

# YAZILIM LABORATUVARI

## PROJE 1

Anıl Engin Keretli

Ahmet Burak Karkaç

Kocaeli Üniversitesi

Kocaeli Üniversitesi

Kocaeli/TÜRKİYE

Kocaeli/TÜRKİYE

230201128

220201173

### I. Giriş

Tarif ve Malzeme Yönetim Sistemi, tariflerin ve malzemelerin düzenlenmesini kolaylaştırmak için tasarlanmıştır. Bu proje, Kocaeli Üniversitesi Bilgisayar Mühendisliği Bölümü'nün belirlediği gereksinimler doğrultusunda geliştirilmiştir. Sistem, tarifleri kaydetme, kategorize etme, hazırlık adımlarını tanımlama ve her tarif için gerekli malzemeleri ve miktarları ilişkilendirme imkanı sağlar.

### II. Sistem Gereksinimleri

#### A. Fonksiyonel Gereksinimler

Tarif Yönetimi: Tarif ekleme, güncelleme ve silme işlemleri.  
Malzeme Yönetimi: Malzeme miktarlarını ve maliyetlerini takip etme.

Tarif-Malzeme İlişkisi: Her tarif için gerekli malzeme miktarlarının belirtilmesi.

#### B. Fonksiyonel Olmayan Gereksinimler

Kullanılabilirlik: Sistem kullanıcı dostu olmalıdır.

Performans: Çok sayıda tarif ve malzemeyi verimli bir şekilde yönetebilmelidir.

Genişletilebilirlik: Gelecekte daha fazla tarif ve malzeme eklemek kolay olmalıdır.

### III. Veri Tabanı Tasarımı

Sistem, tarifleri, malzemeleri ve bunların ilişkilerini yönetmek için üç tabloya sahip bir veri tabanı yapısına sahiptir. Tabloların yapısı şu şekildedir:

#### A. Tarif Tablosu

Alan	Tür	Açıklama
TarifID	int	Her tarif için benzersiz bir kimlik numarası.
TarifAdı	varchar	Tarifin adı.
Kategori	varchar	Tarifin kategorisi (örneğin, Ana Yemek, Tatlı).
HazırlamaSuresi	int	Hazırlık süresi (dakika cinsinden).
Talimatlar	text	Tarifin hazırlanma adımları.

#### B. Malzeme Tablosu

Alan	Tür	Açıklama
MalzemeID	int	Her malzeme için benzersiz bir kimlik numarası.
MalzemeAdı	varchar	Malzememin adı.
ToplamMiktar	varchar	Malzememin depodaki toplam miktarı.
MalzemeBirim	varchar	Malzememin birimi (örneğin, kg, litre, gram).
BirimFiyat	float	Malzememin birim maliyet değeri.

#### C. Tarif-Malzeme İlişkisi Tablosu

Alan	Tür	Açıklama
TarifID	Foreign Key	İlgili tarifin kimlik numarası.
MalzemeID	Foreign Key	İlgili malzemenin kimlik numarası.
MalzemeMiktar	float	Tarif için gerekli malzeme miktarı.

#### Uygulama

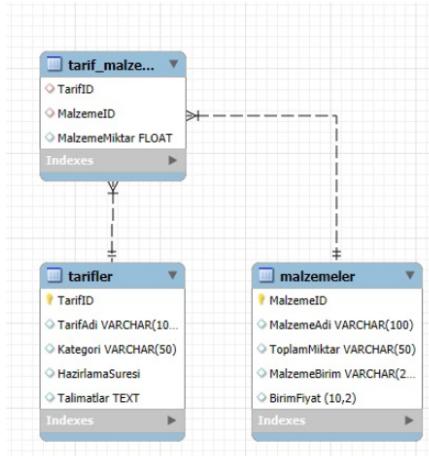
Sistem Java dili ile geliştirilmiş olup, veri tabanı tablolarını temsil eden sınıflardan oluşmaktadır. Temel sınıflar ve görevleri şu şekildedir:

**Tarif Sınıfı:** Tarif bilgilerini saklar ve tarif verilerini yönetme işlevlerini içerir.

**Malzeme Sınıfı:** Malzeme bilgilerini içerir ve stok seviyelerini ve maliyetlerini yönetme işlevlerini içerir.

**TarifMalzeme Sınıfı:** Tarif ile malzemeler arasındaki ilişkiyi yönetir, her tarif için gerekli malzeme miktarlarını takip eder.

