LAPORAN TUGAS BESAR

MATAKULIAH
PEMROGRAMAN BERORIENTASI OBJEK
CDK3BAB4

Universitas
Telkom

Kelompok:
Aisya Mufidah Najwa (1206230026)
Fadiya Zahra Qatranada (1206230034)
Inaya Revalina Putri Mujana (1206230038)

PROGRAM STUDI S1 SAINS DATA
DIREKTORAT KAMPUS SURABAYA
UNIVERSITAS TELKOM
GANJIL 2025/2026

DAFTAR ISI

BAB I
PENDAHULUAN

1.1. Deskripsi Aplikasi dan Latar Belakang

CookBook merupakan aplikasi berbasis web yang dirancang untuk membantu pengguna dalam mengelola, menyimpan, dan berbagi resep masakan secara terstruktur dan kolaboratif. Aplikasi ini menyediakan berbagai fitur utama, antara lain pencarian resep berdasarkan nama maupun bahan yang ingin digunakan atau dihindari (include dan exclude), pengunggahan resep oleh pengguna, pemberian tanda disukai (like), penyimpanan resep ke dalam koleksi pribadi yang dapat dikategorikan, serta penyusunan rencana menu mingguan dari hari Senin hingga Minggu.

CookBook mengadopsi konsep *user-generated content*, di mana konten resep dibuat dan dikelola langsung oleh pengguna aplikasi. Untuk menjaga kualitas dan validitas data resep yang ditampilkan secara publik, sistem menyediakan peran Admin yang bertugas melakukan proses moderasi berupa persetujuan (*approval*) atau penolakan (*rejection*) terhadap resep yang diunggah oleh pengguna sebelum resep tersebut dapat diakses oleh pengguna lain.

Dari sisi teknis, aplikasi CookBook dibangun dengan pendekatan arsitektur Model–View–Controller (MVC) yang terpisah antara Front-End dan Back-End. Front-End berperan sebagai antarmuka pengguna, sedangkan Back-End menangani logika bisnis, pengelolaan data, dan komunikasi dengan basis data. Pada sisi Back-End, aplikasi ini menerapkan konsep Pemrograman Berorientasi Objek (PBO) menggunakan bahasa pemrograman Java, termasuk penerapan inheritance, abstract class, dan interface untuk membangun sistem yang modular, terstruktur, dan mudah dikembangkan.

1.2. Tujuan dan Manfaat Aplikasi

1.2.1 Tujuan

1. Mengembangkan aplikasi web sederhana berbasis Java yang menerapkan pola MVC dan konsep PBO (inheritance, abstract class, dan interface).
2. Menyediakan sarana bagi pengguna untuk mencari, menyimpan, dan mengorganisasikan resep masakan secara digital.
3. Memudahkan pengguna dalam menyusun rencana menu mingguan sehingga proses memasak menjadi lebih terencana.

1.2.2 Manfaat

1. Bagi pengguna:
   - Mempermudah menemukan resep yang sesuai dengan bahan yang tersedia maupun preferensi tertentu (misalnya menghindari bahan tertentu).
   - Mempermudah menyimpan resep favorit dan resep yang akan dicoba, sehingga tidak perlu mencari ulang.
   - Membantu menyusun rencana masakan mingguan, sehingga belanja bahan dan pengaturan waktu memasak lebih efisien.

2. Bagi pengembang (mahasiswa):
    - Melatih penerapan konsep PBO, MVC, dan perancangan basis data dalam konteks aplikasi nyata.
    - Melatih kolaborasi tim melalui pembagian fitur yang jelas dan terukur.
    - Menjadi portofolio awal pengembangan aplikasi web berbasis Java.

BAB II
ANALISIS DAN PERANCANGAN SISTEM

2.1. Analisis Kebutuhan Sistem

Analisis kebutuhan sistem dilakukan untuk mengidentifikasi kebutuhan fungsional dan non-fungsional yang harus dipenuhi oleh aplikasi CookBook agar dapat berjalan sesuai tujuan pengembangan.

a. Kebutuhan Fungsional

Kebutuhan fungsional menggambarkan fungsi utama yang dapat dilakukan oleh sistem, antara lain:

1. Sistem menyediakan fitur registrasi dan login bagi pengguna.
2. Pengguna dapat mencari resep berdasarkan:
   - nama resep,
   - bahan yang ingin digunakan (include),
   - bahan yang ingin dihindari (exclude).
3. Pengguna dapat melihat detail resep yang telah disetujui (approved).
4. Pengguna dapat memberikan like pada resep.
5. Pengguna dapat menyimpan resep ke dalam kategori tertentu.
6. Pengguna dapat mengunggah resep dengan status awal *pending*.
7. Admin dapat menyetujui (approve) atau menolak (reject) resep yang diunggah user.
8. Pengguna dapat menyusun dan mengelola meal plan mingguan dari hari Senin sampai Minggu.

b. Kebutuhan Non-Fungsional

Kebutuhan non-fungsional meliputi aspek pendukung sistem, antara lain:

1. **Keamanan**
   Sistem membatasi hak akses berdasarkan role (User dan Admin).
2. **Kemudahan penggunaan (usability)**
   Antarmuka dirancang sederhana dan mudah dipahami oleh pengguna.
3. **Kinerja sistem**
   Sistem mampu menampilkan hasil pencarian dan data resep secara responsif.
4. **Maintainability**
   Sistem dibangun menggunakan konsep OOP dan arsitektur MVC sehingga mudah dikembangkan.

2.2. Deskripsi User Role

Aplikasi CookBook memiliki dua jenis peran pengguna, yaitu User dan Admin.

1. User

User merupakan pengguna utama aplikasi dengan hak akses sebagai berikut:

- Melakukan login dan logout.

- Mencari dan melihat resep yang telah disetujui.
- Memberikan like pada resep.
- Menyimpan resep ke dalam kategori tertentu.
- Mengunggah resep baru.
- Menyusun dan mengelola meal plan mingguan.

2. Admin

Admin merupakan pengguna dengan hak akses khusus, antara lain:

- Melihat daftar resep dengan status *pending*.
- Menyetujui (approve) resep agar dapat ditampilkan secara publik.
- Menolak (reject) resep yang tidak memenuhi kriteria.
  Peran admin bertujuan untuk menjaga kualitas dan validitas data resep dalam sistem.

2.3. Arsitektur Sistem

Aplikasi CookBook menerapkan arsitektur Client-Server yang terpisah (Decoupled Architecture) berbasis RESTful API. Meskipun konsep dasar Model–View–Controller (MVC) tetap menjadi landasan, implementasinya terbagi secara fisik antara sisi Front-End dan Back-End:

1. View (Front-End): Diimplementasikan menggunakan React (dengan TypeScript). Bagian ini bertanggung jawab penuh atas antarmuka pengguna dan interaksi. Data tidak diproses di sini, melainkan hanya ditampilkan setelah diambil dari server.

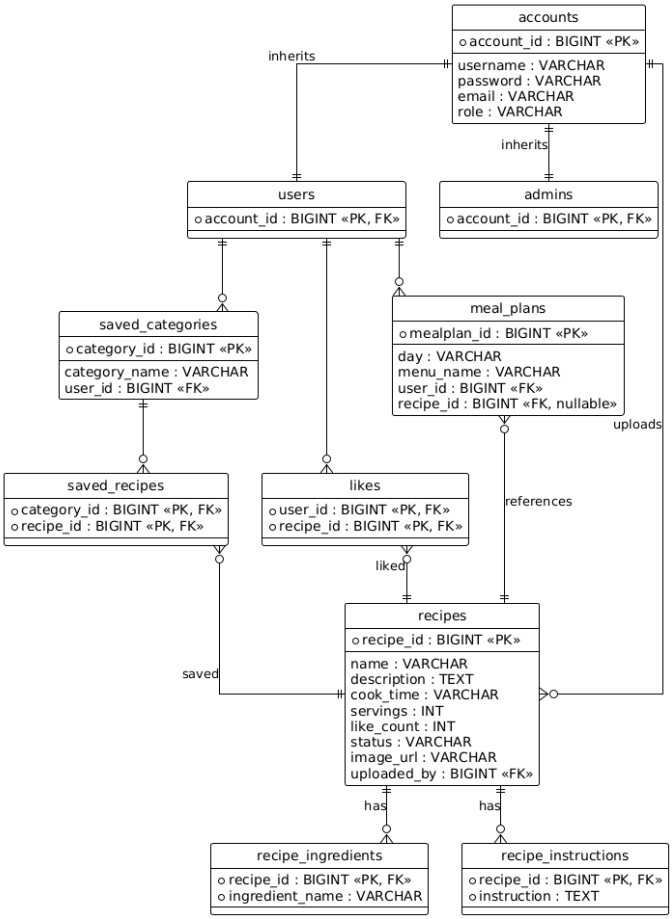
2. Controller & Model (Back-End): Diimplementasikan menggunakan Spring Boot.
   - Controller: Mengatur alur logika bisnis dan menyediakan *endpoints* API untuk diakses oleh *client*.
   - Model: Merepresentasikan struktur data (seperti Account, Recipe, MealPlan) dan aturan bisnis yang berlaku.

3. REST API (Penghubung): Komunikasi antara *View* dan *Controller* dilakukan melalui protokol HTTP dengan format data JSON. Arsitektur ini dipilih karena memisahkan tanggung jawab (*Separation of Concerns*) secara tegas, memungkinkan pengembangan fitur yang lebih fleksibel, serta memudahkan skalabilitas aplikasi di masa depan.

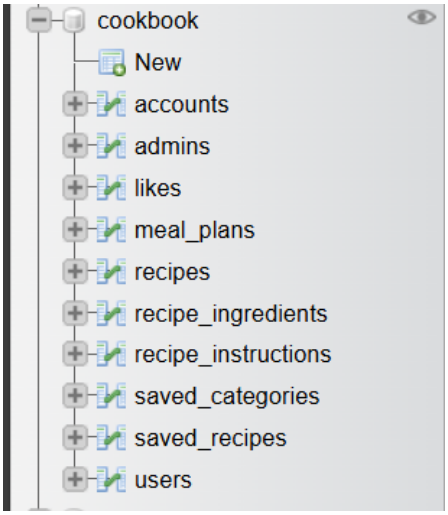
2.4. Perancangan Basis Data (jika ada)

Perancangan basis data pada aplikasi CookBook bertujuan untuk mendukung proses penyimpanan dan pengelolaan data yang berkaitan dengan akun pengguna, resep masakan, interaksi pengguna terhadap resep, serta penyusunan meal plan mingguan. Basis data dirancang menggunakan model relasional agar data tersimpan secara terstruktur, konsisten, dan mudah dikelola.

Perancangan ini mengacu pada Entity Relationship Diagram (ERD) yang menggambarkan struktur tabel, atribut, serta relasi antar tabel dalam sistem.



Secara konseptual, basis data CookBook terdiri dari beberapa tabel utama yang saling berelasi sebagai berikut:



1. **Tabel accounts**

Tabel accounts digunakan untuk menyimpan data autentikasi seluruh akun dalam sistem, baik user maupun admin.
Atribut utama:

- account_id sebagai Primary Key
- username
- password
- email
- role (USER atau ADMIN)

Tabel ini berfungsi sebagai tabel induk (parent table) dalam implementasi inheritance pada basis data.

2. **Tabel users**

Tabel users menyimpan data pengguna aplikasi yang berperan sebagai user biasa.
Atribut utama:

- account_id sebagai Primary Key sekaligus Foreign Key yang mereferensikan tabel accounts

Relasi:

- Satu data pada tabel accounts berelasi satu-ke-satu dengan tabel users
- User dapat:
  - mengunggah resep,
  - menyukai resep,
  - menyimpan resep,
  - serta memiliki meal plan mingguan

3. **Tabel admins**

Tabel admins menyimpan data akun administrator sistem.
Atribut utama:

- account_id sebagai Primary Key sekaligus Foreign Key ke tabel accounts

Relasi ini digunakan untuk memisahkan hak akses admin dan user, meskipun keduanya berasal dari tabel akun yang sama.

