

# RECIPE SHARE SYSTEM

DISUSUN OLEH KELompok 1:  
CUT DAHLIANA 2208107010027  
ZAHRA ZAFIRA 2208107010040



# RECIPE SHARE SYSTEM

Sistem berbasis konsol yang dirancang untuk mempermudah pengguna dalam berbagi, mencari, dan mengelola resep masakan. Pengguna dapat mendaftar, masuk, menambahkan resep sendiri, mencari resep berdasarkan bahan, diet, atau waktu persiapan, serta melihat detail resep yang dibagikan oleh pengguna lain. Sistem ini juga menyertakan data resep awal dan mendukung eksplorasi resep secara menyeluruh.

# PRE AND POST-CONDITION

Use Case Name	Menambah Resep (AddRecipe)
Aktor	User (Pengguna)
Deskripsi	Pengguna dapat menambahkan resep baru ke dalam sistem.
Precondition	Pengguna telah terdaftar dan login ke dalam sistem.
Postcondition	Resep baru berhasil disimpan dalam database dan tersedia untuk dibagikan.
Flow Utama	<ol style="list-style-type: none"> <li>1. Pengguna memilih opsi "Tambah Resep".</li> <li>2. Sistem menampilkan form untuk mengisi detail resep.</li> <li>3. Pengguna mengisi nama, bahan, langkah-langkah, dan kategori resep.</li> <li>4. Pengguna mengkonfirmasi penambahan resep.</li> <li>5. Sistem menyimpan resep baru dan mengkonfirmasi keberhasilan.</li> </ol>

Use Case Name	Mencari Resep (SearchRecipe)
Aktor	User (Pengguna)
Deskripsi	Pengguna dapat mencari resep berdasarkan kata kunci atau kategori.
Precondition	Terdapat resep yang tersimpan dalam database.
Postcondition	Sistem menampilkan daftar resep yang sesuai dengan kriteria pencarian.
Flow Utama	<ol style="list-style-type: none"> <li>1. Pengguna mengakses fitur pencarian resep.</li> <li>2. Sistem menampilkan form pencarian.</li> <li>3. Pengguna memasukkan kata kunci atau memilih kategori.</li> <li>4. Sistem memproses pencarian dan menampilkan hasil yang relevan.</li> <li>5. Pengguna dapat melihat detail resep yang dicari.</li> </ol>