Database'in UML diyagramı aşağıdaki gibidir:

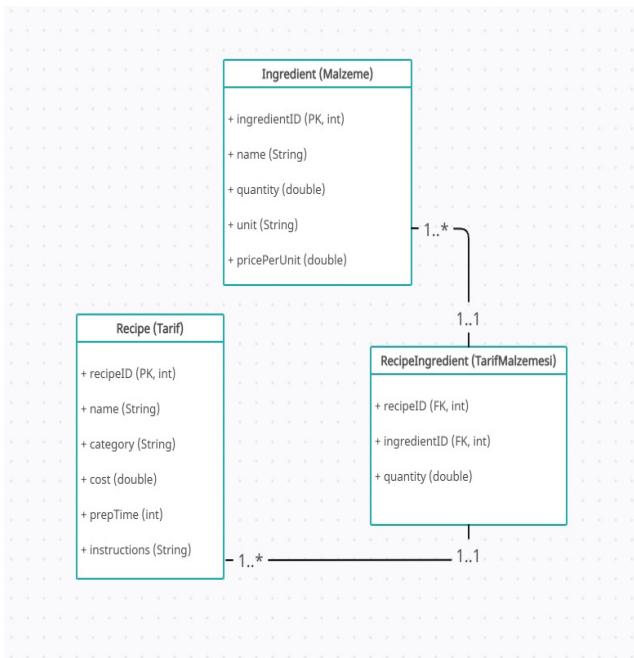


Bu proje, tariflerin ve malzemelerin yönetimini sağlamak için ilişkisel bir veri tabanı yapısını kullanarak sağlam bir yaklaşımı ortaya koymaktadır. Sistem, malzeme stoklarını, tarif hazırlık adımlarını ve gerekli miktarları verimli bir şekilde takip etmektedir. Gelecekte kullanıcı arayüzü eklenmesi ve envanter uyarıları gibi özelliklerin eklenmesi ile sistemin geliştirilmesi mümkündür.

## Tarif Yönetim Sistemi Java Sınıf Tasarımı

Bu rapor, tarif ve malzemeler arasındaki ilişkiyi modelleyen bir Java uygulaması için sınıf yapısını açıklamaktadır. Tarif Yönetim Sistemi'nde Ingredient, Recipe, ve RecipeIngredient sınıfları, tariflerin malzemelerle ilişkili bir şekilde saklanması sağlamak amacıyla tasarlanmıştır. Uygulamada, tarifler ve malzemeler arasındaki çoktan-çoka ilişkiyi yönetmek için bir köprü sınıfı olan RecipeIngredient kullanılmıştır.

### Java sınıfı UML diyagramı:



### Sınıflar ve Özellikler:

#### 1. Ingredient (Malzeme) Sınıfı:

Amaç: Malzemeleri temsil eder ve her malzeme için kimlik, isim, miktar, birim ve birim başına fiyat gibi bilgileri içerir.

#### Özellikler:

- int ingredientID: Malzeme kimliği (PK).
- String name: Malzemenin adı.
- double quantity: Malzeme miktarı.
- String unit: Malzemenin birimi.
- double pricePerUnit: Malzemenin birim fiyatı.

Yöntemler: Getter ve setter yöntemleriyle her özelliğin erişim ve değişimi sağlanmaktadır. Ayrıca, tam özellikli ve belirli özelliklerini içeren yapıcı (constructor) yöntemler sunulmuştur.

#### 2. Recipe (Tarif) Sınıfı:

Amaç: Tarifleri temsil eder ve her tarif için kimlik, isim, kategori, maliyet, hazırlama süresi ve talimatları içeren bilgiler içerir.

### Özellikler:

- int recipeID: Tarif kimliği (PK).
- String name: Tarifin adı.
- String category: Tarifin kategorisi.
- double cost: Tarifin toplam maliyeti.
- int prepTime: Tarifin hazırlama süresi.
- String instructions: Tarifin adım adım talimatları.

Yöntemler: Tüm özellikler için getter ve setter yöntemleri, tarif bilgilerini güncellemeye ve almaya olanak tanır.

