Laporan Individu Tugas Besar

# 1. Identitas

Nama: Muhammad Gholan Firmansyah  
NIM: 1201220031  
Modul Transaksi / Checkout  
Fitur yang Dikerjakan:   
- konfirmasi pesanan  
- mengubah pesanan  
- hapus pesanan

# 2. Teknik Konstruksi yang Digunakan

Jelaskan teknik konstruksi (Unit Testing / Performance Testing / Design by Contract) yang digunakan.

|  |  |  |
| --- | --- | --- |
| Teknik | Penjelasan Singkat | Contoh di Proyek Kamu |
| Code Reuse | Menggunakan kembali kode yang sudah dibuat agar tidak perlu menulis ulang | Fungsi fungsi **PesananService** seperti **Tambah()**, **Hapus()**, dan **Ubah()** dipanggil ulang di **PesananController tanpa membuat ulang logikanya** |
| API | Antarmua yang memungkinkan aplikasi lain mengakses fitur/fungsi yang dibuat lewat internet | **PesananController** menyediakan endpoint HTTP (seperti POST, GET, DELETE) agar aplikasi luar bisa tambah, hapus, atau ambil data p |

# 3. Unit Testing

Tuliskan kode unit testing yang telah dibuat beserta penjelasan singkat.



Tambah\_ShouldAddNewPesanan

Menguji apakah metode Tambah berhasil menambahkan pesanan baru dengan benar.

Dicek bahwa properti Id, Produk, Jumlah, dan Harga sesuai.

GetAll\_ShouldReturnAllPesanan

Menguji apakah semua pesanan yang ditambahkan bisa diambil lewat GetAll.

Dicek jumlah total pesanan yang dikembalikan.

GetById\_ShouldReturnCorrectPesanan

Menguji apakah pesanan bisa diambil berdasarkan ID-nya.

Dicek apakah data pesanan yang dikembalikan sesuai dengan yang ditambahkan.

Konfirmasi\_ShouldMarkPesananAsConfirmed

Menguji apakah pesanan bisa dikonfirmasi (Terkonfirmasi = true).

Dicek bahwa status Terkonfirmasi benar-benar berubah.

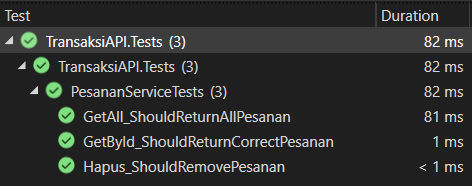
Hapus\_ShouldRemovePesanan

Menguji apakah pesanan bisa dihapus.

Dicek bahwa setelah dihapus, pesanan tidak ditemukan lagi (null).

# 4. Performance Testing

Tuliskan hasil dan penjelasan pengujian performa (jika dikerjakan).

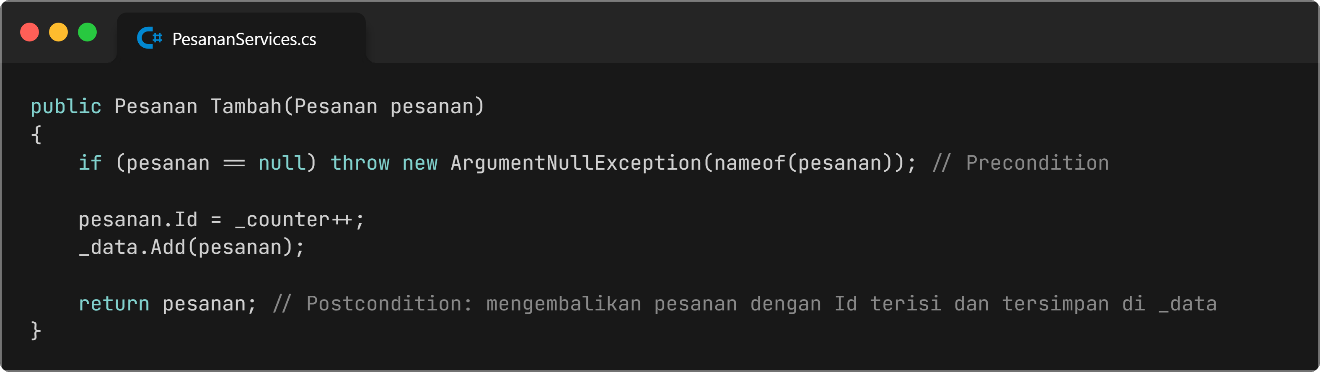


GetAll\_ShouldReturnAllPesanan: Paling lama (81 ms) – mungkin karena test ini menambahkan beberapa data dulu sebelum GetAll(), sehingga ada lebih banyak operasi.