# PRE AND POST-CONDITION

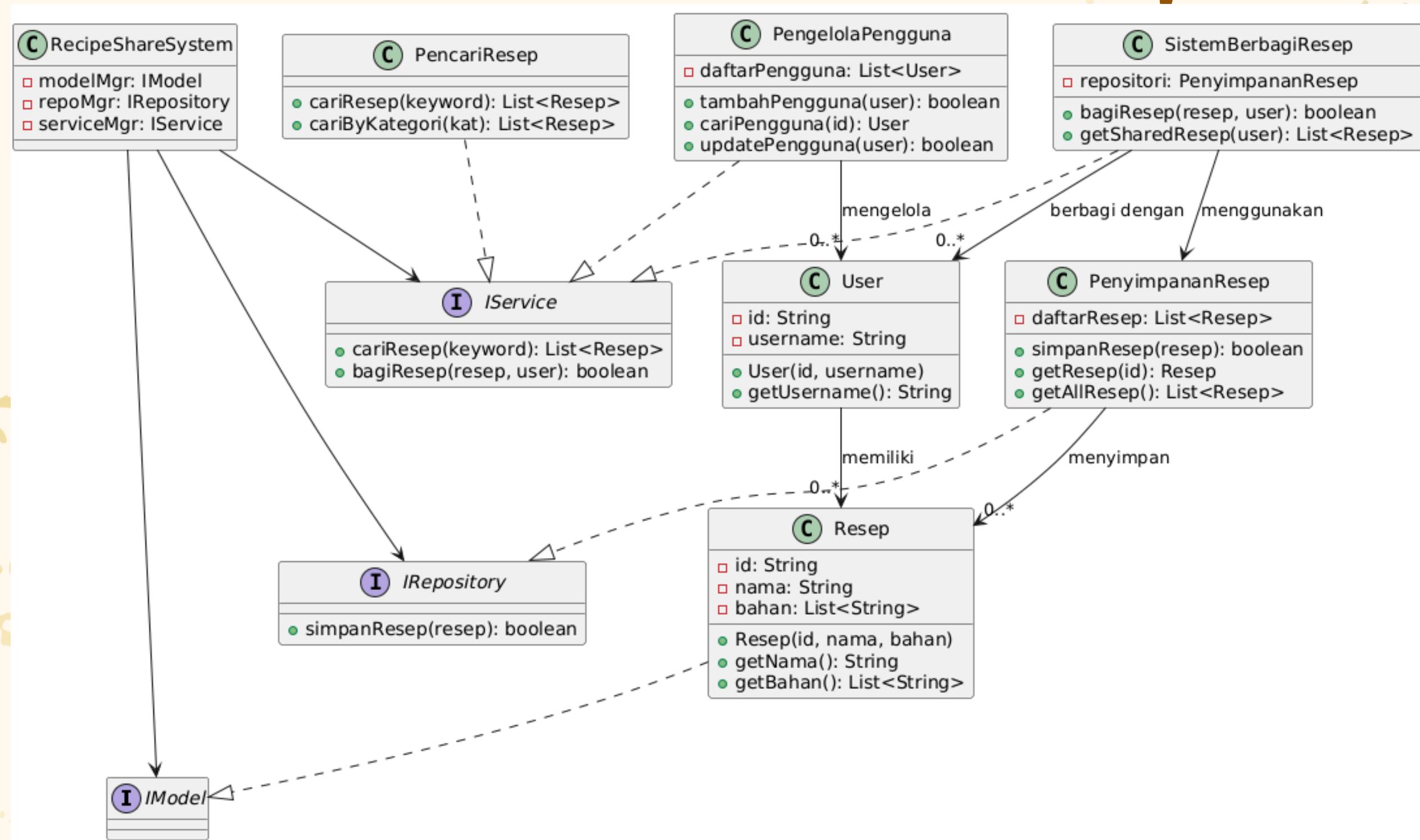
Use Case Name	Berbagi Resep (ShareRecipe)
Aktor	User (Pengguna)
Deskripsi	Pengguna dapat membagikan resep dengan pengguna lain.
Precondition	Pengguna telah memiliki resep yang ingin dibagikan.
Postcondition	Resep berhasil dibagikan dan dapat diakses oleh pengguna yang dituju.
Flow Utama	<ol style="list-style-type: none"> <li>1. Pengguna memilih resep yang ingin dibagikan.</li> <li>2. Sistem menampilkan opsi berbagi.</li> <li>3. Pengguna memilih metode berbagi (pengguna lain, grup, atau publik).</li> <li>4. Pengguna mengkonfirmasi pembagian resep.</li> <li>5. Sistem memproses dan mengkonfirmasi keberhasilan pembagian resep.</li> </ol>

Use Case Name	Menyimpan Resep (SaveRecipe)
Aktor	User (Pengguna)
Deskripsi	Pengguna dapat menyimpan resep yang disukai untuk akses mudah di kemudian hari.
Precondition	Pengguna telah menemukan resep yang ingin disimpan.
Postcondition	Resep berhasil disimpan dalam koleksi pribadi pengguna.
Flow Utama	<ol style="list-style-type: none"> <li>1. Pengguna menemukan resep yang ingin disimpan.</li> <li>2. Pengguna memilih opsi "Simpan Resep".</li> <li>3. Sistem menambahkan resep ke koleksi pribadi pengguna.</li> <li>4. Sistem mengkonfirmasi bahwa resep telah berhasil disimpan.</li> <li>5. Resep tersimpan dapat diakses melalui menu "Resep Tersimpan" pengguna.</li> </ol>

