

```

import java.util.ArrayList;
import java.util.Comparator;
import java.util.Scanner;
import java.util.Date;

public class ReuseMarket {
    static ArrayList<Anggota> daftarAnggota = new ArrayList<>();
    static ArrayList<Barang> daftarBarang = new ArrayList<>();
    static ArrayList<Transaksi> daftarTransaksi = new ArrayList<>();
    static Scanner scanner = new Scanner(System.in);
    static final String ADMIN_USERNAME = "admin";
    static final String ADMIN_PASSWORD = "12345";
    static Anggota currentUser;
    static int lastBarangId = 0;

    public static void main(String[] args) {
        while (true) {
            System.out.println("\n=====");
            System.out.println("=== Selamat Datang di Reuse Market ===");
            System.out.println("=====");
            System.out.println("1. Daftar Anggota");
            System.out.println("2. Login");
            System.out.println("3. Keluar");
            System.out.println("=====");
            System.out.print("Pilih menu (1-3): ");
            int pilihan = scanner.nextInt();
            scanner.nextLine();

            switch (pilihan) {
                case 1:
                    daftarAnggota();
                    break;
                case 2:
                    login();
                    break;
                case 3:
                    System.out.println("Terima kasih telah menggunakan Reuse Market. Sampai jumpa!");
                    System.exit(0);
                default:
                    System.out.println("\nError: Pilihan tidak valid! Silakan pilih angka antara 1 hingga 3.");
            }
        }
    }

    static void daftarAnggota() {
        System.out.println("\n=====");
    }
}

```

```

        System.out.println("===      Daftar Anggota      ===");
        System.out.println("=====");
        System.out.print("Masukkan Username (contoh: john123): ");
        String username = scanner.nextLine().trim();
        if (username.isEmpty()) {
            System.out.println("\nError: Username tidak boleh kosong!");
            return;
        }
        for (Anggota a : daftarAnggota) {
            if (a.getId().equals(username)) {
                System.out.println("\nError: Username sudah terdaftar, silakan langsung
login.");
                return;
            }
        }

        System.out.print("Masukkan Nama (contoh: John Doe): ");
        String nama = scanner.nextLine().trim();
        if (nama.isEmpty()) {
            System.out.println("\nError: Nama tidak boleh kosong!");
            return;
        }
        System.out.print("Masukkan Password (minimal 6 karakter): ");
        String password = scanner.nextLine().trim();
        if (password.isEmpty() || password.length() < 6) {
            System.out.println("\nError: Password tidak boleh kosong dan minimal 6
karakter!");
            return;
        }
        System.out.print("Masukkan No HP (contoh: 081234567890): ");
        String noHp = scanner.nextLine().trim();
        if (noHp.isEmpty()) {
            System.out.println("\nError: No HP tidak boleh kosong!");
            return;
        }
        daftarAnggota.add(new Anggota(username, nama, password, noHp));
        System.out.println("\nAkun berhasil didaftarkan!");
    }
}

```

```

static void login() {
    System.out.println("\n=====");
    System.out.println("===      Login      ===");
    System.out.println("=====");
    System.out.print("Masukkan Username: ");
    String username = scanner.nextLine().trim();
    System.out.print("Masukkan Password: ");
    String password = scanner.nextLine().trim();
    if (username.equals(ADMIN_USERNAME) && password.equals(ADMIN_PASSWORD)) {
        System.out.println("\nLogin sebagai Admin berhasil!");
        menuAdmin();
        return;
    }
    for (Anggota a : daftarAnggota) {

```

```

        if (a.getId().equals(username) && a.getPassword().equals(password)) {
            System.out.println("\nLogin sebagai Anggota berhasil!");
            currentUser = a; //  Ubah jadi objek Anggota
            menuAnggota();
            return;
        }
    }

    System.out.println("\nError: Username atau Password salah!");
}

static void menuAdmin() {
    while (true) {
        System.out.println("\n=====");
        System.out.println("==          Menu Admin          ==");
        System.out.println("=====");
        System.out.println("1. Lihat Anggota");
        System.out.println("2. Lihat Barang untuk Persetujuan Lelang");
        System.out.println("3. Verifikasi Pembayaran");
        System.out.println("4. Hapus Anggota yang bermasalah");
        System.out.println("5. Keluar");
        System.out.println("=====");
        System.out.print("Pilih menu (1-5): ");
        int pilihan = scanner.nextInt();
        scanner.nextLine();

        switch (pilihan) {
            case 1:
                lihatAnggota();
                break;
            case 2:
                lihatBarangPersetujuan();
                break;
            case 3:
                lihatBarangVerifikasi();
                break;
            case 4:
                hapusAnggota();
                break;
            case 5:
                System.out.println("Kembali ke menu utama...");
                return;
            default:
                System.out.println("\nError: Pilihan tidak valid! Silakan pilih angka antara 1 hingga 5.");
        }
    }
}
}

```

```

        static void lihatAnggota() {

System.out.println("-----");
        System.out.println("|
                                Daftar Anggota
|");

System.out.println("-----");

        if (daftarAnggota.isEmpty()) {
            System.out.println("Tidak ada anggota yang terdaftar.");
        } else {
            System.out.printf("%-5s %-10s %-20s %-15s %-10s\n", "No", "ID", "Nama", "No
HP", "Jenis");

System.out.println("-----");

            int no = 1;
            for (Anggota a : daftarAnggota) {
                System.out.printf("| %-5d | %-10s | %-20s | %-15s | %-10s |\n",
                    no++, a.getId(), a.getNama(), a.getNoHp(), a.getJenisAnggota());
            }

System.out.println("-----");
        }
    }

    static void hapusAnggota() {
        System.out.println("\n=== Hapus Anggota ===");
        if (daftarAnggota.isEmpty()) {
            System.out.println("Tidak ada anggota yang terdaftar.");
            return;
        }

System.out.println("-----");
        System.out.printf("%-5s %-15s %-20s %-15s %-10s\n", "No", "Username", "Nama",
            "No HP", "Jenis");

System.out.println("-----");

        for (int i = 0; i < daftarAnggota.size(); i++) {
            Anggota a = daftarAnggota.get(i);
            String jenis = (a instanceof AnggotaPremium) ? "Premium" : "Biasa";
            System.out.printf("%-5d %-15s %-20s %-15s %-10s\n",
                (i + 1), a.getId(), a.getNama(), a.getNoHp(), jenis);
        }

System.out.println("-----");
        System.out.print("Pilih nomor anggota yang ingin dihapus (atau 0 untuk kembali):
");
    }

