UJIAN TENGAH SEMESTER (UTS)
GANJIL 2025/2026

Mata Kuliah:
PEMROGRAMAN BERORIENTASI OBJEK

Kelas D3TI2.B

Studi Kasus: Aplikasi kasir Toko Akun Game Berbasis Console

Oleh:
AHMAD MAUALANA KUDUS
2403099

D3 TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
POLITEKNIK NEGERI INDRAMAYU
OKTOBER 2025

Daftar Isi

Deskripsi Permasalahan

Pada ere modern saat ini, penjualan akun game online semakin viral dan banyak diminati terutama dikalangan anak muda, namun tidak sedikit penjual yang masih menggunakan cara manual seperti mencatat data akun, nama, harga, dan transaksi pembelian di buku catatan pribadi. Hal ini rentangnya typo atau kesalahan pencatatan, kehilangan data, dan sulit untuk memantau riwayat pembelian

Untuk mengatasi permasalahan tersebut, dikembangkanlah aplikasi berbasis console bernama “OhlansStore”. Aplikasi ini berperan untuk membantu admin dalam mengelola data akun game yang di jual,contohnya seperti menambahkan akun game baru yang mau di jual, menampilkan daftar akun yang masih tersedia, serta menampilkan riwayat pembelian sebagai bukti baik buat pembeli ataupun penjual. Aplikasi Console ini juga sebagai simulasi transaksi pembelian akun dengan system otomatis menghitung total dari harga dan kembalian

Dengan adanya aplikasi ini, proses pengelolaan toko akun menjadi lebih efisien, rapi, dan terorganisir tanpa harus menggunakan sistem berbasis database atau tampilan grafis yang kompleks.

Analisis Kebutuhan

1. Menyediakan sistem login admin
   Memungkinkan admin untuk masuk ke dalam fitur utama aplikasi hanya jika berhasil login dengan username dan password yang benar.

2. Menyediakan fitur Kelola
   Memungkinkan admin untuk menambahkan data akun game baru, Dimana di dalamnya mempunyai ID akun, nama, harga akun, dan jumlah stok awal.

3. Menyediakan fitur lihat daftar akun
   Memungkinkan admin dan pembeli dapat melihat daftar akun yang tersedia lengkap Dengan ID, nama, harga, dan stock

4. Menyediakan fitur riwayat pembelian
   Memungkinkan admin dapat melihat seluruh Riwayat transaksi pembelian yang pernah dilakukan oleh pembeli

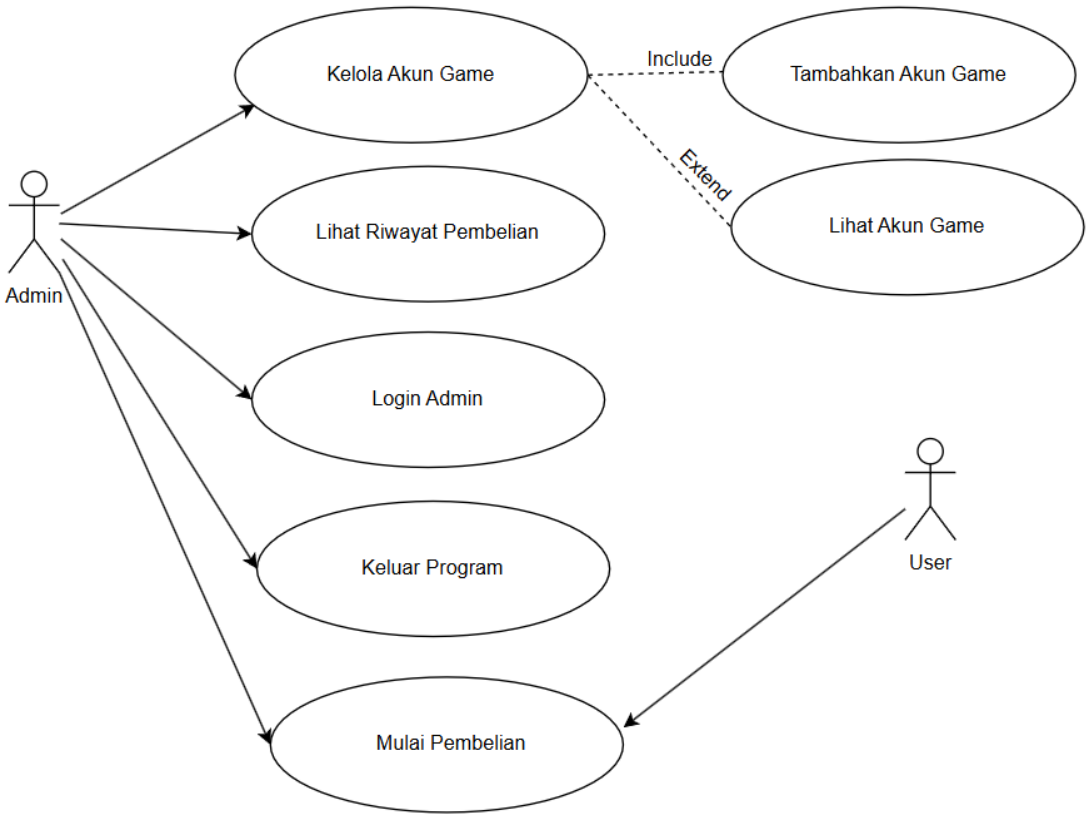
5. Menyediakan fitur pembelian akun
   Memungkinkan pembeli untuk melakukasn transaksi pembelian akun, dengan cara memilih akun berdasarkan ID, nama, melihat total harga, membayar, dan menerima informasi kembalian yang akan di bantu oleh admin.
   - Menyediakan perhitungan otomatis total dan kembalian system secara otomatis berdasarkan nominal pembayaran yang diberikan pembeli.
   - Menampilkan otomatis daftar akun yang tesedia di toko Ohlannsstore

6. Menyediakan menu keluar program
   Memungkinkan admin untuk menghentikan aplikasi denga aman, sekaligus menutup semua proses input yang sedang di jalankan.

