

LAPORAN MINGGU 6

Topik: Desain Arsitektur Sistem dengan UML dan Prinsip SOLID

Nama: Zaki Saputra

NIM: 240202847

Kelas: 3IKRA

Program Studi: Ilmu Komputer

Universitas: Universitas Putra Bangsa Kebumen

A. DESKRIPSI SINGKAT SISTEM

Sistem **Agri-POS (Point of Sale Pertanian)** adalah aplikasi kasir yang dirancang untuk toko atau usaha penjualan produk pertanian (benih, pupuk, alat, obat). Sistem ini bertujuan untuk memfasilitasi manajemen produk, transaksi penjualan tunai dan e-wallet, serta pencetakan struk dan pelaporan penjualan harian/periodik. Sistem membedakan hak akses antara **Kasir** (fokus pada transaksi) dan **Admin** (fokus pada manajemen produk dan laporan).

B. PENJELASAN DIAGRAM DAN KETERKAITANNYA

Catatan Penting untuk Mahasiswa:

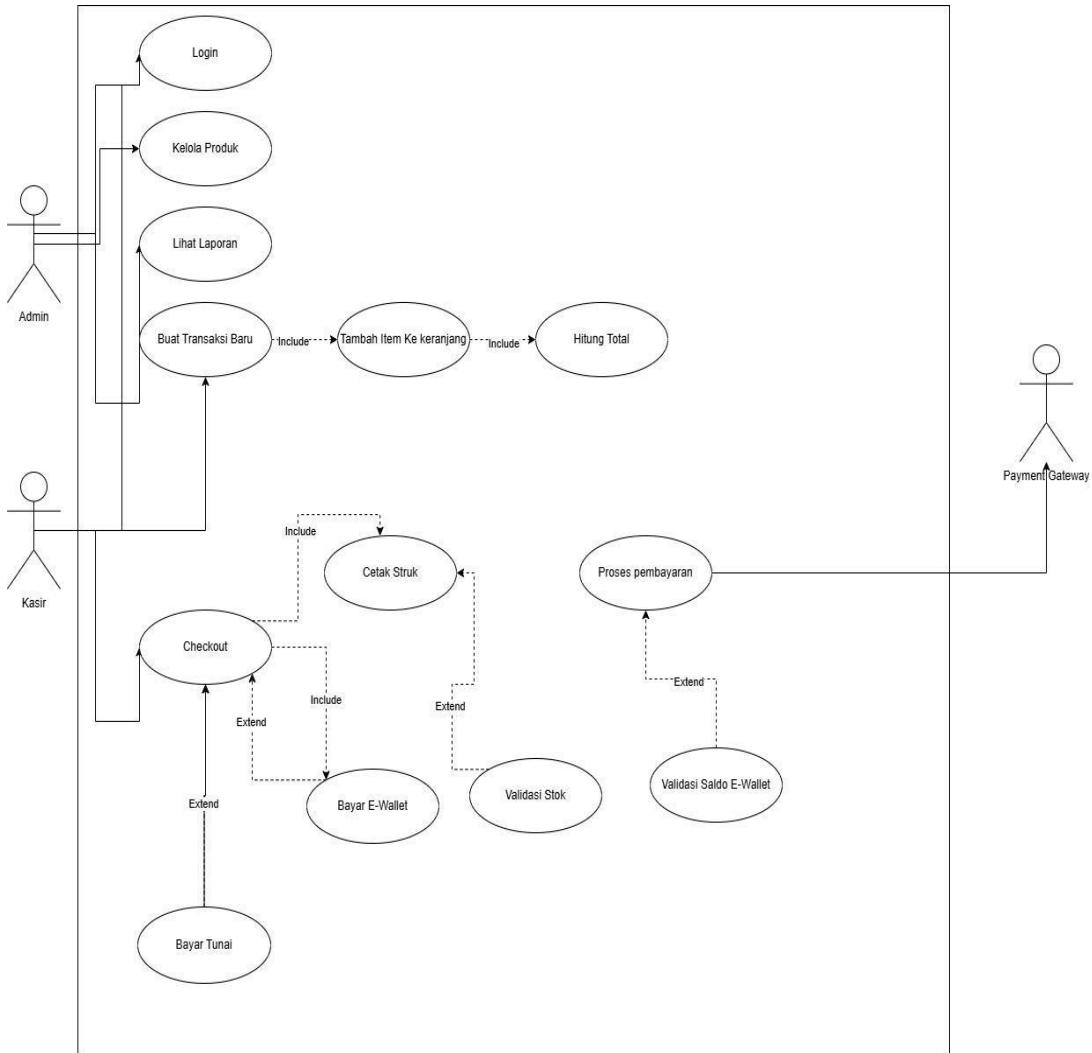
Pada setiap subbab diagram di bawah ini, **WAJIB menyertakan gambar diagram UML**. Letakkan gambar **setelah judul subbab** dan sebelum penjelasan naratif.