```

```

int pilihan = scanner.nextInt();
scanner.nextLine();

if (pilihan == 0) {
    return;
} else if (pilihan > 0 && pilihan <= daftarAnggota.size()) {
    Anggota anggotaDipilih = daftarAnggota.get(pilihan - 1);

    // Cek kalau premium
    if (anggotaDipilih instanceof AnggotaPremium) {
        System.out.println("\nAnggota premium tidak dapat dihapus.");
        return;
    }

    daftarAnggota.remove(anggotaDipilih);
    System.out.println("\nAnggota " + anggotaDipilih.getNama() + " berhasil
dihapus.");
} else {
    System.out.println("Nomor anggota tidak valid.");
}
}

static void lihatBarangPersetujuan() {
    System.out.println("\n=== Barang yang Menunggu Persetujuan ===");

    ArrayList<Barang> barangPremium = new ArrayList<>();
    ArrayList<Barang> barangBiasa = new ArrayList<>();

    // Pisahkan barang yang butuh persetujuan berdasarkan jenis pemiliknya
    for (Barang b : daftarBarang) {
        if (b.isButuhPersetujuan()) {
            if (b.getPemilik() instanceof AnggotaPremium) {
                barangPremium.add(b);
            } else {
                barangBiasa.add(b);
            }
        }
    }

    // Gabungkan dengan prioritas premium di atas
    ArrayList<Barang> barangMenungguPersetujuan = new ArrayList<>();
    barangMenungguPersetujuan.addAll(barangPremium);
    barangMenungguPersetujuan.addAll(barangBiasa);

    if (barangMenungguPersetujuan.isEmpty()) {
        System.out.println("Tidak ada barang yang menunggu persetujuan.");
        return;
    }

    System.out.println("-----");
}

```

```

        System.out.printf("%-5s %-20s %-25s\n", "No", "Nama Barang", "Pemilik");

        System.out.println("-----");

        for (int i = 0; i < barangMenungguPersetujuan.size(); i++) {
            Barang b = barangMenungguPersetujuan.get(i);
            String namaPemilik = b.getPemilik().getNama();
            if (b.getPemilik() instanceof AnggotaPremium) {
                namaPemilik += " ★"; // Tanda bintang untuk premium
            }
            System.out.printf("%-5d %-20s %-25s\n",
                (i + 1), b.getNama(), namaPemilik);
        }

        System.out.println("-----");
        System.out.print("Pilih nomor barang yang ingin diproses (atau 0 untuk kembali):");

        int pilihan = scanner.nextInt();
        scanner.nextLine();

        if (pilihan == 0) {
            return;
        } else if (pilihan > 0 && pilihan <= barangMenungguPersetujuan.size()) {
            Barang barangDipilih = barangMenungguPersetujuan.get(pilihan - 1);
            System.out.println("\n=== Detail Barang ===");
            barangDipilih.tampilkanInfo();
            System.out.println("\nPilih opsi:");
            System.out.println("1. Setujui Lelang");
            System.out.println("2. Tolak Lelang");
            System.out.print("Pilih opsi: ");
            int opsi = scanner.nextInt();
            scanner.nextLine();

            if (opsi == 1) {
                barangDipilih.setButuhPersetujuan(false);
                System.out.println("Lelang untuk barang " + barangDipilih.getNama() + " telah disetujui.");
            } else if (opsi == 2) {
                barangDipilih.setButuhPersetujuan(false);
                barangDipilih.setSudahDilelang(false);
                System.out.println("Lelang untuk barang " + barangDipilih.getNama() + " ditolak.");
            } else {
                System.out.println("Opsi tidak valid.");
            }
        } else {
            System.out.println("Nomor barang tidak valid.");
        }
    }
}

```

```

static void lihatBarangVerifikasi() {
    System.out.println("\n=== Barang yang Menunggu Verifikasi Pembayaran ===");
    ArrayList<Barang> barangMenungguVerifikasi = new ArrayList<>();
    for (Barang b : daftarBarang) {
        if (b.isButuhVerifikasiPembayaran()) {
            barangMenungguVerifikasi.add(b);
        }
    }
    if (barangMenungguVerifikasi.isEmpty()) {
        System.out.println("Tidak ada barang yang menunggu verifikasi pembayaran.");
        return;
    }

    System.out.println("-----");
    System.out.printf("%-5s %-20s %-15s %-20s\n", "No", "Nama Barang", "Harga Tertinggi", "Penawar Tertinggi");

    System.out.println("-----");

    for (int i = 0; i < barangMenungguVerifikasi.size(); i++) {
        Barang b = barangMenungguVerifikasi.get(i);
        System.out.printf("%-5d %-20s Rp%,12d %-20s\n",
            (i + 1), b.getNama(), b.getHargaTertinggi(), b.getPenawarTertinggi());
    }

    System.out.println("-----");
    System.out.print("Pilih nomor barang yang ingin diverifikasi (atau 0 untuk kembali): ");
    int pilihan = scanner.nextInt();
    scanner.nextLine();

    if (pilihan == 0) {
        return;
    } else if (pilihan > 0 && pilihan <= barangMenungguVerifikasi.size()) {
        Barang barangDipilih = barangMenungguVerifikasi.get(pilihan - 1);
        System.out.println("\n=== Detail Barang ===");
        barangDipilih.tampilkanInfoLelang();
        System.out.println("\nPilih opsi:");
        System.out.println("1. Verifikasi Pembayaran");
        System.out.println("2. Tolak Pembayaran");
        System.out.print("Pilih opsi: ");
        int opsi = scanner.nextInt();
        scanner.nextLine();

        if (opsi == 1) {
            barangDipilih.setButuhVerifikasiPembayaran(false);
            System.out.println("Pembayaran untuk barang " + barangDipilih.getNama()
+ " telah diverifikasi.");
            daftarTransaksi.add(new Transaksi(barangDipilih,
barangDipilih.getPenawarTertinggi(), barangDipilih.getHargaTertinggi(), new Date()));
        } else if (opsi == 2) {
            System.out.println("Pembayaran untuk barang " + barangDipilih.getNama()
+ " ditolak.");
            barangDipilih.setButuhVerifikasiPembayaran(false);
            barangDipilih.setPenawarTertinggi(null);
        }
    }
}

