

# Tarif Rehberi Uygulaması

Alperen Karacan 210201096

—Yazılım laboratuvarı dersinin ilk projesi olarak istenen yemek tarifi uygulaması, veritabanı yönetimi, dinamik arama ve filtreleme gibi işlevlerin bir masaüstü uygulamasında ne kadar etkili yapılabileceğinin ölçülmesi için verilmiştir. Biz de bu projeye plan yaparak başladık ve proje sürecimizi netleştirdik. Öncelikle bir arayüz oluşturarak işlevlerin konumlandırılmasına ve buna göre de backend'in şekillenmesine karar verdik. Devamında veritabanı bileşenlerini kendi işlevlerimiz doğrultusunda oluşturduk ve veritabanı kısmını entegre edeceğimiz sonraki aşamaya geçtik. Ardından, fonksiyonları sınıflandırarak teknik altyapımızı oluşturduk ve veritabanıyla iletişim kısmına geçtik. Bu entegrasyon da tamamlandıktan sonra proje neredeyse son haline ulaşmış olduğundan geniş çaplı testler yaparak isterlerimizin daha net bir şekilde gerçekleştiğinden ve olası hatalarımızdan emin olduk. Bu aşamayla birlikte proje, istenilen isterler ve gereklilikler çerçevesinde tamamlanmış ve layığına yerine getirecek bir düzeye ulaşmıştır.

## I. ÖZET

Proje basamaklarımızın belirlenmesi bizim için oldukça rahatlatıcı oldu; bu da proje sürecinin devamlılığı ve deadline kavramının anlamının bizim için daha iyi oturduğu anlamına geliyor. Öncelikle, ikimizin de projeye farklı açılardan yönelerek daha dinamik bir stile geçtiğimizi ve bunun bize çok yarar sağladığını söylemem gerek. Proje sürecine gelecek olursak, arayüz ve veritabanı problemlerine yönelerek başladık. Önceki projelerimizdeki tecrübelerimize göre veritabanı yapısı ve işlevinin oluşturulması daha rahat bir gereklilik olduğundan, öncelikle bu adımı tamamlamada hemfikir olduk. Arayüz gereksinimleri de buna paralel olarak rahat oluşturulabilir bir yapıdaydı ve işlev planımızı daha rahat bir şekilde kurmamızı sağladı. Bu aşamalarla istediğimiz çerçeveyi oluşturduk ve ihtiyacımız doğrultusunda kullanmamızı sağlayacak backend'e yöneldik. Burada Java'nın daha kullanışlı olması bizim için yeterli olmadı; arayüzü önceden tamamlamanın sağladığı rahatlık ve belirleyicilik sayesinde backend üzerinde daha net kararlar alabildik. Böylece program istediğimiz şekli aldı ve

belli başlı test ve denetlemelerin ardından projeyi tamamladık.

## II. GİRİŞ

Projenin aşamalarından bahsedeceğim bu kısımda yolculuğun içinde bulunacak ve projenin gelişim ve değişimine şahit olarak tamamlaması konusunda tam olarak fikir sahibi olacaksınız. Öncelikle yöntemi (yani bu projenin tam olarak nasıl bir üretim aşaması geçirdiğini ), sonucu, deneysel sonuçları ve benzer projeler için gerekebilecek ve bize bu süreçte yardımcı olan kaynakları göreceksiniz. Arayüz ve veri tabanının şekillendirilmesiyle başlayacağımız bu yolculukta bu iki kısmın back end e oop ile nasıl entegre edilerek end to end bir masaüstü uygulaması yapıldığını göstereceğiz.

## III. YÖNTEM

Öncelikle veri tabanının tasarlanması üzerine yöneldik ve bunu aradan çıkarak işlevlerin daha düzenli bir şekilde oluşturulabilmesinin sağlanacağını göstermiş olduk. Tablolarımızı manuel (arayüz aracılığıyla) oluşturarak kolay bir şekilde bu kısmı yaptık .