Analisis Fitur

No	Fitur	Kebutuhan	Alur
1	Login Admin	Admin perlu mengatur akses suapaya tidak sembaranga orang lain bisa masuk dan mengelola data toko	Program meminta admin untuk menginputkan username dan password.Jika terdeteksi cocok dengan data yang ditentukan, kemudian admin bisa masuk ke menu yang ada di dalam; jika tidak, maka program akan menmapilkan Akses ditolak. Program berhenti.
2	Kelola Akun	Admin perlu menambahkan akun game baru dan melihat daftar akun, menambahkan akun game yang kemudian akan di jual di toko serta untuk menambahkan stok yang semakin menipis dan melihat akun yang tersedia untuk memantau stock yang tersedia	Admin memiliki menu “Tambah Akun ”dan”Lihat Daftar Akun”,setelah itu diminta mengisi ID akun, harga, dan stok.Data akun kemudian akan disimpan dalam daftar ArrayList serta bisa memantau stock akun dengan melihat daftar akun yang berisi ID, nama, harga dan stok
4	Lihat Riwayat Pembelian	Admin perlu memantau Riwayat transaksi yang sudah dilakukan pembelian oleh pembeli.	Admin memilih menu “Lihat Riwayat Pembelian”. Sistem menampilkan seluruh transaksi pembelian yang tersimpan dalam daftar Riwayat pembelian.
5	Mulai Pembelian	Pembeli perlu melakukan transaksi pembelian akun berdasarkan ID yang tersedia yang akan dibantu oleh kasir admin.	Pembeli memasukkan jumlah akun yang ingin dibeli, memilih ID akun, dan sistem menampilkan harga otomatis. Setelah itu, pembeli melakukan pembayaran dan sistem menghitung total serta kembalian.
6	Keluar Program	Pengguna perlu menghentikan aplikasi dengan aman setelah selesai menjalankan program.	Pengguna memilih menu “Keluar”. Sistem menampilkan pesan penutupan serta menghentikan program dengan menutup input scanner.

Rancangan UML Use Case Diagram



Rancangan Tampilan Program Aplikasi Console

Tampilan 1 Login Admin

- Berhasil

```
=== LOGIN ADMIN ===
Username: admin
Password: 123
============================
     MENU OHLANS STORE
============================
```

- Gagal

```
=== LOGIN ADMIN ===
Username: admin
Password: 234
Login gagal. Program dihentikan.
PS C:\Users\Asus\OneDrive\Documents\Semester3\PBO\Uts PBO\pemesanan-akun-game>
```

Tampilan 2 Menu Utama

```
============================
     MENU OHLANS STORE
============================
1. Kelola Akun
2. Lihat Riwayat Pembelian
3. Mulai Pembelian
4. Keluar
============================
```

Tampilan 4 Kelola Akun

```
============================
     MENU OHLANS STORE
============================
1. Kelola Akun
2. Lihat Riwayat Pembelian
3. Mulai Pembelian
4. Keluar
============================
Pilih menu: 1
Opsi: 1. Tambah Akun | 2. Lihat Akun
```

Tampilan 4 Tambahkan Akun

```
Opsi: 1. Tambah Akun | 2. Lihat Akun
1
ID: MLBB03
Nama Game: Skpc
Harga: Rp1500000
Stok: 1
Akun baru berhasil ditambahkan!
```

Tampilan 5 Lihat Daftar Akun

```
Opsi: 1. Tambah Akun | 2. Lihat Akun
2

=== DAFTAR AKUN ===
ML01 | Mobile Legends | Rp10000 | Stok: 5
FF01 | Free Fire | Rp15000 | Stok: 4
VL01 | Valorant | Rp20000 | Stok: 5
MLBB03 | Skpc | Rp1500000 | Stok: 1
```

Tampilan 6 Lihat Riwayat Pembelian

```
     MENU OHLANS STORE
============================
1. Kelola Akun
2. Lihat Riwayat Pembelian
     MENU OHLANS STORE
============================
1. Kelola Akun
2. Lihat Riwayat Pembelian
3. Mulai Pembelian
2. Lihat Riwayat Pembelian
3. Mulai Pembelian
4. Keluar
============================
Pilih menu: 2

=== RIWAYAT PEMBELIAN ===
Pembeli ahmet membeli Free Fire (Rp15000)

============================
```

Tampilan 7 Mulai Pembelian

```
Pilih menu: 3
Nama Pembeli: ahmet

=== DAFTAR AKUN ===
ML01 | Mobile Legends | Rp10000 | Stok: 5
FF01 | Free Fire | Rp15000 | Stok: 5
VL01 | Valorant | Rp20000 | Stok: 5
MLBB03 | Skpc | Rp1500000 | Stok: 1

Jumlah akun yang ingin dibeli: 1
Masukkan ID akun: free Fire
Akun tidak ditemukan atau stok habis!
Masukkan ID akun: FF01

=== Daftar Belanja ahmet ===
FF01 | Free Fire | Rp15000 | Stok: 4
Total Bayar: Rp15000
Bayar: Rp16000
Kembalian: Rp1000
Kembalian: Rp1000
Transaksi selesai!

Transaksi selesai!

============================
```

Tampilan 8 Mulai Keluar Program

```
==========================
    MENU OHLANS STORE
==========================
1. Kelola Akun
2. Lihat Riwayat Pembelian
3. Mulai Pembelian
4. Keluar
==========================
Pilih menu: 4
Program selesai. Terima kasih!
PS C:\Users\Asus\OneDrive\Documents\Semester3\PBO\Uts PBO\pemesanan-akun-game>
```

Analisis Class

Identifikasi Class dan Attribute

No	Class (Kata Benda)	Attribute (Kata Benda)
1	Akun	id: String
		namaGame: String
		harga: int
		Stok: int
2	Admin	id: String
		nama: String
		username: String
		password: String
3	Buyer	nama: String
		daftarBelanja: ArrayList<Akun>
		total: int
4	AplikasiKasir	daftarAdmin: ArrayList<Admin>
		daftarAkun: ArrayList<Akun>
		riwayatPembelian: ArrayList<String>
		adminAktif: Admin
5	Main	-

Identifikasi Method

No	Class (Kata Benda)	Method (Kata Kerja)	Parameter (Kata Benda)
1	Akun	getId(): String	-
		getNamaGame(): String	-
		getHarga(): int	-
		getStok(): int	-
		kurangiStok(): void	-
2	Admin	login(): boolean	String user, String pass
		tampilkanData(): void	-
		getNama(): String	-
		Admin():constructor	String id, String nama, String username, String password
3	Buyer	tambahAkun(): void	akun: akun
		tampilkanBelanja(): void	-
		getTotal(): int	-
		buyer():constructor	String: nama
4	AplikasiKasir	inisialisasiData(): void	-
		loginAdmin(): boolean	String username, String password
		mengelolaAkun( ): void	Scanner:scanner
		tampilkanDaftarAkun(): void	-
		tampilkanRiwayat(): void	-
		mulaiPembelian(): void	Scanner:scanner
5.	main	main()	agrs: String[]

Jenis Relasi dan Alasan

a. Dependency

No	Class Awal	Class Tujuan	Alasan
1	Main	AplikasiKasir	Class Main ini menggunakan sebuah class AplikasiKasir untuk menjalankan keseluruhan proses bisnis untama diantarnya inisialisasiData(), loginAdmin(), mengelolaAkun(), tampilkanRiwayat(), dan mulaiPembelian(). Main bertugas sebagai pengendali utama jalannya program yang memanggil method pada class AplikasiKasir.
2	AplikasiKasir	Buyer	AplikasiKasir membuat objek Buyer pada saat proses pembelian akun game mulaiPembelian(), di mana objek Buyer dibuat untuk merepresentasikan pembeli yang sedang melakukan transaksi dan menyimpan daftar akun yang dibeli.