```

```

        barangDipilih.setHargaTertinggi(barangDipilih.getHargaAwal());
        barangDipilih.setJumlahBid(0);
    } else {
        System.out.println("Opsi tidak valid.");
    }

} else {
    System.out.println("Nomor barang tidak valid.");
}
}

static void menuAnggota() {
    while (true) {
        System.out.println("\n=====");
        System.out.println("===          Menu Anggota          ===");
        System.out.println("=====");
        System.out.println("1. Lelang Saya");
        System.out.println("2. Lelang Tersedia");
        System.out.println("3. Lihat Profile");
        System.out.println("4. Upgrade ke Anggota Premium");
        System.out.println("5. Keluar");
        System.out.println("=====");
        System.out.print("Pilih menu (1-3): ");
        int pilihan = scanner.nextInt();
        scanner.nextLine();

        switch (pilihan) {
            case 1:
                menuLelangSaya();
                break;
            case 2:
                menuLelangTersedia();
                break;
            case 3:
                System.out.println("\n--- PROFIL ---");
                currentUser.tampilkanInfo(true);
                System.out.println("Jenis Anggota: " + currentUser.getJenisAnggota());
                break;
            case 4:
                upgradePremium();
                break;
            case 5:
                System.out.println("Terima kasih telah menggunakan sistem lelang. Sampai jumpa!");
                return;
            default:
                System.out.println("\nPilihan tidak valid! Silakan pilih angka antara 1 hingga 5.");
        }
    }
}
}

```



```

static void upgradePremium() {
    // cari objek anggota dari username yg login
    Anggota anggota = currentUser;

    // cek kalau udah premium
    if (anggota instanceof AnggotaPremium) {
        System.out.println("Kamu sudah menjadi anggota premium.");
        return;
    }

    // konfirmasi pembayaran
    System.out.println("\nBiaya upgrade ke anggota premium adalah Rp50.000");
    System.out.print("Apakah kamu sudah membayar? (ya/tidak): ");
    String konfirmasi = scanner.nextLine();

    if (konfirmasi.equalsIgnoreCase("ya")) {
        // ganti anggota biasa jadi premium
        AnggotaPremium premium = new AnggotaPremium(
            anggota.getId(),
            anggota.getNama(),
            anggota.getPassword(),
            anggota.getNoHp()
        );

        daftarAnggota.remove(anggota); // hapus versi biasa
        daftarAnggota.add(premium); // ganti dengan versi premium
        currentUser = premium; // ✅ ganti currentUser ke objek premium

        System.out.println("Selamat! Kamu sekarang menjadi anggota premium 🎉");
    } else {
        System.out.println("Upgrade dibatalkan.");
    }
}

static Anggota cariAnggota(String username) {
    for (Anggota anggota : daftarAnggota) {
        if (anggota.getId().equals(username)) {
            return anggota;
        }
    }
    return null;
}

static void menuLelangSaya() {
    while (true) {
        System.out.println("\n=====");
        System.out.println("===          Lelang Saya          ===");
        System.out.println("=====");
        System.out.println("1. Tambah Barang");
        System.out.println("2. Lihat Barang Milik Sendiri");
    }
}

```

```

        System.out.println("3. Mengajukan Lelang");
        System.out.println("4. Melihat Status Lelang & Tawaran");
        System.out.println("5. Kembali");
        System.out.println("=====");
        System.out.print("Pilih menu (1-5): ");
        int pilihan = scanner.nextInt();
        scanner.nextLine();

        switch (pilihan) {
            case 1:
                tambahBarang();
                break;
            case 2:
                submenuLihatBarang();
                break;
            case 3:
                ajukanLelang();
                break;
            case 4:
                lihatStatusLelang();
                break;
            case 5:
                System.out.println("Kembali ke Menu Anggota...");
                return;
            default:
                System.out.println("\nPilihan tidak valid! Silakan pilih angka
antara 1 hingga 5.");
        }
    }
}

```

```

static void tambahBarang() {
    System.out.println("\n=====");
    System.out.println("==          Tambah Barang          ==");
    System.out.println("=====");

    // Auto-generate ID barang
    lastBarangId++;
    String id = "BRG" + String.format("%03d", lastBarangId);

    // Cek ID barang unik
    for (Barang b : daftarBarang) {
        if (b.getId().equals(id)) {
            System.out.println("Error: ID Barang sudah terdaftar!");
            return;
        }
    }

    // Input Nama Barang
    String nama;
}

```

```

do {
    System.out.print("Masukkan Nama Barang (contoh: Laptop Gaming): ");
    nama = scanner.nextLine().trim();
    if (nama.isEmpty()) {
        System.out.println("Error: Nama barang tidak boleh kosong!");
    }
} while (nama.isEmpty());

// Input Jumlah Barang
int jumlah;
while (true) {
    System.out.print("Masukkan Jumlah Barang (contoh: 1): ");
    if (scanner.hasNextInt()) {
        jumlah = scanner.nextInt();
        scanner.nextLine();
        if (jumlah > 0) break;
        else System.out.println("Error: Jumlah barang harus lebih dari 0!");
    } else {
        System.out.println("Error: Input tidak valid! Masukkan angka.");
        scanner.next();
    }
}

// Input Kategori Barang
String kategori;
while (true) {
    System.out.print("Pilih Kategori (1: Baru, 2: Bekas): ");
    if (scanner.hasNextInt()) {
        int kategoriPilihan = scanner.nextInt();
        scanner.nextLine();
        if (kategoriPilihan == 1) {
            kategori = "Baru";
            break;
        } else if (kategoriPilihan == 2) {
            kategori = "Bekas";
            break;
        } else {
            System.out.println("Error: Pilihan tidak valid! Silakan pilih 1 atau 2.");
        }
    } else {
        System.out.println("Error: Input tidak valid! Masukkan angka.");
        scanner.next();
    }
}

// Input Harga Awal
int hargaAwal;
while (true) {
    System.out.print("Masukkan Harga Awal (contoh: 1000000): ");
    if (scanner.hasNextInt()) {
        hargaAwal = scanner.nextInt();
        scanner.nextLine();
        if (hargaAwal >= 0) break;
        else System.out.println("Error: Harga tidak boleh negatif!");
    } else {