Tablolar: Tarifler Tablosu: Tariflerin adı, kategorisi, hazırlama süresi ve yapılış talimatlarını içerir. Malzemeler Tablosu: Malzemelerin adı, depodaki miktarı ve birim fiyat bilgilerini içerir. Tarif-Malzeme İlişkisi Tablosu: Tarifler ve malzemeler arasında many-to-many ilişkisini temsil eder. Normalizasyon: Aynı malzemelerin birden fazla tarifte kullanılabileceği göz önüne alınarak, veritabanı yapısında normalizasyon sağlanmıştır. Bu sayede, veri tekrarının önüne geçilmiş ve veritabanı daha verimli hale getirilmiştir. Fig 1 de bu tablolardan biri gösterilmektedir.

	malzemeid (PK) Integer	malzemeadi character varying (30)	toplammiktar character varying (20)	malzemebirim character varying (20)	birimfiyat numeric (10,2)
1	1	Un	2 kg	kg	15.50
2	2	Şeker	1 kg	kg	10.00
3	3	Zeytinyağı	500 ml	ml	20.00
4	4	Tuz	500 gr	gr	3.00
5	5	Yoğurt	1 kg	kg	7.50
6	6	Tavuk Göğsü	1 kg	kg	25.00
7	7	Patates	5 kg	kg	12.00
8	8	Soğan	50 kg	kg	2.50
9	9	Et	30 kg	kg	20.00
10	10	Marul	100 adet	adet	2.00
11	11	Pirinç	200 kg	kg	6.00
12	12	Makarna	150 kg	kg	4.00
13	13	Ayçiçek yağı	40 lt	lt	30.00
14	14	Yoğurt	80 kg	kg	5.00
15	15	Salatalık	60 kg	kg	3.00
16	16	Mayonez	25 kg	kg	10.00
17	17	Portakal	300 kg	kg	2.50
18	18	Havuç	120 kg	kg	3.00
19	19	Mantar	70 kg	kg	15.00
20	20	Kıyma	90 kg	kg	40.00
21	22	Su	10 L	kg	1.00
22	23	Limon	10 kg	kg	5.00
23	24	Sarımsak	500 ml	ml	20.00
24	25	Tuz	5 kg	kg	30.00
25	26	Elma	10 kg	kg	15.00

Fig. 1. Veri tabanındaki içerik temsil tablolarından Malzeme tablosu

Fig 1 de yer alan tabloda veri eklemeleri her ne kadar yapılmış olsa da bu veri eklemeleri için ekleme kodunu sql de veri tabanı için yazdık. C içinden de sql komutlarıyla işlemler yapsak da bazı temel işlemleri sql query ile veri tabanı uygulamasında gerçekleştirdik. Bunun örneği de Fig 2 de görülebilir.

Query	Query History
1	INSERT INTO malzemeler (malzemeid, malzemeadi, malzemebirim, birimfiyat) VALUES
2	(1, 'Un', 'kg', 5.0),
3	(2, 'Şeker', 'kg', 7.0),
4	(3, 'Yumurta', 'adet', 1.0),
5	(4, 'Süt', 'lt', 4.0),
6	(5, 'Tuz', 'kg', 2.0),
7	(6, 'Tereyağı', 'kg', 50.0),
8	(7, 'Domates', 'kg', 3.0),
9	(8, 'Soğan', 'kg', 2.5),
10	(9, 'Tavuk Göğsü', 'kg', 20.0),
11	(10, 'Marul', 'adet', 2.0),
12	(11, 'Pirinç', 'kg', 6.0),
13	(12, 'Makarna', 'kg', 4.0),
14	(13, 'Zeytinyağı', 'lt', 30.0),
15	(14, 'Yoğurt', 'kg', 5.0),
16	(15, 'Salatalık', 'kg', 3.0),
17	(16, 'Mayonez', 'kg', 10.0),
18	(17, 'Patates', 'kg', 2.5),
19	(18, 'Havuç', 'kg', 3.0),
20	(19, 'Mantar', 'kg', 15.0),
21	(20, 'Kıyma', 'kg', 40.0);
22	

Fig. 2. Malzemeler tablosu için kullanılan insertion sql query'si

Veri tabanının yapılandırması böyle nihayete erdikten sonra da önyüz üzerine çalışmaya başladık. C dilini seçmemizin sebebi de çok pratik önyüz tasarım imkanları sunmasıydı. Bundan sonuna kadar yararlanarak önyüzü oluşturmaya başladık. Oluştururken de tamamen ister olarak belirtilen

özellikler yönünde geliştirmeler yapmaya dikkat etik ve direktiflere tam olarak uyduk.

Kullanıcı arayüzü, tüm tariflerin ve işlevlerin kullanıcıya kolaylıkla erişebileceği şekilde tasarlanmıştır.

