T.C.

Fırat Üniversitesi

Fen Bilimleri Enstitüsü

|  |
| --- |
| FIRAT SHOP SAHİBİNE ÇEVİRM İÇİ SATIŞ İMKANI SAĞLAYAN ELEKTRONİK TİCARET SİTESİ |
| Mahmut Burak ATAŞ-200260068  Eyüphan ŞAHİN-20026000 |
| **BİTİRME TEZİ** |
| **BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ** |
|  |
|  |
| **Elazığ 2024** |

T.C

FIRAT ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

**FIRAT SHOP SAHİBİNE ÇEVİRM İÇİ SATIŞ İMKANI SAĞLAYAN ELEKTRONİK TİCARET SİTESİ**

MAHMUT BURAK ATAŞ

EYÜPHAN ŞAHİN

BİTİRME TEZİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Bu bitirme tezi …./.…/2017 tarihinde, aşağıda belirtilen jüri tarafından oybirliği/oyçokluğu ile başarılı/başarısız olarak değerlendirilmiştir.

|  |  |  |
| --- | --- | --- |
| (İmza) | (İmza) | (İmza) |
| Danışman | Üye | Üye |

# ÖZGÜNLÜK BİLDİRİMİ

Bu çalışmada başka kaynaklardan yapılan tüm alıntıların, ilgili kaynaklar referans gösterilerek açıkça belirtildiğini, alıntılar dışındaki bölümlerin, özellikle projenin ana konusunu oluşturan teorik çalışmaların ve yazılım/donanımının tarafımızdan yapıldığını bildiririm.

Fırat Üniversitesi 26/07/2024

Bilgisayar Mühendisliği Eyüphan Şahin

23119 Elazığ Mahmut Burak Ataş

# BENZERLİK BİLDİRİMİ



# TEŞEKKÜR

"Fırat Shop Sahibine Çevrim İçi Satış İmkanı Sağlayan Elektronik Ticaret Sitesi" konulu çalışma Fırat Üniversitesi konulu çalışma Fırat Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümünde 2023-2024 yılı bahar döneminde bitirme projesi olarak hazırlanmıştır.

Lisans öğretimim süresince ve tez çalışmamın her aşamasında beni yönlendiren ve desteğini eksik etmeyen danışman hocam Sayın DOKTOR ÖĞRETİM ÜYESİ GÜLŞAH KARADUMAN’ a sonsuz teşekkürlerimi sunarım.

Adı SOYADI

Elazığ 2024

# İçindekiler

Sayfa

[Önsöz 4](#_Toc169290144)

[İçindekiler 5](#_Toc169290145)

[Özet 7](#_Toc169290146)

[Abstract 8](#_Toc169290147)

[Şekiller Listesi 9](#_Toc169290148)

[Tablolar Listesi 12](#_Toc169290149)

[Ekler Listesi 13](#_Toc169290150)

[Simgeler ve Kısaltmalar 14](#_Toc169290151)

[2.1. SOYUTLAMA 16](#_Toc169290152)

[2.3 KALITIM 16](#_Toc169290153)

[2.4. ÇOK BİÇİMLİLİK 17](#_Toc169290154)

[3. GO İLE OOP 17](#_Toc169290155)

[3.1 Sınıflar Ve Nesneler 17](#_Toc169290156)

[3.1.1 Sınıflar (Structs) 17](#_Toc169290157)

[3.1.2 Yöntemler (Methods) 17](#_Toc169290158)

[3.2 Kapsülleme (Encapsulation) 18](#_Toc169290159)

[3.3 Kalıtım (Inheritance) 18](#_Toc169290160)

[3.4 Polimorfizm (Polymorphism) 19](#_Toc169290161)

[3.4.1 Arayüzler (Interface) 19](#_Toc169290162)

[3.5 Sonuç 20](#_Toc169290163)

[4.1 Go 20](#_Toc169290164)

[4.1.1 Go Tarichçe 20](#_Toc169290165)

[4.1.2 Go’nun Temel Özellikleri 20](#_Toc169290166)

[4.1.3 Sonuç 21](#_Toc169290167)

[4.2 Veri Tabanı Seçimi 21](#_Toc169290168)

[4.2.1 SQL ve NOSQL Veri Tabanı Karşılaştırılması 21](#_Toc169290169)

[4.2.2 MongoDB 21](#_Toc169290170)

[4.3 JWT 22](#_Toc169290171)

[4.4 REST 23](#_Toc169290172)

[4.5 Gin 23](#_Toc169290173)

[4.6 HTML 24](#_Toc169290174)

[4.7 CSS 24](#_Toc169290175)

[4.8 JAVASCRİPT 24](#_Toc169290176)

[4.9 BOOSTRAP 25](#_Toc169290177)

[0.10 ANGULAR 25](#_Toc169290178)

[4.11 HTTP 26](#_Toc169290179)

[4.12 TypeScript 26](#_Toc169290180)

[4.12 RxJS Kütüphanesi 27](#_Toc169290181)

[4.12.1. Observable (Gözlemlenebilir) 27](#_Toc169290182)

[4.12.2 Map Operatörü 27](#_Toc169290183)

[4.12.3 Tap Operatörü 27](#_Toc169290184)

[4.12.4 Delay Operatörü 27](#_Toc169290185)

[4.12.5 Take Operatörü 27](#_Toc169290186)

[5.1 Proje Katmanlarının Oluşturulması 28](#_Toc169290187)

[5.1.1 Models Katmanı 28](#_Toc169290188)

[5.1.2 Database Katmanı 30](#_Toc169290189)

[5.1.3 Middleware Katmanı 31](#_Toc169290190)

[5.1.4 Routes Katmanı 31](#_Toc169290191)

[5.1.5 JWT Katmanı 33](#_Toc169290192)

[5.1.6 Controllers Katmanı 34](#_Toc169290193)

[5.1.7 Static Klasörü 37](#_Toc169290194)

[5.2 Kavramların Açıklanması 37](#_Toc169290195)

[5.2.1 Validator Nedir? 37](#_Toc169290196)

[5.2.2 Primitive Nedir? 38](#_Toc169290197)

[5.2.3 Bcrypt Nedir? 39](#_Toc169290198)

[5.2.4 Gomail.v2 nedir ? 39](#_Toc169290199)

[5.3 App Nedir ? 40](#_Toc169290200)

[5.3.1 App.module.ts Nedir ? 40](#_Toc169290201)

[5.3.2 App-routing.module.ts 41](#_Toc169290202)

[5.3.3. Guards Klasörü 42](#_Toc169290203)

[5.3.4 Angular Servis Dosyaları Nedir ? 43](#_Toc169290204)

[5.3.5 product.services.ts 44](#_Toc169290205)

[5.3.6 address.services.ts 46](#_Toc169290206)

[5.3.7 CartService 51](#_Toc169290207)

# ŞEKİLLER LİSTESİ

**Şekil 2.1** [Nesne yönelimli proramlama süreci 4](#_Toc24147221)

[**Şekil 3.1.** Ürün Yapısı 3](#_Toc24147219)

[**Şekil 3.2.** Kapsülleme Örneği 4](#_Toc24147221)

[**Şekil 3.3.** Kaıtım Örneği 5](#_Toc24147222)

[**Şekil 3.4.** Interface Örneği 5](#_Toc24147223)

[**Şekil 5.1.** Klasör Yapısı 8](#_Toc24147224)

[**Şekil 5.2.** Ürün Yapısı 3](#_Toc24147219)

[**Şekil 5.3.** Ürün Yapısı 3](#_Toc24147220)

[**Şekil 5.4.** Sepet Yapısı 4](#_Toc24147221)

[**Şekil 5.5.** Veri tabanı bağlantı işlemi 5](#_Toc24147222)

[**Şekil 5.6.** Middleware yapısı 5](#_Toc24147223)

[**Şekil 5.7.** User routes yapısı örneği 8](#_Toc24147224)

[**Şekil 5.8.** Kullanıcı ürün işlemleri 3](#_Toc24147219)

[**Şekil 5.9.** JWT tokenleri oluşturma 3](#_Toc24147220)

[**Şekil 5.10.** Token doğrulama işlemi 4](#_Toc24147221)

[**Şekil 5.11.** Ürün ekleme 5](#_Toc24147222)

[**Şekil 5.12.** Tüm ürünleri listeleme 5](#_Toc24147223)

[**Şekil 5.13.** Login işlemi örneği 8](#_Toc24147224)

[**Şekil 5.14.** Hashleme ve doğrulama işlemi 3](#_Toc24147219)

[**Şekil 5.15.** Abone ol için mail gönderme işlemi 3](#_Toc24147220)

[**Şekil 5.16.** Adres güncelleme işlemi 4](#_Toc24147221)

[**Şekil 5.17.** Ürün silme işlemi 5](#_Toc24147222)

[**Şekil 5.18.** Validator türetme 5](#_Toc24147223)

[**Şekil 5.19.** Kategori silme işlemi 8](#_Toc24147224)

[**Şekil 5.20.** App-routing.module.ts dosyası 3](#_Toc24147219)

[**Şekil 5.21.** Admin.guard.ts dosyası 3](#_Toc24147220)

[**Şekil 5.22.** GetProducts methodu 4](#_Toc24147221)

[**Şekil 5.23.** GetProductById methodu 5](#_Toc24147222)

[**Şekil 5.24.** CreateProduct methodu 5](#_Toc24147223)

[**Şekil 5.25.** DeleteProudct methodu 8](#_Toc24147224)

[**Şekil 5.26.** SaveImage methodu 3](#_Toc24147219)

[**Şekil 5.27.** Constructor 3](#_Toc24147220)

[**Şekil 5.28.** Addaddress methodu 4](#_Toc24147221)

[**Şekil 5.29.** GetToaddress methodu 5](#_Toc24147222)

[**Şekil 5.30** Gettoadress ve Removetoadress metodu 8](#_Toc24147224)

[**Şekil 5.31** Register , Logut , Login , IsLoggedIn ve removetoadress Metodları 8](#_Toc24147224)

[**Şekil 5.32** AutoLogin ve HandleError Metodları 8](#_Toc24147224)

[**Şekil 5.33** Cart.service.ts 8](#_Toc24147224)

[**Şekil 5.34** Cart.service.ts 8](#_Toc24147224)

[**Şekil 5.35** Category.service.ts 8](#_Toc24147224)

[**Şekil 5.36** GetIsconto.service.ts 8](#_Toc24147224)

[**Şekil 5.37** Myorder.service.ts 8](#_Toc24147224)

[**Şekil 5.38** Navbar.component.t 8](#_Toc24147224)

[**Şekil 5.39** Navbar.component.html 8](#_Toc24147224)

**Şekil 5.40** Home.component.ts 8

**Şekil 5.41** Home.component.html 8

[**Şekil 5.42** Pdcreate.component.cs 8](#_Toc24147224)

[**Şekil 5.43** Pdcreate.component.html 8](#_Toc24147224)

[**Şekil 5.44** Pdcreate.component.html 8](#_Toc24147224)

[**Şekil 5.45** Productlist.component.cs 8](#_Toc24147224)

[**Şekil 5.46** Productlist.component.html 8](#_Toc24147224)

[**Şekil 5.47** Productlist.component.cs 8](#_Toc24147224)

[**Şekil 5.48** Product.ts 8](#_Toc24147224)

[**Şekil 5.49** AuthResponse.t 8](#_Toc24147224)

[**Şekil 5.50** Anasayfa 8](#_Toc24147224)

[**Şekil 5.51** Popüler ürünler 8](#_Toc24147224)

[**Şekil5.54** Kategoriler 8](#_Toc24147224)

[**Şekil 5.55** Tüm ürünler 8](#_Toc24147224)

[**Şekil 5.56** Kategoriye göre ürünler 8](#_Toc24147224)

[**Şekil 5.57** Ürün detay sayfası 8](#_Toc24147224)

[**Şekil 5.58** Ürüne benzer ürünler sayfası 8](#_Toc24147224)

[**Şekil 5.59** Giriş yap sayfası 8](#_Toc24147224)

[**Şekil 5.60** Kayıt ol sayfası 8](#_Toc24147224)

[**Şekil 5.61** Adres bilgileri sayfası 8](#_Toc24147224)

[**Şekil 5.62** Hesap ayaraları sayfası 8](#_Toc24147224)

[**Şekil 5.63** Ürün oluşturma sayfasıı 8](#_Toc24147224)

[**Şekil 5.64** Kategori oluşturma sayfası 8](#_Toc24147224)

# TABLOLAR LİSTESİ

Sayfa

[Tablo 4.2.1 Sql Nosql karşılaştırılması 6](#_Toc23342567)

# Ekler Listesi

Sayfa

# Simgeler ve Kısaltmalar

**Kısaltmalar**

HTTP: Hypertext Transform Protocol (Hipermetin Aktarım Protokolü)

HTTPS: Hypertext Transfer Protocol Secure (Güvenli Hipermetin Aktarım Protokolü)

URL: Uniform Resource Locator (Birleşik Kaynak Konumlayıcı)

API: Application Programming Interface (Uygulama Programlama Arayüzü)

JSON: JavaScript Object Notation (JavaScript Nesne Gösterimi)

XML: Extensible Markup Language (Genişletilebilir İşaretleme Dili)

CRUD: Create, Read, Update, Delete (Oluşturma, Okuma, Güncelleme, Silme)

REST: Representational State Transfer (Temsil Durumu Aktarımı)

JWT: JSON Web Token (JSON Web Belgesi)

AOP: Aspect-Oriented Programming (Yönlendirilmiş Programlama)

ORM: Object-Relational Mapping (Nesne İlişkisel Eşleme)

HTML: Hyper Text Markup Language (Hiper Metin İşaretleme Dili)

CSS: Cascading Style Sheets (Basamaklı Stil Şablonları)

SQL: Structured Query Language (Yapılandırılmış Sorgulama Dili)

IoC: Invercion of Control (Kontrolün Tersine Çevrilmesi)

DI: Dependency Injection (Bağımlılık Enjeksiyonu)

IoT: Internet of Things (Nesnelerin İnterneti)

URI: Uniform Resource Identifier (Tek Tip Kaynek Tanımlayıcı)

SMTP: Simple Mail Transfer Protocol ( Basit Posta Aktarım Protokolü)

# ÖZET

**FIRAT SHOP SAHİBİNE ÇEVİRM İÇİ SATIŞ İMKANI SAĞLAYAN**

**ELEKTRONİK TİCARET SİTESİ**

Bu çalışma, Go, Angular ve MongoDB teknolojileri kullanılarak bir e-ticaret sitesi geliştirme sürecini ele almaktadır. E-ticaret sektöründe müşteri memnuniyetini artırmak ve kullanıcı dostu bir deneyim sunmak amacıyla tasarlanacak bu web sitesi, kullanıcıların ürün seçimini ve satın alma sürecini kolaylaştırmayı hedeflemektedir.

Bu çalışma kapsamında, Go, Angular ve MongoDB teknolojilerinin e-ticaret sektöründe nasıl kullanılabileceği ve işletmelere nasıl değer sağlayabileceği detaylı bir şekilde araştırılacaktır.

Go, güçlü bir programlama dili olarak, veritabanı yönetimi, web uygulama geliştirme ve güvenlik gibi önemli konularda işletmelere çeşitli avantajlar sunmaktadır. Angular ise modern, ölçeklenebilir ve kullanıcı dostu arayüzlerin oluşturulmasında etkili bir framework olarak öne çıkmaktadır. MongoDB, esnek ve yüksek performanslı veritabanı yönetimiyle dikkat çekmektedir. Bu üç teknolojinin bir araya getirilerek kullanılmasıyla e-ticaret sektöründe kullanıcılar için etkileyici bir çevrimiçi deneyim sağlanması amaçlanmaktadır.

Fırat Shop, kullanıcıların ürünleri kolayca filtrelemesine, mevcut ürünlerin detaylarını görüntülemesine ve alışveriş yapmasına olanak tanıyan bir arayüz sunacaktır. Ayrıca, kullanıcıların ödeme sürecini sorunsuz bir şekilde tamamlamasını sağlamak için güvenli ve kullanıcı dostu ödeme sistemleri entegre edilecektir.