GetById\_ShouldReturnCorrectPesanan: Hanya 1 ms – cepat karena hanya ambil 1 data.

Hapus\_ShouldRemovePesanan: Sangat cepat (<1 ms), karena hanya satu data dihapus dan dicek.

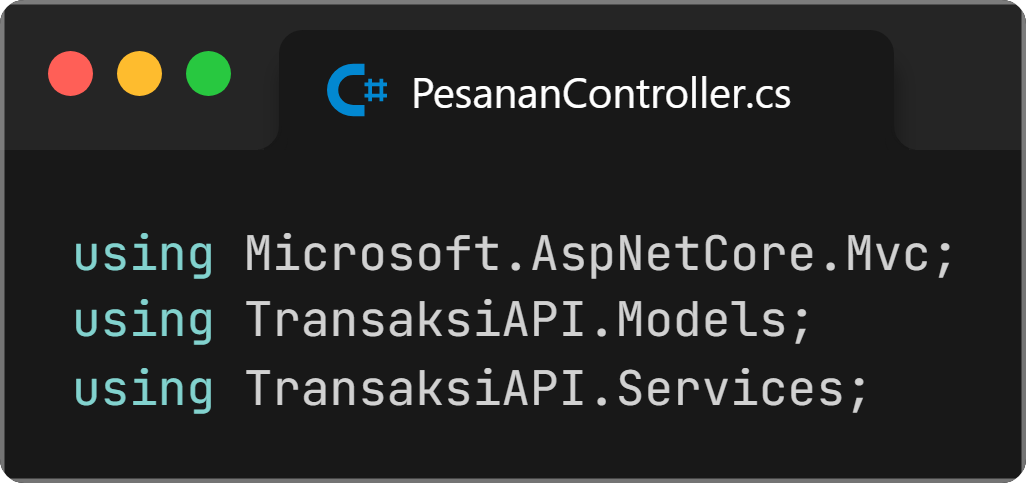
# 5. Design by Contract

Jelaskan implementasi design by contract yang digunakan dalam fitur Anda.

Implementasi **Design by Contract (DbC)** dalam fitur PesananService dapat dilihat dari cara metode seperti Tambah menetapkan *precondition*, *postcondition*, dan *invariant*. Sebagai contoh, dalam metode Tambah, terdapat *precondition* berupa pengecekan bahwa objek pesanan tidak boleh null (throw new ArgumentNullException). Setelah itu, *postcondition*-nya memastikan bahwa objek pesanan memiliki Id yang valid dan berhasil ditambahkan ke dalam koleksi internal \_data. Selain itu, terdapat *invariant* tidak langsung berupa konsistensi data pada \_data dan kenaikan nilai \_counter secara berurutan, yang dijaga di sepanjang siklus hidup objek PesananService. Pendekatan ini memastikan bahwa metode bekerja dalam batas kontrak yang jelas dan dapat diprediksi.

# 6. Penjelasan Source Code

Jelaskan bagian kode yang Anda kerjakan, dan bagaimana bagian tersebut bekerja.

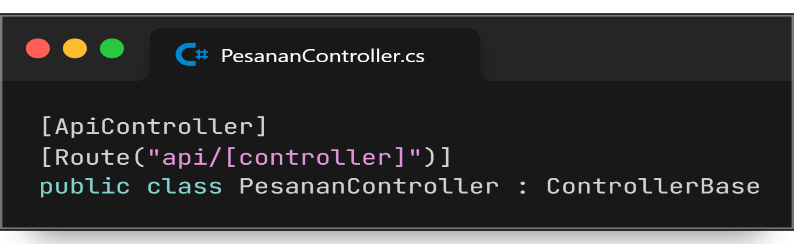
Pada bagian **Pesanan Controller**, ada bagian bagian kode yang saya kerjakan  


Ini adalah import file-file penting:

Microsoft.AspNetCore.Mvc → untuk membuat API.

