**USE CASE DIAGRAM**

Berikut adalah deskripsi umum Use Case Diagram untuk Sistem Informasi Akuntansi sebuah Apotek:

\*\*Aktor:\*\*

1. \*\*Admin\*\* - Bertanggung jawab untuk mengelola sistem, seperti menambah atau memperbarui data produk, transaksi, dan keuangan.

2. \*\*Kasir\*\* - Mengelola transaksi penjualan obat.

3. \*\*Pemilik Apotek\*\* - Memantau laporan keuangan dan stok barang.

4. \*\*Supplier\*\* - Mengelola pengiriman stok barang.

\*\*Use Cases:\*\*

1. \*\*Admin:\*\*

- Menambah data produk.

- Memperbarui data produk.

- Menambahkan data transaksi keuangan.

- Mengelola data supplier.

- Mengelola laporan keuangan.

2. \*\*Kasir:\*\*

- Melakukan transaksi penjualan.

- Mengelola stok barang.

- Mencetak struk penjualan.

3. \*\*Pemilik Apotek:\*\*

- Melihat laporan keuangan.

- Melihat laporan penjualan.

- Memantau stok barang.

4. \*\*Supplier:\*\*

- Mengirim barang.

\*\*Diagram Use Case:\*\*

1. Admin berinteraksi dengan Use Case:

- Menambah data produk

- Memperbarui data produk

- Menambah data transaksi keuangan

- Mengelola data supplier

- Mengelola laporan keuangan

2. Kasir berinteraksi dengan Use Case:

- Melakukan transaksi penjualan

- Mengelola stok barang

- Mencetak struk penjualan

3. Pemilik Apotek berinteraksi dengan Use Case:

- Melihat laporan keuangan

- Melihat laporan penjualan

- Memantau stok barang

4. Supplier berinteraksi dengan Use Case:

- Mengirim barang

Jika ini sesuai, saya bisa menggambarkannya dalam bentuk diagram UML.

Berikut adalah kode Mermaid untuk Use Case Diagram Sistem Informasi Akuntansi Apotek:

```mermaid

%% Use Case Diagram Sistem Informasi Akuntansi Apotek

%% Aktor dan relasi dengan use case

actor Admin

actor Kasir

actor PemilikApotek as "Pemilik Apotek"

actor Supplier

%% Daftar Use Case

Admin --> (Menambah data produk)

Admin --> (Memperbarui data produk)

Admin --> (Menambah data transaksi keuangan)

Admin --> (Mengelola data supplier)

Admin --> (Mengelola laporan keuangan)

Kasir --> (Melakukan transaksi penjualan)

Kasir --> (Mengelola stok barang)

Kasir --> (Mencetak struk penjualan)

PemilikApotek --> (Melihat laporan keuangan)

PemilikApotek --> (Melihat laporan penjualan)

PemilikApotek --> (Memantau stok barang)

Supplier --> (Mengirim barang)

```

Kode di atas bisa digunakan di tool yang mendukung Mermaid, seperti editor teks atau platform yang mendukung Markdown dengan rendering Mermaid.

**SEQUENCE DIAGRAM**

Berikut adalah deskripsi umum \*\*Sequence Diagram\*\* untuk Sistem Informasi Akuntansi sebuah Apotek. Saya akan menjelaskan interaksi utama, misalnya antara \*\*Kasir\*\*, \*\*Sistem\*\*, dan \*\*Pemilik Apotek\*\*, dalam proses penjualan dan pembuatan laporan keuangan.

### Deskripsi Sequence Diagram:

1. \*\*Kasir\*\* melakukan transaksi penjualan obat.

2. \*\*Sistem\*\* mencatat transaksi penjualan, menghitung total biaya, dan mengurangi stok barang.

3. \*\*Sistem\*\* memberikan konfirmasi kepada \*\*Kasir\*\* tentang transaksi yang berhasil.

4. \*\*Sistem\*\* menghasilkan laporan penjualan.

5. \*\*Pemilik Apotek\*\* meminta laporan keuangan dari sistem.

6. \*\*Sistem\*\* mengirimkan laporan keuangan kepada \*\*Pemilik Apotek\*\*.

### Kode Mermaid untuk Sequence Diagram:

```mermaid

sequenceDiagram

participant Kasir

participant Sistem

participant PemilikApotek as "Pemilik Apotek"

Kasir->>Sistem: Memasukkan data transaksi

Sistem-->>Kasir: Validasi data transaksi

Sistem->>Sistem: Menghitung total biaya

Sistem->>Sistem: Mengurangi stok barang

Sistem-->>Kasir: Konfirmasi transaksi berhasil

Kasir->>Sistem: Meminta cetak struk

Sistem-->>Kasir: Mencetak struk penjualan

PemilikApotek->>Sistem: Meminta laporan penjualan

Sistem-->>PemilikApotek: Mengirimkan laporan penjualan

PemilikApotek->>Sistem: Meminta laporan keuangan

Sistem-->>PemilikApotek: Mengirimkan laporan keuangan

```

### Penjelasan Diagram:

1. \*\*Kasir\*\* berinteraksi dengan \*\*Sistem\*\* untuk memasukkan data transaksi penjualan.