Ana Ekran: Tüm tariflerin listelendiği ve arama, filtreleme seçeneklerinin bulunduğu bir alan oluşturuldu.

Menü Seçenekleri: Tarif ekleme, güncelleme, silme işlemleri için menüler eklendi.

Arama ve Filtreleme Alanı: Tarif ve malzemeye göre arama yapılabilecek bir alan tasarlandı. Önyüz kodlarından bir parça

```

<StackPanel Orientation="Horizontal" VerticalAlignment="Top" Grid.Row="0" Margin="10,5,10,5">
    <TextBox x:Name="SearchBox" Width="200" Height="30" VerticalAlignment="Center">
        Text="Tarif Arama" Foreground="Gray" GotFocus="SearchBox_GotFocus" LostFocus="SearchBox_LostFocus"/>
    <ComboBox x:Name="FilterCategory" Width="150" Height="30" Margin="10,0,0,0" SelectionChanged="OnFilterChanged" VerticalAlignment="Center">
    <TextBox x:Name="FilterTime" Width="100" Height="30" Margin="10,0,0,0">
        TextChanged="OnFilterChanged" VerticalAlignment="Center"
        Text="Max Süre" Foreground="Gray"
        GotFocus="FilterTime_GotFocus" LostFocus="FilterTime_LostFocus"/>
    <TextBox x:Name="FilterCost" Width="100" Height="30" Margin="10,0,0,0">
        TextChanged="OnFilterChanged" VerticalAlignment="Center"
        Text="Max Maliyet" Foreground="Gray"
        GotFocus="FilterCost_GotFocus" LostFocus="FilterCost_LostFocus"/>
    <ComboBox x:Name="SortOption" Width="150" Height="30" Margin="10,0,0,0" SelectionChanged="OnFilterChanged" VerticalAlignment="Center">
        <ComboBoxItem Content="Varsayılan" Tag="Varsayılan" IsSelected="True"/>
        <ComboBoxItem Content="Hazırlama Süresi (Artan)" Tag="SureArtan"/>
        <ComboBoxItem Content="Hazırlama Süresi (Azalan)" Tag="SureAzalan"/>
        <ComboBoxItem Content="Maliyet (Artan)" Tag="MaliyetArtan"/>
        <ComboBoxItem Content="Maliyet (Azalan)" Tag="MaliyetAzalan"/>
    </ComboBox>
    <Button Content="Ara" Width="100" Height="30" Click="OnSearchClick"/>
</StackPanel>

<!-- Orta kısım: Tarif Listesi -->
<DataGrid x:Name="RecipeList" Margin="0,5,0,5" AutoGenerateColumns="False" SelectionMode="Single" IsReadOnly="True">
    <ScrollViewer.VerticalScrollBarVisibility="Auto" Grid.Row="1" Height="200">
        <DataGrid.Columns>
            <DataGridTextColumn Header="Tarif Adı" Binding="{Binding TarifAdi}" Width="*"/>
            <DataGridTextColumn Header="Kategori" Binding="{Binding KategoriAdi}" Width="*"/>
            <DataGridTextColumn Header="Hazırlama Süresi (dakika)" Binding="{Binding HazirlamaSuresi}" Width="*"/>
            <DataGridTextColumn Header="Maliyet" Binding="{Binding ToplamMaliyet}" Width="*"/>
        </DataGrid.Columns>
    </DataGrid>
</DataGrid>

<!-- Alt kısım: Malzeme Seçimi ve Tarif Önerileri -->
<StackPanel Orientation="Vertical" Grid.Row="2" Margin="10,0,10,5">
    <TextBlock Text="Malzemeleriniz" FontWeight="Bold" Margin="0,10,0,0" />
    <ScrollViewer Height="100" HorizontalScrollBarVisibility="Auto" VerticalScrollBarVisibility="Auto">
        <ItemsControl x:Name="IngredientList"/>
    </ScrollViewer>
    <Button Content="Tarif Öner" Width="150" Height="30" Margin="0,10,0,0" Click="OnSuggestRecipeClick"/>
</StackPanel>

```

Fig. 3. Önyüz tasarımının kodlarının bir kısmı

Projenin en önemli kısmı olan fonksiyonların yazımına ise önyüz tasarımını bitirdikten sonra geçtik.Bu noktada, asıl işlevselliğin, yani arayüzün ardındaki karmaşık yapıyı kurma zamanı gelmişti. Her buton, her menü ve her liste – birer boş kabuk olmaktan çıkıp arka planda kodlarla işleyen canlı bir yapıya dönüşmeyi bekliyordu. Tüm bu tasarım, kullanıcıyı rahatça yönlendirecek bir

yüzey sunduğundan, şimdi onu altyapıyla destekleyerek her dokunuşun arka planda doğru işlevleri çalıştırmasını sağlamaya koyulduk.