```

```

        System.out.println("Error: Input tidak valid! Masukkan angka.");
        scanner.next();
    }
}

// CARI objek anggota dari currentUser ID
Anggota pemilik = currentUser;
if (pemilik == null) {
    System.out.println("Gagal menambahkan barang: akun tidak ditemukan.");
    return;
}

// Cek status anggota
boolean isPremium = (pemilik instanceof AnggotaPremium);

// Tambah ke daftar barang
Barang barangBaru = new Barang(id, nama, jumlah, kategori, hargaAwal, pemilik,
isPremium, false, false);
daftarBarang.add(barangBaru);

System.out.println("\n=====");
System.out.println("Barang berhasil ditambahkan!");
System.out.println("ID Barang: " + id);
System.out.println("Nama Barang: " + nama);
System.out.println("Jumlah: " + jumlah);
System.out.println("Kategori: " + kategori);
System.out.println("Harga Awal: Rp" + hargaAwal);
System.out.println("Pemilik: " + pemilik.getNama() + (isPremium ? " (Premium)" :
""));
System.out.println("=====");
}

static void ajukanLelang() {
    System.out.println("\n=====");
    System.out.println("===          Mengajukan Lelang          ===");
    System.out.println("=====");

    // Hitung jumlah barang yang sudah diajukan oleh currentUser
    int jumlahBarangDiajukan = 0;
    for (Barang b : daftarBarang) {
        if (b.getPemilik() != null && b.getPemilik().equals(currentUser) &&
b.isSudahDilelang()) {
            jumlahBarangDiajukan++;
        }
    }

    // Cek apakah user biasa melewati batas pengajuan
    if (!currentUser.isPremium() && jumlahBarangDiajukan >= 7) {
        System.out.println("Error: Akun biasa hanya bisa mengajukan maksimal 3
barang untuk dilelang.");
    }
}

```

```

        System.out.println("Silakan upgrade ke akun premium untuk mengajukan lebih
banyak.");
        return;
    }

    // Tampilkan barang yang belum dilelang
    ArrayList<Barang> barangBelumDilelang = new ArrayList<>();
    for (Barang b : daftarBarang) {
        if (b.getPemilik() != null && b.getPemilik().equals(currentUser) &&
!b.isSudahDilelang()) {
            barangBelumDilelang.add(b);
        }
    }

    if (barangBelumDilelang.isEmpty()) {
        System.out.println("Tidak ada barang yang bisa diajukan untuk dilelang.");
        return;
    }

    // Tampilkan daftar barang

System.out.println("-----");
        System.out.printf("%-5s %-15s %-20s %-15s %15s\n", "No", "ID Barang", "Nama
Barang", "Kategori", "Harga Awal");

System.out.println("-----");
        for (int i = 0; i < barangBelumDilelang.size(); i++) {
            Barang b = barangBelumDilelang.get(i);
            System.out.printf("%-5d %-15s %-20s %-15s Rp%,12d\n",
                (i + 1), b.getId(), b.getNama(), b.getKategori(), b.getHargaAwal());
        }

System.out.println("-----");
        System.out.print("Masukkan ID Barang yang ingin dilelang (contoh: BRG001): ");
        String idBarang = scanner.nextLine().trim();

        boolean found = false;
        for (Barang b : daftarBarang) {
            if (b.getId().equals(idBarang) && b.getPemilik() != null &&
b.getPemilik().equals(currentUser) && !b.isSudahDilelang()) {
                b.setButuhPersetujuan(true);
                b.setSudahDilelang(true);
                System.out.println("\n=====");
                System.out.println("Barang berhasil diajukan untuk dilelang!");
                System.out.println("ID Barang: " + b.getId());
                System.out.println("Nama Barang: " + b.getNama());
                System.out.println("Status: Menunggu persetujuan admin.");
                System.out.println("=====");
                found = true;
                break;
            }
        }

        if (!found) {
            System.out.println("\nError: ID Barang tidak ditemukan atau sudah diajukan
untuk lelang.");

```

```

    }
}

static void lihatStatusLelang() {
    System.out.println("\n=====");
    System.out.println("==          Status Lelang & Tawaran          ==");
    System.out.println("=====");

    boolean found = false;
    for (Barang b : daftarBarang) {
        if (b.getPemilik() != null && b.getPemilik().equals(currentUser) &&
            b.isSudahDilelang() && !b.isButuhPersetujuan()) {
            b.tampilkanInfoLelang();
            found = true;
        }
    }
    if (!found) {
        System.out.println("Tidak ada barang lelang yang aktif.");
    }
}

static void submenuLihatBarang() {
    while (true) {
        System.out.println("\n=====");
        System.out.println("==          Lihat Barang Milik Sendiri          ==");
        System.out.println("=====");
        System.out.println("1. Barang Belum Dilelang");
        System.out.println("2. Barang Sedang Dilelang (dengan sisa waktu)");
        System.out.println("3. Barang yang Menunggu Persetujuan");
        System.out.println("4. Barang yang Sudah Dilelang (pemenang bid / opsi lelang ulang)");
        System.out.println("5. Kembali");
        System.out.println("=====");
        System.out.print("Pilih menu (1-5): ");
        int pilihan = scanner.nextInt();
        scanner.nextLine();

        switch (pilihan) {
            case 1:
                lihatBarangBelumDilelang();
                break;
            case 2:
                lihatBarangSedangDilelang();
                break;
            case 3:
                lihatBarangMenungguPersetujuan();
                break;
        }
    }
}

```

```

        case 4:
            lihatBarangSudahDilelang();
            break;
        case 5:
            System.out.println("Kembali ke menu sebelumnya...");
            return;
        default:
            System.out.println("\nError: Pilihan tidak valid! Silakan pilih
angka antara 1 hingga 5.");
    }
}
}

```

```

static void lihatBarangMenungguPersetujuan() {
    System.out.println("\n=====");
    System.out.println("==          Barang Menunggu Persetujuan          ==");
    System.out.println("=====");
    boolean found = false;
    for (Barang b : daftarBarang) {
        if (b.getPemilik() != null && b.getPemilik().equals(currentUser) &&
b.isButuhPersetujuan()) {
            b.tampilkanInfo();
            found = true;
        }
    }
    if (!found) {
        System.out.println("Tidak ada barang yang menunggu persetujuan admin.");
    }
}

```

```

static void lihatBarangBelumDilelang() {
    System.out.println("\n=====");
    System.out.println("==          Barang Belum Dilelang          ==");
    System.out.println("=====");
    boolean found = false;
    for (Barang b : daftarBarang) {
        if (b.getPemilik() != null && b.getPemilik().equals(currentUser) &&
!b.isSudahDilelang()) {
            b.tampilkanInfo();
            found = true;
        }
    }
    if (!found) {
        System.out.println("Tidak ada barang yang belum dilelang.");
    }
}