Bu çalışma, Go, Angular ve MongoDB teknolojilerinin e-ticaret sektöründe uygulanabilirliğini ve işletmelere nasıl katma değer sağlayabileceğini ortaya koymayı hedeflemektedir. Araştırma sonuçları, e-ticaret işletmelerine, rekabetçi bir avantaj elde etmek ve müşteri memnuniyetini artırmak için en iyi uygulamalar ve stratejiler konusunda yol gösterici bir kaynak sunacaktır.

**Anahtar Kelimeler:** . Go, Angular, MongoDB, E-ticaret, Web sitesi, Ürün Seçimi, Ödeme Sistemleri

# ABSTRACT

**E-COMMERCE SITE PROVIDING THE OWNER OF FIRAT SHOP WITH THE OPPORTUNITY FOR ONLINE SALES**

This study addresses the process of developing an e-commerce site using Go, Angular, and MongoDB technologies. The aim of this website, designed to increase customer satisfaction in the e-commerce sector and provide a user-friendly experience, is to facilitate the product selection and purchasing process for users.

Within the scope of this study, the potential uses of Go, Angular, and MongoDB technologies in the e-commerce sector and how they can provide value to businesses will be thoroughly investigated.

Go, as a powerful programming language, offers various advantages to businesses in important areas such as database management, web application development, and security. Angular stands out as an effective framework for creating modern, scalable, and user-friendly interfaces. MongoDB is noted for its flexible and high-performance database management. The aim is to provide an impressive online experience for users in the e-commerce sector by using these three technologies together.

Fırat Shop will offer an interface that allows users to easily filter products, view the details of available products, and make purchases. Additionally, secure and user-friendly payment systems will be integrated to ensure users complete the payment process smoothly.

This study aims to demonstrate the applicability of Go, Angular, and MongoDB technologies in the e-commerce sector and how they can add value to businesses. The research results will provide a guiding resource for e-commerce businesses on best practices and strategies to gain a competitive advantage and increase customer satisfaction.

**Keywords**: Go, Angular, MongoDB, E-commerce, Website, Product Selection, Payment Systems

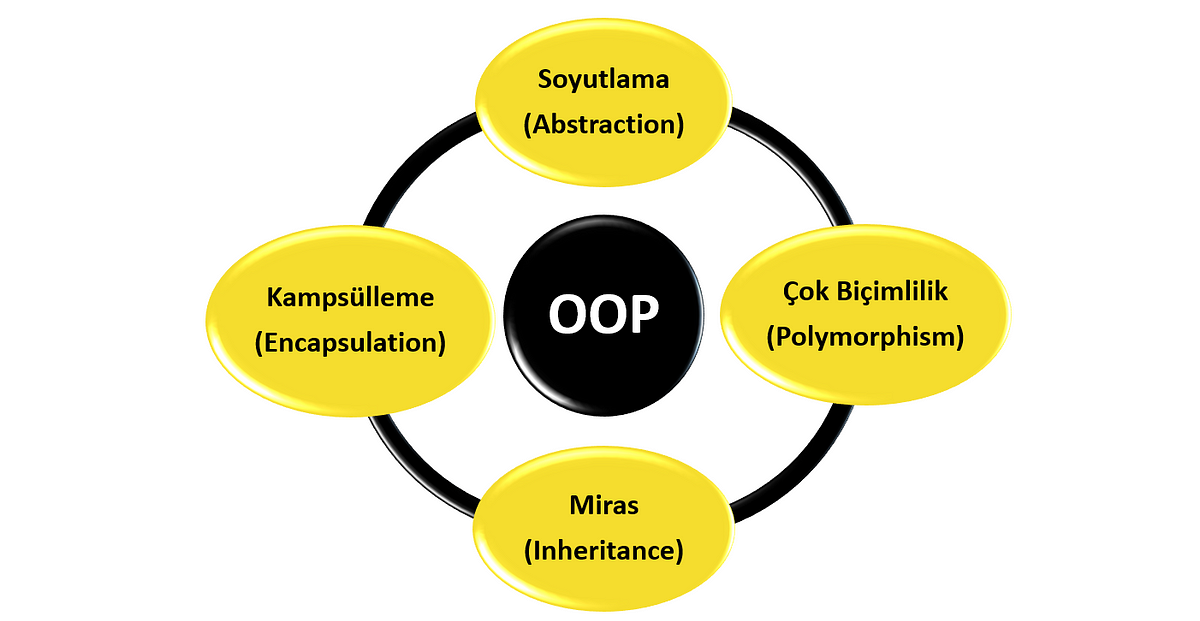
# 1.GİRİŞ

E-ticaret, günümüzde hızla büyüyen ve gelişen bir sektör haline gelmiştir. Teknolojinin ilerlemesi ve dijitalleşmenin artmasıyla birlikte, geleneksel mağazaların sayısı azalmıştır ve bunun sonucunda birçok işletme çevrimiçi alışverişe geçiş yapmıştır. Bu dönüşüm, işletmelerin çalışan sayısını azaltarak ürünleri daha uygun hale getirir ve bununla birlikte daha geniş bir müşteri kitlesine ulaşmalarına olanak tanır.

Çevrimiçi alışveriş, tüketicilere daha fazla seçenek, fiyat karşılaştırma imkanı ve evlerinin konforunda alışveriş yapma kolaylığı sunarak popülerliğini artırmıştır. Bu değişimle birlikte, kullanıcıların çevrimiçi alışveriş deneyimlerini iyileştirmek amacıyla çeşitli web geliştirme teknolojileri kullanılmaktadır. Angular ve GoLang gibi modern teknolojiler, performans ve kullanıcı deneyimi açısından güçlü çözümler sunarak e-ticaret uygulamalarının geliştirilmesinde önemli rol oynamaktadır.

Bu tezde, Angular , GoLang ve mongodb kullanılarak geliştirilen bir e-ticaret sitesinin tasarım, geliştirme ve değerlendirme süreci ele alınacaktır. Projenin amacı, kullanıcı dostu, güvenli ve yüksek performanslı bir e-ticaret platformu oluşturmaktır. Bu bağlamda, teknolojilerin seçimi, sistem tasarımı, geliştirme aşamaları ve karşılaşılan zorluklar detaylı bir şekilde incelenecektir.

# 2. NESNE YÖNELİMLİ PROGRAMLAMA



**Şekil 2.1** Nesne yönelimli programlama süreci

Nesne odaklı programlama, verileri ve nesneleri temel alarak yazılım tasarımını organize eden bir bilgisayar programlama felsefesidir. Bu yaklaşım, fonksiyonlardan ve mantıktan ziyade nesnelerin etrafında şekillenen bir yaklaşımı benimser. Bir nesne, kendine özgü özelliklere ve davranışlara sahip bir veri alanıdır. Nesne odaklı programlamada, her şey ayrı ayrı ayakta durabilen nesneler olarak gruplandırılır. Bu programlama modeli, yazılım geliştiriciler arasında oldukça popülerdir ve özellikle büyük, karmaşık ve sürekli güncellenen veya bakımı yapılan projeler için uygundur. Soyutlama, kalıtım, çok biçimlilik ve kapsülleme gibi temel kavramlar, yazılım geliştirme ve bakım süreçlerini kolaylaştırmaya yardımcı olur. Bu temel kavramlar, nesne odaklı programlamayı destekler ve benzersiz bir yaklaşım sunar.

## 2.1. SOYUTLAMA

Soyutlama, kullanıcının uygulamanın iç işleyişini bilmesine gerek olmadan sadece gerekli bilgileri görme yeteneğini sağlar. Bu sayede, uygulama içinde yapılan değişiklikler zamanla daha kolay yapılabilir. Soyutlama, kullanıcıya sadece temel bilgileri sunar ve ayrıntıları gizler, böylece kullanıcı sadece işlevselliği anlamak ve kullanmak için gerekli olan bilgilere odaklanabilir. Bu da yazılımın daha esnek, sürdürülebilir ve genişletilebilir olmasını sağlar.

# 2.2 KAPSÜLLEME

Kapsülleme, bir varlık içinde verileri ve ilgili fonksiyonları bir araya getirerek bütünlüğü ve güvenliği sağlayan bir mekanizmadır. Bu mekanizma, veriye ve fonksiyonlara dışarıdan erişimi sınıflar ve doğrudan değiştirilmesini veya yanlış kullanılmasını engeller. Böylece, verilerin tutarlılığı ve güvenliği korunur. Kapsülleme aynı zamanda yazılım bileşenlerinin birbirinden bağımsız olarak çalışmasını ve birlikte çalışmasını sağlayarak daha modüler bir yapı oluşturur. Bu da yazılımın daha iyi bir yapıya sahip olmasını ve bakımını kolaylaştırır.

## 2.3 KALITIM

Kalıtım, bir sınıfın var olan bir sınıftan özelliklerini ve davranışlarını devralmasıdır. Bu, kodun tekrar kullanılmasını sağlar ve yeni bir sınıfın, var olan bir sınıfın sahip olduğu özellikleri ve davranışları otomatik olarak edinmesini mümkün kılar. Kalıtım, sınıf hiyerarşileri oluşturmayı ve kodun daha organize ve yeniden kullanılabilir olmasını sağlar. Üst sınıf veya ebeveyn sınıf, alt sınıf veya çocuk sınıf tarafından genişletilir ve değiştirilebilir. Bu sayede, kodun tekrarlanması önlenir, kodun daha kolay bakımı yapılır ve yazılım geliştirme süreci hızlanır.

## 2.4. ÇOK BİÇİMLİLİK

Çok biçimlilik kavramı, aynı isme sahip ancak farklı parametrelerle çalışan yöntemlerin oluşturulabilmesini sağlar. Bu, kodun daha okunabilir ve anlaşılır olmasını sağlar. Örneğin, farklı nesnelerin aynı metodu çağırması durumunda, her nesnenin kendi özel implementasyonunu kullanması mümkündür. Bu sayede, kodun yeniden kullanılabilirliği artar ve sınıflar arasındaki ilişkiler daha esnek hale gelir. Çok biçimlilik, yazılımın genel tasarımını geliştirir ve kodun daha sürdürülebilir ve ölçeklenebilir olmasını sağlar. [1]

## 3. GO İLE OOP

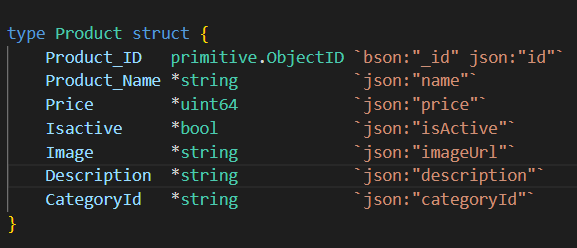
Go, tip güvenliği, eşzamanlılık ve basitlik özellikleriyle bilinen bir sistem programlama dilidir. Nesne yönelimli programlama (OOP) prensiplerini tamamen desteklemese de, OOP'nin bazı kavramlarını benimsemiştir. Go dilinde OOP'yi anlamak için temel OOP prensipleri ve Go'nun nasıl çalıştığına bakalım

## 3.1 Sınıflar Ve Nesneler

Go dilinde geleneksel sınıflar (classes) ve nesneler (objects) kavramları yoktur. Ancak, yapılar (structs) ve yöntemler (methods) kullanarak benzer bir işlevsellik elde edilebilir.

### 3.1.1 Sınıflar (Structs)

Go dilinde, yapı (struct), veri alanlarını bir arada tutan bir veri tipidir. Bir yapı, OOP'deki sınıflara benzer. Örneğin product yapısı şekil 3.1 de gösterilmiştir.



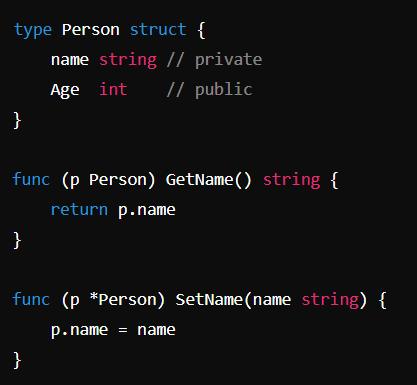
**Şekil 3.1** Ürün Struct Yapısı

### 3.1.2 Yöntemler (Methods)

Bir yapıya bağlı fonksiyonlar tanımlayarak yöntemler oluşturabiliriz. Bu yöntemler, OOP'deki sınıf yöntemlerine benzer.

### 3.2 Kapsülleme (Encapsulation)

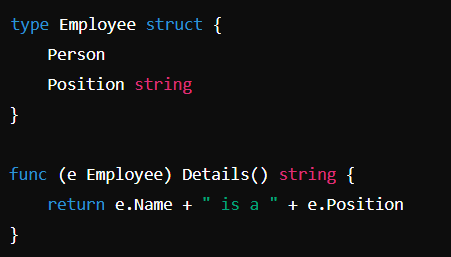
Kapsülleme, verilerin gizlenmesi ve sadece belirli fonksiyonlar aracılığıyla erişilmesi prensibidir. Go dilinde, bu prensip, alanların büyük harfli (public) veya küçük harfli (private) olarak tanımlanmasıyla sağlanır. Şekil 3.2 kapsülleme örneği gösterilmiştir.



**Şekil 3.2** Kapsülleme Örneği

### 3.3 Kalıtım (Inheritance)

Go dilinde doğrudan sınıf tabanlı kalıtım yoktur. Ancak, kompozisyon (composition) kullanılarak benzer bir yapı kurulabilir. Yani, bir yapı başka bir yapıyı alan olarak içerir. Bu yapı aşağıdaki gibi oluşturulur



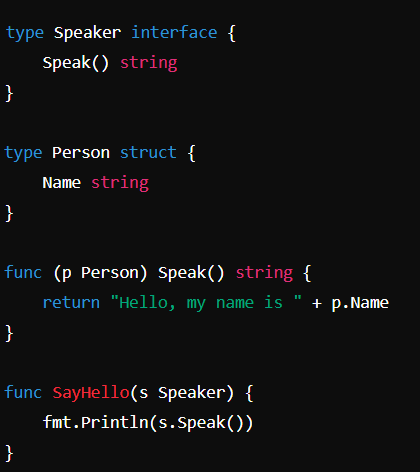
**Şekil 3.3** Örnek Kalıtım

### 3.4 Polimorfizm (Polymorphism)

Go dilinde polimorfizm, arayüzler (interfaces) kullanılarak sağlanır. Bir arayüz, belirli bir yöntem setini tanımlar ve bu yöntemleri uygulayan her yapı, o arayüz tipinde kullanılabilir.

### 3.4.1 Arayüzler (Interface)

Bir arayüz, belirli yöntemleri içeren bir yadır. Herhangi bir yapının bir arayüzü implemente edip etmediği, açıkça belirtilmez. Yapı, arayüzdeki yöntemleri içeriyorsa, o arayüzü implemente etmiş sayılır. Örnek bir arayüz tasarımı şekil 3.4 gösterilmiştir.



**Şekil 3.4** Interface örneği

## 3.5 Sonuç

Go, klasik OOP dillerindeki bazı özellikleri benimsememiştir, fakat yapılar, yöntemler, kompozisyon ve arayüzler aracılığıyla OOP prensiplerinin birçoğunu destekler. Bu, Go dilini öğrenenlerin tip güvenliği, eşzamanlılık ve basitlik avantajlarından yararlanırken OOP'nin güçlü yönlerini de kullanabilmelerini sağlar.

# 4. KULLANILAN TEKNOLOJİ VE KÜTÜPHANELERİN TANIMIı

## 4.1 Go

Google'da 2007 yılından itibaren geliştirilmeye başlayan açık kaynak programlama dilidir. İlk web sitesi golang.org alan adına sahip olduğundan golang ismiyle anılsa da doğru adı Go'dur. Daha çok sistem programlama için tasarlanmış olup, derlenmiş ve statik tipli bir dildir. Kasım 2009'da çıkmıştır. Go derleyicisi "gc", açık kaynak yazılım olarak, Linux, OS X, Windows, bazı BSD ve Unix versiyonları ve ayrıca 2015'ten itibaren akıllı telefonlar için geliştirilmiştir. [2]

### 4.1.1 Go Tarichçe

Go'nun geliştiricileri, Rob Pike, Ken Thompson ve Robert Griesemer, dilin temellerini atarken, mevcut dillerin eksikliklerini gidermeyi hedeflemişlerdir. C ve C++ gibi düşük seviyeli dillerin performans avantajlarını korurken, Python ve JavaScript gibi yüksek seviyeli dillerin kolay okunabilirliğini ve yazılabilirliğini sağlamak istediler. Bu hedeflere ulaşmak için Go, basitlik, verimlilik ve güvenilirlik ilkelerine sıkı sıkıya bağlı kalacak şekilde tasarlanmıştır.