4. **Tabel recipes**

Tabel recipes merupakan tabel inti yang menyimpan data resep masakan.
Atribut utama:

- recipe_id sebagai Primary Key
- name
- description
- cook_time
- servings
- like_count
- status (PENDING, APPROVED, REJECTED)
- image_url
- uploaded_by sebagai Foreign Key ke tabel accounts

Relasi:

- Satu akun user dapat mengunggah banyak resep (one-to-many)
- Resep dapat berelasi dengan user melalui fitur like dan penyimpanan

5. **Tabel recipe_ingredients**

Tabel ini digunakan untuk menyimpan daftar bahan dari setiap resep.
Atribut utama:

- recipe_id sebagai Foreign Key ke tabel recipes
- ingredient_name

Relasi:

- Satu resep dapat memiliki banyak bahan (one-to-many)

6. **Tabel recipe_instructions**

Tabel ini menyimpan langkah-langkah pembuatan resep.
Atribut utama:

- recipe_id sebagai Foreign Key ke tabel recipes
- instruction

Relasi:

- Satu resep dapat memiliki banyak instruksi (one-to-many)

7. **Tabel likes**

Tabel likes digunakan untuk mencatat interaksi like antara user dan resep.
Atribut utama:

- user_id sebagai Foreign Key ke tabel users
- recipe_id sebagai Foreign Key ke tabel recipes

Relasi:

- Merepresentasikan hubungan many-to-many antara user dan resep

8. **Tabel saved_categories**

Tabel saved_categories menyimpan kategori penyimpanan resep yang dibuat oleh user.
Atribut utama:

- category_id sebagai Primary Key
- category_name
- user_id sebagai Foreign Key ke tabel users

Relasi:

- Satu user dapat memiliki banyak kategori penyimpanan (one-to-many)

9. **Tabel saved_recipes**

Tabel saved_recipes berfungsi sebagai tabel penghubung antara kategori dan resep.
Atribut utama:

- category_id sebagai Foreign Key ke tabel saved_categories
- recipe_id sebagai Foreign Key ke tabel recipes

Relasi:

- Satu kategori dapat berisi banyak resep
- Satu resep dapat disimpan pada lebih dari satu kategori
  (Relasi many-to-many)

10. **Tabel meal_plans**

Tabel meal_plans digunakan untuk menyimpan rencana menu mingguan pengguna.

Atribut utama:

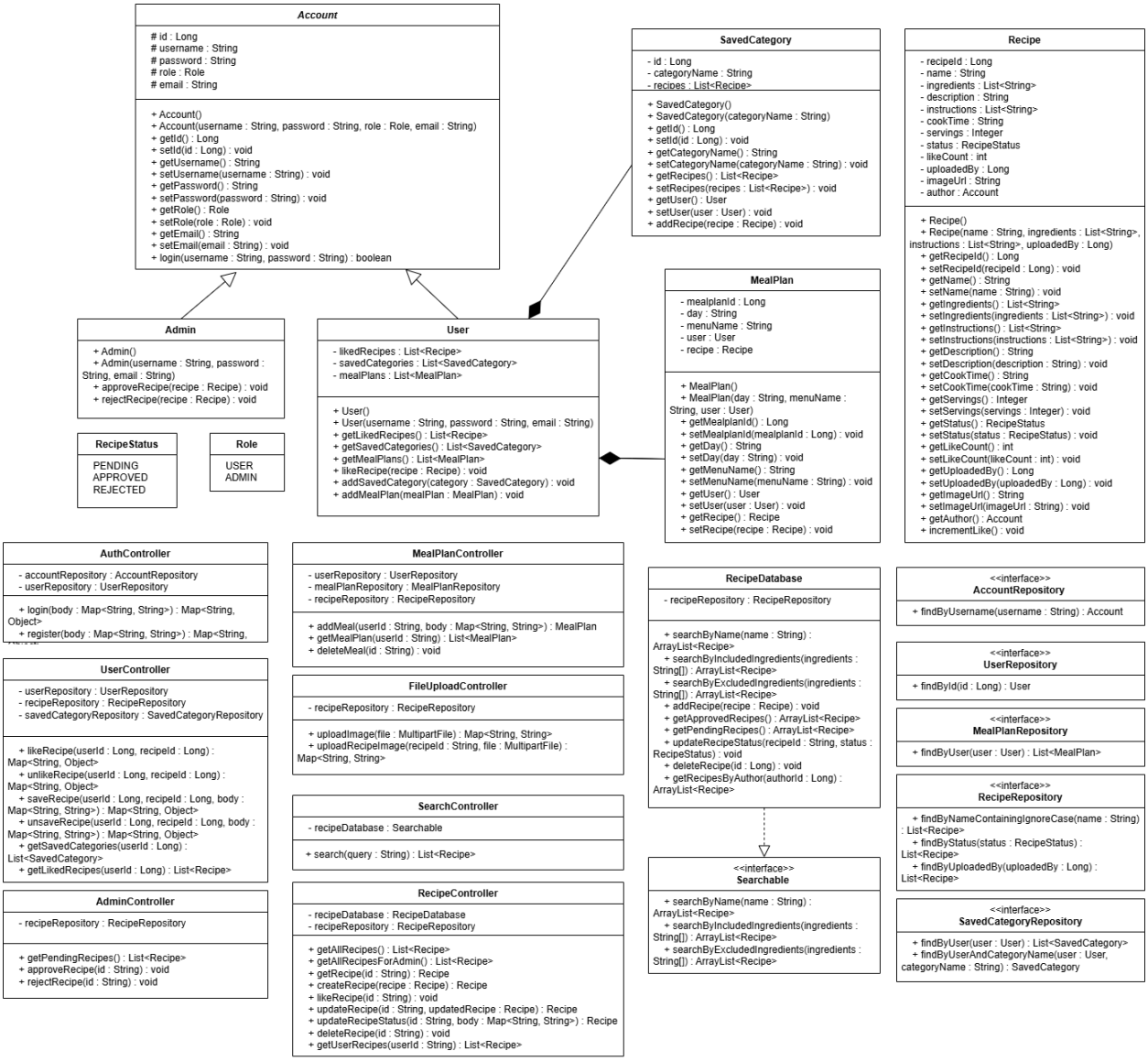
- mealplan_id sebagai Primary Key
- day (Senin–Minggu)
- menu_name
- user_id sebagai Foreign Key ke tabel users
- recipe_id sebagai Foreign Key (nullable) ke tabel recipes

Relasi:

- Satu user dapat memiliki banyak data meal plan
- Meal plan dapat mereferensikan resep tertentu atau berupa menu manual

2.5. Class Diagram Aplikasi

Class diagram pada aplikasi Cookbook digunakan untuk menggambarkan struktur kelas, atribut, method, serta hubungan antar kelas yang membentuk sistem secara keseluruhan. Diagram ini mencerminkan implementasi kebutuhan fungsional aplikasi, mulai dari pengelolaan akun pengguna, resep, hingga penyusunan meal plan.



A. Struktur Kelas

Aplikasi Cookbook terdiri dari beberapa kelompok kelas utama, yaitu:

1. **Model**: Account, User, Admin, Recipe, MealPlan, SavedCategory
2. **Service**: RecipeDatabase, Searchable
3. **Repository**: AccountRepository, UserRepository, RecipeRepository, MealPlanRepository, SavedCategoryRepository
4. **Controller**: AuthController, RecipeController, UserController, AdminController, MealPlanController, FileUploadController, dan SearchController

Setiap kelas memiliki atribut dan method yang merepresentasikan data serta perilaku sesuai dengan perannya dalam sistem.

1) Domain Model (Entity Layer)

1. Account

Kelas Account merupakan kelas abstrak yang berfungsi sebagai entitas dasar akun pengguna. Kelas ini menyimpan atribut umum yang dimiliki oleh seluruh jenis akun dalam sistem, yaitu:

- id
- username
- password
- email
- role

Selain itu, kelas ini menyediakan metode login() yang merepresentasikan proses autentikasi pengguna. Sebagai kelas abstrak, Account tidak dapat diinstansiasi secara langsung dan hanya berfungsi sebagai parent class bagi kelas turunannya.

2. User

Kelas User merupakan turunan dari kelas Account yang merepresentasikan pengguna akhir aplikasi. Kelas ini memiliki atribut tambahan berupa:

- daftar resep yang disukai (likedRecipes),
- kategori resep yang disimpan (savedCategories),
- serta daftar rencana makan (mealPlans).

Kelas User menyediakan metode untuk melakukan interaksi terhadap resep, seperti menyukai resep, menyimpan resep ke dalam kategori tertentu, dan menyusun meal plan.

3. Admin

Kelas Admin juga merupakan turunan dari kelas Account yang memiliki otoritas khusus dalam sistem. Kelas ini bertanggung jawab terhadap proses moderasi konten resep melalui metode:

- approveRecipe()
- rejectRecipe()

Peran admin memastikan bahwa hanya resep yang telah disetujui yang dapat ditampilkan kepada pengguna umum.

4. Recipe

Kelas Recipe merupakan entitas utama yang merepresentasikan resep masakan dalam sistem. Kelas ini menyimpan informasi detail resep, seperti:

- nama resep,
- daftar bahan,
- instruksi memasak,
- waktu memasak,
- jumlah porsi,
- status resep (RecipeStatus),
- jumlah like,
- serta informasi pengunggah resep.

Kelas ini juga memiliki relasi dengan kelas Account sebagai author dan menyediakan metode incrementLike() untuk menambah jumlah like.

5. **MealPlan**

Kelas MealPlan merepresentasikan rencana makan pengguna berdasarkan hari tertentu. Kelas ini memetakan hubungan antara User, Recipe, dan jadwal harian (Senin–Minggu). Meal plan hanya dapat dimiliki oleh satu user dan tidak dapat berdiri sendiri tanpa user tersebut.

6. **SavedCategory**

Kelas SavedCategory digunakan untuk mengelompokkan resep yang disimpan oleh pengguna ke dalam kategori tertentu. Kelas ini memiliki relasi dengan User dan Recipe, di mana satu kategori dapat menyimpan banyak resep, namun resep tetap dapat berdiri secara independen.

2) Controller Layer

Lapisan controller bertanggung jawab menangani HTTP request dari sisi klien, memproses input, dan mengembalikan respons yang sesuai.

- **AuthController**: Mengelola proses autentikasi dan registrasi pengguna.
- **RecipeController**: Mengelola operasi CRUD (Create, Read, Update, Delete) terhadap data resep.
- **UserController**: Menangani interaksi spesifik pengguna terhadap resep, seperti menyukai dan menyimpan resep.
- **MealPlanController**: Mengelola penyusunan dan penghapusan meal plan pengguna.
- **AdminController**: Menangani proses persetujuan dan penolakan resep oleh admin.

3) Service & Repository Layer

- **RecipeDatabase**

  Berfungsi sebagai lapisan service yang mengimplementasikan logika bisnis pencarian dan pengelolaan resep. Kelas ini memisahkan logika aplikasi dari detail teknis akses data.

- **Repository Interfaces**

  Interface repository (AccountRepository, RecipeRepository, dan lainnya) digunakan sebagai abstraksi akses data ke basis data, sehingga mendukung prinsip separation of concerns dan loose coupling.

B. Attribute dan Method

Setiap kelas pada diagram memiliki:

- **Attribute** untuk menyimpan data, seperti:
  - username, password, role pada kelas Account
  - name, ingredients, instructions, status pada kelas Recipe
  - day, menuName pada kelas MealPlan

- **Method** untuk menjalankan fungsi sistem, seperti:
    - login() pada Account
    - likeRecipe() dan addSavedCategory() pada User
    - approveRecipe() dan rejectRecipe() pada Admin
    - searchByName() pada Searchable

Keberadaan atribut dan method ini memastikan setiap kelas memiliki tanggung jawab yang jelas sesuai prinsip Object-Oriented Programming (OOP).

C. Relationship Antar Kelas

Class diagram aplikasi Cookbook mencakup beberapa jenis hubungan antar kelas, yaitu:

1. **Association**

Association menunjukkan hubungan antar kelas yang bersifat saling berinteraksi tanpa ketergantungan siklus hidup.

- Hubungan antara User dan Recipe merupakan association, karena pengguna dapat berinteraksi dengan resep (melihat, menyukai, menyimpan) tanpa memiliki ketergantungan langsung terhadap keberadaan resep tersebut.
- Hubungan antara Recipe dan Account sebagai author juga merupakan association.

2. **Composition**

Composition menunjukkan hubungan kepemilikan kuat, di mana objek tidak dapat berdiri sendiri tanpa objek induknya.

- Hubungan antara User dengan MealPlan dan SavedCategory merupakan composition, karena meal plan dan kategori tersimpan hanya ada jika user tersebut ada.

3. **Aggregation**

Aggregation digunakan pada relasi yang bersifat kepemilikan lemah.

- Hubungan antara SavedCategory dan Recipe bersifat aggregation, karena satu kategori dapat menyimpan banyak resep, namun resep tetap dapat berdiri sendiri di luar kategori tersebut.