TransaksiAPI.Models dan Services → mengambil model dan service buatan sendiri.

### Nama Controller



[ApiController] artinya file ini dipakai untuk API (bukan halaman web biasa).

[Route("api/[controller]")] artinya URL-nya nanti jadi api/pesanan (otomatis ambil nama controller).

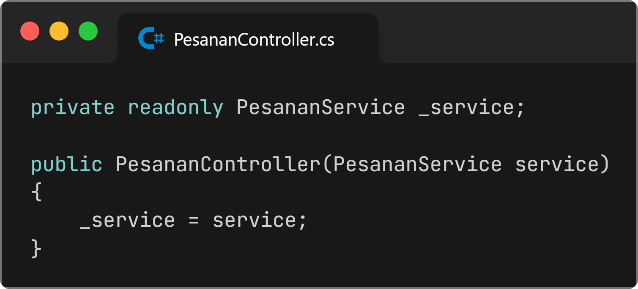
### Contructor dan Service

Di sini kita simpan PesananService ke dalam variabel \_service.

Fungsinya: semua urusan data pesanan (ambil, simpan, hapus, dll) dilakukan lewat service ini.

Berkut ini beberapa bagian bagian penting yang dipakaai untuk mengakses API nya

### Ambil Semua Pesanan



Kalau user akses GET api/pesanan

Maka akan dikembalikan semua pesanan.

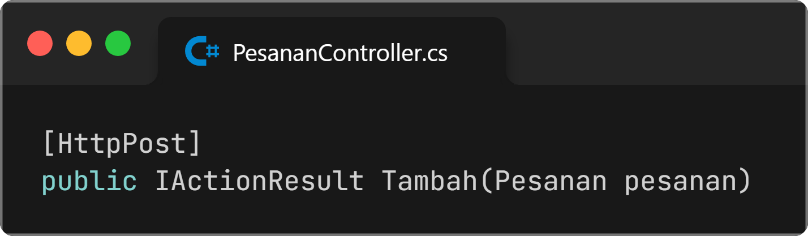
### Ambil Satu Pesanan Berdasarkan ID

Kalau user akses GET api/pesanan/2

Maka API akan cari pesanan dengan ID 2.

Kalau ketemu → dikembalikan, kalau tidak → NotFound() (tidak ditemukan).

### Tambah Pesanan Baru



Kalau user kirim data POST ke api/pesanan

Maka pesanan baru akan dibuat.

Responnya: data yang baru ditambah + link untuk akses data itu.

### Ubah Pesanan Berdasarkan ID

Kalau user kirim PUT api/pesanan/5 dengan data baru,

Maka pesanan dengan ID 5 akan diubah.

Kalau berhasil → Ok, kalau tidak ketemu → NotFound().

### Konfirmasi Pesanan

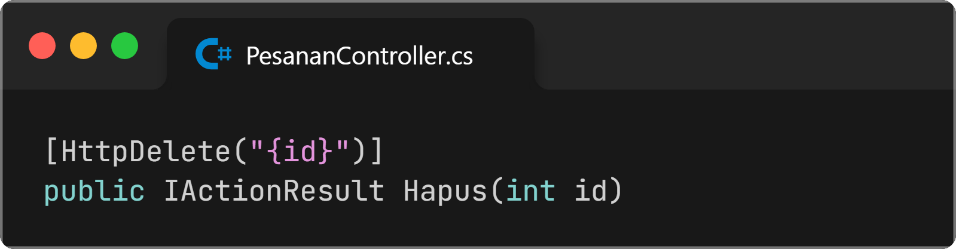


Kalau user akses PATCH api/pesanan/5/konfirmasi

Maka pesanan dengan ID 5 akan dikonfirmasi.

Misalnya status-nya berubah dari "pending" jadi "dikonfirmasi".

### Hapus Pesanan



Kalau user akses DELETE api/pesanan/3

Maka pesanan dengan ID 3 akan dihapus.