### 4.1.2 Go’nun Temel Özellikleri

Go'nun temel özellikleri arasında ilk olarak statik tip kontrolü ve tip güvenliği gelir. Go, statik olarak tip denetimi yapılan bir dildir ve bu sayede hataların derleme zamanında yakalanmasını sağlar. Bu özellik, dinamik dillerde sıkça rastlanan runtime hatalarının önüne geçer. İkinci olarak, Go otomatik bellek yönetimi sağlayan garbage collection mekanizmasına sahiptir. Bu, geliştiricilerin bellek yönetimiyle ilgili karmaşık detaylarla uğraşmadan verimli ve güvenilir kod yazabilmelerini mümkün kılar. Üçüncü olarak, Go basit ve temiz bir söz dizimi sunar. Karmaşıklığı en aza indirmek amacıyla tasarlanmış olan bu söz dizimi, hem öğrenme sürecini hızlandırır hem de kodun bakımını kolaylaştırır. Dördüncü olarak, Go'nun eşzamanlılık (concurrency) modeli dikkat çeker. Özellikle çok çekirdekli işlemcilerin verimli kullanılmasını sağlayan Goroutine'ler hafif iş parçacıklarıdır ve kanallar, bu iş parçacıkları arasında güvenli veri iletişimini sağlar. Son olarak, Go'nun zengin ve kapsamlı bir standart kütüphanesi vardır. Dosya işlemleri, ağ programlama, kriptografi ve daha birçok konuda güçlü araçlar sunan bu kütüphane, geliştiricilere geniş bir yelpazede imkanlar tanır.

### 4.1.3 Sonuç

Go, modern yazılım geliştirme ihtiyaçlarını karşılamak için güçlü, verimli ve kullanıcı dostu bir dil olarak kendini kanıtlamıştır. Geliştiricilere sunduğu performans, kolaylık ve eşzamanlılık özellikleri, onu birçok farklı alanda tercih edilen bir dil haline getirmiştir. Gelişen ekosistemi ve aktif topluluğu ile Go, gelecekte de yazılım dünyasında önemli bir rol oynamaya devam edecektir

## 4.2 Veri Tabanı Seçimi

## 4.2.1 SQL ve NOSQL Veri Tabanı Karşılaştırılması

SQL (Structured Query Language) ve NoSQL (Not Only SQL) veritabanları, farklı kullanım senaryoları ve ihtiyaçlar için tasarlanmışlardır.

|  |  |  |
| --- | --- | --- |
|  | **SQL** | **NOSQL** |
| **Veri Yapısı** | Yapısal Veri, Schema | Esnek Yapı, Schema-Free |
| **Veri Tutarlılığı** | Atomicity,Consistency solation, Durability (ACID) | Basic Availability, Soft-state,Eventual Consistency (BASE) |
| **Performans ve Ölçeklenebilirlik** | Yavaş Sorgu | Hızlı Sorgu |
| **Güvenlik** | Yerleşik Güvenlik Özellikleri | Güvenlik Eksiklikleri |

**Tablo 4.2.1**  Sql NoSql Karşıaştırılması

SQL ve NoSQL veritabanları, farklı ihtiyaçlar ve kullanım senaryoları için özelleştirilmiş çözümler sunar. E-ticaret gibi dinamik ve geniş veri kümelerinin yönetildiği sistemlerde, her iki veri tabanı türünün avantajları ve dezavantajları dikkatlice değerlendirilerek en uygun çözüm seçilmelidir.

### 4.2.2 MongoDB

MongoDB, NoSQL veritabanı sistemleri arasında en popüler olanlardan biridir. Esnek veri yapısı ve yüksek performans sunma yetenekleri ile bilinir. MongoDB, verileri JSON benzeri belgeler (dokümanlar) olarak depolar ve yönetir. Bu esneklik, veritabanı yapısının hızlı bir şekilde değişen ihtiyaçlara uyum sağlamasını mümkün kılar.

MongoDB'nin temel özellikleri arasında doküman tabanlı yapı öne çıkar. Veriler, hızlı performans sunan BSON (Binary JSON) formatında saklanır. Bu yapı, esnek bir şemaya sahiptir; belgeler farklı yapıda olabilir ve veritabanı şeması kolayca değiştirilebilir. Yatay ölçeklenebilirlik, MongoDB'nin büyük veri kümeleri üzerinde yüksek performans sağlamasını sağlar; veritabanı yatay olarak bölünerek (sharding) verilerin birden çok sunucuya dağıtılmasını mümkün kılar. Yüksek performans, çeşitli indeksleme seçenekleri ve bellek etkin kullanımıyla elde edilir. Gelişmiş sorgulama yetenekleri, güçlü bir sorgulama dili ve toplama çerçevesi aracılığıyla veri analizi ve işleme imkanı sunar. Esnek veri modeli, yerleşik ilişkiler ve referanslar sayesinde karmaşık veri yapılarını yönetmeyi sağlar. Güvenlik, kullanıcı kimlik doğrulama ve rol tabanlı erişim kontrolü ile sağlanır. Yedeklilik ve yüksek erişilebilirlik ise veritabanının yedekli kopyalarını oluşturarak veri bütünlüğünü ve dayanıklılığını sağlar.

#### 4.2.2.1 BSON Nedir

BSON, Binary JSON'un (Binary JSON) kısaltmasıdır ve JSON formatının ikili bir temsilidir. Özellikle MongoDB gibi NoSQL veritabanları tarafından yaygın olarak kullanılmaktadır. BSON'ın temel özellikleri arasında şunlar bulunur: İkili formatta olması, verilerin daha az yer kaplamasını ve daha hızlı işlenmesini sağlar. Ayrıca, BSON tarih, tamsayı, kayan nokta gibi ek veri tiplerini destekler. Bu özellikler sayesinde veri işleme süreçleri daha hızlı ve etkin bir şekilde gerçekleştirilebilir.

## 4.3 JWT

JWT (JSON Web Token), web uygulamalarında kimlik doğrulama ve yetkilendirme için kullanılan bir standarttır. JWT'ler, verileri güvenli bir şekilde taşımak ve doğrulamak için kullanılan kompakt ve güvenli bir formattır. Bir JWT, üç bölümden oluşan bir dizedir: başlık, talep ve imza. Başlık, JWT'nin algoritma ve tip gibi bilgilerini içerir. Talep bölümü, kullanıcıya veya uygulamaya özgü bilgileri içerir ve talepleri temsil eder. Örneğin, kullanıcının kimliği, rolleri veya ekstra verileri içerebilir. İmza bölümü ise JWT'nin doğrulama amacıyla kullanılan bir kriptografik imzadır. JWT'nin en büyük avantajlarından biri, sunucu tarafında oturum durumunu tutmamaktır. Her istekle birlikte gönderilen JWT, sunucu tarafında doğrulama işlemlerini gerçekleştirmek için kullanılır. Bu, sunucunun her bir isteği bağımsız olarak işleyebilmesini ve ölçeklenebilir olmasını sağlar. JWT'lerin kullanımı kolaydır ve taşınabilirlik sağlar. JWT'leri kullanarak farklı uygulamalar arasında veri paylaşımı yapabilir ve kimlik doğrulama süreçlerini basitleştirebilirsiniz. Ayrıca, JWT'lerin süresi sınırlı olabilir, bu sayede güvenlik önlemlerini artırabilirsiniz. Sonuç olarak, JWT'ler, kimlik doğrulama ve yetkilendirme için kullanılan bir standarttır. Güvenli bir şekilde veri taşımak ve doğrulamak için kullanılır. Kullanımı kolaydır ve farklı uygulamalar arasında veri paylaşımını sağlar. [3]

## 4.4 REST

REST (Representational State Transfer), ilk olarak Roy Fielding tarafından tanıtılan bir terimdir. REST, istemci-sunucu mimarisi üzerinde çalışan bir mimaridir. İstemci, sunucuya istekler 10 gönderir ve sunucu bu istekleri işleyerek yanıtlarını verir. Bu istekler ve yanıtlar, kaynakların temsilleri etrafında oluşturulmuştur. Bir kaynak, URI ile tanımlanan bir şeydir. Kaynağın temsili genellikle kaynağın mevcut veya amaçlanan durumunu yansıtan bir belgedir. REST, SOAP'tan farklı olarak daha esnek bir yapısı vardır. REST'in dilinde, isimler ve fiiller önemli bir rol oynar. SOAP mesajlarındaki gibi zarf ve başlık gibi mesaj formatları gerektirmez. Bu nedenle, XML ayrıştırma gibi işlemlere ihtiyaç duyulmaz ve bant genişliği tasarrufu sağlanır. REST'in tasarım ilkeleri arasında erişilebilirlik, durumsuzluk ve birleşik arayüz bulunur. Erişilebilirlik, REST'in kaynakları URI ile modellenmesini içerir. Birleşik ve standart bir arayüz, RESTful hizmetlere erişmek için belirli HTTP yöntemlerini kullanır. Her işlem bağımsızdır ve önceki işlemlerle ilişkili değildir. İşlem için gereken tüm veri isteğin içinde bulunur ve sunucu, istemci oturum verilerini tutmaz, bu nedenle sunucu yanıtları da bağımsızdır. Bu ilkeler, REST uygulamalarını basit ve hafif hale getirir. REST mimarisini izleyen web uygulamalarına RESTful web servisleri denir. RESTful web servisleri, kaynakları almak, oluşturmak, güncellemek ve silmek için HTTP yöntemlerini kullanır. RESTful API’nin 3 temel prensibi vardır. İlk olarak, durumsuzluk ilkesiyle her istek birbirinden bağımsızdır ve sunucuda istemciye özgü bilgi tutulmaz. İkincisi, kaynaklar URI ile tanımlanır ve temsilciler aracılığıyla istemciye iletilir. Üçüncüsü, http metotları (GET, POST, PUT, DELETE vb.) kaynak üzerindde belirli işlemleri gerçekleştirmek için kullanılır. [4]

## 4.5 Gin

Gin, Go programlama dilinde geliştirilen hafif, hızlı ve verimli bir web framework'üdür. Minimalist tasarımıyla dikkat çeker; temelde sadece HTTP yönlendirmesi, middleware'ler, gruplama ve filtreleme gibi temel işlevleri sunar. Bu minimalist yaklaşım, geliştiricilere esneklik sağlar, çünkü sadece ihtiyaç duydukları özellikleri kullanabilirler. Bu da uygulama performansını artırır ve gereksiz karmaşıklığı önler.

Gin, hızlı performansıyla öne çıkar. Paralel işleme yeteneklerini kullanarak yüksek talep yüklerine bile hızlı yanıtlar verebilir. Ayrıca, JSON kodlaması ve sıkıştırma gibi özelliklerle veri manipülasyonunu ve işleme süreçlerini optimize eder. Middleware'lerle HTTP taleplerini kolayca işleyebilir, talep zincirini genişletebilir ve işlevselliği esnek bir şekilde genişletebilirsiniz.

RESTful API'lerin oluşturulmasından, basit web sitelerine kadar çeşitli uygulamalar için uygun bir seçenektir. Geliştiricilere hızlı bir şekilde API'ler ve web uygulamaları oluşturma imkanı sunar. Ayrıca, Go'nun doğal özelliklerinden ve performansından yararlanırken, düşük bellek tüketimi ve yüksek performans sağlar. Bu da ölçeklenebilir ve güvenilir uygulamalar geliştirmek için ideal bir seçenek yapar.

## 4.6 HTML

HTML: HTML, (HyperText Markup Language), internet web sayfalarının oluşturulmasını sağlayan bir programlama dilidir. Metin, grafik, ses, video gibi farklı veri türlerini biçimlendirmek için kullanılan bu dil, hypertext belgelerini oluşturmak için özel komutlar içerir. Temel amacı, kullanıcıların farklı içerikleri etkileşimli ve yapılandırılmış bir şekilde görmesini sağlayarak web deneyimini zenginleştirmektir. Hypertext kavramı, belgelerin birbirine referans vermesini ve böylece kullanıcıların belgeler arasında kolayca gezinmesini sağlar. HTML, bu referansları linkler aracılığıyla oluşturarak farklı belgeleri birbirine bağlayarak web sayfalarının temel yapı taşlarını oluşturur. HTML, zaman içinde farklı sürümlerle gelişmiş ve yeni özellikler eklenmiştir. İlk başlarda sadece basit yapısal etiketler kullanılırken, sonraki sürümlerle birlikte metin yapılarına yapılan eklemeler, formlar için teknik etiketler, tablolar, matematiksel formüller ve çoklu ortam desteği gibi pek çok özellik eklenmiştir. Güvenlik, web sayfalarının önemli bir konusudur ve HTML'de de bu konuda çalışmalar yapılmıştır. Özellikle web uygulamalarının güvenliği için çeşitli önlemler alınmış ve güvenlik standartları belirlenmiştir. Bununla birlikte, hala web sayfaları ve HTTP protokolü üzerinde güvenlik konusunda geliştirmeler ve iyileştirmeler devam etmektedir. [5]

## 4.7 CSS

"Cascading Style Sheets" kelimelerinin kısaltmasıdır ve Türkçe'de "Basamaklı Stil Şablonları" veya "Basamaklı Biçim Sayfaları" anlamına gelir. HTML etiketlerine görsel olarak stil ve biçim kazandırmak için kullanılır. CSS, web sayfalarının görünümünü özelleştirmek için kullanılan bir teknolojidir. Renkler, yazı tipleri, arka planlar, boyutlar, pozisyonlar gibi görsel özellikleri kontrol etmek için kullanılır. CSS, HTML etiketlerine atanmış olan id ve class seçicileri sayesinde belirli etiketleri hedef alır ve onlara stil uygular. Bu sayede web tasarımcılar, sayfaları istedikleri şekilde biçimlendirebilirler. CSS'in gücü, web sayfalarının stil ve içeriğini ayrı olarak yönetebilmesidir. Bu sayede tasarım değişiklikleri yapmak daha kolay hale gelir ve sayfaların tutarlılığı sağlanır. CSS, web tasarımında önemli bir role sahiptir ve web sayfalarının profesyonel ve çekici görünmesini sağlar. [6]

## 4.8 JAVASCRİPT

JavaScript, bir betik dili olarak kullanılan ve web sayfalarında kullanımı yaygın olan bir programlama dilidir. HTML dokümanlarına gömülerek kullanılır ve web sayfalarına 8 etkileşimli özellikler eklemek için kullanılır. JavaScript, kullanıcıların web sayfalarında dinamik içerikler oluşturmasını ve etkileşimli öğelerle etkileşime geçmesini sağlar. Bu dil, web tarayıcıları üzerinde çalışır ve tarayıcı tarafında çalıştığı için sunucu taraflı dillerden farklılık gösterir. JavaScript, web sayfalarında HTML öğelerini manipüle etme, formları doğrulama, kullanıcı etkileşimleri için olaylar oluşturma gibi birçok işlemi gerçekleştirebilir. Ayrıca, web uygulamalarında veri doğrulaması, animasyonlar, veri görselleştirme gibi çeşitli alanlarda da kullanılabilir. JavaScript'in güçlü yanlarından biri, hızlı çalışması ve tarayıcılar tarafından desteklenmesidir. Tarayıcıların JavaScript motorları sayesinde, kullanıcıların web sayfalarında hızlı ve etkileyici deneyimler yaşamasını sağlar. Ayrıca, JavaScript'in sürekli olarak gelişmekte olan bir dil olması, yeni özelliklerin eklenmesi ve dilin kullanımının genişlemesi anlamına gelir. [7]