```

```

static void lihatBarangSedangDilelang() {
    System.out.println("\n=====");
    System.out.println("===          Barang Sedang Dilelang          ===");
    System.out.println("=====");
    boolean found = false;
    for (Barang b : daftarBarang) {
        if (b.getPemilik() != null && b.getPemilik().equals(currentUser) &&
            b.isSudahDilelang() && !b.isButuhPersetujuan()) {
            b.tampilkanInfoLelang();
            found = true;
        }
    }
    if (!found) {
        System.out.println("Tidak ada barang yang sedang dilelang.");
    }
}

public static void lihatBarangSudahDilelang() {
    System.out.println("\n=====");
    System.out.println("===          Barang yang Sudah Dilelang          ===");
    System.out.println("=====");

    ArrayList<Barang> barangSudahDilelang = new ArrayList<>();
    for (Barang b : daftarBarang) {
        if (b.getPemilik() != null && b.getPemilik().equals(currentUser) &&
            b.isSudahDilelang()) {
            barangSudahDilelang.add(b);
        }
    }
    if (barangSudahDilelang.isEmpty()) {
        System.out.println("Tidak ada barang yang sudah dilelang.");
        return;
    }

    System.out.println("-----");
    System.out.printf("%-5s %-20s %-15s %-20s\n", "No", "Nama Barang", "Harga
    Tertinggi", "Penawar Tertinggi");

    System.out.println("-----");
    for (int i = 0; i < barangSudahDilelang.size(); i++) {
        Barang b = barangSudahDilelang.get(i);
        System.out.printf("%-5d %-20s Rp%,12d %-20s\n",
            (i + 1), b.getNama(), b.getHargaTertinggi(), b.getPenawarTertinggi());
    }

    System.out.println("-----");

```



```

        System.out.print("Pilih nomor barang yang ingin diproses (atau 0 untuk kembali):
");
        int pilihan = scanner.nextInt();
        scanner.nextLine();
        if (pilihan == 0) {
            return;
        } else if (pilihan > 0 && pilihan <= barangSudahDilelang.size()) {
            Barang barangDipilih = barangSudahDilelang.get(pilihan - 1);
            System.out.println("\n== Detail Barang ==");
            barangDipilih.tampilkanInfoLelang();

            if (barangDipilih.getJumlahBid() == 0 && System.currentTimeMillis() >
barangDipilih.getWaktuBerakhir()) {
                System.out.println("\nBarang ini tidak memiliki penawar. Apa yang ingin
Anda lakukan?");
                System.out.println("1. Lelang ulang");
                System.out.println("2. Hapus barang");
                System.out.print("Pilih opsi: ");
                int opsi = scanner.nextInt();
                scanner.nextLine();

                if (opsi == 1) {
                    barangDipilih.setButuhPersetujuan(true);
                    barangDipilih.setSudahDilelang(false);
                    barangDipilih.setWaktuBerakhir(System.currentTimeMillis() + (7 * 24
* 60 * 60 * 1000)); // Reset waktu
                    System.out.println("\nBarang berhasil diajukan untuk lelang
ulang.");
                } else if (opsi == 2) {
                    daftarBarang.remove(barangDipilih);
                    System.out.println("\nBarang berhasil dihapus.");
                } else {
                    System.out.println("\nOpsi tidak valid.");
                }

                } else if (barangDipilih.getJumlahBid() > 0 && System.currentTimeMillis() >
barangDipilih.getWaktuBerakhir()) {
                    System.out.println("\nBarang ini sudah dilelang dan memiliki penawar.");
                    System.out.println("Penawar Tertinggi: " +
barangDipilih.getPenawarTertinggi() +
" | Harga Tertinggi: " +
barangDipilih.getHargaTertinggi());
                }

                } else {
                    System.out.println("\nNomor barang tidak valid.");
                }
            }

        public static void menuLelangTersedia() {
            while (true) {

```

```

        System.out.println("\n=====");
        System.out.println("===          Lelang Tersedia          ===");
        System.out.println("=====");
        System.out.println("1. Lihat Barang Lelang dari Anggota Lain (Sorting
Harga/Kategori)");
        System.out.println("2. Melakukan Penawaran");
        System.out.println("3. Aktivitas Lelang (cek status menang/kalah &
verifikasi transaksi)");
        System.out.println("4. Lihat Riwayat Lelang & Transaksi");
        System.out.println("5. Kembali");
        System.out.println("=====");
        System.out.print("Pilih menu (1-5): ");
        int pilihan = scanner.nextInt();
        scanner.nextLine();

        switch (pilihan) {
            case 1:
                lihatBarangLelang();
                break;
            case 2:
                lakukanPenawaran();
                break;
            case 3:
                aktivitasLelang();
                break;
            case 4:
                lihatRiwayatTransaksi();
                break;
            case 5:
                System.out.println("Kembali ke menu sebelumnya...");
                return;
            default:
                System.out.println("\nError: Pilihan tidak valid! Silakan pilih
angka antara 1 hingga 5.");
        }
    }
}

```

```

public static void lihatSemuaBarangLelang() {
    System.out.println("\n=====");
    System.out.println("===          Semua Barang Lelang          ===");
    System.out.println("=====");
    ArrayList<Barang> barangLelang = new ArrayList<>();
    for (Barang b : daftarBarang) {
        if (b.isSudahDilelang() && !b.isButuhPersetujuan()) {
            barangLelang.add(b);
        }
    }

    if (barangLelang.isEmpty()) {
        System.out.println("Tidak ada barang lelang yang tersedia.");
    }
}