D. Inheritance

Inheritance diterapkan untuk menghindari duplikasi kode dan meningkatkan reusabilitas.

- Kelas User dan Admin merupakan turunan dari kelas abstrak Account.
- Dengan inheritance ini, atribut umum seperti username, password, dan email cukup didefinisikan sekali pada kelas Account.

Dengan demikian, class diagram aplikasi Cookbook telah menampilkan atribut, method, relationship (association, aggregation, composition), serta inheritance secara lengkap sesuai kebutuhan sistem.

2.6. Implementasi Inheritence dan Abstract Class/Interface

Pengembangan aplikasi Cookbook menerapkan paradigma Object-Oriented Programming (OOP) secara komprehensif untuk menghasilkan sistem yang modular, reusable, dan mudah dipelihara.

A. Inheritance

Inheritance pada aplikasi Cookbook diterapkan melalui kelas abstrak Account sebagai parent class dan kelas User serta Admin sebagai child class.

- **Account (Abstract Class)**
  Menyediakan atribut dan method umum seperti: id, username, password, role, login()

- **User (Child Class)**
  Mewarisi seluruh atribut dan method dari Account serta menambahkan fungsionalitas khusus pengguna, seperti:
  - menyukai resep
  - menyimpan resep
  - mengelola meal plan

- **Admin (Child Class)**
  Mewarisi Account dan memiliki tanggung jawab khusus untuk:
  - menyetujui resep
  - menolak resep

Penggunaan inheritance ini membuat struktur kode lebih terorganisir dan memudahkan pengembangan fitur di masa depan.

B. Abstract Class

Kelas Account didefinisikan sebagai abstract class karena tidak digunakan secara langsung untuk membuat objek, melainkan sebagai kerangka dasar bagi kelas User dan Admin. Kelas ini berperan sebagai blueprint dasar bagi seluruh jenis akun. Kelas ini tidak dapat diinstansiasi secara langsung dan mengenkapsulasi atribut umum, yaitu:

- id sebagai unique identifier,
- username, password, dan email sebagai kredensial akses,
- role sebagai penanda peran pengguna.

```java
@Entity
@Inheritance(strategy = InheritanceType.JOINED)
@Table(name = "accounts")
public abstract class Account {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "account_id")
    protected Long id;

    @Column(nullable = false)
    protected String username;
```

Hal ini memastikan bahwa setiap akun dalam sistem memiliki struktur data dan perilaku dasar yang sama.

Child Classes (Subclasses): User dan Admin

```java
@Table(name = "admins")
@PrimaryKeyJoinColumn(name = "account_id")
public class Admin extends Account {

    // Constructors
    public Admin() {}

    public Admin(String username, String password, String email) {
        super(username, password, Role.ADMIN, email);
    }

    // Business Methods
    public void approveRecipe(Recipe recipe) {
```

```java
@Table(name = "users")
@PrimaryKeyJoinColumn(name = "account_id")
public class User extends Account {

    @ManyToMany
    @JoinTable(
        name = "likes",
        joinColumns = @JoinColumn(name = "user_id", referencedColumnName = "account_id"),
        inverseJoinColumns = @JoinColumn(name = "recipe_id")
    )
    @JsonIgnore
    private List<Recipe> likedRecipes = new ArrayList<>();

    @OneToMany(mappedBy = "user", cascade = CascadeType.ALL)
    @JsonIgnore
    private List<SavedCategory> savedCategories = new ArrayList<>();
```

- Kelas User mewarisi (extends) kelas Account dan memperluas fungsionalitasnya dengan fitur personalisasi resep dan pengelolaan meal plan.
- Kelas Admin mewarisi (extends) kelas Account dan memperluas fungsionalitasnya dengan kemampuan moderasi konten.

Hubungan pewarisan ini digambarkan pada class diagram sebagai relasi Generalization, di mana User dan Admin merupakan spesialisasi dari Account.

C. Interface

Konsep interface digunakan untuk menerapkan prinsip abstraction dan polymorphism, serta memisahkan kontrak layanan dari implementasi teknisnya.

- **Searchable Interface**

  Interface Searchable mendefinisikan kontrak perilaku pencarian resep dalam sistem melalui metode:
  - searchByName(String name)
  - searchByIncludedIngredients(String[] ingredients)
  - searchByExcludedIngredients(String[] ingredients)

```java
public interface Searchable {
    ArrayList<Recipe> searchByName(String name);
    ArrayList<Recipe> searchByIncludedIngredients(String[] ingredients);
    ArrayList<Recipe> searchByExcludedIngredients(String[] ingredients);
}
```

```java
public class RecipeDatabase implements Searchable {

    @Autowired
    private RecipeRepository recipeRepository;

    @Override
    public ArrayList<Recipe> searchByName(String name) {
        return new ArrayList<>(recipeRepository.findByNameContainingIgnoreCase(name));
    }

    @Override
    public ArrayList<Recipe> searchByIncludedIngredients(String[] ingredients) {
        if (ingredients.length == 0) return new ArrayList<>();
```

Kelas RecipeDatabase mengimplementasikan (implements) interface Searchable dan menerjemahkan metode abstrak tersebut ke dalam logika bisnis konkret yang berinteraksi dengan lapisan repository. Implementasi ini meningkatkan fleksibilitas sistem (loose coupling) dan memungkinkan pengembangan atau perubahan algoritma pencarian di masa depan tanpa memengaruhi komponen lain yang bergantung pada interface tersebut.

- **Repository Interface**
  Interface seperti RecipeRepository dan UserRepository digunakan untuk memisahkan logika akses data dari logika bisnis, sesuai dengan prinsip arsitektur berlapis (layered architecture).

BAB III
IMPLEMENTASI SISTEM

3.1. Implementasi Front-End

Implementasi front-end pada aplikasi CookBook bertujuan untuk menyediakan antarmuka pengguna (User Interface) yang interaktif, responsif, dan ramah pengguna (user-friendly). Lapisan ini berfungsi sebagai View dalam arsitektur Model–View–Controller (MVC), bertugas memvisualisasikan data dan menangani interaksi pengguna sebelum diproses lebih lanjut oleh back-end.

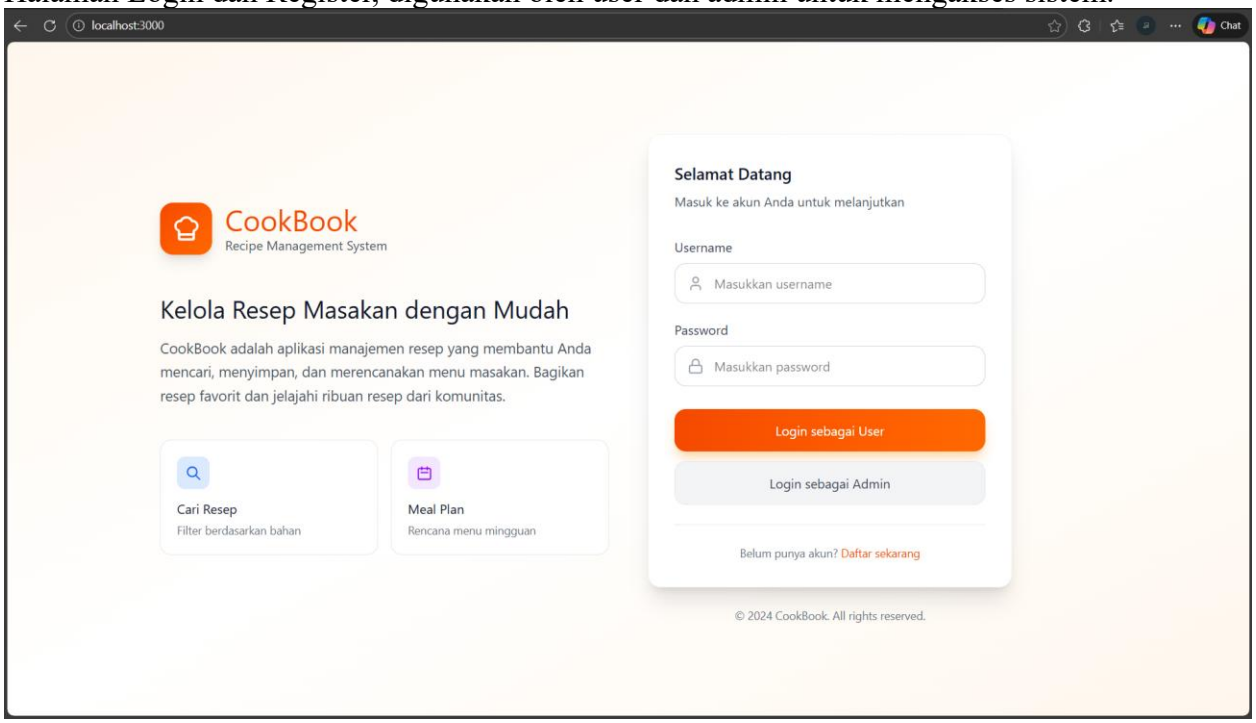
3.1.1. Desain Antarmuka Pengguna

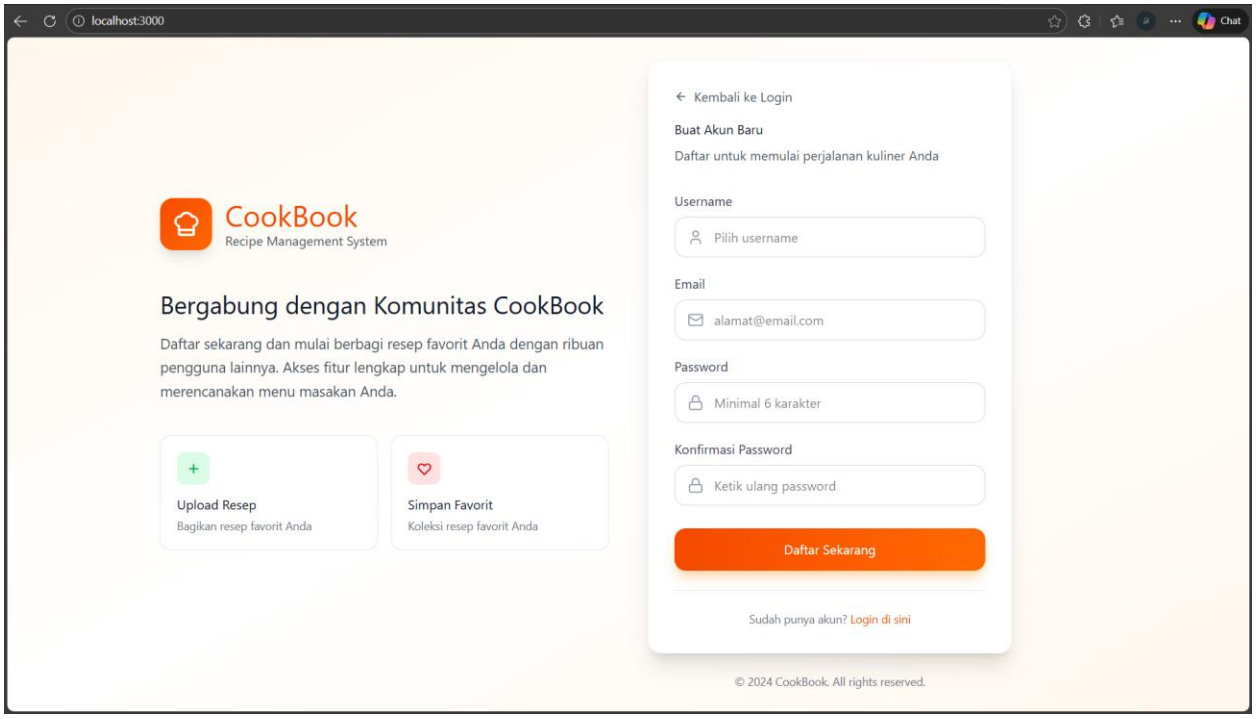
Tampilan antarmuka pada aplikasi CookBook dirancang dengan konsep sederhana dan mudah dipahami oleh pengguna. Desain antarmuka menampilkan elemen utama seperti menu navigasi, form input, tombol aksi, dan area informasi resep. Struktur tata letak disusun secara konsisten menggunakan komponen-komponen reusable seperti Sidebar Navigation, Card Layout untuk resep, dan Modal Dialog untuk input data sehingga pengguna dapat dengan mudah berpindah antar fitur seperti pencarian resep, pengelolaan koleksi resep, dan meal plan mingguan. Tampilan ini berfungsi sebagai media interaksi langsung antara pengguna dan sistem.