## 4.9 BOOSTRAP

Bootstrap, responsive (duyarlı) web siteleri oluşturmak için yaygın olarak kullanılan bir HTML, CSS ve JavaScript framework'üdür. Bu framework sayesinde web geliştiriciler, tarayıcılar arasında (Chrome, Firefox, Opera, Internet Explorer) ve farklı cihazlarda (mobil ve masaüstü) sorunsuz bir şekilde çalışan web siteleri tasarlayabilirler. Responsive Web Design, bir web sitesinin görünümünün ve içeriğinin otomatik olarak kullanılan platforma uyum sağlayabilmesini sağlayan bir tasarım yaklaşımıdır. 9 Bootstrap, içerisinde CSS, JavaScript ve jQuery dosyalarını barındıran bir dizi araç sunar. Bu araçlar, menüler, kartlar, formlar, jumbotron, medya nesneleri, gezinme çubuğu, sayfalama, ilerleme çubukları, dönen simgeler gibi farklı web bileşenlerinin oluşturulmasını kolaylaştırır. Bunun yanı sıra, Bootstrap aynı zamanda daha çekici ve responsive bir web sitesi oluşturmayı destekleyen düzen ve içerik formatları da sunar. Bootstrap 4, Bootstrap framework'ünün en son sürümüdür ve daha önceki sürümlerden daha fazla özellik ve geliştirilmiş performans sunar. Yenilikçi grid sistemi, esnek bileşenleri ve modern tasarımıyla web geliştiricilere daha verimli bir çalışma ortamı sağlar. [8]

## 0.10 **ANGULAR**

Angular, ilk olarak 2008 yılında Google çalışanları Misko Hevery ve Adam Abrons tarafından oluşturuldu. O dönemde AngularJS adıyla bilinen bu framework, saf JavaScript kullanılarak geliştiriliyordu. Bu zamanlarda web sitelerinin çoğu çok sayfalı uygulama yaklaşımına dayanıyordu. Kullanıcı bir bağlantıya tıkladığında, tarayıcının sunucudan istenen HTML belgesini alması gerekiyordu. Ancak bu işlem, internet bağlantısı ve sunucunun tepkisine bağlı olarak zaman alabiliyordu. Zamanla kullanıcı cihazlarının performansı arttıkça, uygulama mantığı tarayıcıda yürütülebilir hale geldi. Bu da Tek Sayfa Uygulamalarının (SPA) ortaya çıkmasına yol açtı. 10 AngularJS, SPA geliştirme için ilk frameworklerden biri olarak öne çıktı. İki yönlü veri bağlama ve harici betikleri içe aktarma gibi özellikler sunarak geliştiricilere kolaylık sağladı. Diğer rakiplerine göre erişilebilir olması da büyük avantajlarından biriydi. Sadece CDN bağlantısını HTML belgesine ekleyerek ve etiketine ng-app direktifini ekleyerek uygulama kullanıma hazır hale getirilebiliyordu. Angular güçlü bir frameworktür ve geliştiricilere geniş ölçekli ve karmaşık web uygulamaları geliştirme imkanı sunar. İleri düzey veri bağlama, bileşen tabanlı mimari, modülerlik ve etkileşimli kullanıcı arabirimleri gibi özellikler içerir. Performansı ve kullanım kolaylığı sayesinde büyük projelerde sıkça tercih edilen bir seçenektir.

## 4.11 HTTP

HTTP, web sunucuları ve istemciler arasındaki iletişim için kullanılan bir protokoldür. WWW’ün temelidir ve web sayfalarını yüklemek ve internet üzerinden veri transferi yapmakla ilgilidir. HTTP isteği istemci tarafından sunucuya yapılan taleptir ve istenen eylemi ve gerekli bilgileri belirtir. İstek, HTTP sürümü, URL, HTTP yöntemi (GET veya POST gibi), istek başlıkları ve isteğe bağlı istek gövdesi içerir. HTTP yöntemleri, sunucuda gerçekleştirilmek istenen eylemi belirtir. Ortak yöntemler arasında GET (bir kaynağı almak), POST (işlenmek üzere veri göndermek), PUT (bir kaynağı güncellemek) ve DELETE (bir kaynağı silmek) bulunur. HTTP yanıtı, sunucunun istemcinin isteğine verdiği cevaptır. Bu cevap, bir HTTP durum kodu içerir ve isteğin başarılı olup olmadığını belirtir. 2xx aralığındaki durum kodları başarıyı gösterirken, 4xx aralığındaki kodlar istemci hatalarını ve 5xx aralığındaki kodlar sunucu hatalarını temsil eder. [9]

## 4.12 TypeScript

TypeScript, JavaScript'e alternatif olarak geliştirilen bir programlama dilidir. Microsoft tarafından desteklenen ve hala geliştirilmeye devam eden TypeScript, ECMAScript (ES) 6'nın üzerine inşa edilmiştir. TypeScript, kodun derleme zamanında düz JavaScript'e dönüştürülmesiyle çalışır ve daha güçlü bir yazım denetimi sağlar. TypeScript'in en büyük avantajlarından biri, güçlü bir tip sistemi sunmasıdır. Bu sayede, kodun daha tutarlı ve hatasız olmasını sağlar. Aynı zamanda, modern JavaScript dil özelliklerini destekler. Örneğin, 'for...of' döngüsü, ok fonksiyonları (lambda ifadeleri) ve dekoratörler gibi özellikleri kullanabilirsiniz. Dekoratörler, TypeScript'in Angular framework'üyle birlikte kullanıldığında önemli bir rol oynar. Dekoratörler, bileşenlere veya diğer öğelere ek özellikler eklemek için kullanılır ve kodun daha esnek ve özelleştirilebilir olmasını sağlar. TypeScript, büyük ve karmaşık projelerde geliştirme sürecini kolaylaştırır. Tip kontrolleri sayesinde hataların erken tespit edilmesi ve daha güvenli bir kod yazma deneyimi sunar. Ayrıca, TypeScript'in JavaScript'e dönüştürülmesi, tarayıcılar tarafından kolaylıkla çalıştırılabilir hale gelmesini sağlar. Sonuç olarak, TypeScript, geliştiricilere JavaScript'e kıyasla daha güçlü bir yazım denetimi ve gelişmiş dil özellikleri sunar. Angular gibi popüler framework'lerle uyumlu çalışabilir ve daha güvenilir ve özelleştirilebilir projeler oluşturmanıza yardımcı olur.[14]

## 4.12 RxJS Kütüphanesi

RxJS (Reactive Extensions for JavaScript), reaktif programlama paradigmasını kullanarak veri akışlarını ve olayları yönetmek için kullanılan güçlü bir kütüphanedir. Angular uygulamalarında sıkça kullanılır ve asenkron veri işleme işlemlerini daha kolay ve yönetilebilir hale getirir. RxJS, veri akışlarını ve olayları temsil eden Observable (gözlemlenebilir) sınıfı ile birlikte gelir.

### 4.12.1. Observable (Gözlemlenebilir)

Observable, RxJS kütüphanesinin temel yapı taşıdır. Asenkron veri akışlarını temsil eder ve bu akışlara abone olarak (subscribe) veri olaylarını izleyebiliriz. Bir Observable, veri yayınlar ve bu verilere abone olan gözlemciler (subscribers) bu verileri alır.

### 4.12.2 Map Operatörü

map operatörü, bir Observable'dan gelen her bir veriyi alır ve belirli bir işlevi uygulayarak yeni bir veri döner. Bu, veri dönüştürme işlemlerinde kullanılır.

### 4.12.3 Tap Operatörü

tap operatörü, Observable üzerindeki verilere müdahale etmeden yan etkiler yaratmak için kullanılır. Genellikle loglama veya ek yan işlemler yapmak için kullanılır.

### 4.12.4 Delay Operatörü

delay operatörü, bir Observable'ın yaydığı her bir değeri belirtilen süre kadar geciktirir. Bu, zamanlama kontrolleri yapmak için kullanılır.

### 4.12.5 Take Operatörü

take operatörü, bir Observable'dan belirli sayıda değeri alır ve sonrasında tamamlanır. Bu, sınırlı sayıda veri olaylarını işlemek için kullanılır.

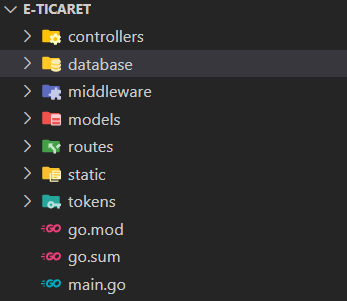
**4.12.6 ExhaustMap Operatörü**

exhaustMap operatörü, iç içe Observable'larla çalışmak için kullanılır. Yeni bir iç Observable başlatıldığında, önceki iç Observable tamamlanana kadar yeni iç Observable'ları yok sayar.

# 5. PROJENİN GERÇEKLEŞTİRME AŞAMALARI

## 5.1 Proje Katmanlarının Oluşturulması

İlk olarak go için belirlediğimiz çalışma dosyasında (genelde /go/src klasörü) bir proje oluşturup pro için go mod init komutuyla bir modül dosyası oluşturmamız lazım. Daha sonra gerekli klasör yapılarını entegre edelim. Projede kullanılan golang klasör yapısı aşağıdaki gibidir



**Şekil 5.1**  Klasör Yapısı

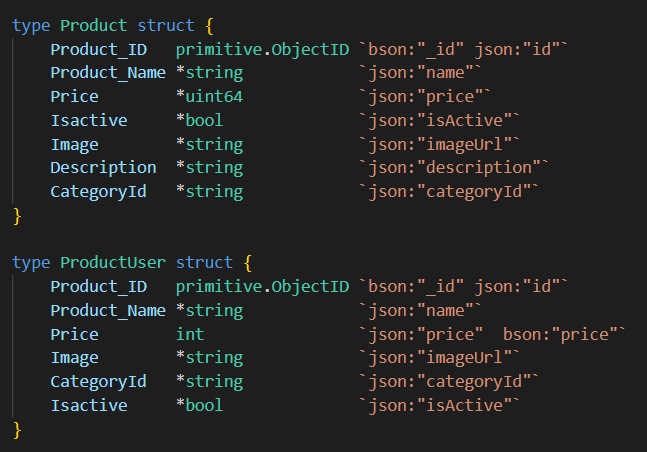
### 5.1.1 Models Katmanı

Models katmanı uygulamanın veri yapısını ve veritabanı ile etkileşimini tanımlayan kritik bir bileşendir. Bu katman, veritabanı tablolarına karşılık gelen yapıların (struct'ların) tanımlandığı yerdir; örneğin, bir "User" tablosunu temsil eden bir "User" struct'ı burada yer alır. Ayrıca, models katmanı verilerin doğruluğunu sağlamak için doğrulama kurallarını içerir ve ORM (Object-Relational Mapping) kütüphaneleri ile entegrasyon sağlar. Bu entegrasyon sayesinde, veritabanı işlemleri struct'lar aracılığıyla kolayca yönetilebilir.

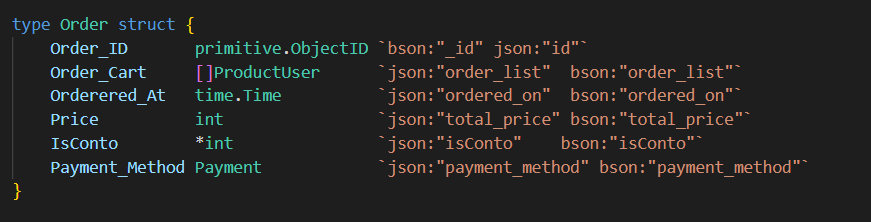
Bakım kolaylığı, kodun modülerliği ve anlaşılabilirlik açısından büyük öneme sahip olan models katmanı, tüm veri yapıları ve veritabanı işlemlerinin merkezi bir konumda bulunması sayesinde kodun bakımını ve güncellenmesini kolaylaştırır. Ayrıca, bu katman uygulamanın diğer katmanlarından bağımsız olduğu için kodun modüler ve yeniden kullanılabilir olmasını sağlar ve veritabanı şeması ile uygulama arasındaki ilişkiyi net bir şekilde göstererek yeni geliştiricilerin projeye dahil olmasını kolaylaştırır.



**Şekil 5.2** Örnek User Yapısı



**Şekil 5.3** Ürün Yapısı

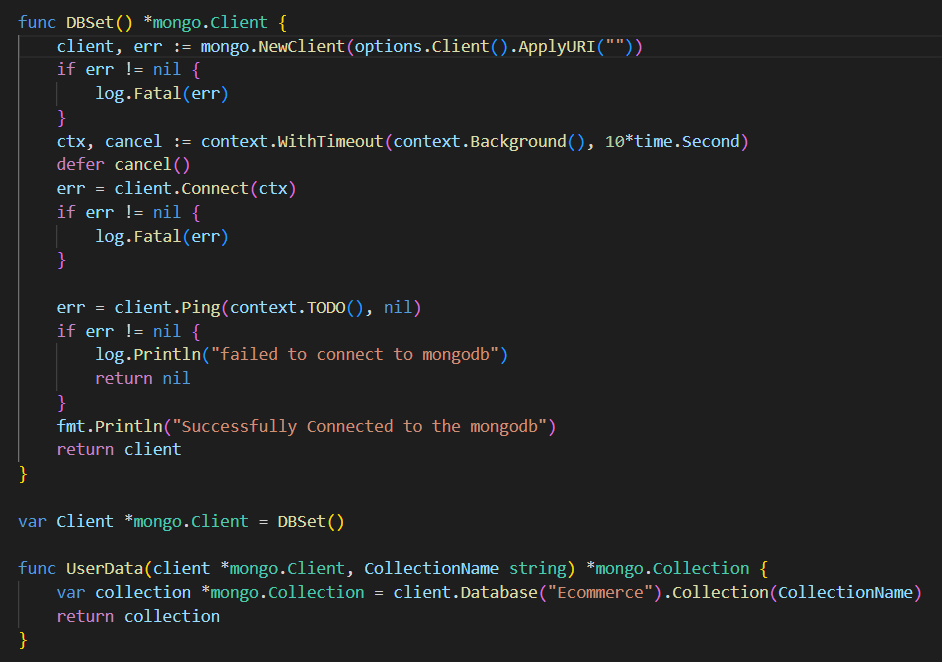


**Şekil 5.4** Sepet Yapısı

### 5.1.2 Database Katmanı

Database katmanı uygulamanın MongoDB veritabanı ile doğrudan etkileşimini yöneten önemli bir bileşendir. Bu katman, MongoDB bağlantılarının kurulumunu ve yönetimini, MongoDB sorgularının yürütülmesini sağlar

Database katmanı ayrıca, MongoDB sorgularının ve işlemlerinin yürütülmesini sağlar. Bu, MongoDB'ye doğrudan yapılan sorguların yanı sıra, ODM (Object-Document Mapping) kütüphaneleri kullanılarak daha yüksek seviyeli veri işlemlerinin gerçekleştirilmesini içerir.



**Şekil 5.5** Veri Tabanı Bağlantı İşlemi

### 5.1.3 Middleware Katmanı

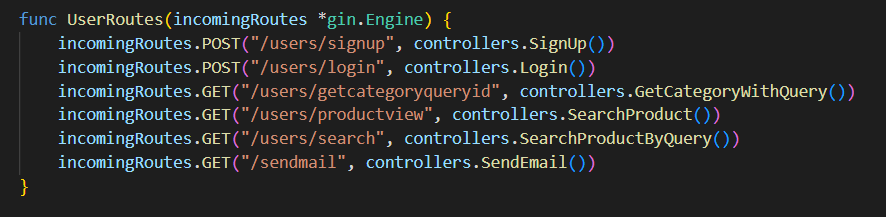
Middleware, bir web uygulamasında gelen istekleri işleyen ve yanıtları oluşturan bir ara yazılım katmanıdır. Bu katman, isteklerin sunucuya ulaşmadan önce veya sunucudan yanıt gönderilmeden önce geçtiği noktadır. Middleware, uygulamanın istekleri işleme sürecine ek işlevsellik eklemek için kullanılır ve genellikle bir dizi işlevin birleştirilmesiyle oluşturulur. Örneğin, yetkilendirme, güvenlik denetimleri, günlüğe kayıt tutma, oturum yönetimi, önbellekleme, sıkıştırma gibi işlevler middleware'ler aracılığıyla uygulanabilir. Middleware, istek nesnesi ve yanıt nesnesi üzerinde değişiklikler yapabilir veya isteği tamamen durdurabilir (örneğin, yetkilendirme başarısız olduğunda). Bu, uygulamanın temel işlevselliğini değiştirmeden veya karmaşıklaştırmadan önce, gelen isteklerin işlenmesi veya yanıtların oluşturulması için önemli bir noktadır. Middleware'ler genellikle web framework'leri tarafından desteklenir ve HTTP isteklerinin işlenme sürecinde birden çok middleware sıralanabilir veya yığılabilir. Bu, uygulamanın esnekliğini artırır ve farklı gereksinimlere uygun işlevselliğin kolayca eklenmesini sağlar.