b. Unidirectional Association

No	Class Awal	Class Tujuan	Alasan
1	AplikasiKasir	Admin	Jadi pada AplikasiKasir menyimpan daftar sebuah admin dan dapat memanggil fungsi login admin.
2.	AplikasiKasir	Akun	Jadi Pada AplikasiKasir dapat mengelola daftar Akun untuk ditampilkan atau diupdate stoknya.
3	Buyer	Akun	Karena Buyer memiliki sebuah daftar akun yang dibeli, tetapi Akun tidak mengenal Buyer.

c. Bidirectional Association

No	Class Awal	Class Tujuan	Alasan
1	-	-	Tidak memiliki hubungan dua arah; semua hubungan hanya dikenal satu arah.

d. Aggregation

No	Class Awal	Class Tujuan	Alasan
1	AplikasiKasir	Admin	Jadi pada AplikasiKasir memiliki sebuah kumpulan Admin yang dapat ada secara independen dari sistem.
2	AplikasiKasir	Akun	Jadi pada AplikasiKasir memiliki sebuah daftar Akun, tetapi Akun bisa eksis terpisah dari AplikasiKasir.
3	Buyer	Akun	Karena Buyer memiliki sebuah daftar Akun yang dibeli, namun Akun tidak bergantungan pada Buyer.

e. Composition

No	Class Awal	Class Tujuan	Alasan
1	AplikasiKasir	Buyer	Karena pada Buyer dibuat serta di Kelola penuh oleh AplikasiKasir saat proses transaksi berjalan. Setelah transaksi selesai,Buyer tidak lagi eksis

f. Generalization

No	Class Awal	Class Tujuan	Alasan

1	-	-	Karena tidak memiliki inheritance antar class dalam sistem ini. Semua class berdiri sendiri.

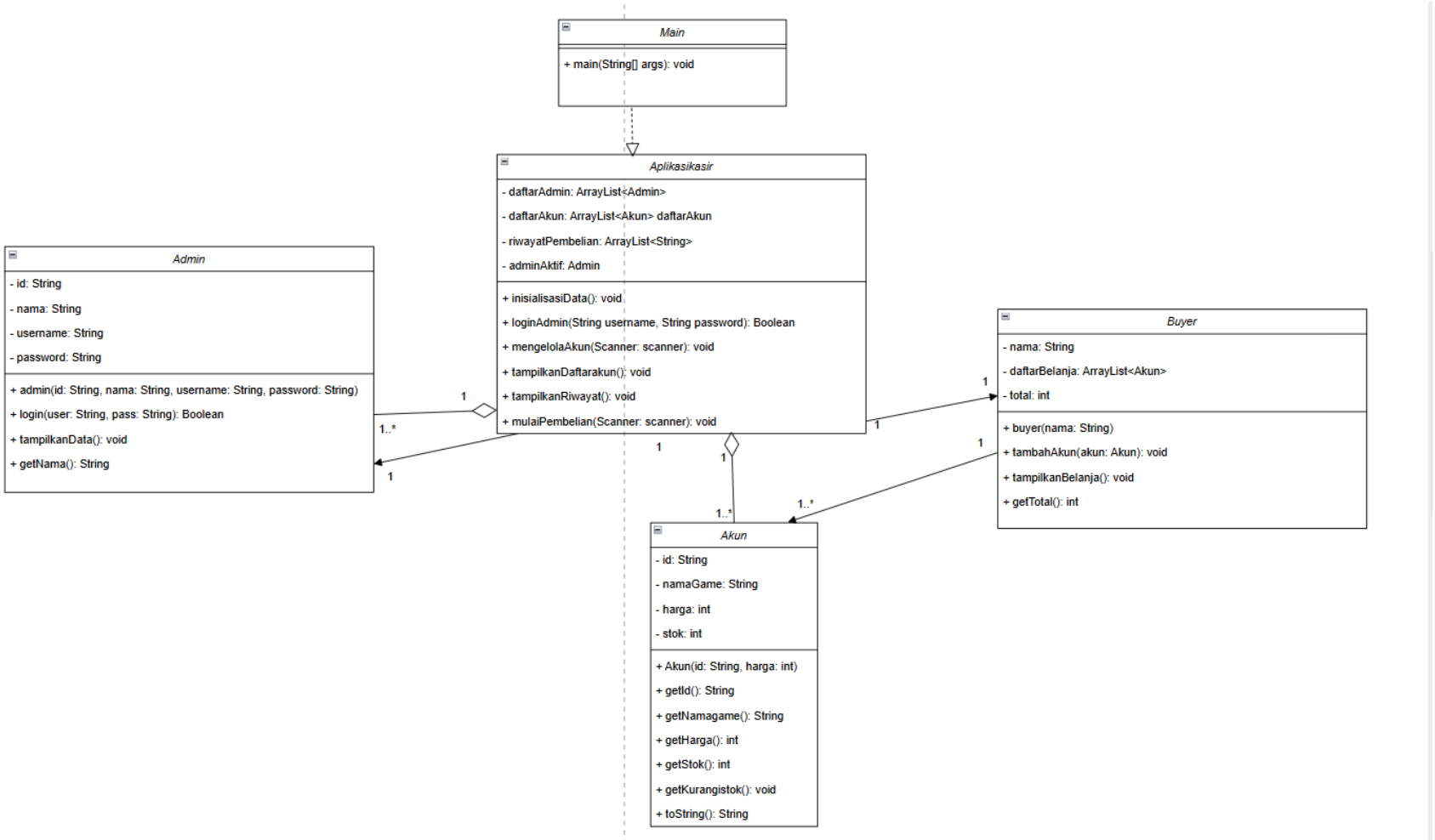
g. Realization

No	Class Awal	Class Tujuan	Alasan
1	-	-	Karena tidak ada sebuah class yang mengimplementasikan interface, sehingga tidak terdapat hubungan realization.