3.1.2. Halaman Utama Aplikasi

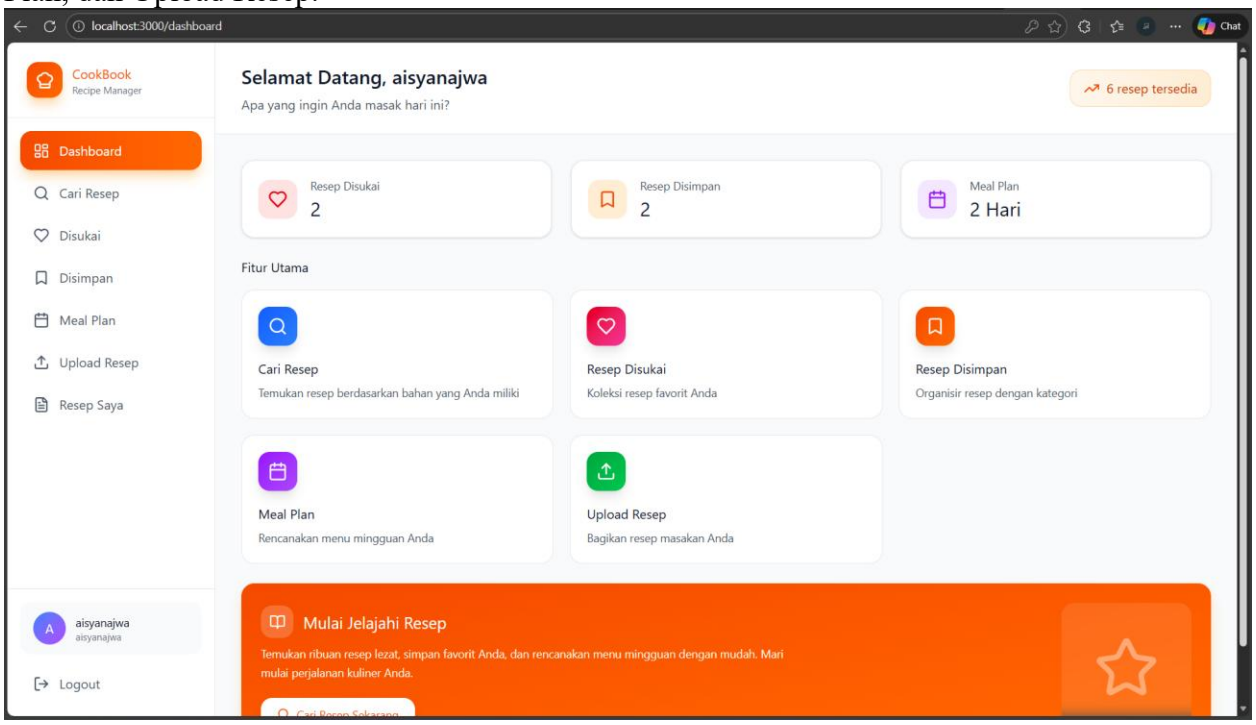
Halaman atau form utama yang diimplementasikan pada aplikasi CookBook meliputi:

1. Halaman Login dan Register, digunakan oleh user dan admin untuk mengakses sistem.

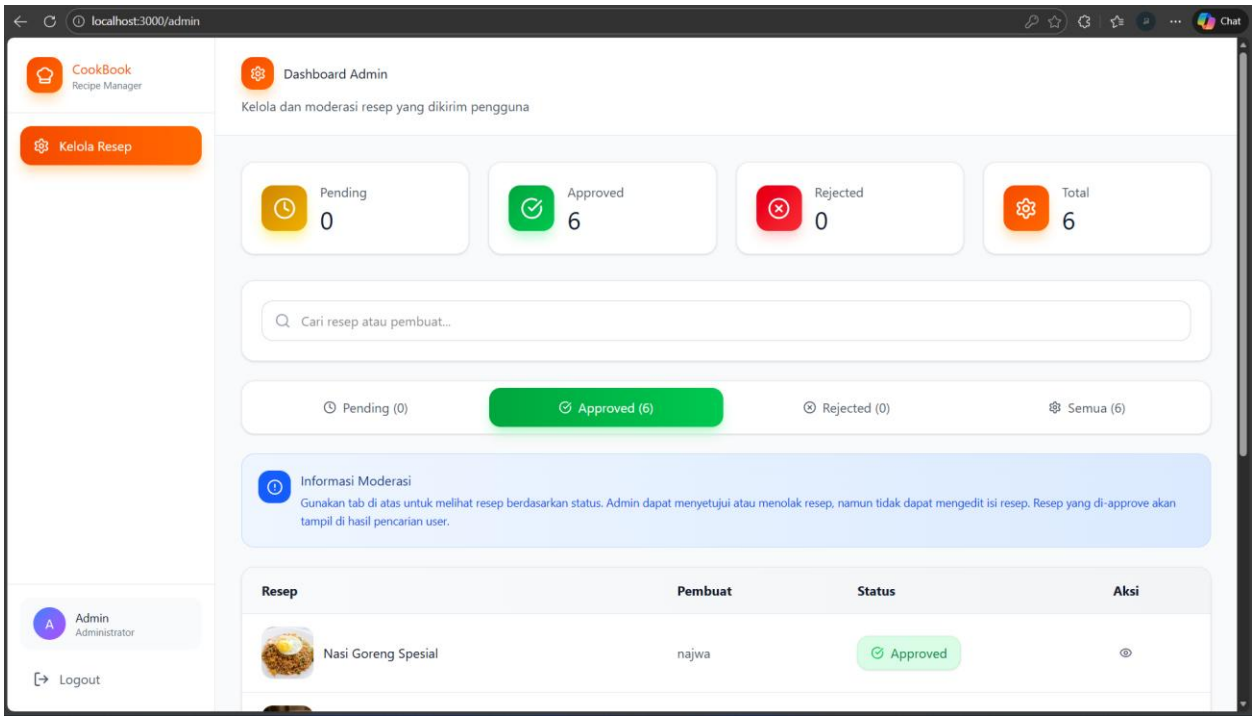




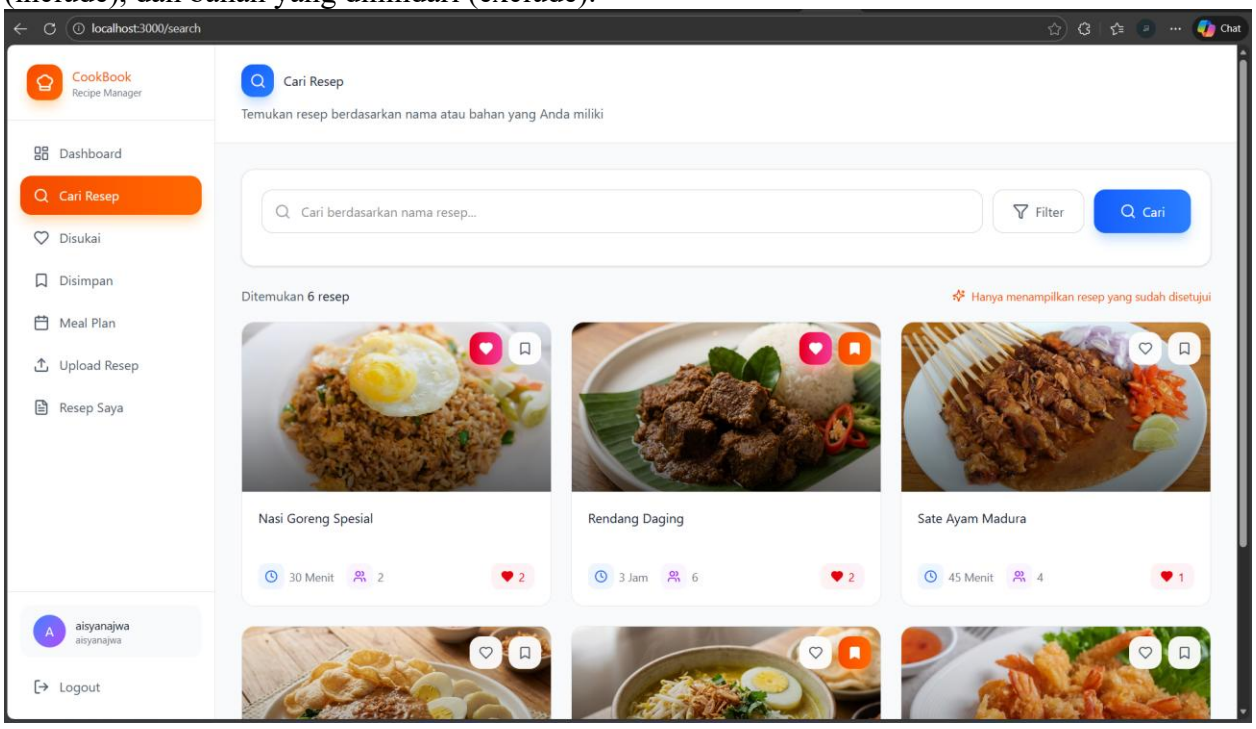
2. Dashboard User, sebagai pusat navigasi menuju fitur Cari Resep, Disukai, Disimpan, Meal Plan, dan Upload Resep.



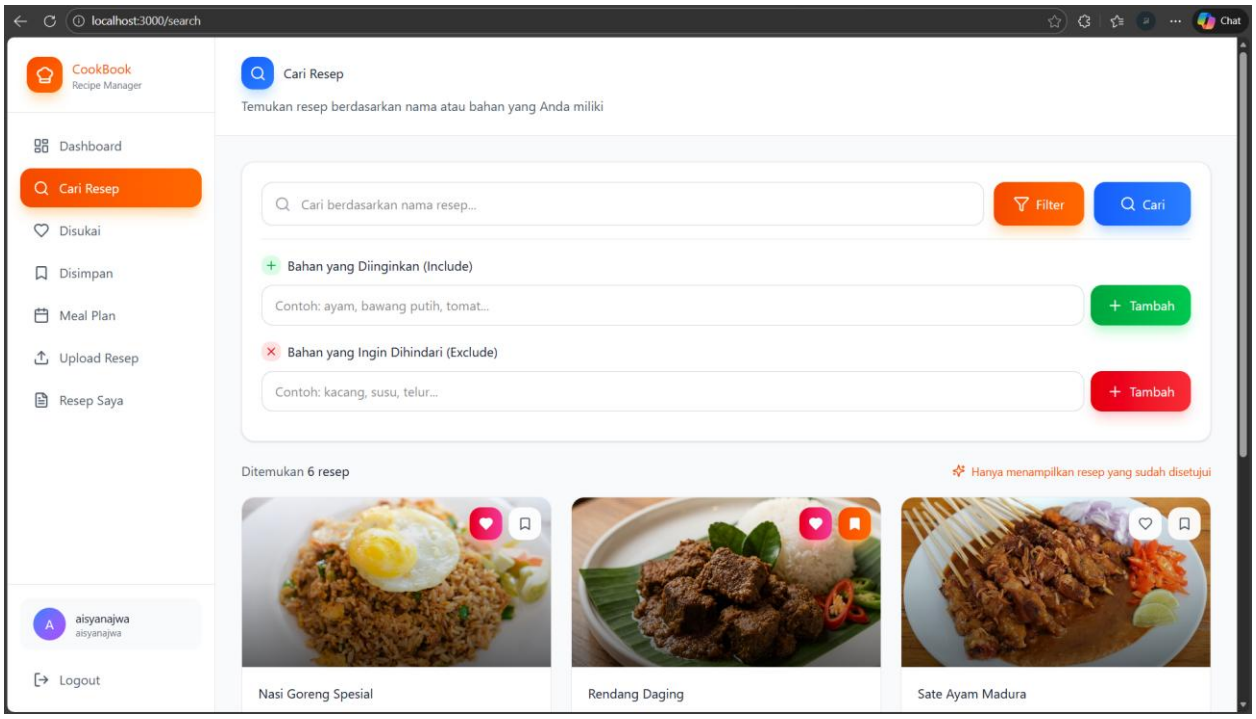
3. Dashboard Admin, antarmuka khusus administrator untuk melakukan moderasi konten, meliputi persetujuan (approval) atau penolakan resep yang diunggah pengguna.