Format penamaan gambar yang disarankan:

Gambar X.X Nama Diagram – Sistem Agri-POS

Desain sistem Agri-POS direpresentasikan melalui empat diagram UML utama untuk memetakan kebutuhan fungsional dan non-fungsional, serta memastikan struktur yang modular dan *maintainable*.

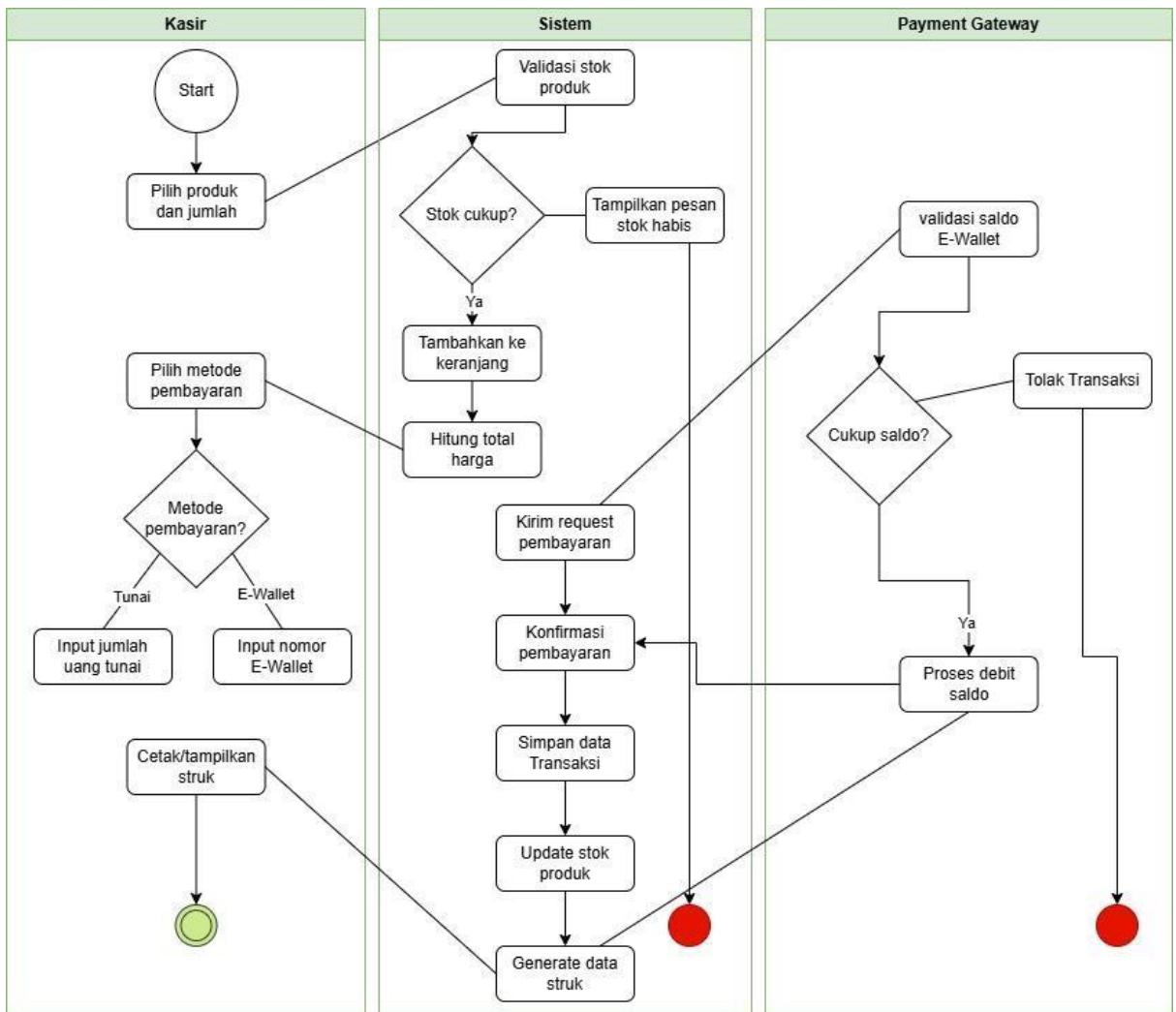
1. Use Case Diagram (UCD)