Fonksiyonel özelliklerin kodlanmasına geçişimizle birlikte, kullanıcı deneyiminin derinliklerine inen bir süreç başladı. Arayüzde tek bir tıklama ile başlatılan "Tarif Ekle" ya da "Tarif Önerisi" gibi işlevlerin, kodların ardında adım adım nasıl bir döngüyle işlediğini görmek projeye yeni bir boyut kattı. Arayüzde görünen bu sade ve kullanışlı yüzey, artık her bir dokunuşla veri tabanına bağlanıyor, dinamik arama ve filtreleme özelliklerini çalıştırıyor ve kullanıcıya kişiselleştirilmiş sonuçlar sunuyordu. Önyüzü tamamladıktan sonra fonksiyonel detaylara geçmek, projeyi gözle görülür bir tasarımdan yaşam bulan bir yazılım haline getirdi; her satır kodla birlikte, projemiz tamamlanmaya bir adım daha yaklaşıyordu.

```
// Tarif güncelleme
using NpgsqlConnection con = new NpgsqlConnection(connectionString);
{
    con.Open();

    NpgsqlCommand guncelleme = new NpgsqlCommand("UPDATE tarifler SET tarifadi = @tarifadi, kategorisi = @kategorisi, malzemesi = @malzemesi, talimatlar = @talimatlar WHERE tarifid = @tarifid", con);
    guncelleme.Parameters.AddWithValue("tarifadi", tarifadi);
    guncelleme.Parameters.AddWithValue("kategorisi", ((ComboBoxItem)MalzemeListe.SelectedItem).Tag);
    guncelleme.Parameters.AddWithValue("malzemesi", malzemesi);
    guncelleme.Parameters.AddWithValue("talimatlar", talimatlar);
    guncelleme.Parameters.AddWithValue("tarifid", tarifid);
    guncelleme.ExecuteNonQuery();

    NpgsqlCommand silme = new NpgsqlCommand("DELETE FROM tarifler WHERE tarifid = @tarifid", con);
    silme.Parameters.AddWithValue("tarifid", tarifid);
    silme.ExecuteNonQuery();

    // Yeni malzemeleri ekleme
    foreach (var malzeme in malzemeListe.Items)
    {
        NpgsqlCommand ekleme = new NpgsqlCommand("INSERT INTO tarifler (tarifid, malzemesi, malzemeadi) VALUES (@tarifid, @malzemesi, @malzemeadi)", con);
        ekleme.Parameters.AddWithValue("tarifid", tarifid);
        ekleme.Parameters.AddWithValue("malzemesi", malzemesi);
        ekleme.Parameters.AddWithValue("malzemeadi", malzemeadi);
        ekleme.ExecuteNonQuery();
    }

    MessageBox.Show("Tarif başarıyla güncellendi.");
    this.DialogResult = true;
    this.Close();
}
```

Fig. 4. Tarif güncelleme fonksiyonunun back end kodu

Diğer bir kilit özellik olan dinamik arama bir sonraki aşamamızdı. Dinamik arama, kullanıcı deneyimini daha akıcı ve verimli hale getiren temel bir özellik olarak, tarif uygulamamızda öne çıkıyor. Kullanıcı, aradığı tarifi ya da malzemeyi birkaç kelimeyle bile ifade etse, arama motoru, veritabanında hızlı bir tarama yaparak en alakalı sonuçları anında listeye getiriyor. Bu özellik, kullanıcıların yalnızca tarif isimlerine göre değil, tariflerde kullanılan malzemelere göre de arama yapabilmesine olanak tanıyor. Böylece, eldeki malzemelere göre yapılabilecek tarifleri bulmak isteyen kullanıcılar

için çok yönlü bir arama deneyimi sunuluyor. Örneğin, kullanıcı "yumurta" malzemesiyle yapılan tüm tarifleri görmek istiyorsa, bu malzemeyi arama alanına yazması yeterli; sistem, ilgili tüm tarifleri sırayla sunuyor.