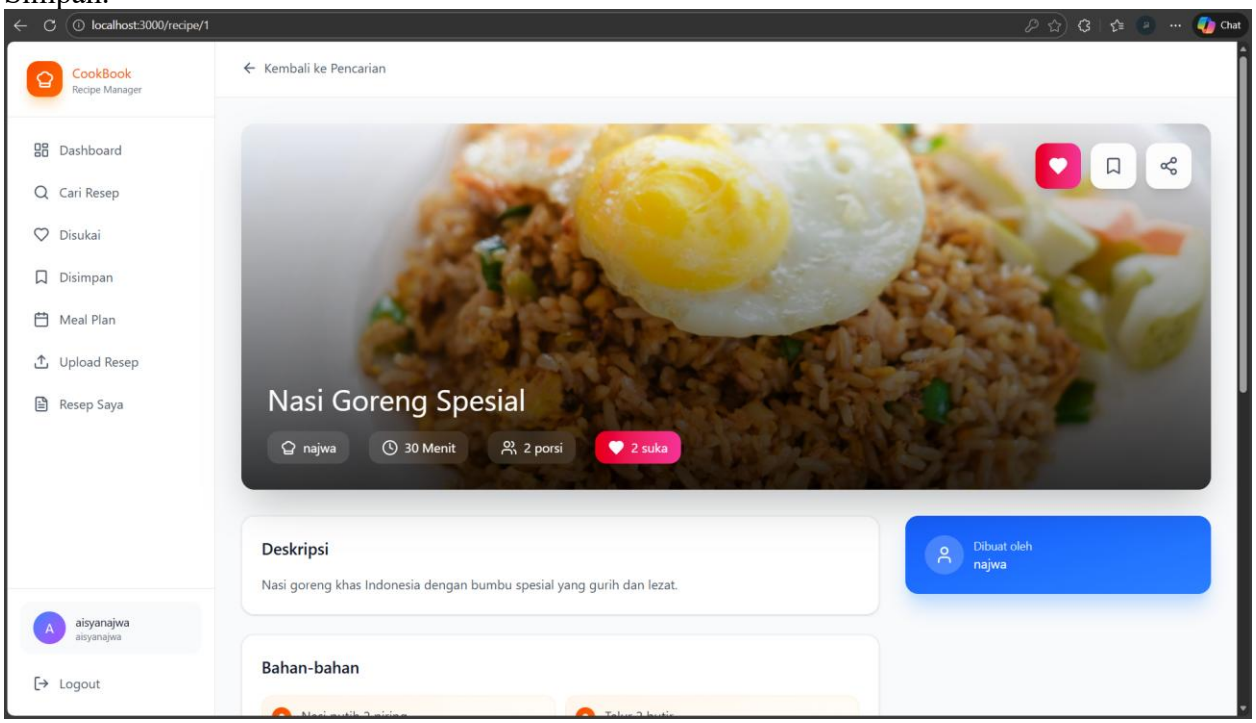
4. Halaman Cari Resep, berisi form pencarian resep berdasarkan nama, bahan yang digunakan (include), dan bahan yang dihindari (exclude).



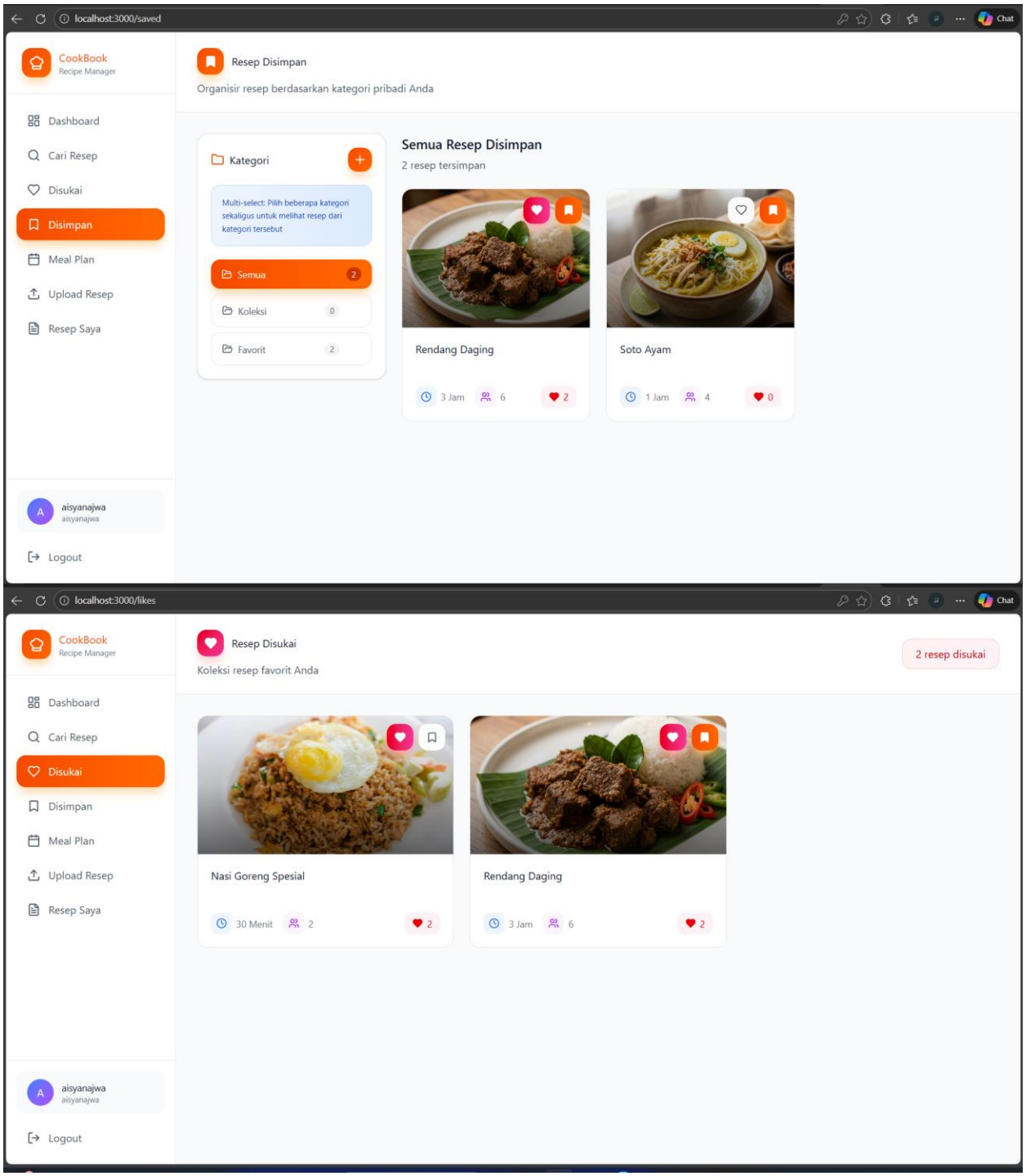
Filter Resep:



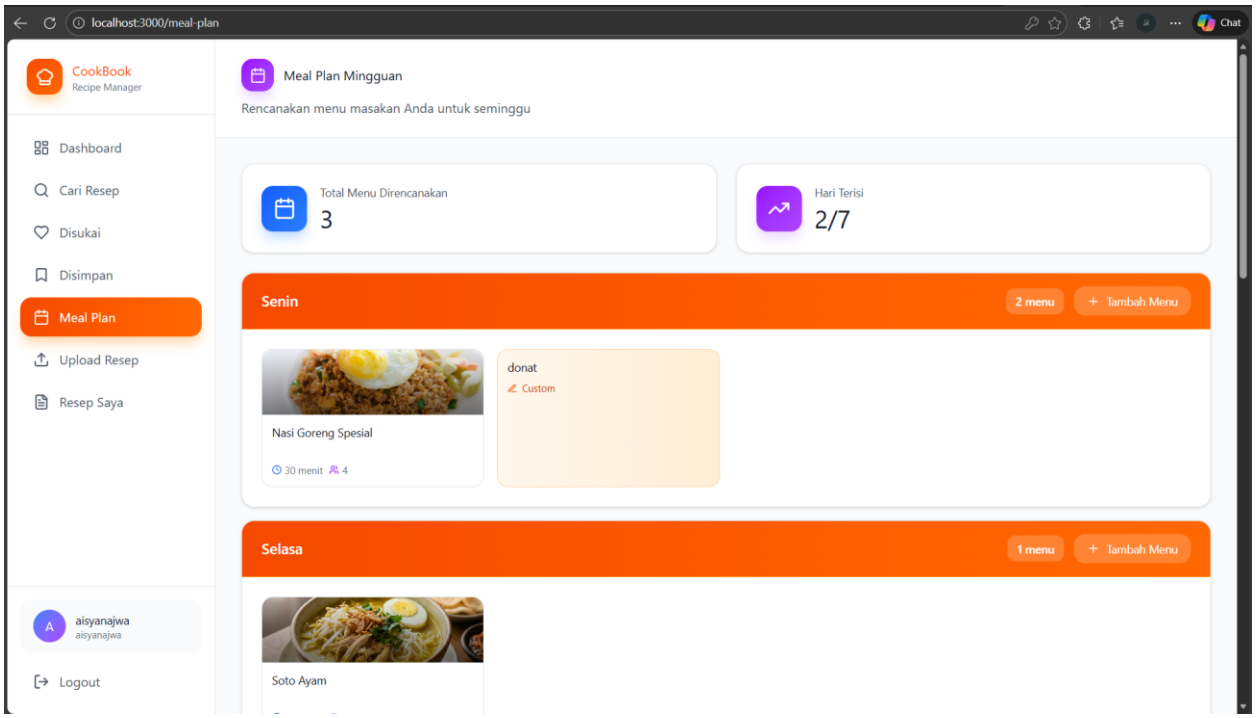
5. Halaman Detail Resep, menampilkan informasi lengkap resep serta tombol aksi Like dan Simpan.



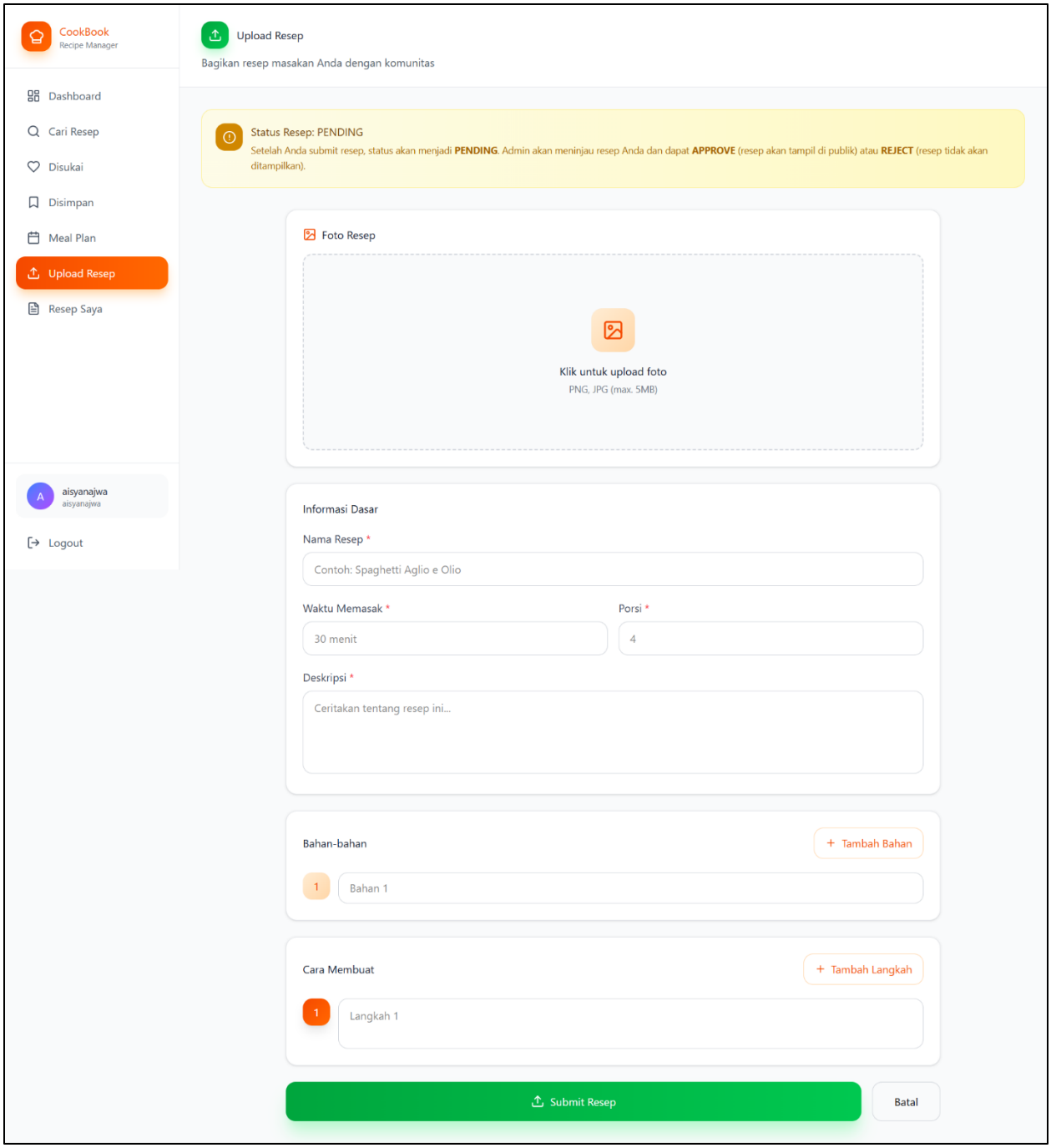
6. Halaman Disimpan dan Disukai, digunakan untuk melihat resep yang disukai dan disimpan pengguna.