(Diagram berisi aktor Admin dan Kasir beserta seluruh use case yang telah didefinisikan)

- **Fungsi:** Menjelaskan fungsionalitas sistem dari sudut pandang pengguna (aktor).
- **Aktor:**
 - **Admin:** Mengelola produk dan melihat laporan.
 - **Kasir:** Melakukan login, transaksi, dan pembayaran.
- **Keterkaitan:**
 - a) Proses *Buat Transaksi* dan *Tambah Produk ke Keranjang* mengarah ke *Selesaikan Pembayaran (Checkout)*.
 - b) *Selesaikan Pembayaran (Checkout)* mencakup (*include*) *Cetak Struk* dan dapat diperluas (*extend*) menjadi *Pembayaran Tunai* atau *Pembayaran E-Wallet*.
 - c) *Pembayaran E-Wallet* berinteraksi (*uses*) dengan *Payment Gateway*.
 - d) *Pembayaran Tunai* menyertakan (*include*) *Hitung Total*.

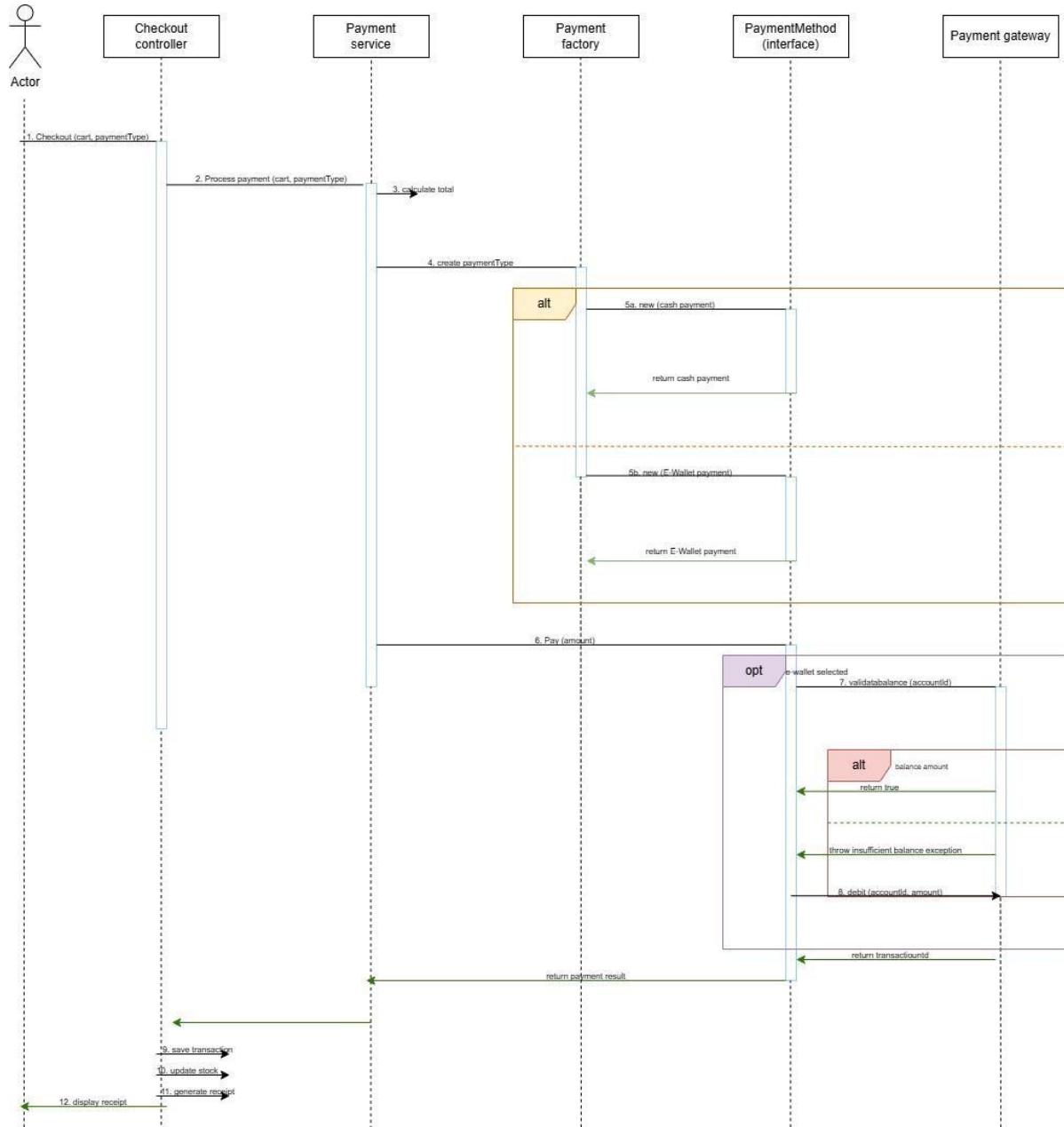
2. Activity Diagram (ACD)



- **Fungsi:** Menggambarkan alur kerja proses utama sistem, yaitu proses *Checkout* dan *Pembayaran*.
- **Swimlane:** Kasir, Sistem, dan Payment Gateway.
- **Alur Utama:** Kasir memulai dari *Scan Produk* hingga *Pilih Metode Pembayaran*.
- **Skenario Pembayaran:**
 - **Tunai:** Sistem *Hitung Total* → Kasir *Input Nominal Pembayaran* → Sistem mengecek *Uang Cukup?*
 - **YA:** Proses tunai selesai.
 - **TIDAK:** Opsi *Tolak & Minta Uang* atau *Batalkan Transaksi*.
 - **E-Wallet:** Sistem menampilkan QRIS → Interaksi dengan *Payment Gateway* untuk *Verifikasi Saldo & Status Akun* → Sistem mengecek *Otorisasi Berhasil?*
 - **YA:** Proses e-wallet selesai.
 - **TIDAK:** Opsi *Tolak & Minta Ulang* atau *Batalkan Transaksi*.
- **Akhir Proses:** Pembayaran berhasil dilanjutkan ke *Update Stok Produk*, *Catat Transaksi Selesai*, dan *Cetak Struk*.

- **Keterkaitan:** ACD merinci UC *Selesaikan Pembayaran (Checkout)*, termasuk *decision node* untuk alur sukses/gagal.