Aramanın dinamik oluşu, kullanıcıya gerçek zamanlı bir geri bildirim sağlıyor ve veritabanındaki tüm tariflerde anında arama yaparak sonuçları hızlıca ekrana yansıtıyor. Bu esneklik, kullanıcıların filtreleme ve arama işlevlerini kolayca birleştirmesine de olanak tanıyor; örneğin, belirli bir sürede hazırlanabilecek "yumurtalı" tarifleri aramak gibi. Dinamik aramanın bu hızlı ve çok yönlü yapısı sayesinde, kullanıcılar tek bir arama çubuğunda tüm tarifleri kolayca keşfedebiliyor ve tarif seçim süreçleri çok daha etkili hale geliyor.

#### IV. SONUÇ

Bu proje süreci, yazılım geliştirme sürecindeki adımların her birinin nasıl önemli olduğunu görmemizi sağladı. Özellikle kullanıcı odaklı bir yaklaşımın, arayüz tasarımından backend işlevlerine kadar projeye yön verdiğini gözlemledik. Dinamik arama ve filtreleme gibi işlevlerin başarılı bir şekilde uygulanabilmesi için veritabanı yapısının etkili bir şekilde tasarlanması gerektiğini fark ettik. Veritabanı ile arayüz arasında sağlanan uyum, kullanıcıların ihtiyaçlarına hızlı bir şekilde yanıt vermemizi kolaylaştırdı. Ayrıca, veritabanı işlemlerinin optimize edilmesinin performans açısından ne denli önemli olduğunu anladık. Projeyi geliştirirken edindiğimiz tecrübe, kullanıcı dostu bir arayüz tasarmanın ve arka planda sağlam bir altyapı kurmanın başarılı bir yazılımın temel unsurları arasında olduğunu gösterdi.

Deneyisel sonuçlar kısmında detaylı olarak değinildiği üzere, uygulamanın her bir fonksiyonunu test etmek, bu fonksiyonların işlevselliğini ve kullanıcıya sağladığı kolaylığı gözler önüne serdi. Proje boyunca yaptığımız testlerle, dinamik arama özelliğinin arama sonuçlarını hızla güncelleyebildiğini ve kullanıcı girdilerine duyarlı olduğunu gözlemledik. Bu süreç, projenin tamamlanmasının ardından kapsamlı testlerle doğrulanmış ve tüm işlevlerin istenilen gereksinimleri karşıladığı görülmüştür. Bu projeden çıkarımlarımız, yazılım geliştirme sürecinde yapılandırma, test etme ve kullanıcı odaklı

düşünme becerilerimizi geliştirmiştir ve benzer projelerde daha etkin çözümler üretebilmek için sağlam bir temel oluşturmuştur.