```

```

        return;
    }

    System.out.println("-----");
    System.out.printf("%-10s %-20s %-15s %-15s %-15s\n", "ID", "Nama Barang",
        "Kategori", "Harga Awal", "Pemilik");

    System.out.println("-----");

    for (Barang b : barangLelang) {
        String namaPemilik = b.getPemilik().getNama(); //  jika getPemilik()
        mengembalikan Anggota
        String jenis = getJenisAnggotaByNama(namaPemilik);

        // Tambah simbol  kalau premium
        if (jenis.equalsIgnoreCase("Premium")) {
            namaPemilik += " ";
        }

        System.out.printf("%-10s %-20s %-15s Rp%,12d %-15s\n",
            b.getId(), b.getNama(), b.getKategori(), b.getHargaAwal(), namaPemilik);
    }

    System.out.println("-----");
}

public static void lihatBarangLelang() {
    System.out.println("\n=====");
    System.out.println("===      Barang Lelang dari Anggota Lain      ===");
    System.out.println("=====");

    ArrayList<Barang> barangLelang = new ArrayList<>();
    for (Barang b : daftarBarang) {
        if (b.getPemilik() != null && !b.getPemilik().equals(currentUser) &&
            b.isSudahDilelang() && !b.isButuhPersetujuan()) {
            barangLelang.add(b);
        }
    }
    if (barangLelang.isEmpty()) {
        System.out.println("Tidak ada barang lelang yang tersedia.");
        return;
    }
    System.out.println("Pilih metode sorting:");
    System.out.println("1. Melihat Semua Barang Lelang");
}

```

```

        System.out.println("2. Berdasarkan Harga (Murah ke Mahal)");
        System.out.println("3. Berdasarkan Harga (Mahal ke Murah)");
        System.out.println("4. Berdasarkan Kategori (Baru)");
        System.out.println("5. Berdasarkan Kategori (Bekas)");
        System.out.print("Pilih opsi: ");
        int sortOption = scanner.nextInt();
        scanner.nextLine();
        switch (sortOption) {
            case 1:
                break;
            case 2:
                barangLelang.sort(Comparator.comparingInt(b -> b.getHargaAwal()));
                break;
            case 3:
                barangLelang.sort((b1, b2) -> Integer.compare(b2.getHargaAwal(),
b1.getHargaAwal()));
                break;
            case 4:
                barangLelang.removeIf(b -> !b.getKategori().equalsIgnoreCase("Baru"));
                break;
            case 5:
                barangLelang.removeIf(b -> !b.getKategori().equalsIgnoreCase("Bekas"));
                break;
            default:
                System.out.println("Opsi tidak valid, menampilkan semua barang tanpa
sorting.");
        }

        System.out.println("-----");
        System.out.printf("%-10s %-20s %-15s %15s\n", "ID", "Nama Barang", "Kategori",
"Harga Awal");

        System.out.println("-----");
        for (Barang b : barangLelang) {
            System.out.printf("%-10s %-20s %-15s Rp%,12d\n",
                b.getId(), b.getNama(), b.getKategori(), b.getHargaAwal());
        }

        System.out.println("-----");
    }

    public static void lakukanPenawaran() {
        System.out.println("\n=====");
        System.out.println("===          Barang yang Bisa Ditawar          ===");
        System.out.println("=====");
        ArrayList<Barang> barangLelang = new ArrayList<>();
        for (Barang b : daftarBarang) {
            if (!b.getPemilik().equals(currentUser) && b.isSudahDilelang() &&
!b.isButuhPersetujuan()) {
                barangLelang.add(b);
            }
        }
    }

```

```

    }
}
if (barangLelang.isEmpty()) {
    System.out.println("Tidak ada barang lelang yang tersedia untuk ditawar.");
    return;
}

System.out.println("-----");
System.out.printf("%-10s %-20s %-15s %15s%n", "ID", "Nama Barang", "Kategori",
"Harga Tertinggi");

System.out.println("-----");
for (Barang b : barangLelang) {
    System.out.printf("%-10s %-20s %-15s Rp%,12d%n",
        b.getId(), b.getNama(), b.getKategori(), b.getHargaTertinggi());
}

System.out.println("-----");
System.out.print("Masukkan ID barang yang ingin Anda bid (atau tekan Enter untuk
kembali): ");
String idBarang = scanner.nextLine().trim();
if (idBarang.isEmpty()) return;
boolean found = false;
for (Barang b : barangLelang) {
    if (b.getId().equals(idBarang)) {
        System.out.println("\nBarang ditemukan!");
        System.out.print("Masukkan jumlah bid Anda (minimal " +
(b.getHargaTertinggi() + 1) + "): ");
        int bid = scanner.nextInt();
        scanner.nextLine();
        if (bid > b.getHargaTertinggi() && bid >= b.getHargaAwal()) {
            b.setHargaTertinggi(bid);
            b.setPenawarTertinggi(currentUser);
            b.setJumlahBid(b.getJumlahBid() + 1);

            System.out.println("\nBid berhasil! Anda sekarang penawar
tertinggi.");
        } else {
            System.out.println("\nBid harus lebih tinggi dari harga tertinggi
saat ini dan tidak boleh kurang dari harga awal!");
        }
        found = true;
        break;
    }
}

if (!found) {
    System.out.println("\nBarang tidak ditemukan atau tidak bisa dibid.");
}
}

```

```

public static void aktivitasLelang() {
    System.out.println("\n=====");
    System.out.println("===          Aktivitas Lelang          ===");
    System.out.println("=====");

    ArrayList<Barang> barangMenang = new ArrayList<>();
    for (Barang b : daftarBarang) {
        if (b.getPenawarTertinggi().equals(currentUser)) {
            barangMenang.add(b);
        }
    }
    if (barangMenang.isEmpty()) {
        System.out.println("Tidak ada aktivitas lelang yang ditemukan.");
        return;
    }

    System.out.println("-----");
    System.out.printf("%-5s %-20s %15s\n", "No", "Nama Barang", "Harga Tertinggi");

    System.out.println("-----");

    for (int i = 0; i < barangMenang.size(); i++) {
        Barang b = barangMenang.get(i);
        System.out.printf("%-5d %-20s Rp%,12d\n",
            (i + 1), b.getNama(), b.getHargaTertinggi());
    }

    System.out.println("-----");
    System.out.print("Pilih nomor barang yang ingin diproses (atau 0 untuk kembali):");

    int pilihan = scanner.nextInt();
    scanner.nextLine();
    if (pilihan == 0) {
        return;
    } else if (pilihan > 0 && pilihan <= barangMenang.size()) {
        Barang barangDipilih = barangMenang.get(pilihan - 1);
        System.out.println("\n=== Detail Barang ===");
        barangDipilih.tampilkanInfoLelang();
        if (System.currentTimeMillis() > barangDipilih.getWaktuBerakhir()) {
            if (barangDipilih.isButuhVerifikasiPembayaran()) {
                System.out.println("\nStatus: Menunggu Verifikasi Transaksi");
            } else if (!barangDipilih.isButuhVerifikasiPembayaran() &&
barangDipilih.isSudahDilelang()) {
                System.out.println("\nStatus: Transaksi Sudah Terverifikasi");
            } else {
                System.out.println("\nStatus: Anda menang lelang!");
                System.out.println("1. Ajukan Verifikasi Transaksi untuk barang
ini");

                System.out.println("2. Lewati barang ini");
                System.out.print("Pilih opsi: ");
                int opsi = scanner.nextInt();
                scanner.nextLine();
                if (opsi == 1) {
                    barangDipilih.setButuhVerifikasiPembayaran(true);
                    System.out.println("\nPermintaan verifikasi transaksi telah