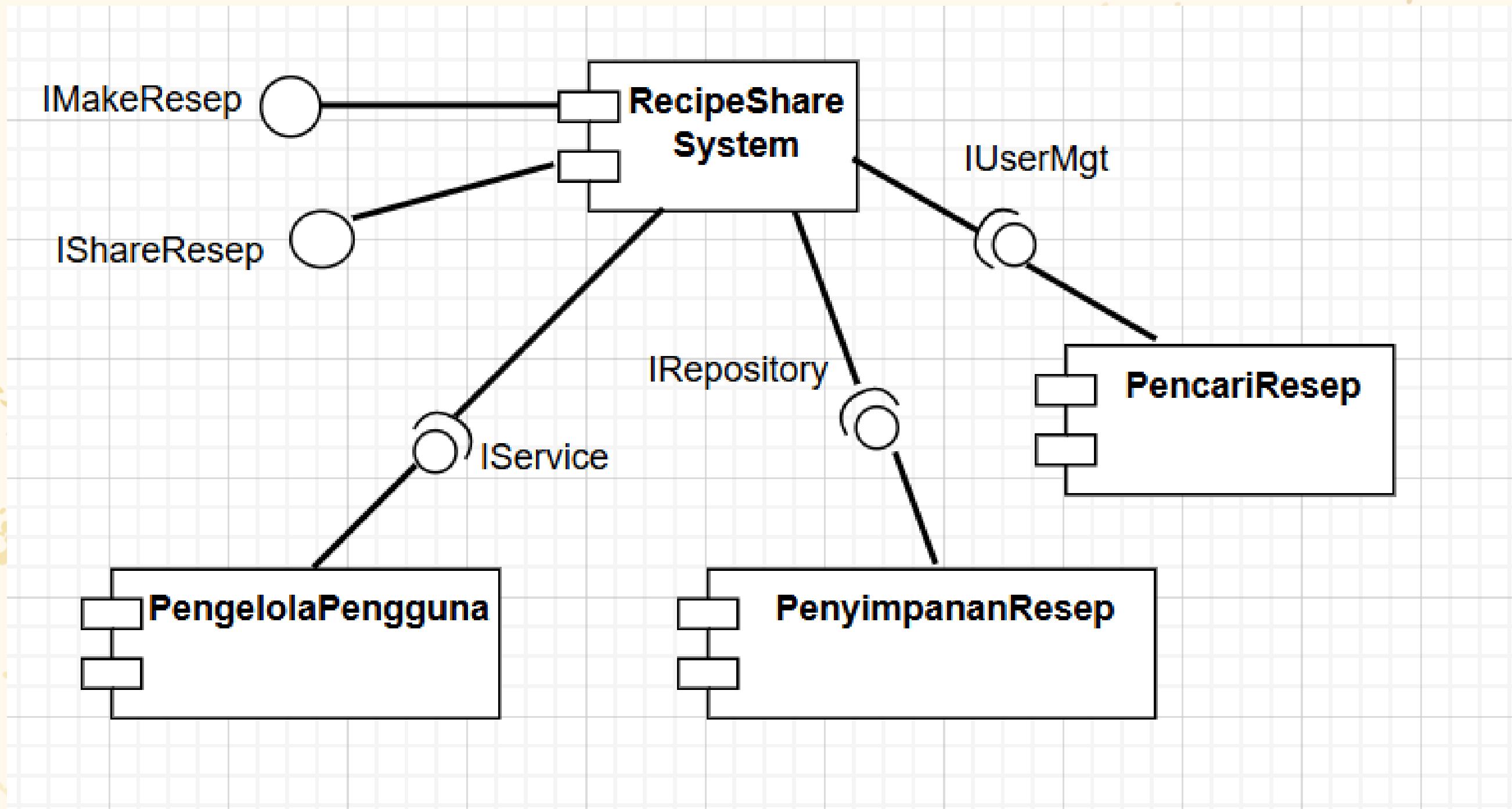
# PRE AND POST-CONDITION

Use Case Name	Mengelola Pengguna (ManageUser)
Aktor	Admin
Deskripsi	Admin dapat mengelola akun pengguna dalam sistem.
Precondition	Admin telah login dengan kredensial yang valid.
Postcondition	Perubahan status atau informasi pengguna berhasil diperbarui.
Flow Utama	<ol style="list-style-type: none"><li>1. Admin mengakses panel pengelolaan pengguna.</li><li>2. Sistem menampilkan daftar pengguna terdaftar.</li><li>3. Admin melakukan tindakan (tambah, edit, nonaktifkan) pada akun pengguna.</li><li>4. Sistem memproses perubahan dan mengkonfirmasi keberhasilan.</li><li>5. Sistem memperbarui status dan informasi pengguna dalam database.</li></ol>

# UML CLASS DIAGRAMS



# UML COMPONENT DIAGRAMS



# ARSITEKTUR SISTEM

"Sistem ini menggunakan arsitektur modular dengan pembagian sebagai berikut:"