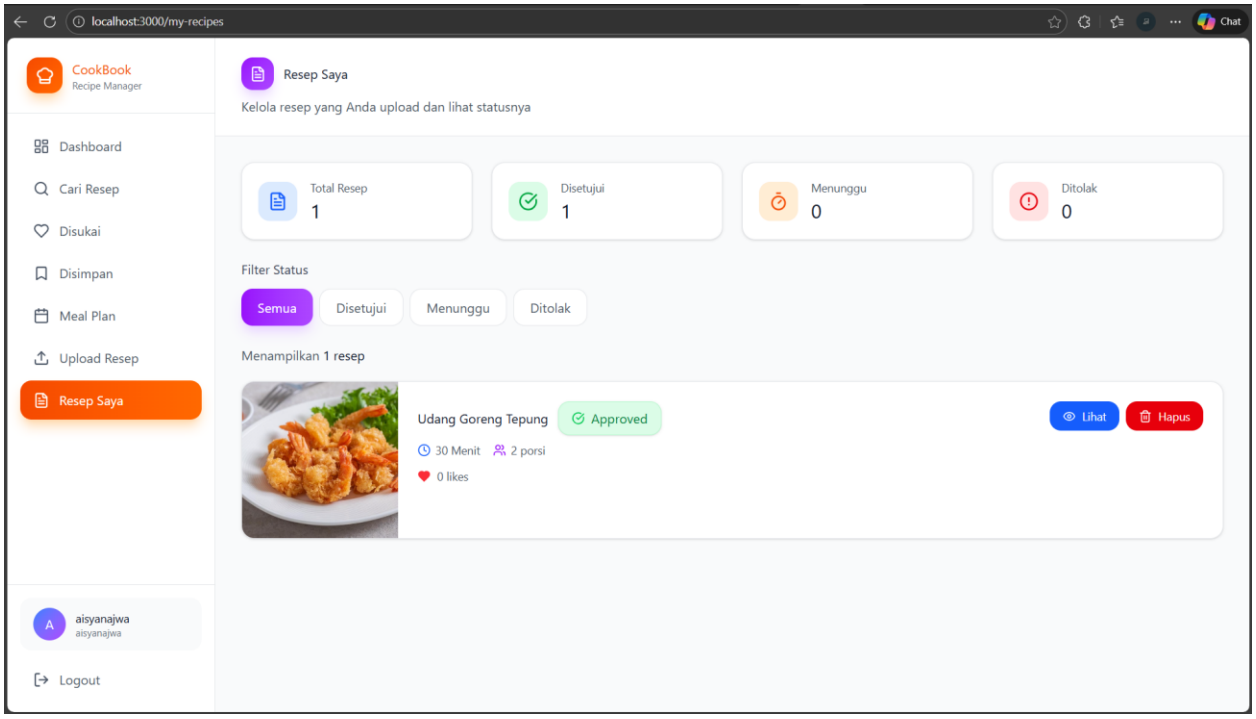
7. Halaman Meal Plan, digunakan untuk menyusun rencana menu mingguan dari hari Senin hingga Minggu.



8. Form Upload Resep, memfasilitasi pengguna untuk berkontribusi dengan mengunggah resep baru beserta foto masakan.



9. Halaman Resep Saya, yang menampilkan resep yang telah di upload



Setiap halaman hanya bertugas menampilkan data dan menerima input dari pengguna, sedangkan pemrosesan data dilakukan oleh controller pada sisi back-end.

3.1.3. Teknologi Front-End yang Digunakan

Pengembangan antarmuka aplikasi CookBook memanfaatkan teknologi web modern untuk memastikan performa yang cepat dan kode yang terstruktur:

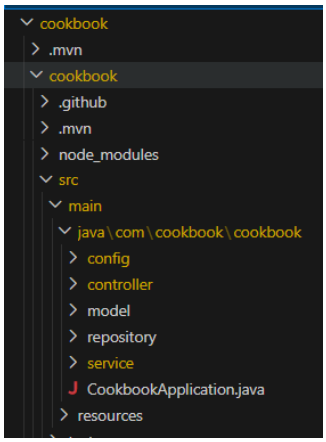
1) **HTML5**: Bahasa markah standar yang digunakan sebagai kerangka dasar struktur halaman web. Dalam ekosistem React, struktur ini direpresentasikan melalui *JSX (JavaScript XML)*.
2) **React (dengan TypeScript)**: Pustaka JavaScript berbasis komponen yang digunakan untuk membangun antarmuka pengguna yang dinamis.
   Penggunaan TypeScript memberikan keamanan tipe data (*type safety*) yang ketat, meminimalkan *runtime error*, dan mempermudah pemeliharaan kode.
3) **Tailwind CSS**: Kerangka kerja CSS *utility-first* yang digunakan untuk mempercepat proses *styling* dan memastikan desain yang konsisten serta responsif di berbagai ukuran layar.
4) **Vite**: *Build tool* generasi baru yang digunakan untuk mempercepat waktu pengembangan (*serve*) dan proses *bundling* aplikasi.
5) **Lucide React**: Pustaka ikon vektor ringan yang digunakan untuk memperkaya visual antarmuka pengguna.
6) **Fetch API**: Standar browser modern yang digunakan untuk melakukan permintaan HTTP (*HTTP Requests*) secara asinkron guna menjembatani komunikasi data antara *Front-End* dan *Back-End* tanpa memerlukan dependensi tambahan.

3.2. Implementasi Back-End

Implementasi back-end pada aplikasi CookBook berfungsi sebagai pusat logika bisnis dan manajemen data yang menghubungkan antarmuka pengguna (Front-End) dengan basis data. Sistem ini dikembangkan menggunakan framework Spring Boot dengan bahasa pemrograman Java, serta menerapkan arsitektur RESTful API dan paradigma Pemrograman Berorientasi Objek (Object-Oriented Programming – OOP). Pendekatan ini bertujuan untuk menghasilkan sistem yang modular, terstruktur, dan mudah dikembangkan.

3.2.1 Struktur Proyek

Struktur proyek back-end diorganisasikan ke dalam beberapa *package* yang modular sesuai dengan pola Layered Architecture untuk mendukung prinsip Separation of Concerns.



Pembagian *package* utama pada aplikasi CookBook adalah sebagai berikut:

- **com.cookbook.cookbook.model**
  Berisi kelas-kelas entitas (*Entity*) yang merepresentasikan struktur data dalam basis data, seperti Account, User, Admin, Recipe, MealPlan, SavedCategory, serta enumerasi Role dan RecipeStatus.

- **com.cookbook.cookbook.repository**
  Merupakan lapisan akses data (*Data Access Layer*) yang berisi antarmuka (*interface*) turunan dari JpaRepository. Package ini mencakup UserRepository, RecipeRepository, MealPlanRepository, SavedCategoryRepository, dan AccountRepository.

- **com.cookbook.cookbook.service**
  Berisi logika bisnis aplikasi, termasuk kelas RecipeDatabase yang mengimplementasikan antarmuka Searchable untuk fitur pencarian resep. Lapisan ini berperan sebagai penghubung antara controller dan repository.

- **com.cookbook.cookbook.controller**
  Bertindak sebagai *API endpoint* yang menerima permintaan HTTP (*request*) dari klien dan mengembalikan respons. Kelas utama pada package ini meliputi AuthController, RecipeController, UserController, AdminController, dan MealPlanController.

- **com.cookbook.cookbook.config**
  Berisi konfigurasi aplikasi, seperti WebConfig untuk pengaturan CORS (Cross-Origin Resource Sharing) serta DataSeeder untuk inisialisasi data awal pada sistem.

3.2.2 Alur Proses Data

Alur pemrosesan data pada aplikasi CookBook mengikuti arsitektur REST API, dengan tahapan sebagai berikut:

1. **Request**
   Pengguna melakukan aksi pada sisi *Front-End* (misalnya mencari resep atau menyukai resep). Aksi tersebut dikirimkan sebagai HTTP request (GET, POST, PUT, atau DELETE) ke *controller* yang sesuai.

```java
// File: RecipeController.java
@PostMapping
public Recipe createRecipe(@RequestBody Recipe recipe) {
    // Controller menerima input JSON yang otomatis dikonversi menjadi objek Recipe
```

2. **Processing**
   Controller menerima permintaan dan memprosesnya dengan memanggil *service* atau *repository* yang relevan. Logika bisnis utama dieksekusi pada tahap ini.

```java
// Logika bisnis: Set status awal menjadi PENDING
    recipe.setStatus(RecipeStatus.PENDING);
    recipe.setRecipeId(null); // Pastikan buat ID baru

    // Memanggil Service/Database layer
    recipeDatabase.addRecipe(recipe);
```

3. **Data Access**
   Repository berinteraksi dengan basis data MySQL menggunakan Spring Data JPA untuk melakukan operasi pengambilan, penyimpanan, atau pembaruan data secara persisten.

```java
// File: RecipeDatabase.java
public Recipe addRecipe(Recipe recipe) {
    // Repository melakukan query INSERT ke database
    return recipeRepository.save(recipe);
}
```

4. **Response**
   Data hasil pemrosesan dikemas dalam format JSON dan dikirimkan kembali ke *Front-End* untuk ditampilkan kepada pengguna.

```
// Kembali ke Controller
    return recipe; // Objek ini dikirim sebagai HTTP Response (JSON 200 OK)
}
```

3.2.3 Implementasi Kelas dan Method Utama

Berikut adalah implementasi kelas dan metode utama yang mendukung fungsionalitas inti sistem:

A. Model (Entity Classes)

7) **Account (Abstract Class)**
   Kelas Account merupakan kelas induk abstrak yang merepresentasikan data dasar seluruh jenis akun dalam sistem. Kelas ini menerapkan konsep Abstraction dan Inheritance dengan strategi JOINED, sehingga atribut umum seperti id, username, password, email, dan role hanya didefinisikan satu kali.

```
@Entity
@Inheritance(strategy = InheritanceType.JOINED)
@Table(name = "accounts")
public abstract class Account {
```

```
public Account(String username, String password, Role role, String email) {
    this.username = username;
    this.password = password;
    this.role = role;
    this.email = email;
}
```

```
public Long getId() {
    return id;
}

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

public String getUsername() {
    return username;
}

public void setUsername(String username) {
    this.username = username;
}

public String getPassword() {
    return password;
}

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

```
public Role getRole() {
    return role;
}

public void setRole(Role role) {
    this.role = role;
}

public String getEmail() {
    return email;
}

public void setEmail(String email) {
    this.email = email;
}

// Business Methods
public boolean login(String username, String password) {
    return this.username.equals(username) && this.password.equals(password);
}
```

Method Utama:

- login(String username, String password): Memverifikasi kesesuaian kredensial pengguna.
- getId(), getUsername(): Digunakan untuk mengakses data atribut yang telah dienkapsulasi.

Kelas ini menjadi dasar bagi kelas User dan Admin, sehingga mendukung prinsip *code reuse*.

- **User (extends Account)**
  Kelas User merupakan turunan dari kelas Account yang merepresentasikan pengguna

umum aplikasi. Kelas ini memiliki atribut tambahan berupa likedRecipes untuk menyimpan daftar resep yang disukai, savedCategories untuk koleksi resep tersimpan, serta mealPlans untuk pengelolaan jadwal makan.

Metode utama pada kelas ini meliputi likeRecipe(Recipe recipe) untuk menyukai resep, addSavedCategory(SavedCategory category) untuk membuat dan menambahkan kategori resep, serta addMealPlan(MealPlan mealPlan) untuk menambahkan rencana makan baru.