```

```

diajukan untuk barang ID: " + barangDipilih.getId());
        daftarTransaksi.add(new Transaksi(barangDipilih, currentUser,
        barangDipilih.getHargaTertinggi(), new Date()));
    }
}
} else {
    System.out.println("\nStatus: Lelang masih berlangsung.");
}

} else {
    System.out.println("\nNomor barang tidak valid.");
}
}
}

```

```

public static void lihatRiwayatTransaksi() {
    System.out.println("\n=====");
    System.out.println("==          Riwayat Lelang & Transaksi          ==");
    System.out.println("=====");

    ArrayList<Transaksi> riwayatTransaksiUser = new ArrayList<>();
    for (Transaksi t : daftarTransaksi) {
        if (t.getPenawar().equals(currentUser) &&
        !t.getBarang().isButuhVerifikasiPembayaran() && daftarBarang.contains(t.getBarang())) {
            riwayatTransaksiUser.add(t);
        }
    }

    if (riwayatTransaksiUser.isEmpty()) {
        System.out.println("Tidak ada riwayat transaksi yang ditemukan.");
        return;
    }

    System.out.println("-----");
    System.out.println("-----");
    System.out.printf("%-5s %-20s %-25s %15s %-15s\n", "No", "Barang", "Penjual", "Jenis", "Harga (Rp)", "Tanggal");

    System.out.println("-----");
    System.out.println("-----");

    for (int i = 0; i < riwayatTransaksiUser.size(); i++) {
        Transaksi t = riwayatTransaksiUser.get(i);
        String namaPenjual = t.getBarang().getPemilik().getNama();
        String jenisPenjual = getJenisAnggotaByNama(namaPenjual);

        System.out.printf("%-5d %-20s %-25s Rp%,12d %-15s\n",
            (i + 1),
            t.getBarang().getNama(),

```

```

        namaPenjual + " (" + jenisPenjual + ")",
        t.getHarga(),
        t.getTanggal()
    );
}

System.out.println("-----
-----");
}

// Fungsi bantu untuk cari jenis anggota dari nama
private static String getJenisAnggotaByNama(String nama) {
    for (Anggota a : daftarAnggota) {
        if (a.getNama().equalsIgnoreCase(nama)) {
            return a.getJenisAnggota();
        }
    }
    return "Tidak diketahui";
}
}
}

```

```

public class Anggota {
    private String id;           // ✓ private: akses melalui getter/setter
    protected String nama;      // ✓ protected: bisa diakses
                                // subclass/package
    String noHp;                // ✓ default/package-private
    public String password;     // ✓ public: bebas diakses

    // Constructor
    public Anggota(String id, String nama, String password, String noHp) {
        this.id = id;
        this.nama = nama;
        this.password = password;
        this.noHp = noHp;
    }

    public String getTipe() {
        return "Anggota Umum";
    }

    // === Getter dan Setter untuk id ===
    public String getId() {
        return id;
    }
}

```



```
public void setId(String id) {
    this.id = id;
}

// === Getter dan Setter untuk nama ===
public String getNama() {
    return nama;
}

public void setNama(String nama) {
    this.nama = nama;
}

// === Getter dan Setter untuk noHp ===
public String getNoHp() {
    return noHp;
}

public void setNoHp(String noHp) {
    this.noHp = noHp;
}

// === Getter dan Setter untuk password ===
public String getPassword() {
    return password;
}

public void setPassword(String password) {
    this.password = password;
}

public String getJenisAnggota() {
    return this instanceof AnggotaPremium ? "Premium" : "Biasa";
}

public boolean isPremium() {
    return this instanceof AnggotaPremium;
}
```

```

        // Tampilkan info dengan format rapi
        public void tampilkanInfo() {
            System.out.printf("%-10s %-20s %-15s %-10s\n", id, nama, noHp,
password);

System.out.println("-----");
        }

        // Overload: tampilkan info dengan opsi untuk menampilkan password
        public void tampilkanInfo(boolean tampilkanPassword) {
            if (tampilkanPassword) {
                System.out.printf("ID: %s, Nama: %s, No HP: %s, Password: %s\n", id,
nama, noHp, password);
            } else {
                System.out.printf("ID: %s, Nama: %s, No HP: %s\n", id, nama, noHp);
            }
        }
    }
}

```

```

public class AnggotaBiasa extends Anggota {
    public AnggotaBiasa(String id, String nama, String noHp, String password)
    {
        super(id, nama, noHp, password);
    }

    @Override
    public String getJenisAnggota() {
        return "Biasa";
    }
}

```

```

public class AnggotaPremium extends Anggota {
    private int poinLoyalti;

    public AnggotaPremium(String id, String nama, String noHp, String
password) {
        super(id, nama, noHp, password);
    }
}

```

```

        this.poinLoyalti = 0;
    }

    public int getPoinLoyalti() {
        return poinLoyalti;
    }

    public void tambahPoin(int poin) {
        this.poinLoyalti += poin;
    }

    @Override
    public String getJenisAnggota() {
        return "Premium";
    }
}

```

```

public class Barang {
    private String id;                // ✓ private
    public String nama;               // ✓ public
    int jumlah;                      // ✓ default (package-private)
    protected String kategori;       // ✓ protected

    private int hargaAwal;
    private int hargaTertinggi;
    private Anggota pemilik;
    private Anggota penawarTertinggi;
    private int jumlahBid;
    private boolean butuhPersetujuan;
    private boolean sudahDilelang;
    private boolean butuhVerifikasiPembayaran;
    private long waktuBerakhir;

    // === Constructor ===
    public Barang(String id, String nama, int jumlah, String kategori, int
hargaAwal, Anggota pemilik,
                    boolean butuhPersetujuan, boolean butuhVerifikasiPembayaran,
boolean sudahDilelang) {
        this.id = id;
    }
}