Representasi objek bisnis

inti

## MODEL:

Resep.java - Class utama yang merepresentasikan struktur data resep

Penyimpanan data dan operasi CRUD

## REPOSITORY:

- PenyimpananResep.java (interface)
- PenyimpananResepImpl.java (implementasi) - Menangani penyimpanan data resep

Menangani logika bisnis kompleks

## SERVICE:

- PengelolaPengguna.java (interface)
- PengelolaPenggunaImpl.java (implementasi)
- PencariResep.java (interface)
- PencariResepImpl.java (implementasi)

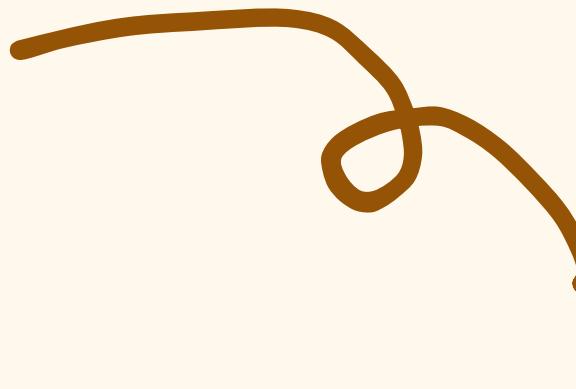
Antarmuka pengguna dan koordinasi alur

## MAIN CLASS:

SistemBerbagiResep.java - Menghubungkan semua komponen dan menampilkan menu



# CODE-STRUKTUR CLASS



Resep.java



```
package model;

import java.util.List;

public class Resep {
    private int id;
    private String judul;
    private String deskripsi;
    private List<String> bahan;
    private List<String> langkah;
    private String kategori;
    private int waktuPersiapan; // dalam menit
    private String preferensiDiet;
    private String penulis;
    private double rating;

    public Resep(int id, String judul, String deskripsi, List<String> bahan,
                List<String> langkah, String kategori, int waktuPersiapan,
                String preferensiDiet, String penulis) {
        this.id = id;
        this.judul = judul;
        this.deskripsi = deskripsi;
        this.bahan = bahan;
        this.langkah = langkah;
        this.kategori = kategori;
        this.waktuPersiapan = waktuPersiapan;
        this.preferensiDiet = preferensiDiet;
        this.penulis = penulis;
        this.rating = 0.0;
    }

    // Getter dan setter
    public int getId() { return id; }
    public String getJudul() { return judul; }
    public String getDeskripsi() { return deskripsi; }
    public List<String> getBahan() { return bahan; }
    public List<String> getLangkah() { return langkah; }
    public String getKategori() { return kategori; }
    public int getWaktuPersiapan() { return waktuPersiapan; }
    public String getPreferensiDiet() { return preferensiDiet; }
    public String getPenulis() { return penulis; }
    public double getRating() { return rating; }

    public void setRating(double rating) { this.rating = rating; }
}
```



# CODE-INTERFACE

## PengelolaPengguna.java

```
package service;

public interface PengelolaPengguna {
    void daftarkanPengguna(String namaPengguna, String kataSandi, String email);
    boolean verifikasiPengguna(String namaPengguna, String kataSandi);
    void perbaruiProfil(String namaPengguna, String dataProfilBaru);
}
```

## PencariResep.java

```
package service;

import java.util.List;
import model.Resep;

public interface PencariResep {
    List<Resep> cariBerdasarkanBahan(List<String> bahan);
    List<Resep> cariBerdasarkanDiet(String preferensiDiet);
    List<Resep> cariBerdasarkanWaktuPersiapan(int maksimalMenit);
}
```



## PenyimpananResep.java (interface)

```
package repository;

import java.util.List;
import model.Resep;

public interface PenyimpananResep {
    void tambahResep(Resep resep);
    Resep dapatkanResepById(int id);
    List<Resep> dapatkanSemuaResep();
    List<Resep> dapatkanResepByKategori(String kategori);
}
```