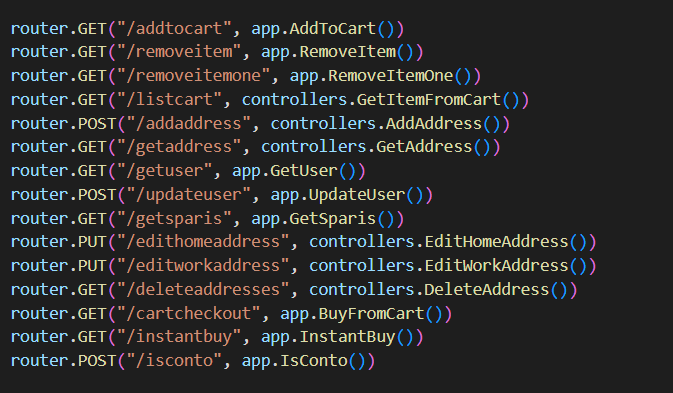
**Şekil 5.6** Middleware Yapısı

### 5.1.4 Routes Katmanı

Routes katmanı, bir Golang (Go) ve Gin web framework'ü kullanarak geliştirilen bir backend projesinde, gelen HTTP isteklerini belirli işleme yollarına yönlendirir. Bu katman, URL rotalarını tanımlar ve istekleri ilgili işlevlere yönlendirerek iş mantığını yürütür. HTTP metotlarına (GET, POST, PUT, DELETE vb.) uygun işlevleri atar, parametre yakalama ve doğrulama gibi işlemleri gerçekleştirir ve middleware'leri entegre ederek özellikler ekler. Bu sayede, uygulama istekleri doğru şekilde işleyebilir ve modüler bir yapı oluşturabilir.



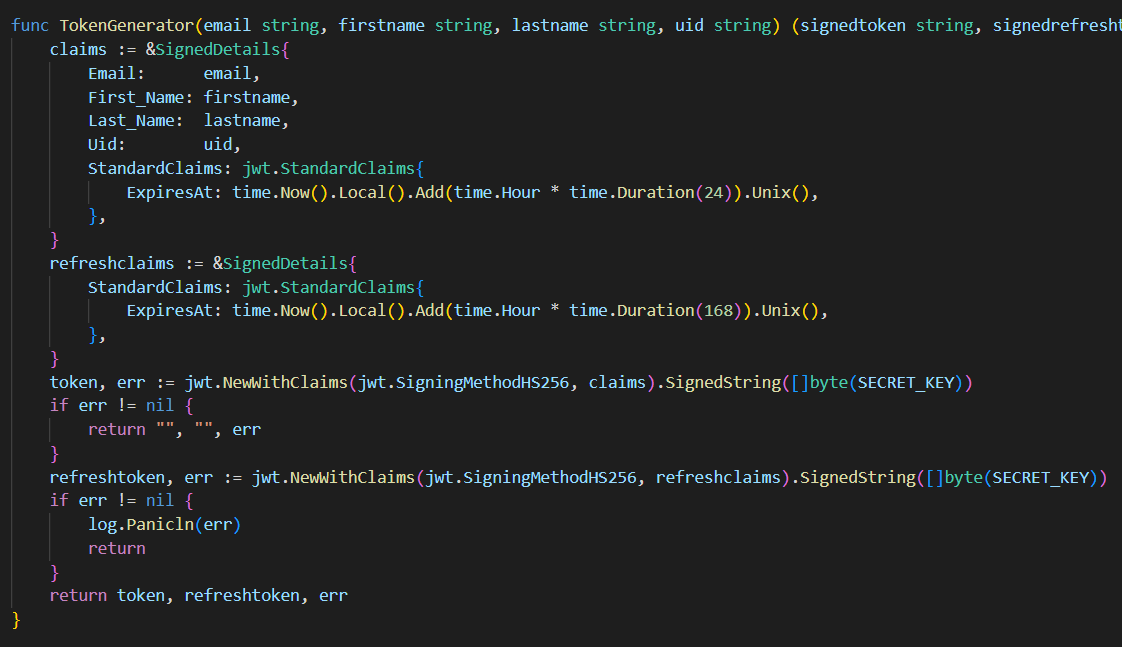
**Şekil 5.7** User Routes Yapısı Örneği



**Şekil 5.8** Kullanıcı ve Ürün işlemleri

### 5.1.5 JWT Katmanı

Tokens katmanı,JWT (JSON Web Token) gibi bir kimlik doğrulama ve yetkilendirme mekanizması kullanılarak uygulamanın güvenliğini sağlayan bir bileşendir. Bu katman, kullanıcıların kimlik doğrulaması yapmasını ve belirli kaynaklara erişim yetkilerini doğrulamasını sağlar. JWT token'ları, kullanıcıların oturumlarını yönetmek ve kimlik doğrulama bilgilerini taşımak için kullanılır. Bu token'lar, sunucu tarafında oluşturulur ve istemciye gönderilir. İstemci, bu token'ı kullanarak sunucudan korunan kaynaklara erişebilir. Tokens katmanı, JWT token'larının oluşturulması, doğrulanması ve yönetilmesinden sorumludur. Bu sayede, uygulama güvenliği artar ve yetkilendirme işlemleri sağlanır.



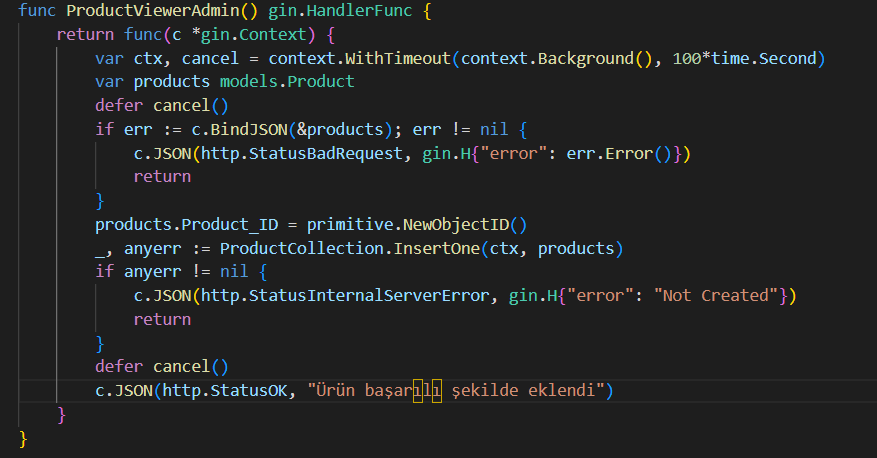
**Şekil 5.9** JWT tokenleri oluşturma



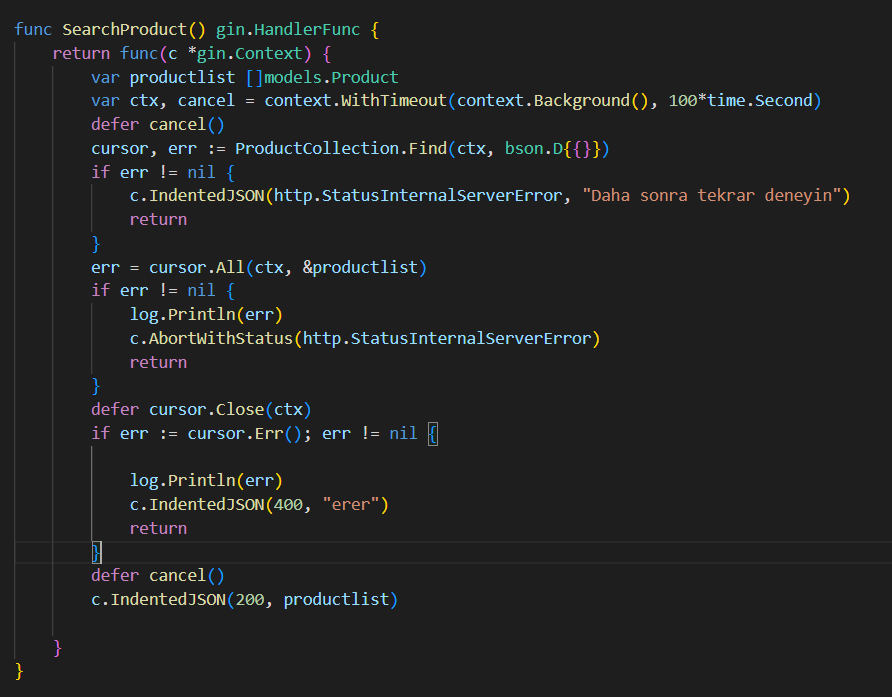
**Şekil 5.10** Token Doğrulama İşlemi

### 5.1.6 Controllers Katmanı

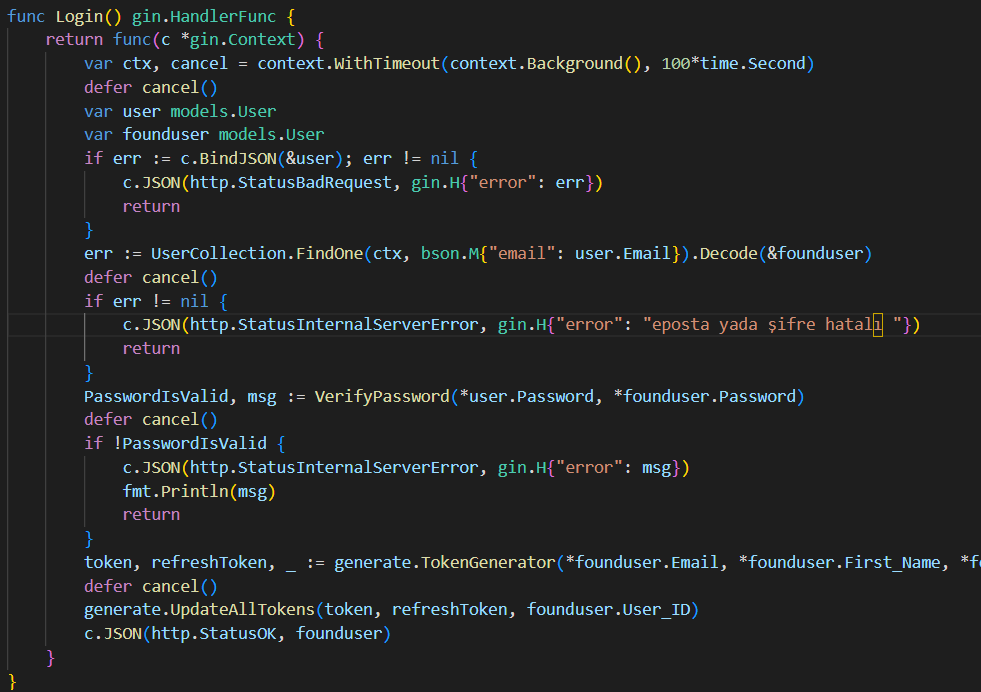
Controllers katmanı, bir web uygulamasında HTTP isteklerini işleyen ve iş mantığını uygulayan bir bileşendir. Bu katman genellikle model ve görünüm (model-view-controller veya MVC) tasarım desenindeki "controller" rolünü üstlenir. HTTP isteklerini alır, işler ve sonuçları HTTP yanıtları olarak döndürür. Kontrolörler genellikle uygulamanın iş mantığını uygular, gerekli veritabanı işlemlerini gerçekleştirir, verileri doğrular ve istemciye uygun yanıtlar üretir. Bu katman, uygulamanın farklı bölümleri arasında köprü görevi görür ve genellikle routes (rotalar) katmanı tarafından belirtilen HTTP isteklerini işlemek için kullanılır. Controllers katmanı, genellikle proje içindeki belirli işlevselliği veya kullanım durumlarını temsil eden farklı dosyalarda organize edilir. Bu katman, uygulamanın temel işlevselliğini sağlayarak, diğer katmanların (örneğin, models ve views) izole kalmasını sağlar ve kodun modülerliğini artırır.



**Şekil 5.11** Ürün ekleme



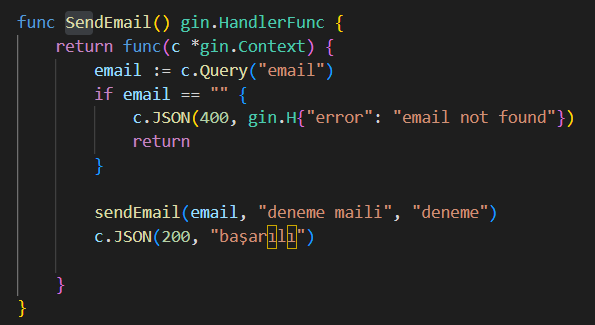
**Şekil 5.12** Tüm ürünleri listeleme



**Şekil 5.13** Login İşlemi Örneği



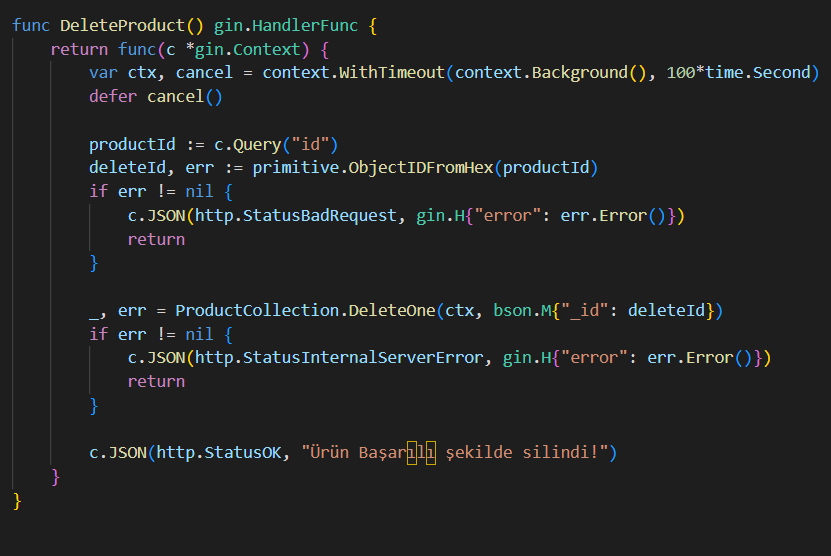
**Şekil 5.14** Hashleme ve doğrulama işlemi



**Şekil 5.15** Abone ol için mail gönderme işlemi



**Şekil 5.16** Adres güncelleme işlemi



**Şekil 5.17** Ürün Silme İşlemi

### 5.1.7 Static Klasörü

Eticaret sitelerinde "static katmanı" genellikle sabit içeriklerin barındırıldığı bir bölümü ifade eder. Bu bölümde genellikle ürün fotoğrafları, logolar, CSS dosyaları, JavaScript dosyaları gibi değişmeyen, sabit içerikler bulunur. Bu tür dosyalar, kullanıcıların tarayıcılarına indirilerek sitenin görsel ve işlevsel bileşenlerinin düzgün bir şekilde görüntülenmesini sağlar. Bu katman, sitenin dinamik kısmından ayrıdır ve genellikle sunucu tarafında depolanır. Dinamik içerikler ise kullanıcı etkileşimine bağlı olarak değişen ve veri tabanından alınan içeriklerdir. Örneğin, kullanıcıların ürünleri araması veya sepete ürün eklemesi gibi işlemler dinamik içeriklere örnektir.

## 5.2 Kavramların Açıklanması

### 5.2.1 Validator Nedir?

Go dilinde yazılmış ve sıklıkla kullanılan bir kütüphanenidir. Bu kütüphane, veri doğrulama (validation) işlemlerini gerçekleştirmek için tasarlanmıştır.

Genellikle web uygulamaları geliştirirken, kullanıcıların girdiği verilerin doğruluğunu kontrol etmek önemlidir. Örneğin, bir kullanıcının bir form aracılığıyla gönderdiği verilerin belirli bir formata uygun olması, belirli kurallara uygun olması gerekebilir. Bu kütüphane, bu tür doğrulama işlemlerini kolaylaştırır ve standart doğrulama işlemlerini hızlıca gerçekleştirmenizi sağlar.