```java
@Entity
@Table(name = "users")
@PrimaryKeyJoinColumn(name = "account_id")
public class User extends Account {
```

```java
// Business Methods
public void likeRecipe(Recipe recipe) {
    likedRecipes.add(recipe);
    recipe.incrementLike();
}

public void addSavedCategory(SavedCategory category) {
    category.setUser(this);
    savedCategories.add(category);
}

public void addMealPlan(MealPlan mealPlan) {
    mealPlans.add(mealPlan);
    mealPlan.setUser(this);
}
```

- **Admin (extends Account)**
  Kelas Admin merupakan turunan dari kelas Account yang digunakan untuk administrator sistem. Kelas ini memiliki kewenangan khusus dalam moderasi konten resep. Metode utama yang disediakan adalah approveRecipe(Recipe recipe) untuk mengubah status resep menjadi APPROVED dan rejectRecipe(Recipe recipe) untuk mengubah status resep menjadi REJECTED.

```java
@Table(name = "admins")
@PrimaryKeyJoinColumn(name = "account_id")
public class Admin extends Account {

    // Constructors
    public Admin() {}

    public Admin(String username, String password, String email) {
        super(username, password, Role.ADMIN, email);
    }

    // Business Methods
    public void approveRecipe(Recipe recipe) {
        recipe.setStatus(RecipeStatus.APPROVED);
    }

    public void rejectRecipe(Recipe recipe) {
        recipe.setStatus(RecipeStatus.REJECTED);
    }
```

- **Recipe**
  Kelas Recipe merupakan entitas utama yang menyimpan informasi resep masakan. Atribut yang dimiliki antara lain judul resep, daftar bahan (*ingredients*), instruksi memasak (*instructions*), status resep (PENDING, APPROVED, atau REJECTED), serta informasi pengunggah (uploadedBy).
  Secara relasional, kelas Recipe memiliki hubungan Many-to-One dengan kelas Account sebagai author, yang memungkinkan satu akun mengunggah banyak resep.

```java
public Recipe(String name, List<String> ingredients, List<String> instructions, Long uploadedBy) {
    this.name = name;
    this.ingredients = ingredients;
    this.instructions = instructions;
    this.uploadedBy = uploadedBy;
    this.status = RecipeStatus.PENDING;
    this.likeCount = 0;
}
```

B. Controller (API Endpoints)

- **RecipeController**
  Kelas RecipeController bertanggung jawab mengelola operasi CRUD terhadap data resep. Controller ini menyediakan endpoint untuk mengambil seluruh resep yang telah berstatus APPROVED untuk ditampilkan kepada publik, menerima data resep baru dari pengguna dengan status awal PENDING, memperbarui status resep dalam proses moderasi admin, serta menghapus resep berdasarkan ID.

- **UserController**
  Kelas UserController menangani interaksi personal pengguna terhadap sistem. Endpoint yang disediakan memungkinkan pengguna untuk menyukai resep, menyimpan resep ke dalam kategori tertentu (misalnya *Sarapan* atau *Makan Malam*), menghapus resep dari koleksi simpanan, serta mengambil daftar resep yang disukai oleh pengguna tertentu. Pada setiap proses, sistem melakukan validasi terhadap keberadaan user dan recipe yang bersangkutan.

- **MealPlanController**
  Kelas MealPlanController digunakan untuk mengelola fitur jadwal makan mingguan. Controller ini menyediakan endpoint untuk menambahkan entitas MealPlan baru yang menghubungkan pengguna, hari (Senin–Minggu), dan menu, mengambil daftar rencana makan pengguna dalam format JSON, serta menghapus item rencana makan tertentu.

- **AuthController dan AdminController**
  AuthController bertanggung jawab terhadap proses otentikasi dan registrasi akun. Endpoint login digunakan untuk memeriksa kombinasi username dan password, sedangkan endpoint register digunakan untuk mendaftarkan akun baru dengan mendeteksi tipe peran (USER atau ADMIN) secara otomatis menggunakan prinsip polimorfisme.

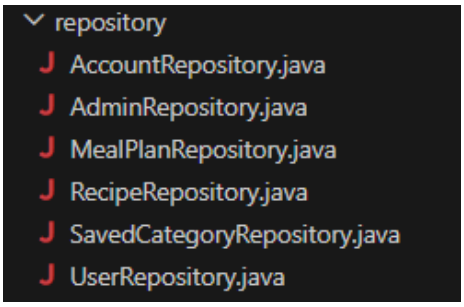
  AdminController digunakan khusus untuk menangani operasi admin, seperti mengambil daftar resep yang masih berstatus PENDING serta melakukan persetujuan atau penolakan resep secara cepat.

C. Service dan Repository

- **RecipeDatabase (Service)**
  Kelas RecipeDatabase berfungsi sebagai lapisan service yang mengimplementasikan antarmuka Searchable. Kelas ini menangani logika bisnis pencarian resep, seperti

pencarian berdasarkan nama dengan memanfaatkan metode findByNameContainingIgnoreCase() pada repository.

- **Repository Interfaces**
  Repository yang digunakan dalam sistem meliputi RecipeRepository, UserRepository, AccountRepository, MealPlanRepository, dan SavedCategoryRepository. Seluruh repository tersebut merupakan turunan dari JpaRepository<T, ID> yang menyediakan metode bawaan seperti save(), findById(), findAll(), dan deleteById(). Dengan pendekatan ini, sistem tidak memerlukan penulisan query SQL secara manual untuk operasi dasar basis data.



3.3. Implementasi Database (Jika ada)

Pada aplikasi CookBook, manajemen data dilakukan menggunakan sistem manajemen basis data (DBMS) MySQL. Interaksi antara aplikasi Back-End dan basis data tidak dilakukan secara manual menggunakan raw SQL, melainkan diabstraksikan melalui Spring Data JPA dan Hibernate. Pendekatan ini memungkinkan pengembang untuk memanipulasi data melalui objek Java (Entity) yang secara otomatis dipetakan ke dalam tabel-tabel basis data menggunakan konsep Object-Relational Mapping (ORM).

Dengan penggunaan ORM, pengelolaan data menjadi lebih terstruktur, aman dari kesalahan penulisan query, serta mendukung pemeliharaan dan pengembangan sistem secara berkelanjutan.

3.3.1 Koneksi Database

Pada aplikasi CookBook, basis data yang digunakan adalah MySQL dan dikelola melalui phpMyAdmin untuk pembuatan database serta pengelolaan tabel. Aplikasi melakukan koneksi ke database dari sisi Back-End (Spring Boot) menggunakan konfigurasi yang didefinisikan dalam file application.properties. Spring Boot menggunakan Java Database Connectivity (JDBC) dengan driver MySQL Connector/J (com.mysql.cj.jdbc.Driver) untuk menjembatani komunikasi antara aplikasi dan basis data.

Detail konfigurasi koneksi yang digunakan dalam proyek ini meliputi:

- Database URL: jdbc:mysql://localhost:3306/cookbook
- Driver: MySQL Connector/J

- DDL Auto: update
  Opsi ini memungkinkan Hibernate untuk memperbarui skema tabel secara otomatis apabila terdapat perubahan pada struktur *Entity*.

```
spring.application.name=cookbook
server.port=8081

# Database Configuration
spring.datasource.url=jdbc:mysql://localhost:3306/cookbook
spring.datasource.username=root
spring.datasource.password=
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# JPA / Hibernate
spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect

