperti BFS (Breadth-First Search), scheduling, dan load balancing.

✅ Menyediakan Berbagai Implementasi → LinkedList, PriorityQueue, ArrayDeque, dan ConcurrentLinkedQueue.

🔹 Kesimpulan: Queue di Java sangat berguna untuk mengelola data yang diproses secara berurutan, baik dalam satu thread maupun lingkungan multithreading. 🚀Dequeue

* Deque

✅ Akses dari Kedua Ujung → Bisa menambah (addFirst, addLast) dan menghapus (removeFirst, removeLast) elemen dari depan & belakang.

✅ Lebih Fleksibel dari Queue → Mendukung FIFO (seperti Queue) & LIFO (seperti Stack).

✅ Efisien dalam Pemrosesan Data → Operasi di ujung koleksi lebih cepat dibandingkan List.

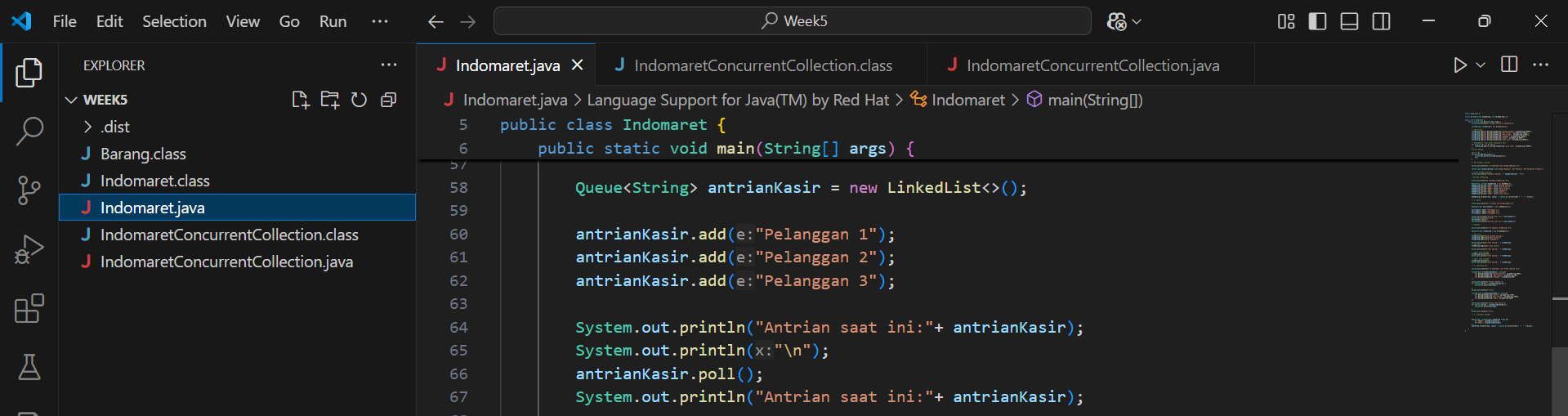
✅ Mendukung Multithreading → ConcurrentLinkedDeque bisa digunakan dalam aplikasi thread-safe.

✅ Digunakan dalam Algoritma Populer → Seperti Sliding Window, BFS (Breadth-First Search), dan Undo/Redo system.

🔹 Kesimpulan: Deque sangat berguna untuk struktur data dinamis yang memerlukan akses cepat dari dua arah. 🚀