Multiplicity

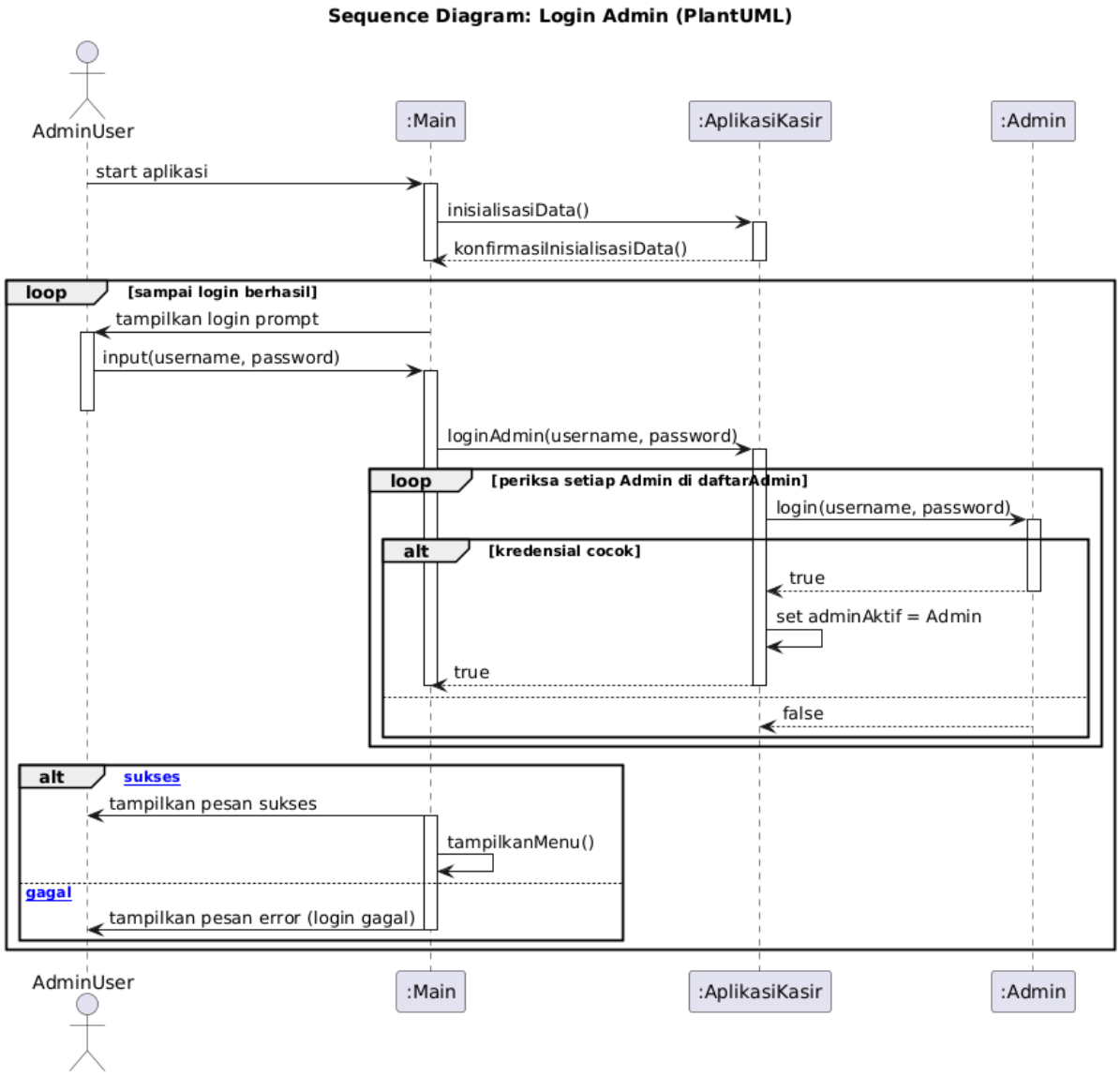
No	Class Awal	Class Tujuan	Relasi	Multiplicity	Keterangan
1	AplikasiKasir	Admin	Aggregation	1..* & 1	Jadi satu atau banyaknya admin menjadi bagian dari satu AplikasiKasir, pada admin hanya tersedia jika selama aplikasi berjalan.Data admin kemudian disimpan sementara di memori sebuah aplikasi
2	AplikasiKasir	Akun	Aggregation	1..* & 1	Jadi satu atau banyaknya akun menjadi bagain dari satu AplikasiKasir, Pada daftar akun hanya tersedia selama aplikasi berjalan dan hilang jika aplikasi ditutup.
3	AplikasiKasir	Buyer	Composition	1 & 1	Jadi Satu Buyer hanya bisa berinteraksi dalam satu AplikasiKasir pada saat transaksi sedang berlangsung. Objek Buyer dibuat serta di hapus dalam lingkup lingkup satu transaksi
4.	Buyer	Akun	Unidirectional Association	1..* & 1	Jadi pada satu Buyer bisa membeli satu atau lebih dari satu Akun, tapi setiap akun yang di beli hanya terdaftar pada satu pembeli pada saat transaksi
5	Main	AplikasiKasir	Dependency	1 & 1	Jadi satu main menjalankan satu instance pada AplikasiKasir, dan seleruh proses aplikasi bergantung pada instance tersebut.