Kütüphane, Go dilinin yapısal ve değer doğrulama işlemlerini destekler. Yani bir struct'ın (yapısal) veya bir değerin (değer) belirli kurallara uygun olup olmadığını kontrol edebilirsiniz. Örneğin, bir alanın boş olmaması, belirli bir uzunlukta olması, belirli bir desene uyması gibi kurallar belirleyebilir ve bu kurallara göre veri doğrulaması yapabilirsiniz.

Bu kütüphane, web uygulamalarında kullanıcı girdilerini işlerken hataları minimize etmek ve güvenilirlik sağlamak için oldukça yararlıdır.



**Şekil 5.18** Validator Türetme

### 5.2.2 Primitive Nedir?

go.mongodb.org/mongo-driver/bson/primitive adresinde bulunan açık kaynaklı kütüphane , MongoDB için Go programlama dilinde kullanılan bir pakettir. Bu paket, MongoDB'nin BSON (Binary JSON) belge formatındaki verileri Go programında kullanırken, BSON veri türlerini Go'nun temel veri türlerine dönüştürmeyi kolaylaştırır.

MongoDB'nin BSON formatı, belgelerin (documents) ve bu belgelerdeki verilerin binary (ikili) formatta temsil edilmesini sağlar. Bu verileri Go dilinde kullanırken, BSON veri tiplerini Go'nun temel veri tiplerine dönüştürmek gerekebilir. Örneğin, MongoDB'de ObjectId olarak bilinen benzersiz kimlikler, Go'da string veya byte dizisi olarak temsil edilemez. Bu durumda, go.mongodb.org/mongo-driver/bson/primitive paketi devreye girer.

Bu paket, BSON veri tiplerini Go'nun temel veri tiplerine dönüştürmek için bir dizi yöntem sağlar. Örneğin, MongoDB'de kullanılan ObjectId'i Go'nun primitive.ObjectID tipine dönüştürmek için bu paketi kullanabilirsiniz. Bu şekilde, MongoDB'den alınan verileri Go programınızda kullanırken, uygun veri tiplerine dönüştürme işlemlerini bu paket aracılığıyla gerçekleştirebilirsiniz. Bu da verilerin işlenmesini ve uygulama mantığınızı kolaylaştırır.

### 5.2.3 Bcrypt Nedir?

golang.org/x/crypto/bcrypt, Go programlama dilinde kullanılan bir kriptografi paketidir. Bu paket, bcrypt algoritması üzerinde çalışan bir kriptografik kütüphanedir.

bcrypt, genellikle parola hashleme için kullanılan güvenli bir algoritmadır. Parolaları güvenli bir şekilde depolamak için hashleme yöntemi olarak tercih edilir. Bu algoritma, saldırılara karşı ekstra koruma sağlamak için yavaş ve hesaplama yoğun bir şekilde çalışır. Bu nedenle, brute force (kaba kuvvet) saldırılarına karşı daha dayanıklıdır.

golang.org/x/crypto/bcrypt paketi, Go'da bcrypt algoritması kullanarak parola hashleme ve doğrulama işlemlerini gerçekleştirmeyi sağlar. Bu paket sayesinde, kullanıcıların parolalarını güvenli bir şekilde hashleyebilir ve veritabanında saklayabilirsiniz. Ayrıca, kullanıcıların girdikleri parolaları doğrulamak için de kullanabilirsiniz.

Bu paket, Go'nun standart kütüphanesinin bir parçası değildir, bu yüzden golang.org/x/ öneki altında yer alır ve harici bir bağımlılık olarak projenize eklenmelidir.

### 5.2.4 Gomail.v2 nedir ?

gopkg.in/gomail.v2, Go programlama dilinde kullanılan bir e-posta gönderme (SMTP) kütüphanesidir. Bu kütüphane, e-posta gönderme işlemlerini kolaylaştırmak için tasarlanmıştır.

Go programlarıyla e-posta göndermek için, genellikle SMTP (Simple Mail Transfer Protocol) sunucularıyla iletişim kurmak gereklidir. Ancak, bu işlemi düşük seviyede gerçekleştirmek zaman alıcı ve karmaşık olabilir. gopkg.in/gomail.v2 paketi, bu işlemi kolaylaştırır ve Go programcılarına basit bir API sağlar.

Bu kütüphane sayesinde, e-posta göndermek için gerekli olan temel işlemleri gerçekleştirebilirsiniz. Örneğin, e-posta başlıklarını, gövdesini ve eklerini belirleyebilir, alıcıları ve gönderen adresini belirleyebilir ve e-postayı SMTP sunucusuna iletebilirsiniz.

gopkg.in/gomail.v2 paketi, Go'nun harici bir bağımlılığıdır ve projenize eklenmesi gereken bir pakettir. E-posta gönderme işlemlerini Go ile gerçekleştirirken, bu paketi kullanarak işlemleri basitleştirebilir ve zaman kazanabilirsiniz.

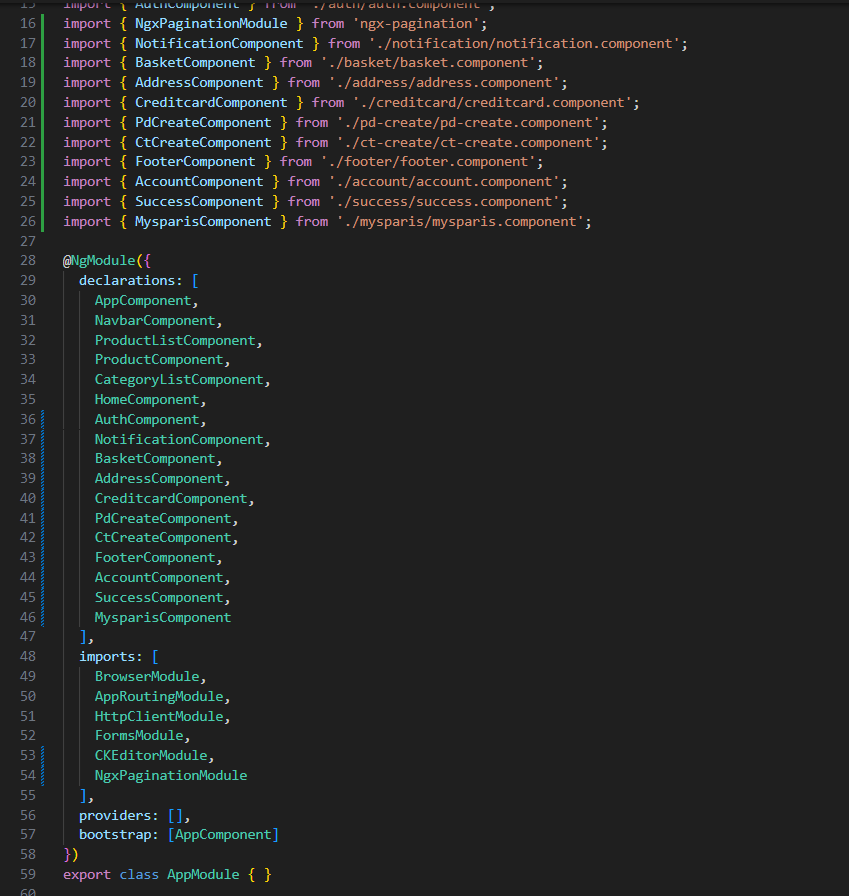
## 5.3 App Nedir ?

Angular projesinde "app" terimi, uygulamanın temel yapı taşını ifade eder. "App" genellikle "src" klasörü altında bulunur ve uygulamanın ana bileşenini, modüllerini, servislerini, yönlendiricilerini ve diğer önemli dosyalarını içerir. Bu yapı, Angular uygulamasının düzgün ve modüler bir şekilde organize edilmesini sağlar. "App" klasörü, projenin geri kalanını birbirine bağlayan ve uygulamanın çalışmasını mümkün kılan ana bileşenleri içerir. Bu nedenle, "app" klasörü, bir Angular projesinde merkezi bir öneme sahiptir ve projenin genel yapısını ve işleyişini belirler.

### 5.3.1 App.module.ts Nedir ?

Bu dosya, Angular uygulamalarının ana modülüdür. Uygulamanın yapılandırmasını ve bileşenlerin, servislerin, yönlendiricilerin ve diğer modüllerin dahil edildiği yerdir. İçe aktarma (imports) bölümü, diğer modüllerin dosyaya eklenmesini sağlar. Bildirimler (declarations) bölümünde uygulamanın bileşenleri belirtilir. Sağlayıcılar (providers) bölümünde servisler tanımlanır. Bootstrap bölümü ise uygulamanın başlangıç bileşenini belirtir ve uygulama bu bileşenle başlatılır.

app.module.ts dosyası, Angular uygulamasının yapılandırılması için kritik bir öneme sahiptir ve uygulama bileşenlerinin, servislerin ve modüllerin birleştirildiği merkezi bir noktadır. Bu dosya, uygulamanın düzgün bir şekilde çalışabilmesi için gerekli olan tüm yapı taşlarını bir araya getirir ve Angular'ın modüler yapısını destekler.



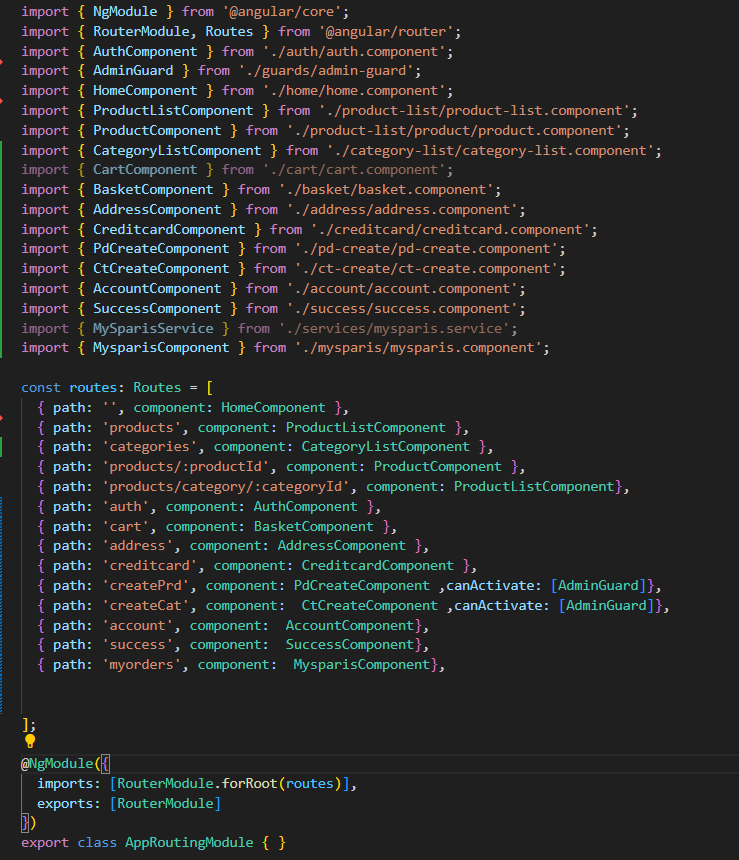
**Şekil 5.19** Kategori silme işlemi

### 5.3.2 App-routing.module.ts

Bu dosya, Angular uygulamasında URL rotalarını ve bu rotaların hangi bileşene yönlendirileceğini belirleyen bir dosyadır. Kullanıcının farklı sayfalar arasında gezinmesini sağlar. app-routing.module.ts dosyası, uygulamanın yönlendirme (routing) mekanizmasını yapılandırır ve bu sayede kullanıcıların farklı URL'lere erişerek ilgili bileşenleri görüntülemesine olanak tanır.

Bu dosya içinde, RouterModule ve Routes sınıfları kullanılarak rota tanımlamaları yapılır. Routes dizisi, her bir rotanın URL yolu ve yönlendirdiği bileşeni belirtir. RouterModule.forRoot(routes) ifadesi ise bu rotaları ana modüle bağlar. Ayrıca, bu dosyada bazen yönlendirme korumaları (route guards) gibi ek yönlendirme özellikleri de tanımlanabilir.

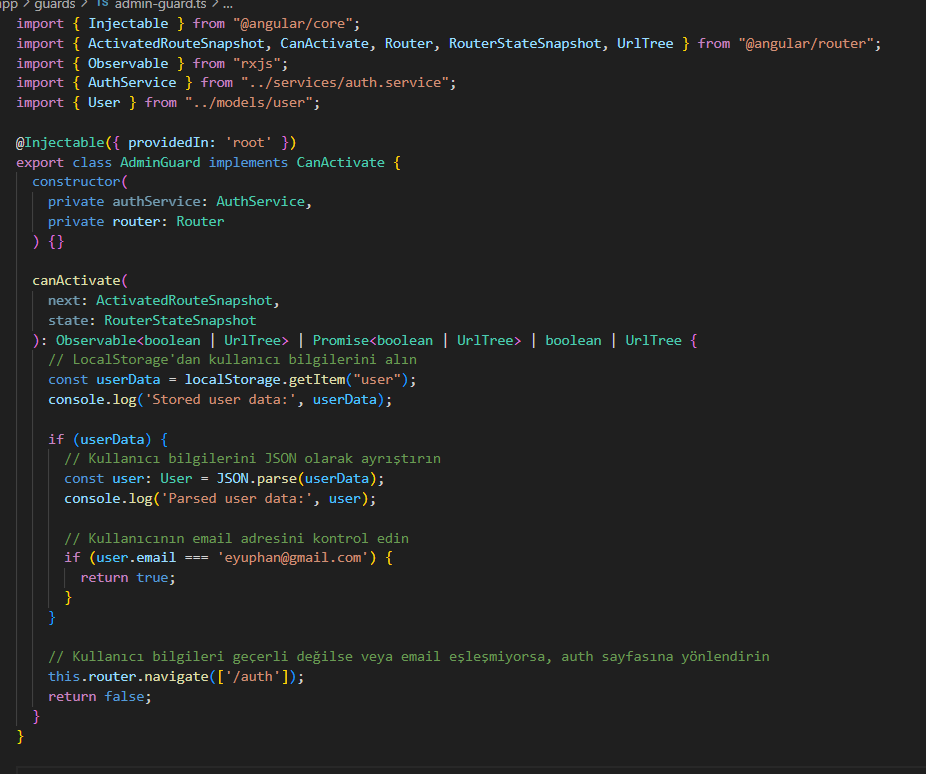
app-routing.module.ts dosyası, Angular uygulamasının kullanıcı dostu bir navigasyon deneyimi sunabilmesi için önemlidir ve uygulamanın farklı bileşenlerine erişimi düzenler.



**Şekil 5.20** App-routing.module.ts dosyası

### 5.3.3. Guards Klasörü

Angular uygulamasında rotalar veya bileşenlere erişim öncesi yapılan koruma kontrollerini içerir. Bu kontroller, kimlik doğrulama, yetkilendirme ve veri kontrolü gibi güvenlik önlemlerini uygulamak için kullanılır.



**Şekil 5.21** Admin.guard.ts dosyası

### 5.3.4 Angular Servis Dosyaları Nedir ?

Angular uygulamalarında, services.ts dosyaları, uygulamanın iş mantığını ve veri yönetimini merkezi bir yerde toplamak için kullanılan önemli bileşenlerdir. Servisler, bileşenler arasında veri paylaşımını ve bağımsız işlevsellikleri yönetmeyi sağlar. Angular'da servisler, uygulamanın her yerinde kullanılabilen, yeniden kullanılabilir ve modüler bir yapı sunar.

Servislerin Rolü ve Kullanımı:

Angular servisleri, veri alma, veri işleme ve bileşenler arasında veri paylaşma gibi işlevleri yerine getirir. Servisler genellikle HTTP istekleri yaparak sunucudan veri alır veya sunucuya veri gönderir. Bu sayede, veri yönetimi bileşenlerin dışına çıkarılarak daha temiz ve okunabilir bir kod yapısı elde edilir.