#### 3. RecipeIngredient (TarifMalzemesi) Sınıfı:

Amaç: Tarif ve malzemeler arasındaki çoktan-çoka ilişkiyi yönetir. Bu sınıf, her tarifin içinde kullanılan belirli miktarda malzemeyi tanımlamak için Recipe ve Ingredient sınıflarıyla ilişkilidir.

#### Özellikler:

- int recipieID: Tarif kimliği (Recipe tablosuna FK).
- int ingredientID: Malzeme kimliği (Ingredient tablosuna FK).
- double quantity: Tarif için gerekli olan malzeme miktarı.

Yöntemler: Her bir tarife ait malzemeyi ve miktarını güncellemeye veya almaya yönelik getter ve setter yöntemleri bulunmaktadır.

### İlişkiler ve Yapının Sağladığı İşlevsellik:

• Recipe ve RecipeIngredient Arasındaki İlişki: Recipe sınıfı ile RecipeIngredient sınıfı arasında 1'e-çok ilişki bulunmaktadır. Bir tarif birden fazla malzeme içerebilir. RecipeIngredient, recipieID aracılığıyla Recipe sınıfına referans vermektedir.

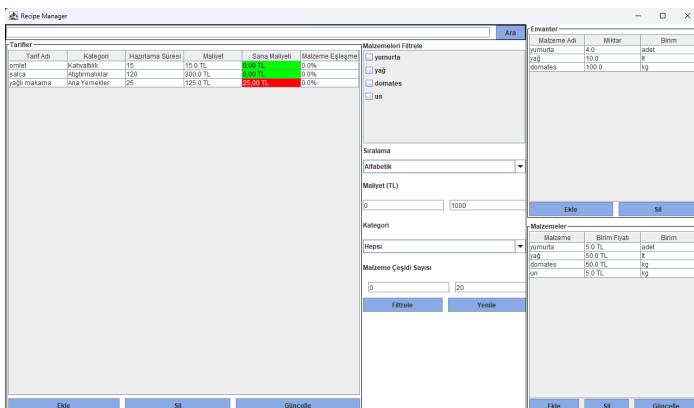
• Ingredient ve RecipeIngredient Arasındaki İlişki: Ingredient sınıfı ile RecipeIngredient sınıfı arasında 1'e-çok ilişki vardır. Bir malzeme birden fazla tarife kullanılabilir. RecipeIngredient, ingredientID aracılığıyla Ingredient sınıfına referans vermektedir.

Bu yapı, RecipeIngredient sınıfının bir köprü görevi görmesini sağlayarak Recipe ve Ingredient sınıfları arasında çoktan-çoka bir ilişki kurulmasını mümkün kılmaktadır. Örneğin, bir tarifte birden fazla malzeme bulunabilir ve aynı malzeme farklı tariflerde kullanılabilir.

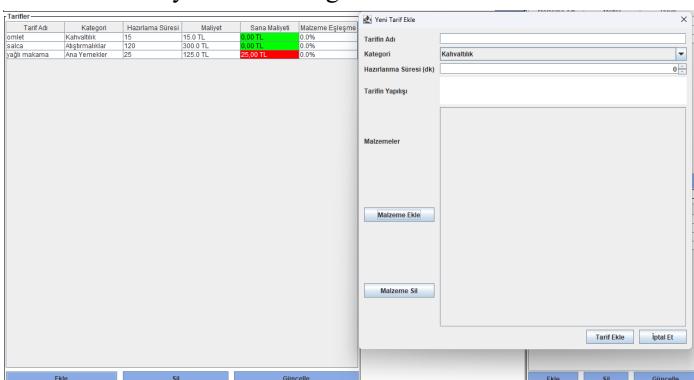
**Sonuç:** Tarif Yönetim Sistemi'nde Ingredient, Recipe, ve RecipeIngredient sınıfları, veri modelini temsil etmek için başarılı bir şekilde tasarlanmıştır. RecipeIngredient sınıfı, tarif ve malzeme ilişkisini yönetmek için işlevsel bir köprü oluşturarak, tariflerde kullanılan malzemelerin miktarları ve detayları hakkında esnek bir yapı sunmaktadır. Bu sınıf yapısı, kullanıcıların tarif ve malzemeleri etkili bir şekilde yönetmesine imkan tanır ve tarif oluşturma, düzenleme, silme gibi işlevlerin kolayca gerçekleştirilemesini sağlar.