3. Sequence Diagram (SD)

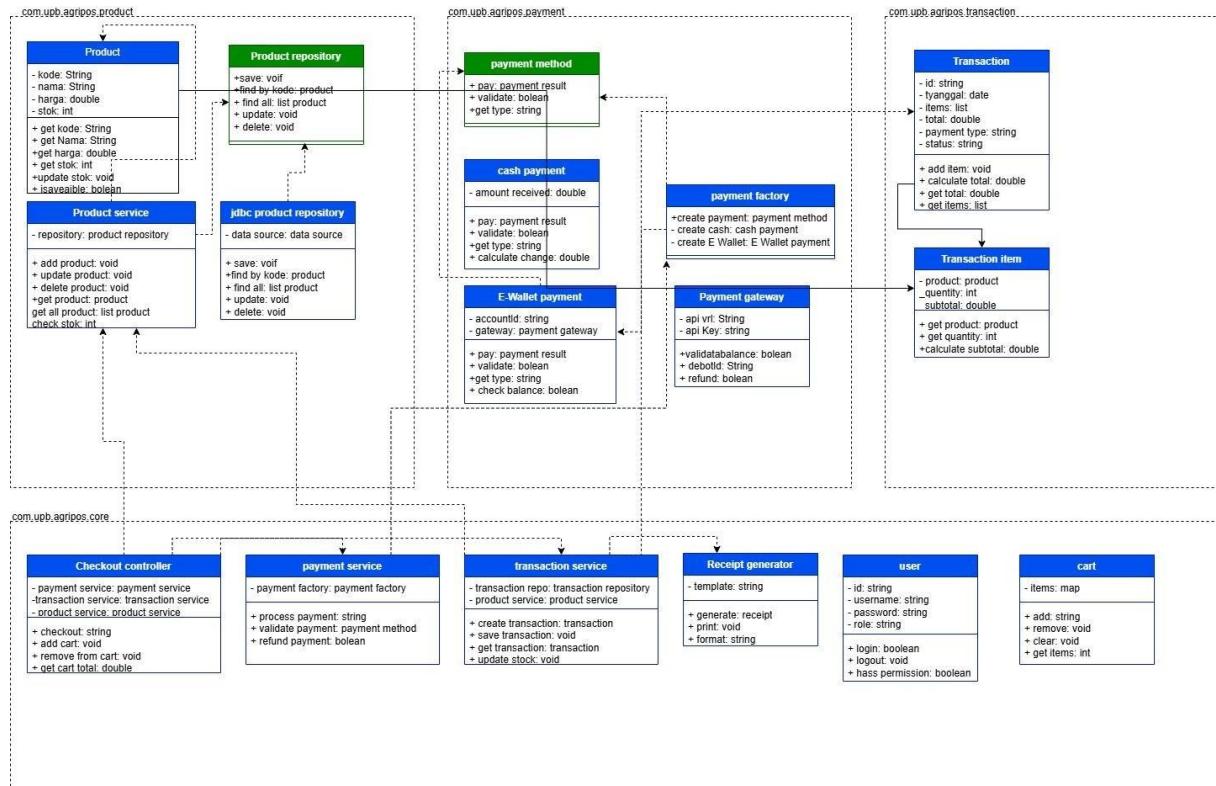


(Diagram memuat skenario pembayaran tunai dan e-wallet beserta alur gagal)

- **Fungsi:** Menunjukkan urutan interaksi pesan antar objek selama proses pembayaran.
- **Lifeline:** Kasir, UI/Sistem POS, PaymentService System, dan Payment Gateway.
- **Skenario Alternatif (alt):** Pembayaran Tunai dan Pembayaran E-Wallet.
- **Error Handling:**
 - **Tunai:** [Uang Kurang] → pesan gagal dan permintaan ulang.

- **E-Wallet:** [Pembayaran Gagal – Saldo Tidak Cukup] → Status FAILED/Kode Gagal dari Payment Gateway.
- **Inti Proses:** PaymentService melakukan validasi, inisiasi transaksi, dan pembaruan stok; Payment Gateway khusus untuk otorisasi e-wallet.
- **Keterkaitan:** SD mengimplementasikan aktivitas pada ACD, misalnya *Request Otorisasi* direalisasikan sebagai pesan dari UI ke Payment Gateway.