### 5.3.5 product.services.ts

product.service.ts dosyası, Angular uygulamasında ürünlerle ilgili işlemleri yönetmek için oluşturduğumuz bir servistir. Bu dosya, ürün verilerini almak, belirli bir ürünü ID ile getirmek, yeni ürünler oluşturmak, ürünleri silmek ve ürün resimlerini kaydetmek gibi işlevleri içerir

**5.3.5.1Ürünleri getirme**

getProducts fonksiyonu, belirli bir kategoriye ait ürünleri veya tüm ürünleri alır. HTTP GET [http://localhost:8000/"users/productview](http://localhost:8000/%22users/productview) adresine istek yaparak ürün verilerini alır ve bu veriler map operatörü ile işler . Ayrıca, sonuçlar tap operatörü ile konsola yazdırılır ve delay operatörü ile gecikme eklenir



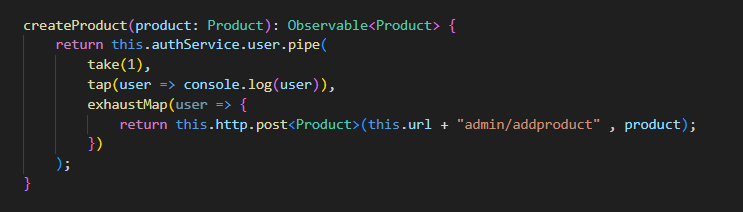
**Şekil 5.22** GetProducts methodu

**5.3.5.2 Ürünü ID İle Getirme**: getProductById methodu, belirli bir ID'ye sahip ürünü alır. HTTP GET isteği ile ürün verisi alınır ve delay operatörü ile gecikme eklenir.



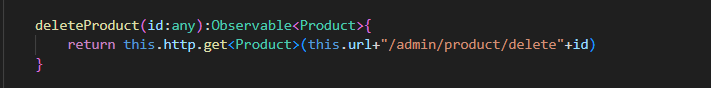
**Şekil 5.23** GetProductById methodu

**5.3.5.3 Ürün Oluşturma**: createProduct yöntemi, yeni bir ürün oluşturur. İlk olarak, authService.user kullanılarak kullanıcı bilgisi alınır ve bu bilgi take ve exhaustMap operatörleri ile işlenir. Daha sonra, HTTP POST isteği ile yeni ürün verisi sunucuya gönderilir.



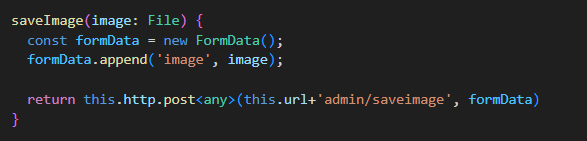
**Şekil 5.24** CreateProduct methodu

**5.3.5.4 Ürün Silme**: deleteProduct methodu, belirli bir ID'ye sahip ürünü siler. HTTP GET isteği ile silme işlemi gerçekleştirilir.



**Şekil 4.25** DeleteProduct methodu

**5.3.5.5 Resim Kaydetme**: saveImage methodu, bir ürüne ait resmi sunucuya kaydeder. FormData kullanılarak resim verisi hazırlanır ve HTTP POST isteği ile sunucuya gönderilir.



**Şekil 5.26** SaveImage methodu

### 5.3.6 address.services.ts

address.service.ts, Angular uygulamamızda adreslerle ilgili işlemleri yönetmek için oluşturulmuş bir servistir. Bu servis, HTTP isteklerini kullanarak adres ekleme, alma, düzenleme ve silme işlemlerini gerçekleştirir. RxJS kütüphanesi ve operatörleri sayesinde bu işlemler asenkron olarak yönetilir ve veri akışları daha etkili bir şekilde kontrol edilir.

#### 5.3.6.1 Constructor ve Token Yönetimi

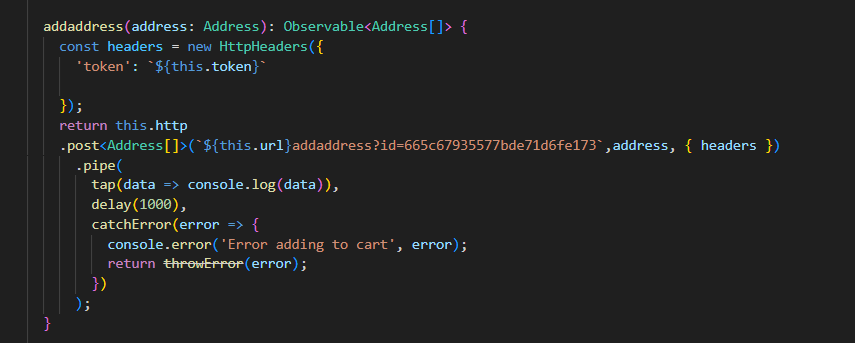
Servis sınıfının yapıcısı (constructor), HttpClient'i enjekte eder ve kullanıcı token'ını localStorage'dan alarak servisin token değişkenine atar.



**Şekil 5.27** Constructor

#### 5.3.6.2 addaddress Metodu

Bu metod, bir adres eklemek için HTTP POST isteği gönderir. İstek başlıklarında (headers) token'ı içerir.



**Şekil 5.28** Addaddress Metodu

#### 5.3.6.3 gettoaddress Metodu

Bu metod, adresleri almak için HTTP GET isteği gönderir.Delay() ile 1000 saniye gecikme verildikten sonra gelen adress bilgisini istemciye gönderir.catchError ile hatalar yakalanır



**Şekil 5.29** GetToaddress Metodu

#### 5.3.6.4 gettoaddress Metodu

Bu metod, mevcut bir adresi düzenlemek için HTTP PUT isteği gönderir

#### 5.3.6.5 removetoaddress Metodu

Bu metod, bir adresi silmek için HTTP GET isteği gönderir.



**Şekil 5.30** Gettoadress ve Removetoadress metodu

#### 5.3.6.6 AuthService

AuthService, Angular uygulamanızda kullanıcı kimlik doğrulama işlemlerini yönetmek için oluşturulan bir servistir. Bu servis, kullanıcı kaydı, giriş, otomatik giriş, çıkış ve kullanıcı oturum durumunun yönetilmesini sağlar. Ayrıca, RxJS kütüphanesi ve operatörleri kullanılarak HTTP istekleri ve hata yönetimi işlemleri gerçekleştirilir

#### 5.3.6.7 register Metodu

Bu metod, kullanıcı kaydı işlemi için HTTP POST isteği gönderir. Kullanıcı bilgileri, istek gövdesinde (body) gönderilir ve yanıt başarılı olduğunda, kullanıcı bilgileri işlenir.

#### 5.3.6.8 logout Metodu

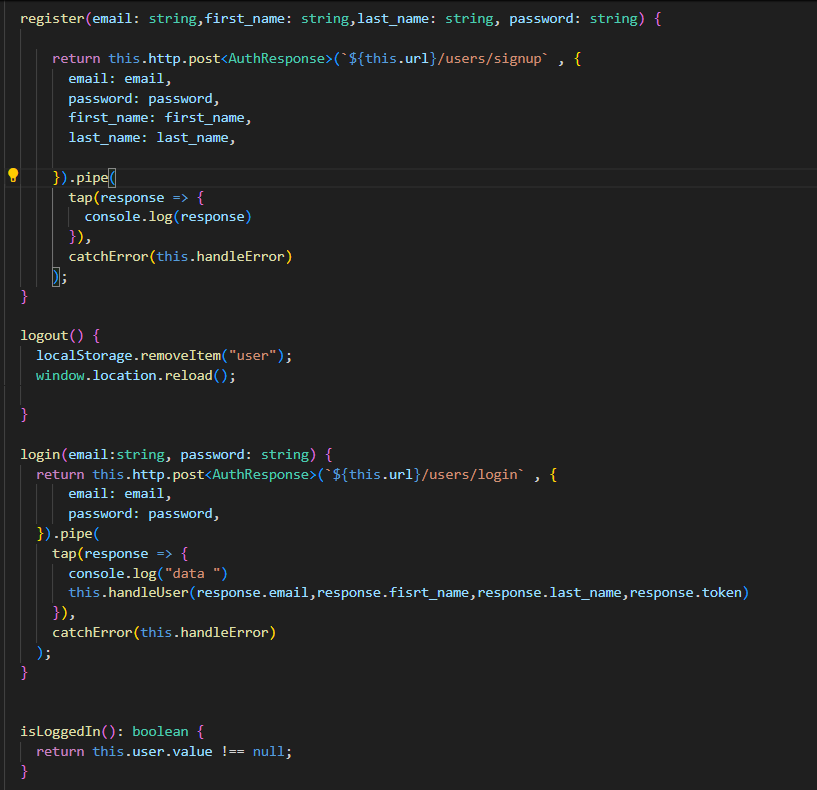
Bu metod, kullanıcının oturumunu sonlandırır. localStorage'da saklanan kullanıcı bilgilerini kaldırır ve sayfayı yeniden yükler.

#### 5.3.6.9 login Metodu

Bu metod, kullanıcı giriş işlemi için HTTP POST isteği gönderir. Kullanıcı bilgileri, istek gövdesinde gönderilir ve yanıt başarılı olduğunda, kullanıcı bilgileri işleni

#### 5.3.6.10 isLoggedIn Metodu

Bu metod, kullanıcının oturum açmış olup olmadığını kontrol eder. BehaviorSubject'in değerini kontrol ederek kullanıcı oturum durumunu belirler.



**Şekil 5.31** Register , Logut , Login , IsLoggedIn ve removetoadress Metodları

#### 5.3.6.11 autoLogin Metodu

Bu metod, tarayıcıda saklanan kullanıcı bilgilerini kullanarak otomatik giriş işlemini gerçekleştirir. localStorage'dan kullanıcı bilgilerini alır ve geçerli bir token varsa kullanıcı oturumunu günceller

#### 5.3.6.12 handleError Metodu

Bu özel metod, HTTP hatalarını yönetir. Hata mesajlarını belirli durumlara göre düzenler ve uygun hata mesajını döner.



**Şekil 5.32** AutoLogin ve HandleError Metodları

### 5.3.7 CartService

CartService sınıfı, Angular uygulamanızda alışveriş sepeti işlemlerini yönetmek için oluşturulan bir servistir . Bu servis, sepetin güncellenmesi, sepetten ürün ekleme, çıkarma, listeleme ve satın alma gibi işlemleri sağlar. Ayrıca, RxJS kütüphanesi ve operatörleri kullanılarak HTTP istekleri ve hata yönetimi işlemleri gerçekleştirilir.

**5.3.7.1 addtocart Metodu**

Bu metod, sepete ürün eklemek için HTTP GET isteği gönderir. Kullanıcı token'ı istek başlıklarına eklenir ve sepetin güncellenmesi için BehaviorSubject güncellenir.

**5.3.7.1 getCartUpdatedListener Metodu**

Bu metod, sepetin güncellenme durumunu gözlemlemek için kullanılan Observable'ı döner.

**5.3.7.2 gettocart Metodu**

Bu metod, sepetin içeriğini almak için HTTP GET isteği gönderir. Kullanıcı token'ı istek başlıklarına eklenir ve sepetin içeriği döner.

**5.3.7.3 buytocart Metodu**

Bu metod, sepeti satın almak için HTTP GET isteği gönderir. Kullanıcı token'ı istek başlıklarına eklenir ve sepetin içeriği döner.

**5.3.7.4 removetocart Metodu**

Bu metod, sepetten bir ürünü çıkarmak için HTTP GET isteği gönderir. Kullanıcı token'ı istek başlıklarına eklenir ve sepetin güncellenmesi için BehaviorSubject güncellenir.

**5.3.7.5 removetocartone Metodu**

Bu metod, sepetten bir ürün miktarını (örneğin birden fazla ürün varsa sadece birini) çıkarmak için HTTP GET isteği gönderir. Kullanıcı token'ı istek başlıklarına eklenir ve sepetin güncellenmesi için BehaviorSubject güncellenir.



**Şekil 5.33** Cart.service.ts



**Şekil 5.34** Cart.service.ts

**5.3.7.6 CategoryService Dosyası**

CategoryService sınıfı, Angular uygulamanızda kategori işlemlerini yönetmek için oluşturduğumuz servistir. Bu servis, kategorileri listeleme, kategori oluşturma, kategori silme ve kategori resimlerini kaydetme gibi işlemleri sağlar

**5.3.7.7 getCategories Metodu**

Bu metod, kategorileri listelemek için HTTP GET isteği gönderir ve gelen veriyi Category tipine dönüştürür.

**5.3.7.8 getCategoryByid Metodu**

Bu metod, belirli bir kategori ID'sine göre kategoriyi getirmek için HTTP GET isteği gönderir.

**5.3.7.9 createCategory Metodu**

Bu metod, yeni bir kategori oluşturmak için HTTP POST isteği gönderir.

**5.3.7.10 saveCategoryImage Metodu**

Bu metod, kategori resmi kaydetmek için HTTP POST isteği gönderir. Form verilerini FormData kullanarak gönderir.

**5.3.7.11 deleteCategory Metodu**

Bu metod, belirli bir kategori ID'sine göre kategoriyi silmek için HTTP GET isteği gönderir.



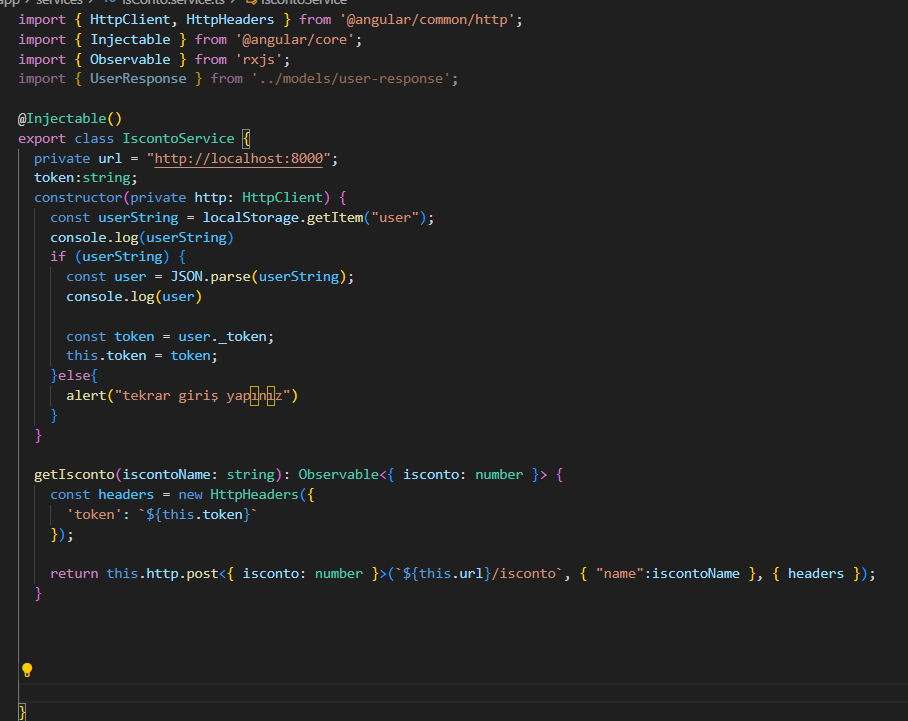
**Şekil 5.35** Category.service.ts

**5.3.8 IscontoService**

IscontoService, Angular uygulamanızda iskontolarla (indirimlerle) ilgili işlemleri yönetmek için oluşturulan bir servistir. Bu servis, belirli bir iskontonun değerini almak için HTTP POST isteği gönderir ve kullanıcının giriş yapıp yapmadığını kontrol eder.

**5.3.8.1 getIsconto Metodu**

Bu metod, belirli bir iskontonun değerini almak için HTTP POST isteği gönderir ve iskontonun değerini döner. Ayrıca, istek sırasında gerekli olan token bilgisini HTTP header'a ekler.