## ARAYÜZ TASARIMI

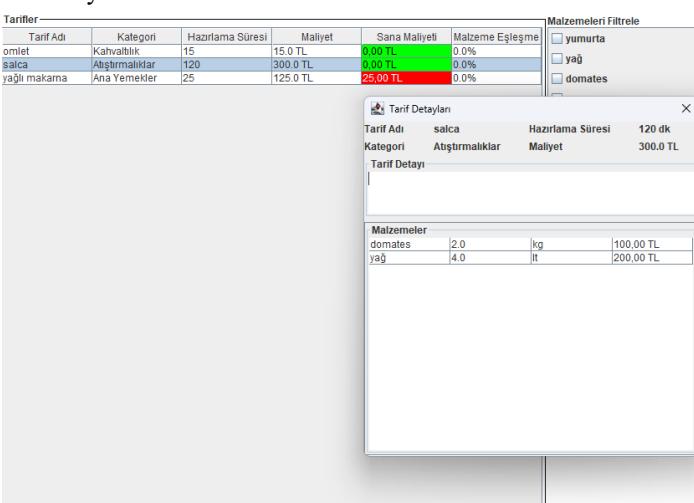
Menü: Ana ekranda kullanıcıya kolay erişim sağlayan bir menü bulunmaktadır. Menüde yer alan seçenekler:



- Tarif Ekleme: Yeni bir tarif eklemek isteyen kullanıcılar bu seçeneği kullanarak, tarifin adını, malzemelerini, hazırlama süresini ve maliyetini sisteme girebilirler.



- Tarif Güncelleme: Kullanıcı, mevcut tariflerden birinin detayını güncellemek istediğiinde, bu seçeneği kullanarak tarif bilgilerini düzenleyebilir.



- Tarif Silme: Kullanıcı, listede yer alan tariflerden herhangi birini silmek için bu seçeneği kullanabilir.

### Arama ve Filtreleme Alanı:

Arama ve filtreleme alanı, ekranın üst kısmında yer almaktır olup, kullanıcıların tarifleri hızlı bir şekilde bulabilmesini sağlar. Kullanıcılar bu alanda:

- Tarif ismine veya hazırlama süresine göre arama yapabilir.
- Belirli maliyet veya kategoriye göre filtreleme yaparak tarif listesinde daha dar bir arama sonucu elde edebilir.

Arama ve filtreleme sonuçlarına göre güncellenen liste, kullanıcıların belirledikleri kriterlere uygun tarifleri hızla görüntülemelerine imkan tanır. Listeleme sonucunda ekrandaki tarifler yalnızca kullanıcı kriterlerine uygun olanları gösterecek şekilde güncellenmektedir.

<b>Malzemeleri Filtrele</b>
<input type="checkbox"/> yumurta
<input type="checkbox"/> yağı
<input type="checkbox"/> domates
<input type="checkbox"/> un
<b>Sıralama</b>
Hazırlama Süresi
<b>Maliyet (TL)</b>
0 <input type="text"/> 1000
<b>Kategori</b>
Hepsi
<b>Malzeme Çeşidi Sayısı</b>
0 <input type="text"/> 20
<b>Filtrele</b> <b>Yenile</b>

**Tarif Yönetim Sistemi GUI tasarımi**, kullanıcıların tarifler üzerinde kapsamlı bir kontrol sağlamaına yardımcı olacak şekilde düzenlenmiştir. Ana ekran, tariflerin hızla görüntülenmesine ve detaylı bilgilere erişime odaklanmıştır. Menü, tarif yönetim işlemlerini kolaylaştırırken, arama ve filtreleme özellikleri, kullanıcının ihtiyaç duyduğu tarifleri kısa sürede bulmasına imkan tanır.