Rancangan UML Class Diagram

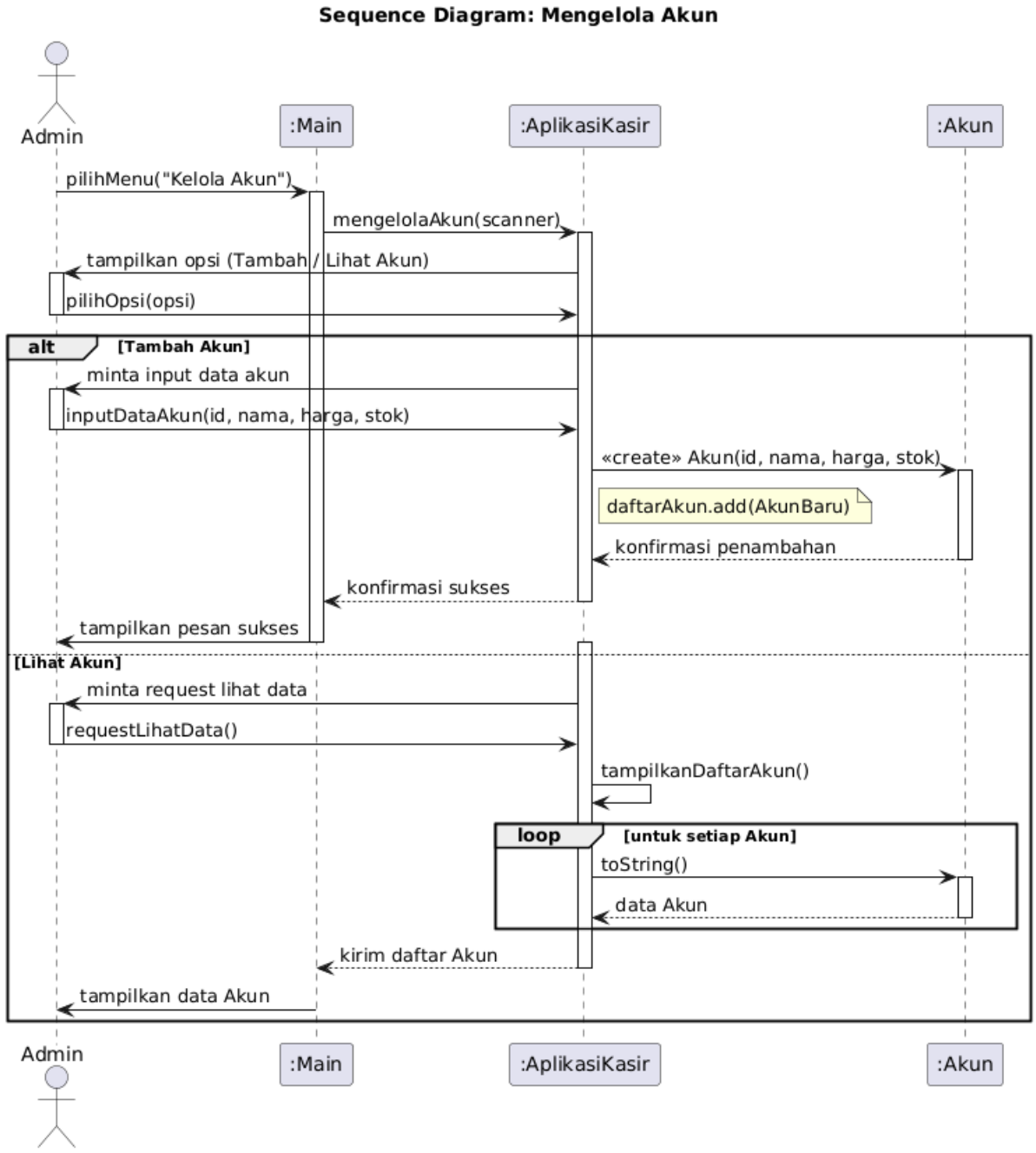


Rancangan UML Sequence Diagram

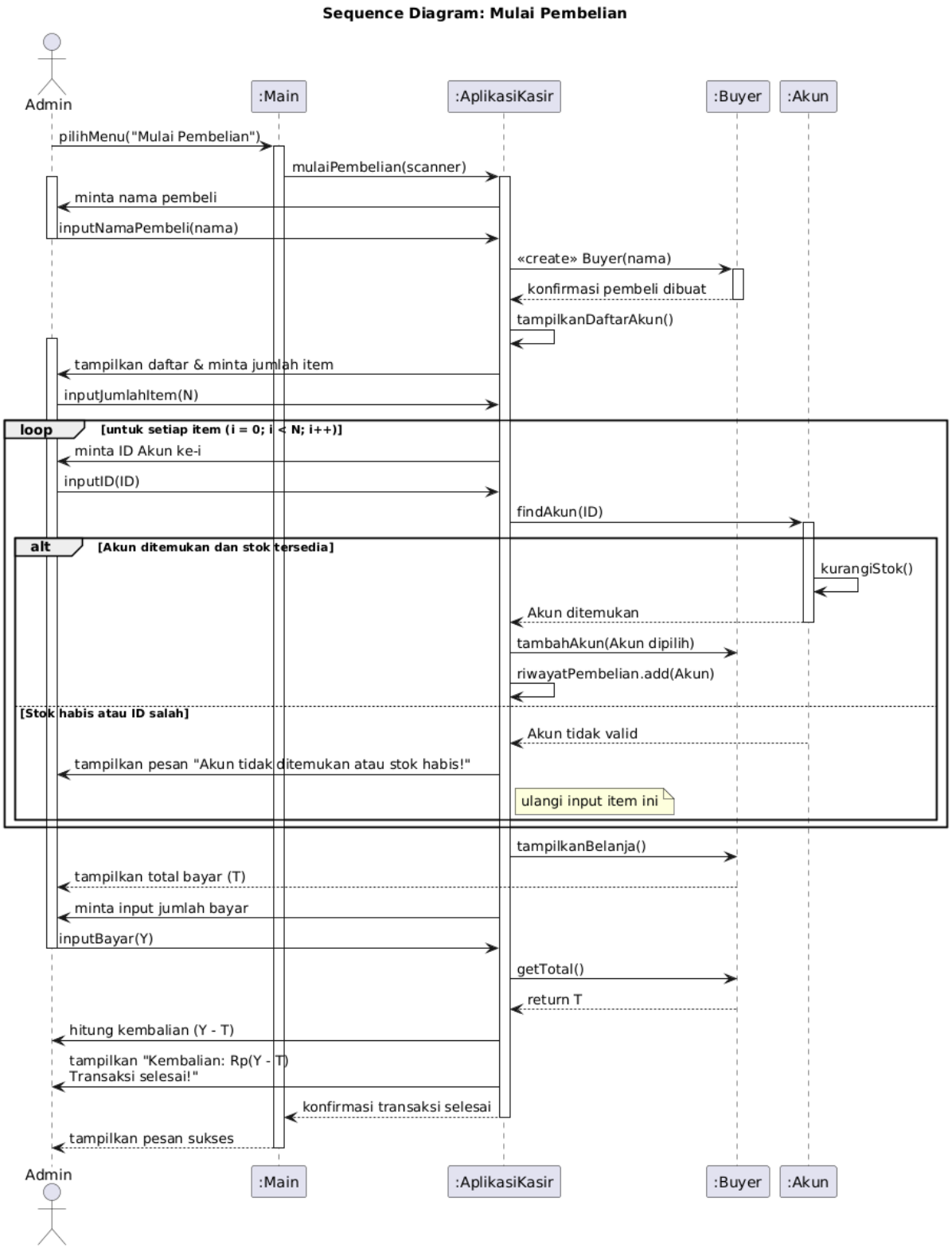
Sequence Diagram 1 : melakukan login admin