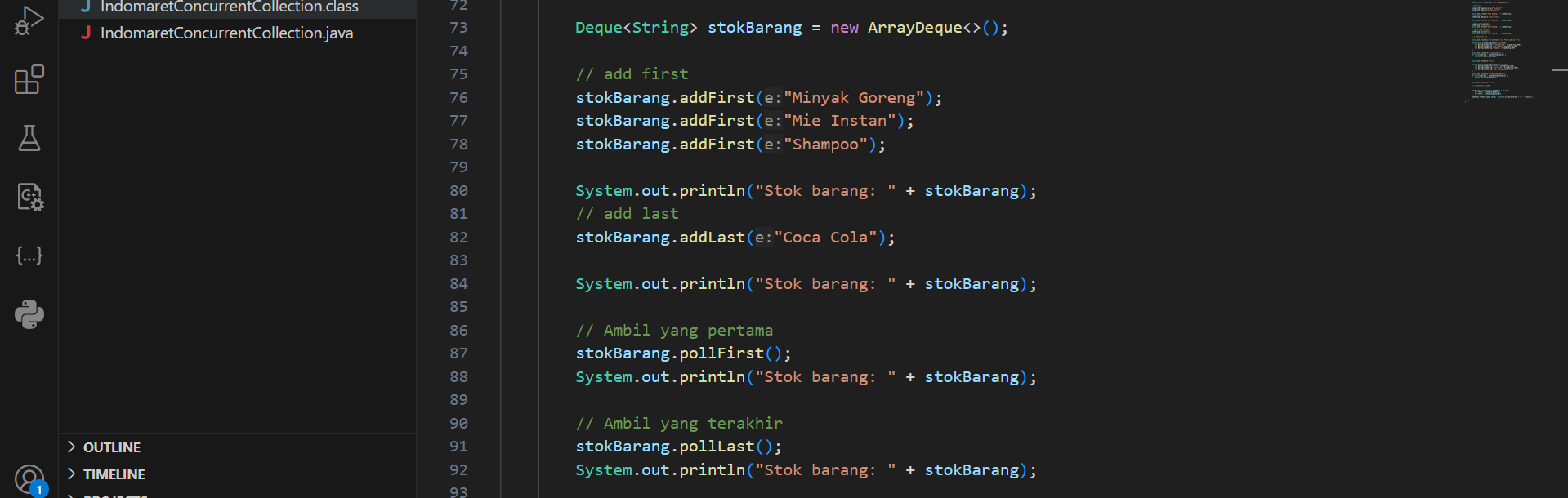
3. Implementasi

* Queue



Untuk implementasi queue di java, saya masih menggunakan kasus yang sama yaitu Indomaret, queue disini digunakan untuk membuat antrian pelanggan di Indomaret

* Deque



Untuk deque sendiri disini saya mengimplementasikannya dalam Deque stokBarang, yang dimana nanti stokBarang dapat diisi Barang dengan posisi dibelakang atau didepan antrian

## 6. Immutable collection List.of, Set.of, Map.of

1. Definisi

Immutable Collection adalah koleksi yang tidak bisa diubah setelah dibuat. Java menyediakan metode List.of(), Set.of(), dan Map.of() untuk membuat koleksi immutable dengan cara yang ringkas.

2. Mengapa diperlukan

✅ Mencegah Modifikasi Tidak Sengaja → Data tidak bisa diubah setelah dibuat, sehingga lebih aman.

✅ Thread-Safe Secara Default → Tidak perlu sinkronisasi tambahan dalam lingkungan multithreading.

✅ Meningkatkan Performa → Tidak ada overhead dari operasi tambah/hapus elemen.

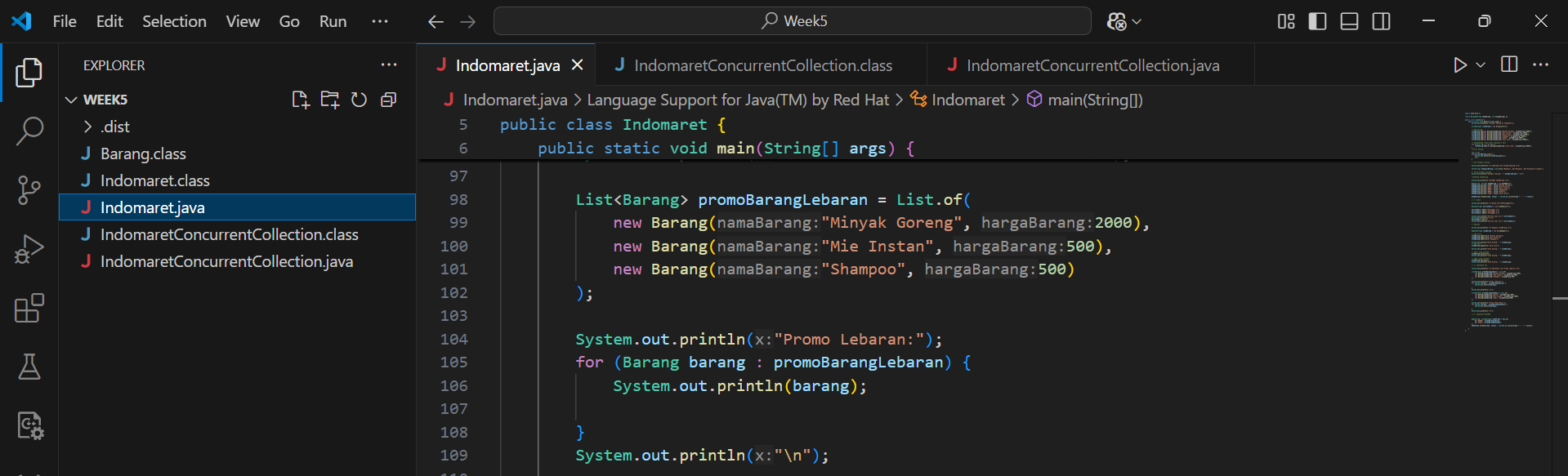
✅ Menghindari Bug dan Race Condition → Mengurangi risiko inkonsistensi data saat digunakan di berbagai bagian program.

✅ Lebih Simpel → Cara cepat membuat koleksi tanpa perlu Collections.unmodifiableXXX().