# Multipart File Upload Configuration
spring.servlet.multipart.enabled=true
spring.servlet.multipart.max-file-size=10MB
spring.servlet.multipart.max-request-size=10MB
```

Konfigurasi tersebut memastikan bahwa seluruh operasi CRUD (Create, Read, Update, Delete) yang dilakukan melalui lapisan *Repository* tersimpan secara persisten pada basis data cookbook. Setelah koneksi berhasil, sistem dapat menjalankan proses penyimpanan dan pengambilan data sesuai kebutuhan fitur seperti login, upload resep, persetujuan admin, like, penyimpanan resep ke kategori, dan meal plan. Koneksi database memastikan data tersimpan secara permanen dan dapat diakses kembali ketika aplikasi dijalankan.

3.3.2 Query Utama

Query utama pada aplikasi CookBook mencakup operasi CRUD (Create, Read, Update, Delete) yang digunakan untuk mengelola data akun, resep, interaksi pengguna, kategori simpan, dan meal plan. Query tersebut diimplementasikan melalui metode bawaan dan derived query methods dari JpaRepository, sehingga tidak memerlukan penulisan query SQL secara manual.

Dalam implementasi Spring Data JPA, mekanisme query dibagi menjadi dua kategori utama, yaitu metode bawaan dari JpaRepository dan Derived Query Methods (query yang dihasilkan secara otomatis berdasarkan penamaan metode).

A. Metode Bawaan (Standard CRUD)

Seluruh *repository* secara otomatis mewarisi metode standar dari antarmuka JpaRepository, antara lain:

- save(Entity)
  Digunakan untuk menyimpan data baru (*INSERT*) atau memperbarui data yang sudah ada (*UPDATE*).

- findById(Long id)
  Digunakan untuk mengambil satu data spesifik berdasarkan *primary key*.
- findAll()
  Digunakan untuk mengambil seluruh data dari tabel.
- deleteById(Long id)
  Digunakan untuk menghapus data berdasarkan ID.

Metode-metode ini memungkinkan pengelolaan data dasar tanpa perlu menuliskan query SQL secara manual.

B. Query Turunan (Derived Query Methods)

Selain metode bawaan, aplikasi ini mendefinisikan beberapa derived query methods di dalam antarmuka *repository* untuk memenuhi kebutuhan bisnis tertentu. Spring Data JPA secara otomatis menerjemahkan metode-metode ini menjadi perintah SQL yang sesuai.

1. AccountRepository

- Method: findByUsername(String username)
  Fungsi: Mencari akun pengguna berdasarkan *username*. Metode ini digunakan pada proses login dan validasi registrasi untuk mencegah duplikasi akun.
  Analogi SQL: SELECT * FROM accounts WHERE username = ?

2. RecipeRepository

- Method: findByStatus(RecipeStatus status)
  Fungsi: Mengambil daftar resep berdasarkan status tertentu, misalnya resep berstatus APPROVED untuk halaman beranda atau PENDING untuk halaman admin.
  Analogi SQL: SELECT * FROM recipes WHERE status = ?

- Method: findByNameContainingIgnoreCase(String name)
  Fungsi: Melakukan pencarian resep berdasarkan nama secara fleksibel (*fuzzy search*) dan tidak sensitif terhadap huruf besar/kecil (*case-insensitive*).
  Analogi SQL: SELECT * FROM recipes
  WHERE LOWER(name) LIKE LOWER('%?%')

- Method: findByUploadedBy(Long uploadedBy)
  Fungsi: Mengambil seluruh resep yang diunggah oleh pengguna tertentu, yang digunakan pada fitur “Resep Saya”.
  Analogi SQL: SELECT * FROM recipes WHERE uploaded_by = ?

BAB IV
KASIMPULAN DAN SARAN

4.1. Kesimpulan

Berdasarkan hasil perancangan dan implementasi aplikasi CookBook, dapat disimpulkan bahwa aplikasi ini berhasil dikembangkan sebagai aplikasi web berbasis Java yang menerapkan konsep Pemrograman Berorientasi Objek (PBO) dan arsitektur Model–View–Controller (MVC). Aplikasi CookBook mampu memenuhi kebutuhan fungsional utama, seperti registrasi dan login pengguna, pencarian resep berdasarkan nama maupun bahan (include dan exclude), pengunggahan resep oleh pengguna, pemberian like, penyimpanan resep ke dalam kategori, serta penyusunan meal plan mingguan.

Dari sisi teknis, penerapan konsep OOP telah diimplementasikan dengan baik melalui penggunaan inheritance, abstract class, dan interface. Kelas abstrak Account digunakan sebagai parent class bagi User dan Admin untuk menghindari duplikasi atribut dan meningkatkan reusabilitas kode. Selain itu, penggunaan interface Searchable pada lapisan service menunjukkan penerapan abstraction dan polymorphism dalam pengelolaan fitur pencarian resep.

Aplikasi ini juga didukung oleh basis data relasional MySQL yang dirancang menggunakan Entity Relationship Diagram (ERD) untuk memastikan keterkaitan data antar entitas tersimpan secara konsisten dan terstruktur. Dengan memanfaatkan Spring Data JPA dan Hibernate, proses pengelolaan data dapat dilakukan secara efisien tanpa penulisan query SQL manual. Secara keseluruhan, aplikasi CookBook telah memenuhi tujuan pengembangan sebagai media pembelajaran penerapan konsep PBO dalam pengembangan aplikasi web.

4.2. Saran

Meskipun aplikasi CookBook telah berjalan sesuai dengan kebutuhan yang dirancang, terdapat beberapa hal yang dapat dikembangkan lebih lanjut di masa mendatang, antara lain:

1. Penambahan fitur autentikasi berbasis token (seperti JWT) untuk meningkatkan aspek keamanan aplikasi.
2. Pengembangan fitur rekomendasi resep secara otomatis berdasarkan preferensi pengguna atau riwayat interaksi.
3. Penyempurnaan tampilan antarmuka pengguna agar lebih interaktif dan optimal pada berbagai perangkat.
4. Pengembangan fitur notifikasi, misalnya pemberitahuan ketika resep yang diunggah telah disetujui atau ditolak oleh admin.
5. Implementasi deployment ke server publik agar aplikasi dapat diakses secara luas, tidak hanya pada lingkungan lokal.

Dengan adanya pengembangan lanjutan tersebut, diharapkan aplikasi CookBook dapat menjadi sistem yang lebih lengkap, aman, dan bermanfaat bagi pengguna serta memiliki nilai praktis yang lebih tinggi.

LAMPIRAN

Lampiran 1. Link Directory Aplikasi (Onedrive atau Github)

1.1 Link Source Code

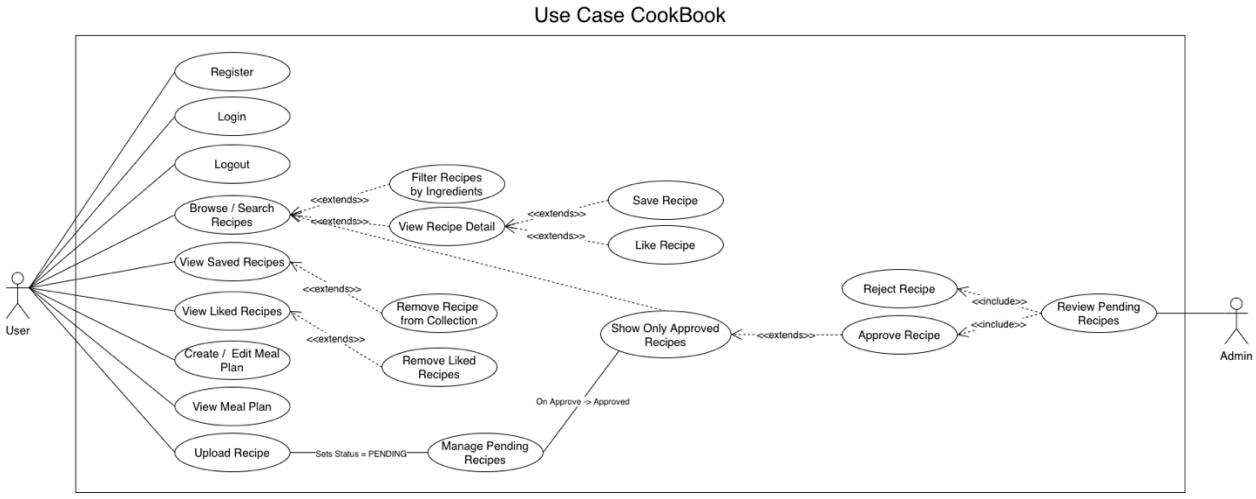
Source Code aplikasi CookBook dapat diakses melalui tautan berikut:
https://github.com/aisyanajwa/cookbookrecipe

Repository tersebut terdiri dari dua bagian utama, yaitu:

- cookbook: backend aplikasi yang dikembangkan menggunakan Spring Boot
- Cookbookwebsiteuidesign: frontend aplikasi yang dikembangkan menggunakan React

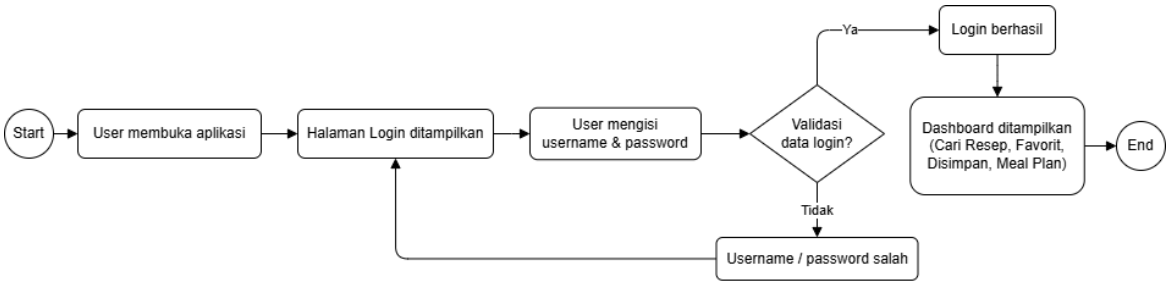
Lampiran 2. Diagram Tambahan

2.1 Use Case Diagram

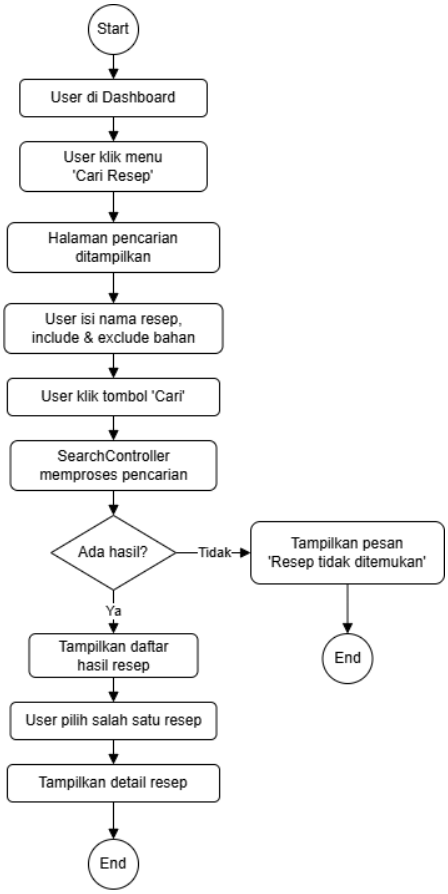


Gambar 2.1 Use Case Diagram

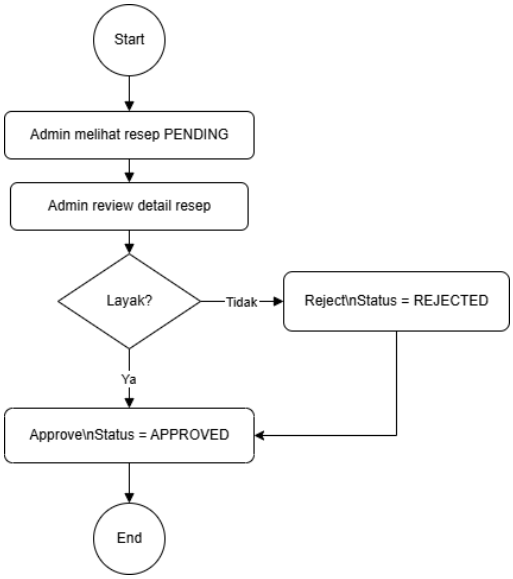
2.2 Activity Diagram



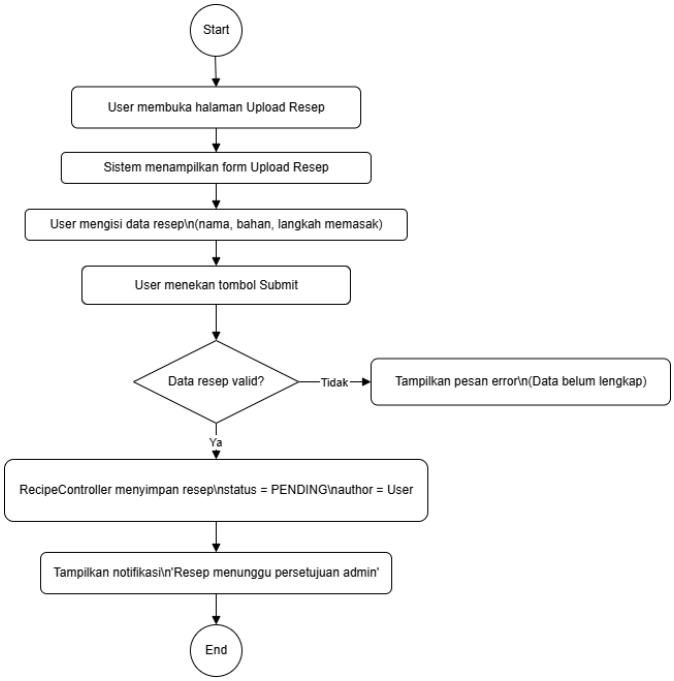
Gambar 2.2.1 Activity Diagram Fitur Login



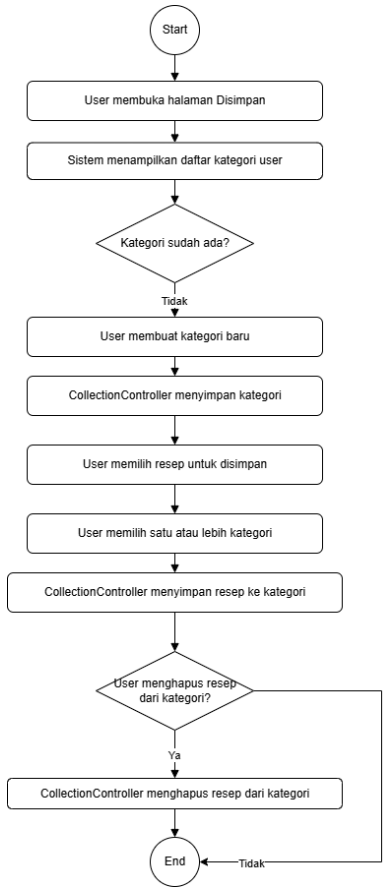
Gambar 2.2.2 Activity Diagram Fitur Pencarian Resep



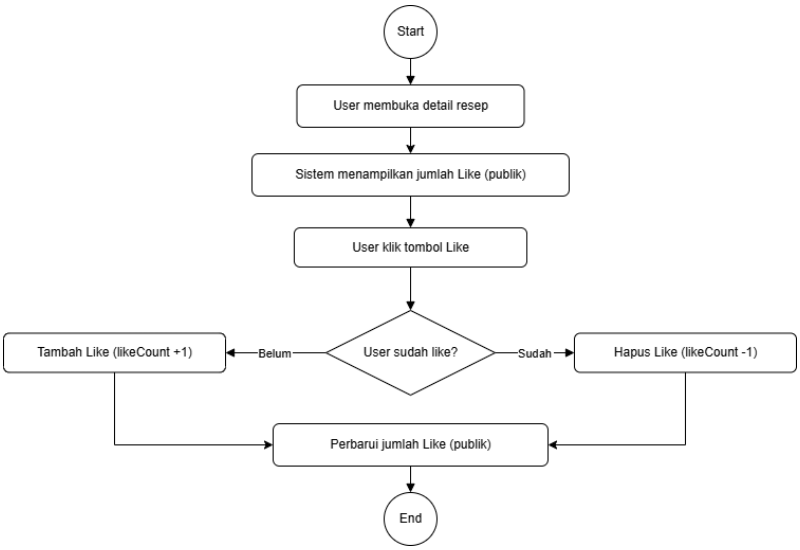
Gambar 2.2.3 Activity Diagram Fitur Persetujuan Resep (Admin)



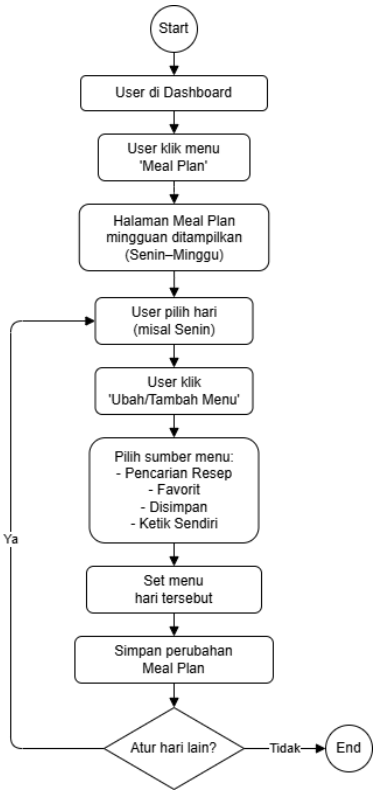
Gambar 2.2.4 Activity Diagram Fitur Upload Resep oleh User



Gambar 2.2.5 Activity Diagram Fitur Disimpan dengan Kategori



Gambar 2.2.6 Activity Diagram Fitur Likes



Gambar 2.2.7 Activity Diagram Meal Plan Mingguan