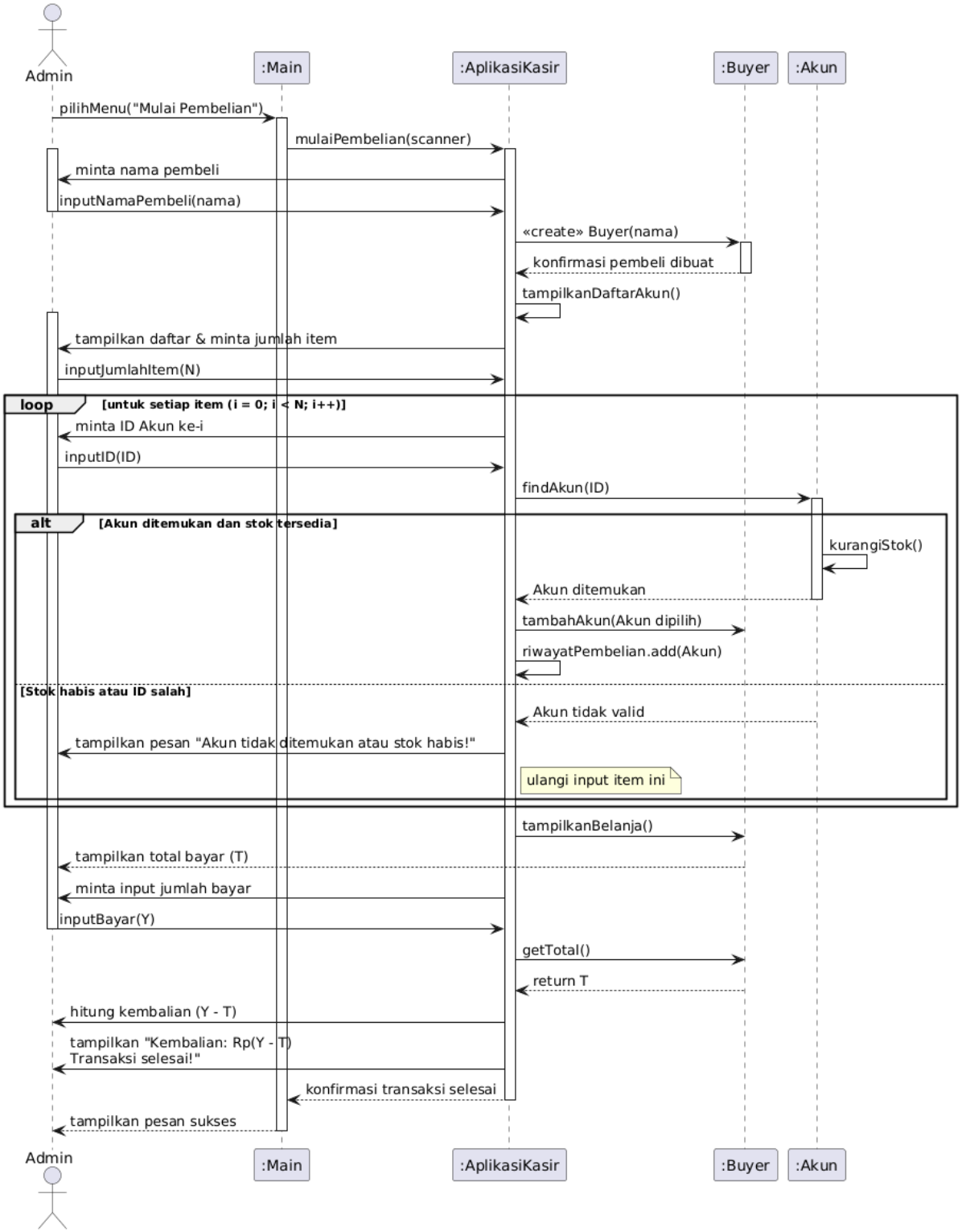
Sequence Diagram 2 : Mengelola Akun



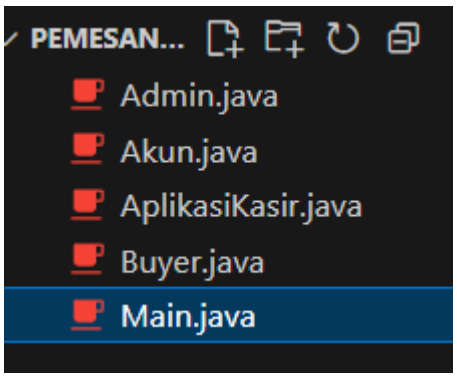
Sequence Diagram 3 : Mulai Pembelian



Sequence Diagram: Mulai Pembelian



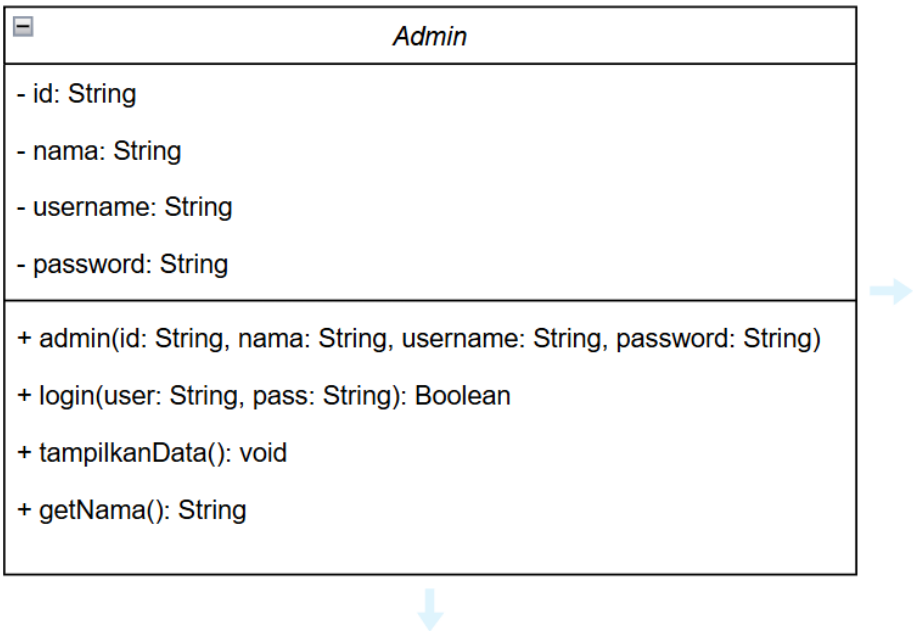
Pra-Kode Program



Kode Program

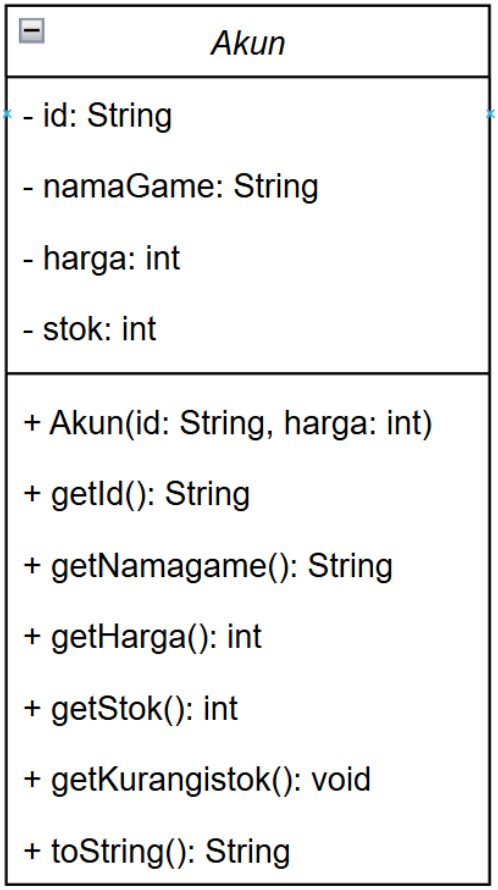
Admin.java

```
public class Admin {
    private String id;
    private String nama;
    private String username;
    private String password;

    public Admin(String id, String nama, String username, String password) {
        this.id = id;
        this.nama = nama;
        this.username = username;
        this.password = password;
    }

    public boolean login(String user, String pass) {
        return username.equals(user) && password.equals(pass);
    }

    public void tampilkanData() {
        System.out.println(id + " | " + nama + " | " + username);
    }

    public String getNama() {
        return nama;
    }
}
```