2. \*\*Sistem\*\* memvalidasi data transaksi, menghitung total biaya, dan memperbarui stok barang.

3. \*\*Sistem\*\* mengirimkan konfirmasi kepada \*\*Kasir\*\* bahwa transaksi berhasil, lalu mencetak struk.

4. \*\*Pemilik Apotek\*\* dapat meminta laporan penjualan dan laporan keuangan dari \*\*Sistem\*\*.

### Langkah Menggunakan Kode Mermaid:

1. \*\*Salin kode di atas\*\* ke editor Mermaid seperti [Mermaid Live Editor](https://mermaid-js.github.io/mermaid-live-editor/).

2. Jalankan kodenya untuk melihat visualisasi sequence diagram.

3. Anda juga bisa menggunakannya di VS Code dengan ekstensi Mermaid Preview untuk melihat diagram langsung di sana.

Apakah ini sesuai dengan kebutuhan Anda atau perlu penyesuaian lebih lanjut pada alur?

**CLASS DIAGRAM**

Berikut adalah deskripsi \*\*Class Diagram\*\* untuk Sistem Informasi Akuntansi sebuah Apotek. Class Diagram menggambarkan struktur sistem perangkat lunak dengan menunjukkan kelas-kelas dalam sistem, atribut, metode, dan hubungan antar kelas.

### Deskripsi Umum:

1. \*\*Kelas Utama:\*\*

- \*\*Produk\*\*: Mengelola data barang (obat).

- \*\*Transaksi Penjualan\*\*: Mencatat detail penjualan.

- \*\*Kasir\*\*: Mengelola aktivitas transaksi penjualan.

- \*\*Laporan Keuangan\*\*: Menyimpan dan menghasilkan laporan keuangan.

- \*\*Supplier\*\*: Mengelola informasi pemasok.

- \*\*Stok Barang\*\*: Mengelola stok barang yang tersedia.

### Detail Kelas:

1. \*\*Produk\*\*

- Atribut: `idProduk`, `namaProduk`, `harga`, `stok`

- Metode: `tambahProduk()`, `ubahProduk()`, `hapusProduk()`

2. \*\*Transaksi Penjualan\*\*

- Atribut: `idTransaksi`, `tanggalTransaksi`, `totalBayar`

- Metode: `buatTransaksi()`, `hitungTotalBayar()`

3. \*\*Kasir\*\*

- Atribut: `idKasir`, `namaKasir`

- Metode: `inputTransaksi()`, `cetakStruk()`

4. \*\*Laporan Keuangan\*\*

- Atribut: `idLaporan`, `periodeLaporan`, `totalPemasukan`

- Metode: `generateLaporan()`, `tampilkanLaporan()`

5. \*\*Supplier\*\*

- Atribut: `idSupplier`, `namaSupplier`, `kontakSupplier`

- Metode: `tambahSupplier()`, `updateSupplier()`, `hapusSupplier()`

6. \*\*Stok Barang\*\*

- Atribut: `idStok`, `jumlahStok`, `tanggalUpdate`

- Metode: `updateStok()`, `cekStok()`

### Relasi Antar Kelas:

1. \*\*Kasir\*\* berhubungan dengan \*\*Transaksi Penjualan\*\*. Setiap kasir dapat melakukan beberapa transaksi.

2. \*\*Transaksi Penjualan\*\* memiliki hubungan dengan \*\*Produk\*\*. Setiap transaksi mencatat barang yang dijual.

3. \*\*Laporan Keuangan\*\* terhubung dengan \*\*Transaksi Penjualan\*\* untuk menghasilkan laporan keuangan.

4. \*\*Produk\*\* memiliki hubungan dengan \*\*Supplier\*\* karena setiap produk dipasok oleh satu atau lebih pemasok.

5. \*\*Stok Barang\*\* terhubung dengan \*\*Produk\*\* untuk mengelola ketersediaan stok.