🔹 Kesimpulan: Immutable Collection sangat berguna untuk data yang bersifat tetap, meningkatkan keamanan, kinerja, dan kejelasan kode. 🚀

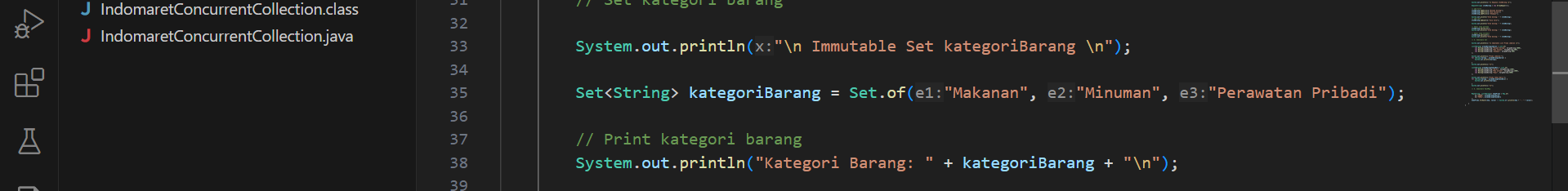
3. Implementasi

* List Immutable



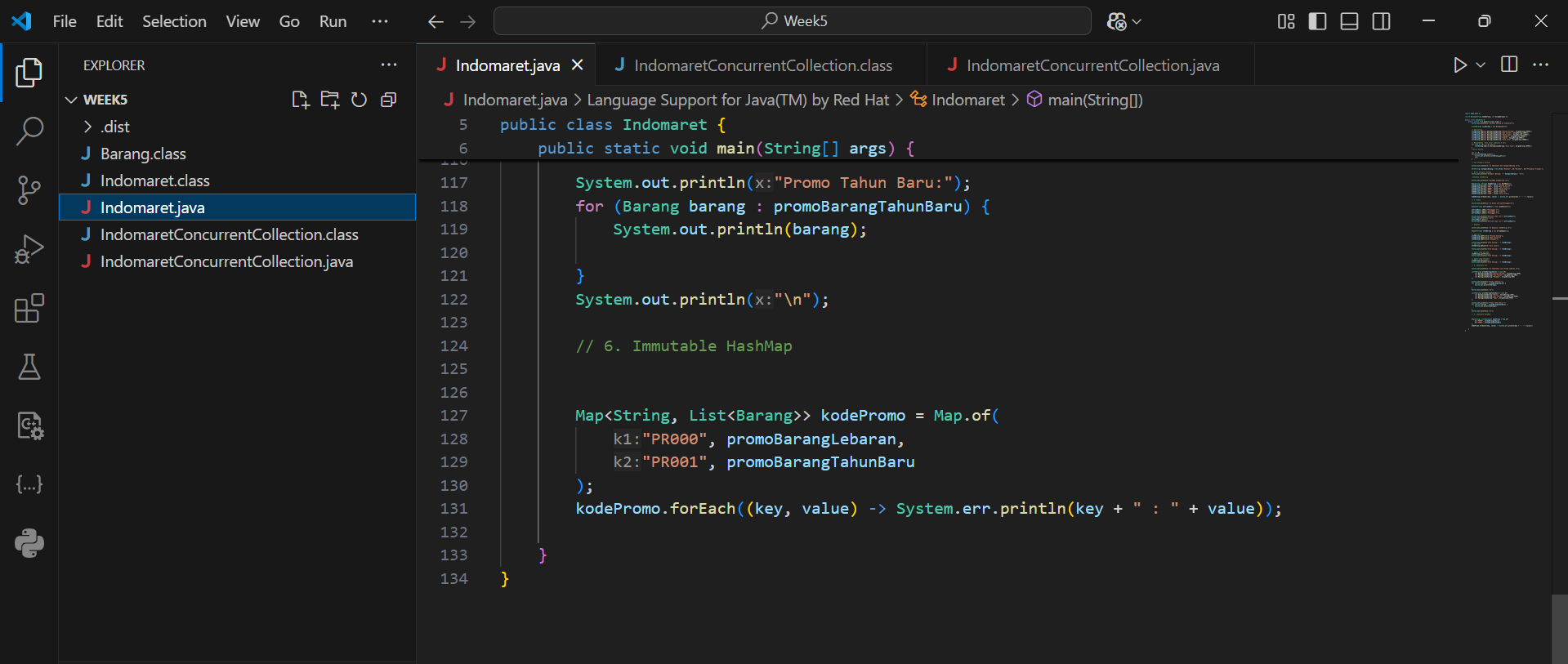
Untuk implementasi List immutable, disini saya membuat immutable list promoBarangLebaran dimana list ini hanya berlaku di hari lebaran saja

* Set Immutable



Untuk set immutable saya disini menggunakan set yang lama yaitu Set kategoriBar karena kategoriBarang tidak akan diubah ubah sampai kapanpun

* Map Immutable



Untuk map immutable disini saya membuat map dengan key: kodePromo dan value yaitu diambil dari Barang promoBarangLebaran