## V. DENEYSEL SONUÇLAR

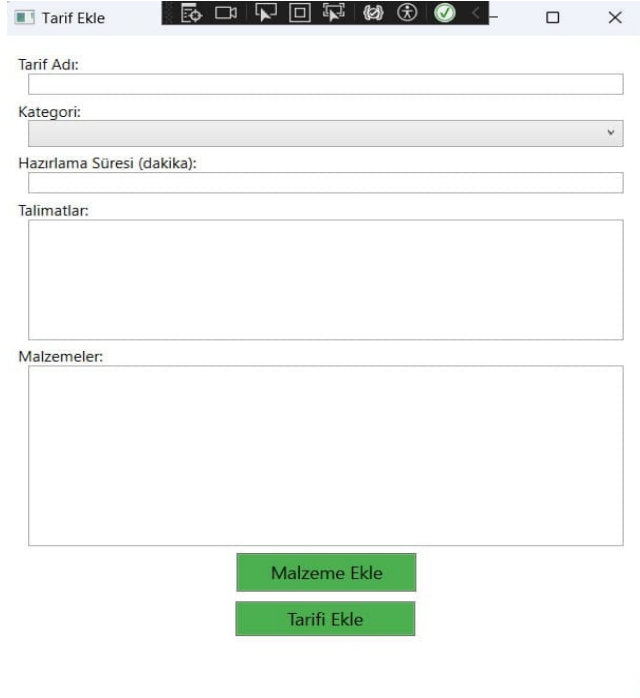


Fig. 5. Tarif Ekleme Ekranı

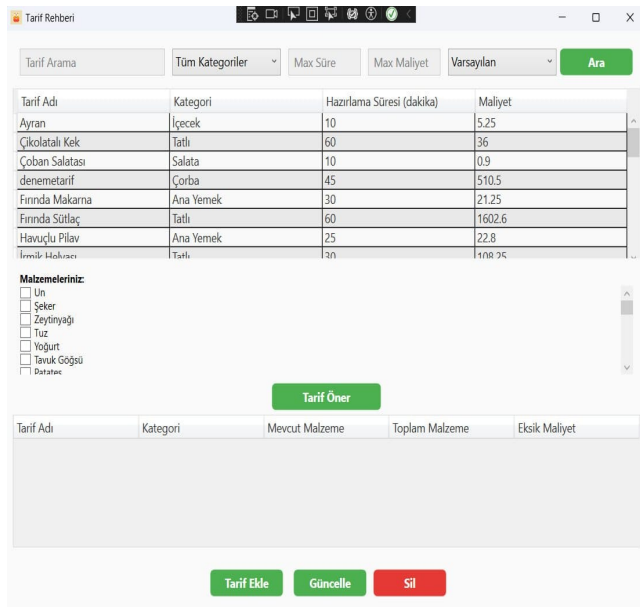


Fig. 6. Tarif Rehberi Ekranı

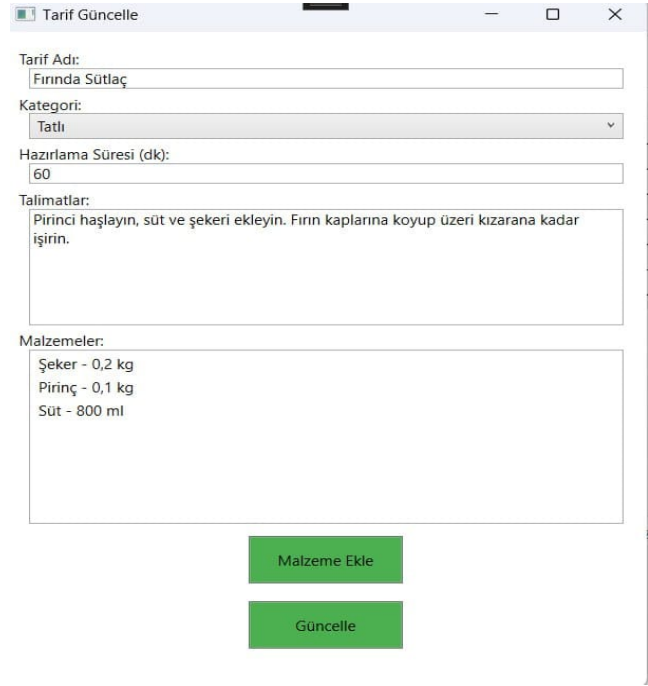


Fig. 7. Tarif Güncelleme Ekranı

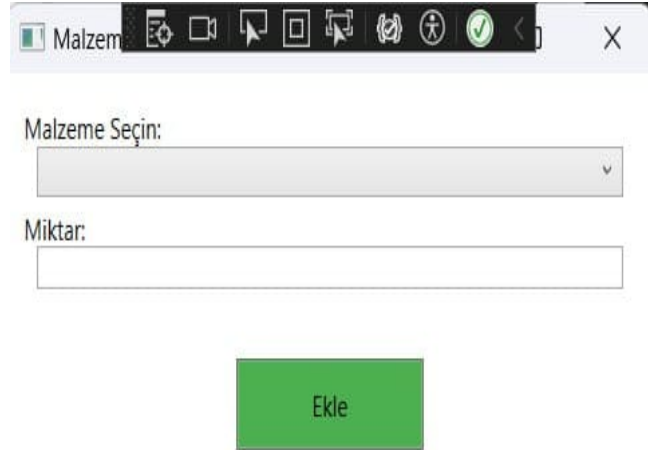


Fig. 8. Malzeme ekleme ekranı

Tarif Detayları

Tarif Adı:

Kategori:

Hazırlama Süresi:

Talimatlar:

Malzemeler:

Fig. 9. Tarif detayları ekranı

## VI. KAYNAKÇA

stackoverflow  
chatGPT  
Youtube