

LAPORAN PERTEMUAN 7

NAMA : AHMAD FIKRI ZAKARIA
NIM : H1D024062
MATERI : INTERFACE

A. Alur Kerja (Workflow)

Pada pertemuan ini, saya mengembangkan simulasi sistem pembayaran digital (*E-Wallet*) yang menerapkan konsep **Interface**. Fokus utamanya adalah memisahkan antara deklarasi layanan (kontrak) dengan implementasi logika bisnisnya. Langkah pengerjaan meliputi:

1. **Pembuatan Interface (PaymentMethod):**
 - Mendefinisikan kontrak method yang wajib ada dalam sistem pembayaran: `processPayment()`, `getPaymentDetails()`, `getTransactionFee()`, dan `getBalance()`.
 - Tidak ada kode logika sama sekali di file ini, murni hanya deklarasi.
2. **Implementasi Class (EWalletPayment):**
 - Membuat class yang mengadopsi interface menggunakan keyword `implements` `PaymentMethod`.
 - Menambahkan atribut `namaLayanan`, `saldo`, dan `nominalPembayaran`.
 - **Mengisi Logika Method:**
 - `processPayment()`: Menambahkan logika pengurangan saldo jika dana mencukupi (termasuk biaya admin).
 - `getTransactionFee()`: Menetapkan biaya admin statis sebesar Rp 2.000.
 - Memastikan semua method dari interface telah di-override agar class dapat dikompilasi.
3. **Pengujian (PaymentTest):**
 - Membuat objek `EWalletPayment` dengan skenario saldo awal Rp 150.000 dan tagihan Rp 50.000.
 - Memanggil method-method interface untuk memverifikasi bahwa saldo berkurang dengan benar dan status pembayaran berhasil ditampilkan.

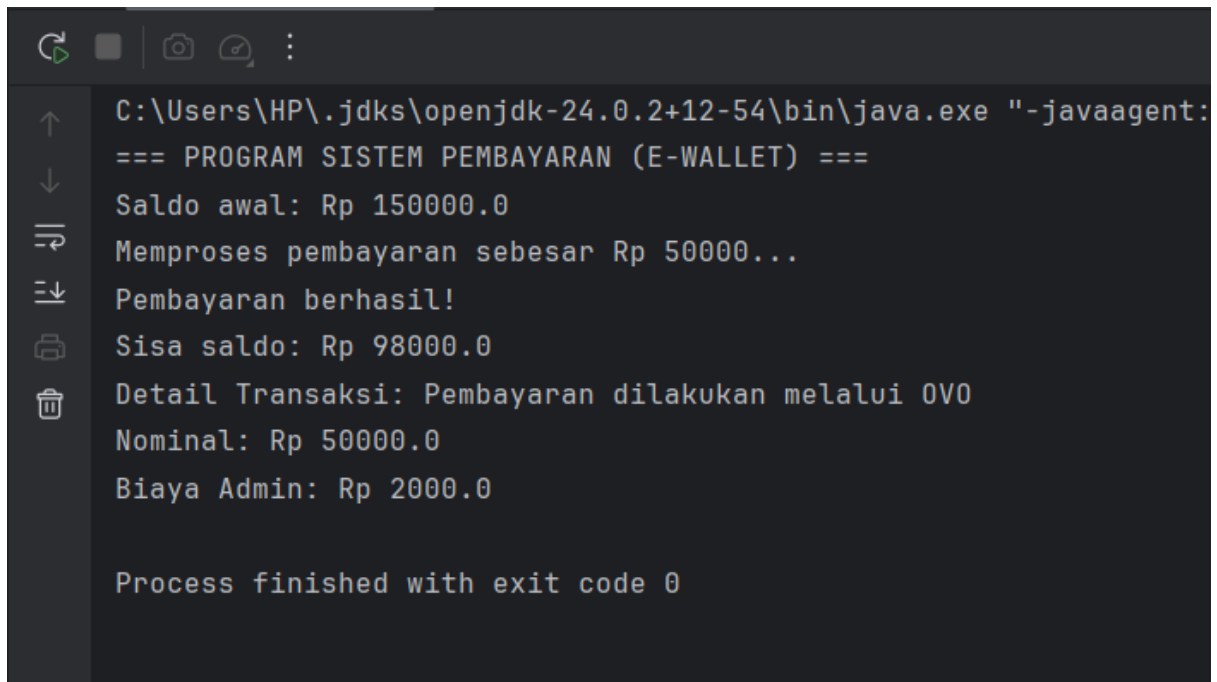
B. Penjelasan Kode & Fungsi

Berikut adalah analisis mendalam mengenai konsep teknis yang diterapkan dalam kode program:

1. **Interface sebagai Kontrak** Interface `PaymentMethod` bertindak sebagai standar baku. Siapapun yang ingin membuat fitur pembayaran (misal nanti ada `CreditCardPayment` atau `BankTransferPayment`) harus memiliki method yang sama (`processPayment`, dll). Ini menjamin konsistensi struktur kode dalam sistem yang besar.
2. **Keyword implements** Class `EWalletPayment` menggunakan `implements` untuk menyepakati kontrak tersebut. Konsekuensinya, class ini **wajib** menuliskan ulang (*Override*) isi dari semua method yang ada di Interface. Jika satu saja terlewat, program akan error dan dianggap class abstrak.
3. **Logika Transaksi** Pada method `processPayment`, diterapkan logika bisnis sederhana: $\text{Saldo Akhir} = \text{Saldo Awal} - (\text{Nominal} + \text{Fee})$. Program melakukan validasi terlebih dahulu; jika saldo kurang, transaksi dibatalkan tanpa mengurangi saldo, menjaga integritas data keuangan pengguna.

C. Hasil Output Program

Berikut adalah tangkapan layar/teks hasil eksekusi program UjiPegguna.java:



```
C:\Users\HP\.jdk\openjdk-24.0.2+12-54\bin\java.exe "-javaagent:  
=== PROGRAM SISTEM PEMBAYARAN (E-WALLET) ===  
Saldo awal: Rp 150000.0  
Memproses pembayaran sebesar Rp 50000...  
Pembayaran berhasil!  
Sisa saldo: Rp 98000.0  
Detail Transaksi: Pembayaran dilakukan melalui OVO  
Nominal: Rp 50000.0  
Biaya Admin: Rp 2000.0  
  
Process finished with exit code 0
```