Kalau berhasil → Ok, kalau gagal (misalnya ID tidak ada) → NotFound().

Berikutnya pada bagian **Pesanan :**

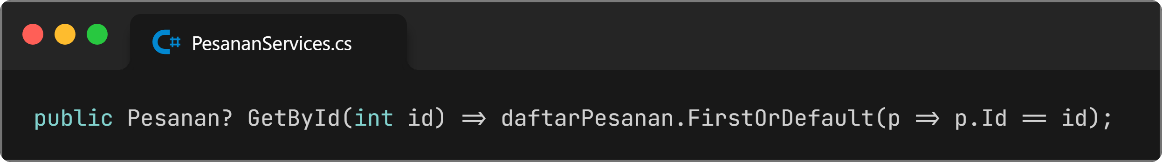
* **public int Id { get; set; } //** nomor unik untuk membedakan setiap pesanan.
* **public string Produk { get; set; } //** Nama produk yang dipesan.
* **public int Jumlah { get; set; } //** Jumlah barang yang dipesan.
* **public decimal Harga { get; set; } //** Harga total dari pesanan.
* **public string Status { get; set; } = "menunggu\_konfirmasi"; //** Menyimpan status pesanan saat ini.
* **public DateTime Tanggal { get; set; } = DateTime.Now; //** Tanggal dan waktu saat pesanan dibuat.

Berikutnya pada bagian **PesananServices**

### Ambil Semua Pesanan

Mengembalikan seluruh isi daftarPesanan.

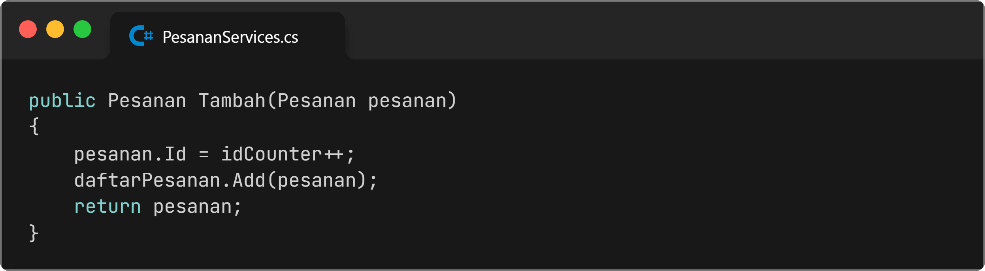
### Cari Pesanan Berdasarkan ID



Mencari satu pesanan yang ID-nya sesuai.

Kalau ketemu, dikembalikan. Kalau tidak, null.

### Tambah Pesanan Baru

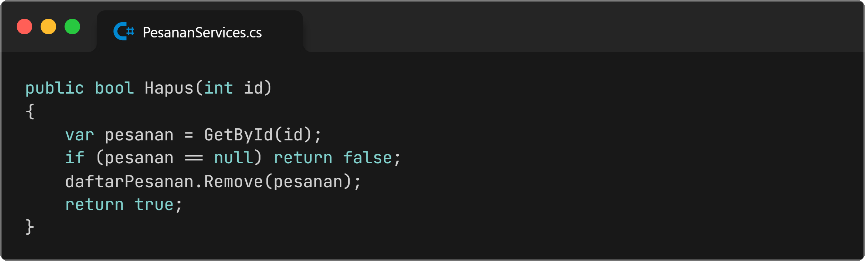


Pesanan baru diberi ID secara otomatis (idCounter).

Lalu dimasukkan ke daftarPesanan.

Lalu dikembalikan hasilnya.

### Hapus Pesanan



Cari dulu pesanan berdasarkan ID.

Kalau ketemu, hapus dari daftar.

Kembalikan true kalau berhasil, false kalau tidak ditemukan.

### Ubah Pesanan

Cari pesanan berdasarkan ID.

Kalau ketemu, update datanya sesuai yang baru (produk, jumlah, harga).

Status dan tanggal tidak diubah di sini.

Kembalikan true kalau berhasil.

### Konfirmasi Pesanan

Cari pesanan berdasarkan ID.

Kalau ketemu, ubah statusnya jadi "dikonfirmasi".

Kembalikan true kalau berhasil.