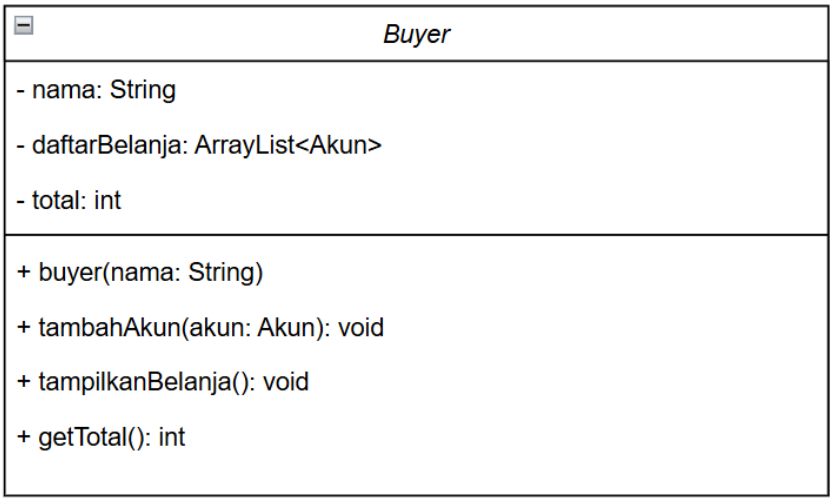
Akun.java

```
public class Akun {
    private String id;
    private String namaGame;
    private int harga;
    private int stok;

    public Akun(String id, String namaGame, int harga, int stok) {
        this.id = id;
        this.namaGame = namaGame;
        this.harga = harga;
        this.stok = stok;
    }

    public String getId() {
        return id;
    }

    public String getNamaGame() {
        return namaGame;
    }

    public int getHarga() {
        return harga;
    }

    public int getStok() {
        return stok;
    }

    public void kurangiStok() {
        if (stok > 0) stok--;
    }

    @Override
    public String toString() {
        return id + " | " + namaGame + " | Rp" + harga + " | Stok: " + stok;
    }
}
```



Buyer.java

```
import java.util.ArrayList;

public class Buyer {
    private String nama;
    private ArrayList<Akun> daftarBelanja = new ArrayList<>();
    private int total = 0;

    public Buyer(String nama) {
        this.nama = nama;
    }

    public void tambahAkun(Akun akun) {
        daftarBelanja.add(akun);
        total += akun.getHarga();
    }

    public void tampilkanBelanja() {
        System.out.println("\n=== Daftar Belanja " + nama + " ===");
        for (Akun akun : daftarBelanja) {
            System.out.println(akun);
        }
        System.out.println("Total Bayar: Rp" + total);
    }

    public int getTotal() {
        return total;
    }
}
```