# CODE IMPLEMENTASI



PenyimpananResepImpl.java -Implementasi konkret

```
package repository;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import model.Resep;

public class PenyimpananResepImpl implements PenyimpananResep {
    private Map<Integer, Resep> daftarResep = new HashMap<>();
    private int idBerikutnya = 1;

    @Override
    public void tambahResep(Resep resep) {
        daftarResep.put(resep.getId(), resep);
        System.out.println("Resep " + resep.getJudul() + " berhasil ditambahkan!");
    }

    @Override
    public Resep dapatakanResepById(int id) {
        return daftarResep.get(id);
    }

    @Override
    public List<Resep> dapatakanSemuaResep() {
        return new ArrayList<>(daftarResep.values());
    }

    @Override
    public List<Resep> dapatakanResepByKategori(String kategori) {
        List<Resep> hasil = new ArrayList<>();
        for (Resep resep : daftarResep.values()) {
            if (resep.getKategori().equalsIgnoreCase(kategori)) {
                hasil.add(resep);
            }
        }
        return hasil;
    }
}
```

PengelolaPenggunaImpl.java

```
package service;

import java.util.HashMap;
import java.util.Map;

public class PengelolaPenggunaImpl implements PengelolaPengguna {
    private Map<String, String> kredensialPengguna = new HashMap<>();
    private Map<String, String> profilPengguna = new HashMap<>();
    private Map<String, String> emailPengguna = new HashMap<>();

    @Override
    public void daftarkanPengguna(String namaPengguna, String kataSandi, String email) {
        if (kredensialPengguna.containsKey(namaPengguna)) {
            System.out.println("Nama pengguna sudah digunakan! Silakan pilih nama lain.");
            return;
        }

        kredensialPengguna.put(namaPengguna, kataSandi);
        profilPengguna.put(namaPengguna, "Email: " + email);
        emailPengguna.put(namaPengguna, email);
        System.out.println("Pengguna " + namaPengguna + " berhasil didaftarkan!");
    }

    @Override
    public boolean verifikasiPengguna(String namaPengguna, String kataSandi) {
        return kredensialPengguna.containsKey(namaPengguna) &&
               kredensialPengguna.get(namaPengguna).equals(kataSandi);
    }

    @Override
    public void perbaruiProfil(String namaPengguna, String dataProfilBaru) {
        if (profilPengguna.containsKey(namaPengguna)) {
            profilPengguna.put(namaPengguna, dataProfilBaru);
            System.out.println("Profil diperbarui untuk " + namaPengguna);
        } else {
            System.out.println("Pengguna tidak ditemukan!");
        }
    }

    public String getEmail(String namaPengguna) {
        return emailPengguna.get(namaPengguna);
    }
}
```

PencariResepImpl.java

```
package service;

import java.util.ArrayList;
import java.util.List;
import model.Resep;
import repository.PenyimpananResep;

public class PencariResepImpl implements PencariResep {
    private PenyimpananResep penyimpananResep;

    public PencariResepImpl(PenyimpananResep penyimpananResep) {
        this.penyimpananResep = penyimpananResep;
    }

    @Override
    public List<Resep> cariBerdasarkanBahan(List<String> bahan) {
        List<Resep> semuaResep = penyimpananResep.dapatakanSemuaResep();
        List<Resep> hasil = new ArrayList<>();

        for (Resep resep : semuaResep) {
            boolean mengandungSatu = false;
            for (String b : bahan) {
                // Periksa jika resep mengandung setidaknya satu dari bahan yang dicari
                if (bahanResep.toLowerCase().contains(b.toLowerCase())) {
                    mengandungSatu = true;
                    break;
                }
            }
            if (mengandungSatu) {
                hasil.add(resep);
            }
        }
        return hasil;
    }

    @Override
    public List<Resep> cariBerdasarkanDiet(String preferensiDiet) {
        List<Resep> semuaResep = penyimpananResep.dapatakanSemuaResep();
        List<Resep> hasil = new ArrayList<>();

        for (Resep resep : semuaResep) {
            if (resep.getPreferensiDiet().equalsIgnoreCase(preferensiDiet)) {
                hasil.add(resep);
            }
        }
        return hasil;
    }

    @Override
    public List<Resep> cariBerdasarkanMakroPersiapan(int maksimalMakro) {
        List<Resep> semuaResep = penyimpananResep.dapatakanSemuaResep();
        List<Resep> hasil = new ArrayList<>();

        for (Resep resep : semuaResep) {
            if (resep.getMakroPersiapan() <= maksimalMakro) {
                hasil.add(resep);
            }
        }
        return hasil;
    }
}
```

# LAYER PRESENTATION

SistemBerbagiResep.java

**THANK YOU  
FOR YOUR  
ATTENTION**