### Kode Mermaid untuk Class Diagram:

```mermaid

classDiagram

class Produk {

+idProduk: int

+namaProduk: String

+harga: float

+stok: int

+tambahProduk()

+ubahProduk()

+hapusProduk()

}

class TransaksiPenjualan {

+idTransaksi: int

+tanggalTransaksi: Date

+totalBayar: float

+buatTransaksi()

+hitungTotalBayar()

}

class Kasir {

+idKasir: int

+namaKasir: String

+inputTransaksi()

+cetakStruk()

}

class LaporanKeuangan {

+idLaporan: int

+periodeLaporan: String

+totalPemasukan: float

+generateLaporan()

+tampilkanLaporan()

}

class Supplier {

+idSupplier: int

+namaSupplier: String

+kontakSupplier: String

+tambahSupplier()

+updateSupplier()

+hapusSupplier()

}

class StokBarang {

+idStok: int

+jumlahStok: int

+tanggalUpdate: Date

+updateStok()

+cekStok()

}

Kasir --> TransaksiPenjualan : "melakukan"

TransaksiPenjualan --> Produk : "terdiri dari"

TransaksiPenjualan --> LaporanKeuangan : "termasuk dalam"

Produk --> Supplier : "dipasok oleh"

Produk --> StokBarang : "memiliki"

```

### Penjelasan Diagram:

- Kelas-kelas utama seperti \*\*Produk\*\*, \*\*Kasir\*\*, \*\*Transaksi Penjualan\*\*, dan \*\*Laporan Keuangan\*\* direpresentasikan beserta atribut dan metode mereka.

- Hubungan antar kelas digambarkan dengan panah, misalnya \*\*Kasir\*\* berhubungan dengan \*\*Transaksi Penjualan\*\*, yang berarti kasir melakukan transaksi penjualan.

Anda dapat menggunakan kode Mermaid ini di editor seperti \*\*Mermaid Live Editor\*\* atau di IDE yang mendukung \*\*Mermaid Preview\*\* (misalnya, Visual Studio Code dengan ekstensi).

**STATE DIAGRAM**

Berikut adalah \*\*State Diagram\*\* untuk Sistem Informasi Akuntansi sebuah Apotek, yang menggambarkan perubahan status dari sistem berdasarkan berbagai peristiwa atau aksi yang dilakukan, seperti transaksi penjualan, pembaruan stok, atau pembuatan laporan.

### Deskripsi Umum:

- Sistem dimulai dalam \*\*State Awal\*\*.

- \*\*Kasir\*\* memulai transaksi penjualan.

- Setelah transaksi dilakukan, sistem akan memvalidasi dan mencatat transaksi, lalu memperbarui stok barang.

- Sistem akan menghasilkan laporan penjualan.

- Pemilik apotek bisa meminta laporan keuangan.

- Sistem akan menghasilkan laporan keuangan berdasarkan transaksi yang tercatat.

- Sistem bisa kembali ke status menunggu input baru dari kasir atau pemilik apotek.

### Kode Mermaid untuk State Diagram:

```mermaid

stateDiagram

[\*] --> MenungguTransaksi

MenungguTransaksi --> TransaksiDilakukan: Transaksi dimulai

TransaksiDilakukan --> ValidasiTransaksi: Validasi data

ValidasiTransaksi --> TransaksiSukses: Transaksi valid

TransaksiSukses --> UpdateStok: Perbarui stok barang

UpdateStok --> CetakStruk: Cetak struk penjualan

CetakStruk --> MenungguTransaksi: Menunggu transaksi baru

MenungguTransaksi --> GenerateLaporanPenjualan: Minta laporan penjualan

GenerateLaporanPenjualan --> MenungguTransaksi: Laporan penjualan selesai

MenungguTransaksi --> GenerateLaporanKeuangan: Minta laporan keuangan

GenerateLaporanKeuangan --> MenungguTransaksi: Laporan keuangan selesai

MenungguTransaksi --> [\*]: Sistem dimatikan

```

### Penjelasan State Diagram:

1. \*\*State Awal\*\*: Sistem dalam kondisi menunggu transaksi atau input dari pengguna.

2. \*\*Transaksi Dilakukan\*\*: Kasir memulai transaksi dengan memasukkan data penjualan.

3. \*\*Validasi Transaksi\*\*: Sistem memvalidasi data transaksi yang dimasukkan.

4. \*\*Transaksi Sukses\*\*: Jika data valid, sistem mencatat transaksi dan memperbarui stok barang.

5. \*\*Update Stok\*\*: Sistem memperbarui jumlah stok barang setelah transaksi berhasil.

6. \*\*Cetak Struk\*\*: Sistem mencetak struk penjualan untuk transaksi tersebut.

7. \*\*Laporan Penjualan & Laporan Keuangan\*\*: Sistem bisa menghasilkan laporan penjualan atau keuangan berdasarkan permintaan dari kasir atau pemilik apotek.

8. \*\*State Final\*\*: Sistem kembali ke status menunggu atau bisa berakhir jika sistem dimatikan.

### Cara Menggunakan Diagram:

- Anda dapat menempelkan kode di atas ke editor yang mendukung \*\*Mermaid\*\*, seperti [Mermaid Live Editor](https://mermaid-js.github.io/mermaid-live-editor/), atau gunakan di VS Code dengan ekstensi \*\*Mermaid Preview\*\*.

Diagram ini menunjukkan bagaimana sistem akuntansi di apotek beralih dari satu status ke status lain, mulai dari menunggu transaksi hingga memproses laporan keuangan.