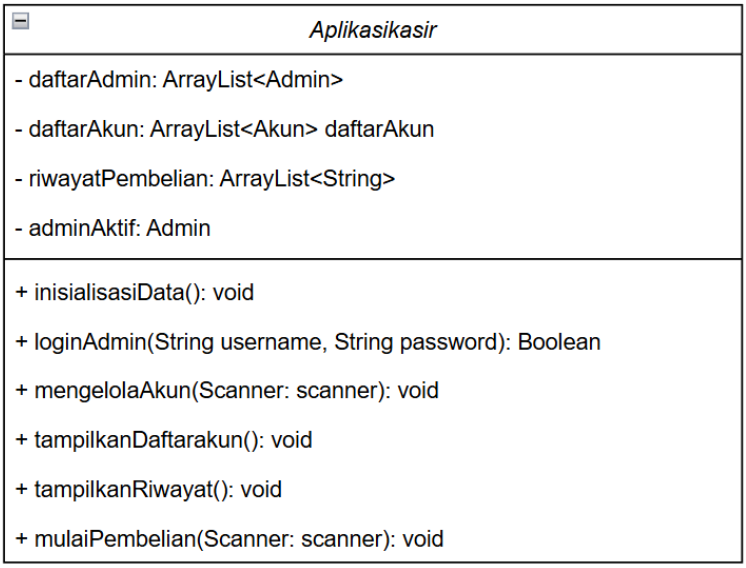


AplikasiKasir.java

```
import java.util.ArrayList;
import java.util.Scanner;

public class AplikasiKasir {
    private ArrayList<Admin> daftarAdmin = new ArrayList<>();
    private ArrayList<Akun> daftarAkun = new ArrayList<>();
    private ArrayList<String> riwayatPembelian = new ArrayList<>();
    private Admin adminAktif;

    public void inisialisasiData() {
        daftarAdmin.add(new Admin("A1", "Admin Utama", "admin", "123"));
        daftarAkun.add(new Akun("ML01", "Mobile Legends", 10000, 5));
        daftarAkun.add(new Akun("FF01", "Free Fire", 15000, 5));
        daftarAkun.add(new Akun("VL01", "Valorant", 20000, 5));
    }

    public boolean loginAdmin(String username, String password) {
        for (Admin admin : daftarAdmin) {
            if (admin.login(username, password)) {
                adminAktif = admin;
                return true;
            }
        }
        return false;
    }

    public void mengelolaAkun(Scanner scanner) {
        System.out.println("Opsi: 1. Tambah Akun | 2. Lihat Akun");
        int opsi = scanner.nextInt();
        scanner.nextLine();

        if (opsi == 1) {
            System.out.print("ID: ");
            String id = scanner.nextLine();
            System.out.print("Nama Game: ");
            String nama = scanner.nextLine();
            System.out.print("Harga: Rp");
            int harga = scanner.nextInt();
            System.out.print("Stok: ");
            int stok = scanner.nextInt();
            scanner.nextLine();

            daftarAkun.add(new Akun(id, nama, harga, stok));
            System.out.println("Akun baru berhasil ditambahkan!\n");
        } else if (opsi == 2) {
            tampilkanDaftarAkun();
        }
    }
```

```
    public void tampilkanDaftarAkun() {
        System.out.println("\n=== DAFTAR AKUN ===");
        for (Akun akun : daftarAkun) {
            System.out.println(akun);
        }
        System.out.println();
    }

    public void tampilkanRiwayat() {
        System.out.println("\n=== RIWAYAT PEMBELIAN ===");
        if (riwayatPembelian.isEmpty()) {
            System.out.println("Belum ada pembelian.");
        } else {
            for (String data : riwayatPembelian) {
                System.out.println(data);
            }
        }
        System.out.println();
    }

    public void mulaiPembelian(Scanner scanner) {
        System.out.print("Nama Pembeli: ");
        String nama = scanner.nextLine();
        Buyer pembeli = new Buyer(nama);

        tampilkanDaftarAkun();

        System.out.print("Jumlah akun yang ingin dibeli: ");
        int jumlah = scanner.nextInt();
        scanner.nextLine();

        for (int i = 0; i < jumlah; i++) {
            System.out.print("Masukkan ID akun: ");
            String id = scanner.nextLine().toUpperCase();

            Akun akunDipilih = null;
            for (Akun akun : daftarAkun) {
                if (akun.getId().equalsIgnoreCase(id) && akun.getStok() > 0) {
                    akunDipilih = akun;
                    break;
                }
            }

            if (akunDipilih != null) {
                akunDipilih.kurangiStok();
                pembeli.tambahAkun(akunDipilih);
                riwayatPembelian.add("Pembeli " + nama + " membeli " +
akunDipilih.getNamaGame() + " (Rp" + akunDipilih.getHarga() + ")");
            } else {
                System.out.println("Akun tidak ditemukan atau stok habis!");
                i--;
            }
        }

        pembeli.tampilkanBelanja();

        System.out.print("Bayar: Rp");
        int bayar = scanner.nextInt();
        int kembali = bayar - pembeli.getTotal();

        System.out.println("Kembalian: Rp" + kembali);
        System.out.println("Transaksi selesai!\n");
    }
}

        total += akun.getHarga();
    }

    public void tampilkanBelanja() {
        System.out.println("\n=== Daftar Belanja " + nama + " ===");
        for (Akun akun : daftarBelanja) {
            System.out.println(akun);
        }
        System.out.println("Total Bayar: Rp" + total);
    }
```

```
    public int getTotal() {
        return total;
    }
}
```



Main.java

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        AplikasiKasir app = new AplikasiKasir();
        app.inisialisasiData();

        System.out.println("=== LOGIN ADMIN ===");
        System.out.print("Username: ");
        String user = scanner.nextLine();
        System.out.print("Password: ");
        String pass = scanner.nextLine();

        if (app.loginAdmin(user, pass)) {
            boolean jalan = true;
            while (jalan) {
                System.out.println("============================");
                System.out.println("     MENU OHLANS STORE     ");
                System.out.println("============================");
                System.out.println("1. Kelola Akun");
                System.out.println("2. Lihat Riwayat Pembelian");
                System.out.println("3. Mulai Pembelian");
                System.out.println("4. Keluar");
                System.out.println("============================");
                System.out.print("Pilih menu: ");
                int pilih = scanner.nextInt();
                scanner.nextLine();

                switch (pilih) {
                    case 1 -> app.mengelolaAkun(scanner);
```

```
                    case 2 -> app.tampilkanRiwayat();
                    case 3 -> app.mulaiPembelian(scanner);
                    case 4 -> {
                        jalan = false;
                        System.out.println("Program selesai. Terima kasih!");
                    }
                    default -> System.out.println("Pilihan tidak valid!");
                }
            }
        } else {
            System.out.println("Login gagal. Program dihentikan.");
        }

        scanner.close();
    }
}

akun);
        }
        System.out.println("Total Bayar: Rp" + total);
    }

    public int getTotal() {
        return total;
    }
}
```

Compile & Run

Compile

```
PS C:\Users\Asus\OneDrive\Documents\Semester3\PBO\Uts PBO\pemesanan-akun-game> javac *.java
PS C:\Users\Asus\OneDrive\Documents\Semester3\PBO\Uts PBO\pemesanan-akun-game>
```

Run

```
PS C:\Users\Asus\OneDrive\Documents\Semester3\PBO\Uts PBO\pemesanan-akun-game> java Main
=== LOGIN ADMIN ===
Username:
```

Testing

Skenario Login Admin (Berhasil)

```
=== LOGIN ADMIN ===
Username: admin
Password: 123
============================
     MENU OHLANS STORE
============================
```

Skenario Login Admin (Gagal)

```
=== LOGIN ADMIN ===
Username: admin
Password: 234
Login gagal. Program dihentikan.
PS C:\Users\Asus\OneDrive\Documents\Semester3\PBO\Uts PBO\pemesanan-akun-game>
```

Skenario Beli Akun Saat Masukan ID (Berhasil)

```
Masukkan ID akun: ML01
Masukkan ID akun: FF01

=== Daftar Belanja aku ===
ML01 | Mobile Legends | Rp10000 | Stok: 4
FF01 | Free Fire | Rp15000 | Stok: 4
Total Bayar: Rp25000
Bayar: Rp
```

Skenario Beli Akun Saat Masukan ID (Gagal)

```
Jumlah akun yang ingin dibeli: 2
Masukkan ID akun: sdew
Akun tidak ditemukan atau stok habis!
```

Skenario Bayar (Berhasil)

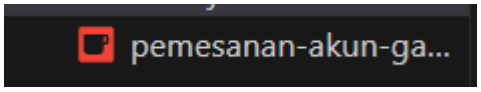
```
=== Daftar Belanja asf ===
ML01 | Mobile Legends | Rp10000 | Stok: 3
Total Bayar: Rp10000
Bayar: Rp100000
Kembalian: Rp90000
Transaksi selesai!
```

Skenario Bayar (Gagal/mines)

```
=== Daftar Belanja aku ===
ML01 | Mobile Legends | Rp10000 | Stok: 4
FF01 | Free Fire | Rp15000 | Stok: 4
Total Bayar: Rp25000
Bayar: Rp2000
Kembalian: Rp-23000
Transaksi selesai!
```

Build (Deploy)

```
PS C:\Users\Asus\OneDrive\Documents\Semester3\PBO\Uts PBO\pemesanan-akun-game> jar cfe pemesanan-akun-game.jar Main *.class
PS C:\Users\Asus\OneDrive\Documents\Semester3\PBO\Uts PBO\pemesanan-akun-game>
```




```
PS C:\Users\Asus\OneDrive\Documents\Semester3\PBO\Uts PBO\pemesanan-akun-game> jar cfe pemesanan-akun-game.jar Main *.class
PS C:\Users\Asus\OneDrive\Documents\Semester3\PBO\Uts PBO\pemesanan-akun-game> java -jar pemesanan-akun-game.jar
=== LOGIN ADMIN ===
Username:
```

URL Repository

Sajikan alamat URL repositori github yang berisi kode program dan gambar-gambar (png / jpg) yang digunakan pada laporan UTS ini

https://github.com/...../.....