```

```

        this.nama = nama;
        this.jumlah = jumlah;
        this.kategori = kategori;
        this.hargaAwal = hargaAwal;
        this.hargaTertinggi = hargaAwal;
        this.pemilik = pemilik;
        this.penawarTertinggi = null; // No bid yet
        this.jumlahBid = 0;
        this.butuhPersetujuan = butuhPersetujuan;
        this.butuhVerifikasiPembayaran = butuhVerifikasiPembayaran;
        this.sudahDilelang = sudahDilelang;
        this.waktuBerakhir = System.currentTimeMillis() + (7 * 24 * 60 * 60 *
1000);
    }

    // === Getter dan Setter untuk id ===
    public String getId() {
        return id;
    }

    public String getName() {
        return nama;
    }

    public void setId(String id) {
        this.id = id;
    }

    // Getter dan Setter untuk kategori
    public String getKategori() {
        return kategori;
    }

    public void setKategori(String kategori) {
        this.kategori = kategori;
    }

    // Getter dan Setter untuk hargaAwal
    public int getHargaAwal() {
        return hargaAwal;
    }

```

```
public void setHargaAwal(int hargaAwal) {
    this.hargaAwal = hargaAwal;
}

// Getter dan Setter untuk hargaTertinggi
public int getHargaTertinggi() {
    return hargaTertinggi;
}

public void setHargaTertinggi(int hargaTertinggi) {
    this.hargaTertinggi = hargaTertinggi;
}

// Getter dan Setter untuk pemilik
public Anggota getPemilik() {
    return pemilik;
}

public void setPemilik(Anggota pemilik) {
    this.pemilik = pemilik;
}

// Getter dan Setter untuk penawarTertinggi
public Anggota getPenawarTertinggi() {
    return penawarTertinggi;
}

public void setPenawarTertinggi(Anggota penawarTertinggi) {
    this.penawarTertinggi = penawarTertinggi;
}

// Getter dan Setter untuk jumlahBid
public int getJumlahBid() {
    return jumlahBid;
}

public void setJumlahBid(int jumlahBid) {
    this.jumlahBid = jumlahBid;
}

// Getter dan Setter untuk butuhPersetujuan
```

```

    public boolean isButuhPersetujuan() {
        return butuhPersetujuan;
    }

    public void setButuhPersetujuan(boolean butuhPersetujuan) {
        this.butuhPersetujuan = butuhPersetujuan;
    }

    // Getter dan Setter untuk sudahDilelang
    public boolean isSudahDilelang() {
        return sudahDilelang;
    }

    public void setSudahDilelang(boolean sudahDilelang) {
        this.sudahDilelang = sudahDilelang;
    }

    // Getter dan Setter untuk butuhVerifikasiPembayaran
    public boolean isButuhVerifikasiPembayaran() {
        return butuhVerifikasiPembayaran;
    }

    public void setButuhVerifikasiPembayaran(boolean
butuhVerifikasiPembayaran) {
        this.butuhVerifikasiPembayaran = butuhVerifikasiPembayaran;
    }

    // Getter dan Setter untuk waktuBerakhir
    public long getWaktuBerakhir() {
        return waktuBerakhir;
    }

    public void setWaktuBerakhir(long waktuBerakhir) {
        this.waktuBerakhir = waktuBerakhir;
    }

    // === Tampilkan Informasi Barang ===
    public void tampilkanInfo() {
        System.out.printf("%-10s %-20s %-10s %-15s %15s %-15s\n",
            "ID", "Nama", "Jumlah", "Kategori", "Harga Awal", "Pemilik");

System.out.println("-----");

```

```

-----");

        System.out.printf("%-10s %-20s %-10d %-15s Rp%,12d %-15s%n",
            id, nama, jumlah, kategori, hargaAwal, pemilik);
    }

    public void tampilkanInfoLelang() {
        long waktuSekarang = System.currentTimeMillis();
        long sisaWaktu = (waktuBerakhir - waktuSekarang) / 1000;
        String statusWaktu = (sisaWaktu > 0) ? (sisaWaktu + " detik tersisa")
: "Lelang sudah berakhir";

        System.out.printf("%-10s %-20s %15s %15s %-20s %-10s %-20s%n",
            "ID", "Nama", "Harga Awal", "Harga Tertinggi", "Penawar
Tertinggi", "Jumlah Bid", "Status Waktu");





System.out.println("-----
-----");

        System.out.printf("%-10s %-20s Rp%,12d Rp%,12d %-20s %-10d %-20s%n",
            id, nama, hargaAwal, hargaTertinggi, penawarTertinggi, jumlahBid,
statusWaktu);
    }
}

```

```

import java.util.Date;

public class Transaksi {
    private Barang barang;           //  private
    protected String penawar;        //  protected
    int harga;                        //  default
    public Date tanggal;             //  public

    // Constructor
    // Add a new constructor to match the required parameters
    public Transaksi(Barang barang, Anggota penawar, int harga, Date tanggal)
{
    // Initialize fields using the provided parameters
    this.barang = barang;
    this.penawar = penawar.getNama(); // Assuming penawar is stored as a

```

```

String (name)
    this.harga = harga;
    this.tanggal = tanggal;
}

// === Getter dan Setter untuk barang ===
public Barang getBarang() {
    return barang;
}

public void setBarang(Barang barang) {
    this.barang = barang;
}

// === Getter dan Setter untuk penawar ===
public String getPenawar() {
    return penawar;
}

public void setPenawar(String penawar) {
    this.penawar = penawar;
}

// === Getter dan Setter untuk harga ===
public int getHarga() {
    return harga;
}

public void setHarga(int harga) {
    this.harga = harga;
}

// === Getter dan Setter untuk tanggal ===
public Date getTanggal() {
    return tanggal;
}

public void setTanggal(Date tanggal) {
    this.tanggal = tanggal;
}

// Menampilkan informasi transaksi

```



```
public void tampilkanInfo() {  
    System.out.printf("%-20s %-15s %-15s %15s %-15s\n",  
        "Barang", "Penjual", "Penawar", "Harga (Rp)", "Tanggal");  
  
    System.out.println("-----  
-----");  
    System.out.printf("%-20s %-15s %-15s Rp%,12d %-15s\n",  
        barang.getNama(), barang.getPemilik(), penawar, harga, tanggal);  
}  
}
```