4. Class Diagram (CD)



(Diagram menampilkan struktur kelas, atribut, method, relasi, dan interface IPaymentMethod)

- **Fungsi:** Menjelaskan struktur statis sistem (kelas, atribut, operasi, dan relasi).
- **Kelas Utama:**
 - **Product:** kode, nama, kategori, harga, stok.
 - **Transaction:** id, date, totalAmount.
 - **CartItem:** product, quantity.
 - **ProductService:** addProduct, editProduct, deleteProduct.
 - **CartService:** addlItem, calculateTotal.
 - **PaymentService:** orkestrasi proses pembayaran.
- **Relasi Penting:**
 - **Inheritance:** CashPayment dan EWalletPayment mewarisi interface IPaymentMethod.

- **Asosiasi/Multiplicity:** Transaction memiliki 1..* CartItem; CartItem terkait dengan 1 Product; PaymentService menggunakan 1 IPaymentMethod.
- **Keterkaitan:** Kelas pada CD adalah objek yang berinteraksi pada SD dan menjalankan fungsi pada ACD dan UCD.

C. PENERAPAN PRINSIP SOLID

1) Single Responsibility Principle (SRP)

Setiap kelas memiliki satu tanggung jawab:

- ProductService: Manajemen produk.
- CartService: Manajemen keranjang.
- PaymentService: Orkestrasi pembayaran.
- Transaction: Penyimpanan status transaksi selesai.

2) Open/Closed Principle (OCP)

- PaymentService bergantung pada **IPaymentMethod**.
- Penambahan metode baru (mis. TransferBankPayment) cukup dengan implementasi baru tanpa mengubah PaymentService.

3) Liskov Substitution Principle (LSP)

- CashPayment dan EWalletPayment dapat saling menggantikan melalui IPaymentMethod tanpa merusak fungsionalitas.

4) Interface Segregation Principle (ISP)

- IPaymentMethod hanya memiliki method *pay()* yang relevan.
- Method khusus e-wallet dipisahkan ke interface lain bila diperlukan.

5) Dependency Inversion Principle (DIP)

- PaymentService bergantung pada abstraksi (IPaymentMethod), bukan implementasi konkret.
- Mendukung *dependency injection* dan *mocking* untuk pengujian.

TABEL TRACEABILITY (FR → DIAGRAM → KELAS/INTERFACE)

FR	Use Case	Activity/Sequence	Class/Interface
Manajemen Produk	UC Tambah/Ubah/Hapus/Lihat Produk	–	ProductService, Product
Transaksi Penjualan	UC Buat Transaksi, Tambah Keranjang, Hitung Total	Seq Pembayaran	CartService, CartItem, Transaction

FR	Use Case	Activity/Sequence	Class/Interface
Perhitungan Total	UC Hitung Total	Activity/Sequence	CartService
Metode Pembayaran	UC Checkout, Tunai, E-Wallet	Activity Checkout, Seq Pembayaran	IPaymentMethod, CashPayment, EWalletPayment, PaymentService
Cetak Struk	UC Cetak Struk	Activity/Sequence	Transaction
Laporan	UC Lihat Laporan	—	Transaction

D. KESIMPULAN DAN REFLEKSI

Keunggulan Sistem:

- Ekstensibilitas tinggi (OCP & DIP).
- Modular dan mudah dirawat (SRP).
- Traceability kuat antar diagram.

Potensi Pengembangan:

- Penerapan *Strategy Pattern* untuk diskon/promosi.
- Penambahan *Repository Pattern* untuk meningkatkan testability dan portability.

JAWABAN QUIZ

1. Perbedaan Aggregation dan Composition

Composition	Pembeda	Aggregation
Objek part ikut hancur	Ketergantungan Hidup	Objek part dapat berdiri sendiri
Berlian solid	Notasi UML	Berlian kosong
Transaction–CartItem	Contoh	ProductService–Product

2. Open/Closed Principle

OCP memisahkan variasi dari inti sehingga fitur baru dapat ditambahkan tanpa memodifikasi kode stabil.

3. Dependency Inversion Principle

DIP meningkatkan testability dengan memungkinkan penggunaan *mock* pada dependency abstrak (IPaymentMethod).