Bu uygulama tasarımı, kullanıcı dostu bir yapıya sahip olup, tariflerin yönetilmesini ve erişimini oldukça pratik hale getirmektedir.

```
public boolean isDuplicateRecipe(String recipeName) { new *  
    try {  
        if (dbManager.getConnection() == null) {  
            dbManager.connect();  
        }  
  
        String query = "SELECT COUNT(*) FROM Tarifler WHERE TarifAdı = ?";  
        PreparedStatement pstmt = dbManager.getConnection().prepareStatement(query);  
        pstmt.setString( parameterIndex: 1, recipeName);  
        ResultSet rs = pstmt.executeQuery();  
  
        if (rs.next()) {  
            return rs.getInt( columnIndex: 1 ) > 0;  
        }  
    } catch (SQLException e) {  
        System.out.println("duplicate check failed: " + e.getMessage());  
    }  
    return false;  
}
```

### Duplicate control

recipeName adlı tarifin veritabanında zaten olup olmadığını kontrol eden bir metoddur. İlk olarak, veritabanı bağlantısı yoksa bağlantı kurulur. Ardından, Tarifler tablosunda TarifAdı sütununda tarif adının sayısı sorgulanır. Sorgu sonucu > 0 ise, tarif mevcut demektir ve true döner; aksi halde false döner.

## Tarif Ekleme Testi

Tarif ekleme işlemi sırasında sistemin, kullanıcının verdiği bilgileri doğru bir şekilde veritabanına kaydettiği doğrulandı.

```
public void addRecipe(String recipeName, String category, int prepTime, String instructions, ArrayList<Ingredient> ingredientsList) {
    try {
        if (isDuplicateRecipe(recipeName)) {
            System.out.println("[HATA] Tarif zaten mevcut.");
            return;
        }
        // Şimdi bir transaction başlatalım
        dbManager.getConnection().setAutoCommit(false);

        // Tarif veritabanına ekleniyor
        String insertRecipeSQL = "INSERT INTO Tarifler (TarifAdi, Kategori, HazirlamaSuresi, Talimatlar) VALUES (?, ?, ?)";
        PreparedStatement recipeStmt = dbManager.getConnection().prepareStatement(insertRecipeSQL, Statement.RETURN_GENERATED_KEYS);
        recipeStmt.setString(1, recipeName);
        recipeStmt.setString(2, category);
        recipeStmt.setInt(3, prepTime);
        recipeStmt.setString(4, instructions);
        recipeStmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

## KATKILAR

Projede Anıl Engin Keretli, tarif ekleme çıkarma ve temel fonksiyonlar üzerine çalıştı; raporlamayı da üstlendi. Ahmet Burak Karkaç ise database yapısı ve gui tasarımını üzerine çalıştı.

## KAYNAKÇA

<https://docs.oracle.com/en/java/javase/22/>

<https://bulutistan.com/blog/iliskisel-veritabani-nedir-nasil-calisir-ozellikleri-ornekleri-modelleri/>

<https://proceedingsoftheieee.ieee.org/journal/>

[https://gad.com/gad\\_source=1&gclid=CjwKCAjwyfe4BhAWEiwAkIL8sE5MgtGRPjzr6J2Gv26\\_ZMIxfEsGVAyYyZ-zF67X-0\\_SFNIIMN9pzhhoCfQIQAvD\\_BwE](https://gad.com/gad_source=1&gclid=CjwKCAjwyfe4BhAWEiwAkIL8sE5MgtGRPjzr6J2Gv26_ZMIxfEsGVAyYyZ-zF67X-0_SFNIIMN9pzhhoCfQIQAvD_BwE)

## YALANCI KOD

-Fonksiyon tarifEkle(tarifAdi, kategori, hazırlamaSuresi, talimatlar)

-Yeni bir Tarif nesnesi oluştur

-Tarif nesnesinin özelliklerini tarifAdi, kategori, hazırlama Suresi ve talimatlar ile ayarla

-Veritabanına Tarif nesnesini ekle

-Son

-Fonksiyon tarifGuncelle(tarifID, yeniAdi, yeniKategori, yeniSure, yeniTalimatlar)

-Veritabanında tarifID ile eşleşen tarif kaydını bul

-Eğer tarif bulunamazsa

-Hata mesajı döndür

-Değilse

-Tarifin adı, kategori, hazırlamaSuresi ve talimatlarını yeniAdi, yeniKategori, yeniSure ve yeniTalimatlar ile güncelle

-Veritabanında tarif kaydını güncelle

-Son