

Laporan Pertemuan 7

Nama : [Al Jatsiya Profitar Taqwala]
NIM : [H1D024095]
Shift KRS : [B]
Shift Baru : [I]

1) Alur Kerja Program

Program dieksekusi secara berurutan mulai dari method main di kelas PaymentTest. Berikut adalah langkah demi langkah apa yang terjadi saat program dijalankan:

a) Inisialisasi Objek (Setup):

- ▶ Program dimulai di PaymentTest. Sebuah objek EWalletPayment dibuat (diinstansiasi). Pada tahap ini, constructor dipanggil untuk menyimpan data awal ke dalam memori, yaitu: nama layanan (misal: "OVO"), nominal yang akan dibayar (50.000), dan saldo awal (150.000),

b) Pengecekan Awal:

- ▶ Program memanggil method getBalance() untuk menampilkan saldo awal pengguna sebelum transaksi terjadi, sesuai dengan format output yang diharapkan.

c) Proses Transaksi:

Program memanggil method utama processPayment(). Di dalam method ini terjadi logika utama:

- ▶ Program mengambil biaya transaksi melalui getTransactionFee() (sebesar 2.000).
- ▶ Program menjumlahkan Nominal Pembayaran + Biaya Transaksi.
- ▶ Program melakukan validasi saldo:
 - Jika Saldo Cukup: Saldo pengguna dikurangi total biaya, dan pesan "Pembayaran berhasil!" ditampilkan.
 - Jika Saldo Kurang: Saldo tidak berubah, dan pesan "Pembayaran gagal!" ditampilkan.

d) Output:

- ▶ Setelah proses selesai, program kembali memanggil getBalance() untuk menampilkan sisa saldo terkini dan getPaymentDetails() untuk mencetak ringkasan transaksi (nama penyedia layanan) ke layar.

2) Encapsulation Methods (Getter & Setter)

Program ini menggunakan empat method utama yang dideklarasikan dalam Interface PaymentMethod dan diimplementasikan ulang (override) di dalam class EWalletPayment.

a) processPayment()

- ▶ Fungsi: Merupakan "otak" dari program ini. Method ini menangani logika bisnis pembayaran.
- ▶ Cara Kerja: Method ini menghitung total pengeluaran (harga barang + pajak/admin) dan membandingkannya dengan saldo yang tersedia. Method ini juga yang bertanggung jawab mengubah (mengurangi) nilai saldo jika transaksi valid .

b) getTransactionFee()

- ▶ Fungsi: Mengembalikan nilai biaya administrasi atau biaya transaksi.
- ▶ Cara Kerja: Dalam studi kasus ini, method ini mengembalikan nilai tetap (hardcoded) sebesar 2000.0 sesuai contoh perhitungan pada output yang diharapkan ($150.000 - 50.000 - 2.000 = 98.000$).

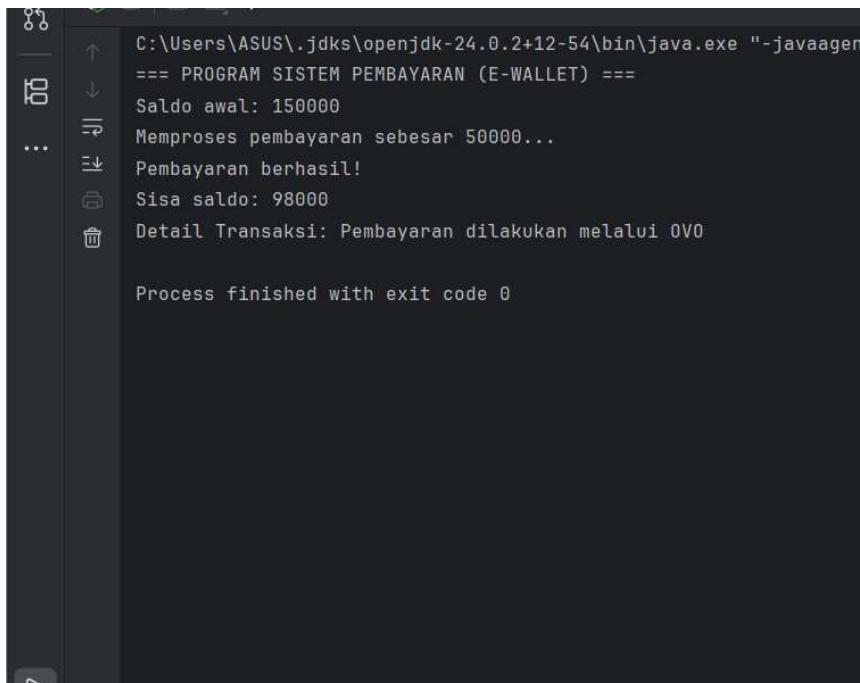
c) getBalance()

- ▶ Fungsi: Sebagai getter untuk mengakses data saldo pengguna yang bersifat private.
- ▶ Cara Kerja: Mengembalikan nilai double yang mewakili sisa uang pengguna saat ini. Digunakan untuk verifikasi sebelum transaksi dan pelaporan setelah transaksi.

d) getPaymentDetails()

- ▶ Fungsi: Memberikan informasi deskriptif mengenai transaksi.
- ▶ Cara Kerja: Mengembalikan String yang berisi teks detail, termasuk nama penyedia layanan e-wallet (misalnya: "Pembayaran dilakukan melalui OVO").

3) Hasil Output



```
C:\Users\ASUS\.jdks\openjdk-24.0.2+12-54\bin\java.exe "-javaagent:C:/Users/ASUS/.jdks/openjdk-24.0.2+12-54/lib/jvm-debugger.jar" -Dfile.encoding=UTF-8 -jar C:/Users/ASUS/Desktop/E-Wallet.jar
== PROGRAM SISTEM PEMBAYARAN (E-WALLET) ==
Saldo awal: 150000
Memproses pembayaran sebesar 50000...
Pembayaran berhasil!
Sisa saldo: 98000
Detail Transaksi: Pembayaran dilakukan melalui OVO

Process finished with exit code 0
```