**Şekil 5.36** GetIsconto.service.ts

**5.3.8.2. MyOrderService**

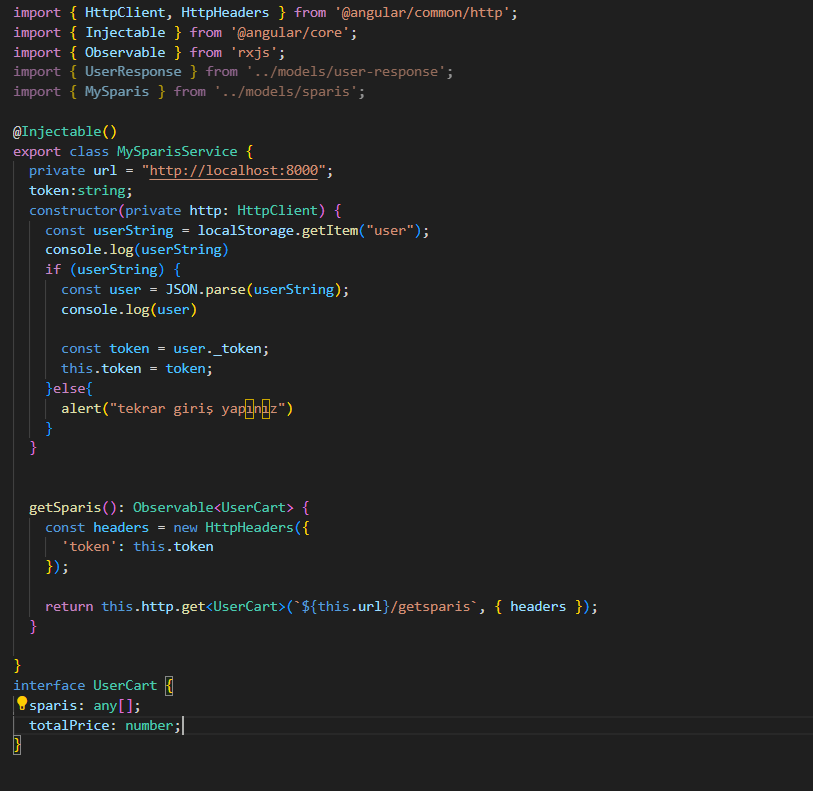
MyOrderService, Angular uygulamanızda sipariş (sipariş) işlemlerini yönetmek için kullanılan bir servistir. Bu servis, belirli bir kullanıcının sparis bilgilerini almak için HTTP GET isteği gönderir ve kullanıcının giriş yapıp yapmadığını kontrol eder

**5.3.8.3 getSparis Metodu**

Bu metod, belirli bir kullanıcının sparis bilgilerini almak için HTTP GET isteği gönderir ve sparis bilgilerini döner. Ayrıca, istek sırasında gerekli olan token bilgisini HTTP header'a ekler.

**5.3.8.4 UserCart Arayüzü**

Bu arayüz, kullanıcıya ait sparis bilgilerini ve toplam fiyatı tanımlar.



**Şekil 5.37** Myorder.service.ts

**5.3.9 Compenent Klasörü Nedir**

Angular uygulamasında kullanılan bileşenlerin yer aldığı bir klasördür. Her bileşen, kendi HTML şablonu, CSS stilini ve TypeScript kodunu içerir. Bu klasördeki bileşenler, kullanıcı arayüzünü oluşturmak için kullanılır ve farklı görevleri yerine getirebilir. Örneğin, bir başlık bileşeni, bir form bileşeni veya bir liste bileşeni gibi farklı bileşenler bulunabilir. Bu bileşenler, uygulamanın farklı parçalarını oluşturarak modüler bir yapı oluşturmayı sağlar.

**5.3.10 NavbarComponent**

NavbarComponent, Angular uygulamamızda gezinme çubuğunu yönetmek için kullanılan bir bileşendir. Bu bileşen, kullanıcı oturumunu ve sepetteki ürünleri kontrol eder.

**5.3.10.1 ngOnInit Yöntemi**

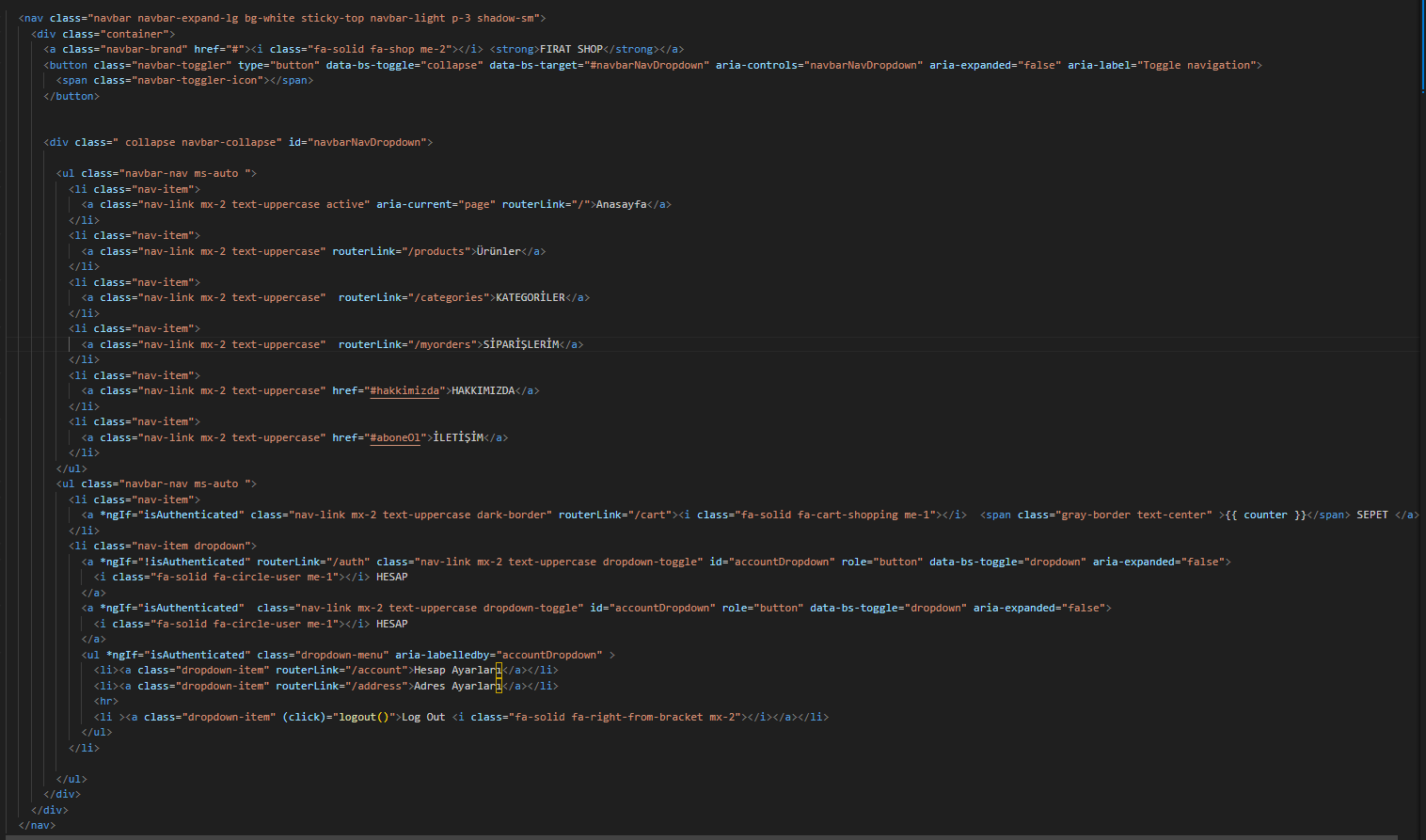
Bu yöntem, bileşen yüklendiğinde çalıştırılır. Kullanıcının oturum durumunu kontrol eder ve sepet bilgilerini günceller.5

**5.3.10.2 updateCart Yöntemi**

Bu yöntem, sepet bilgilerini günceller ve sepetin içeriğini görüntüler



**Şekil 5.38** Navbar.component.ts



**Şekil 5.39** Navbar.component.html

**5.3.11 HomeComponent**

HomeComponent, Angular uygulamamızda ana sayfanın içeriğini yönetmek için kullanılan bir bileşendir. Bu bileşen, ürünleri listelemek ve e-posta göndermek gibi işlevleri içerir. Aşağıda bu bileşenin detaylı açıklamaları ve işlevleri bulunmaktadır.

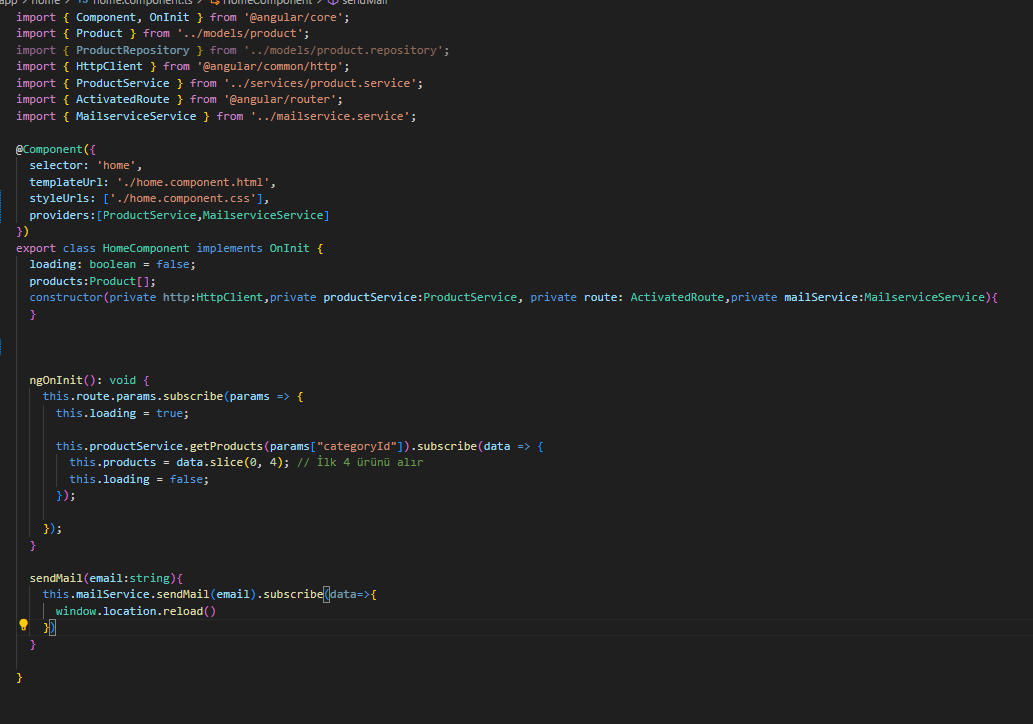
HomeComponent sınıfı, ürünleri listelemek ve e-posta göndermek için ProductService ve MailserviceService servislerini kullanır.

**5.3.11.1 ngOnInit Yöntemi**

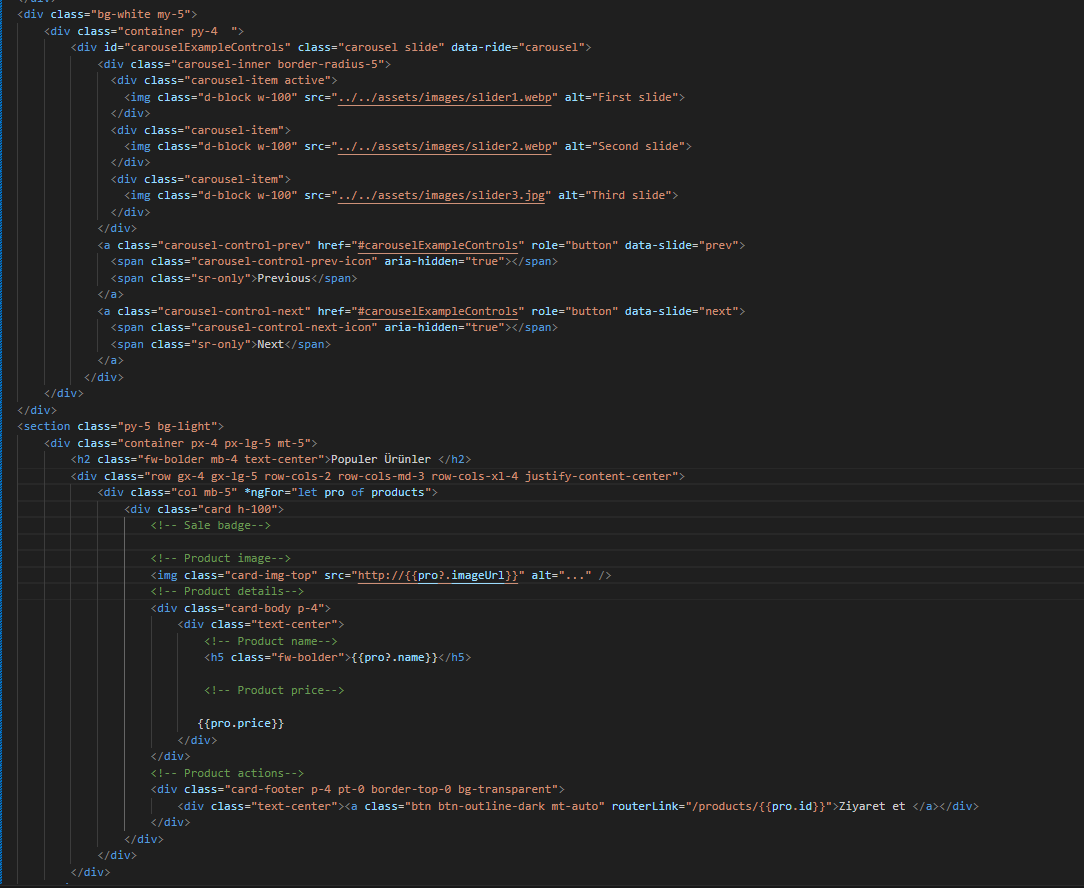
Bu yöntem, bileşen yüklendiğinde çalıştırılır ve URL parametrelerini izler. ProductService kullanarak ürünleri alır ve ilk 4 ürünü görüntüler.

**5.3.11.2 sendMail Yöntemi**

Bu yöntem, e-posta göndermek için MailserviceService kullanır ve başarı durumunda sayfayı yeniden yükler. Kullanıcının token bilgisini içeren HTTP başlıklarını ekler.



**Şekil 5.40** Home.component.ts



**Şekil 5.41**Home.component.html

**5.3.12 PdCreateComponent**

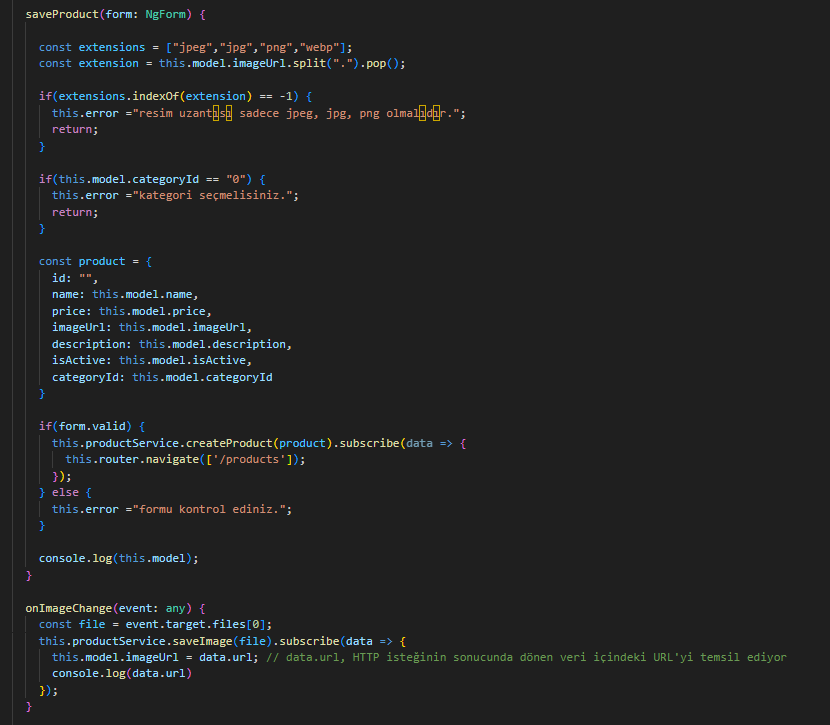
PdCreateComponent, Angular uygulamamızda yeni ürün oluşturmak için kullanılan bir bileşendir. Bu bileşen, kategori bilgilerini almak, form doğrulaması yapmak ve yeni ürünü kaydetmek gibi işlevleri içerir.

**5.3.12.1 saveProduct Yöntemi**

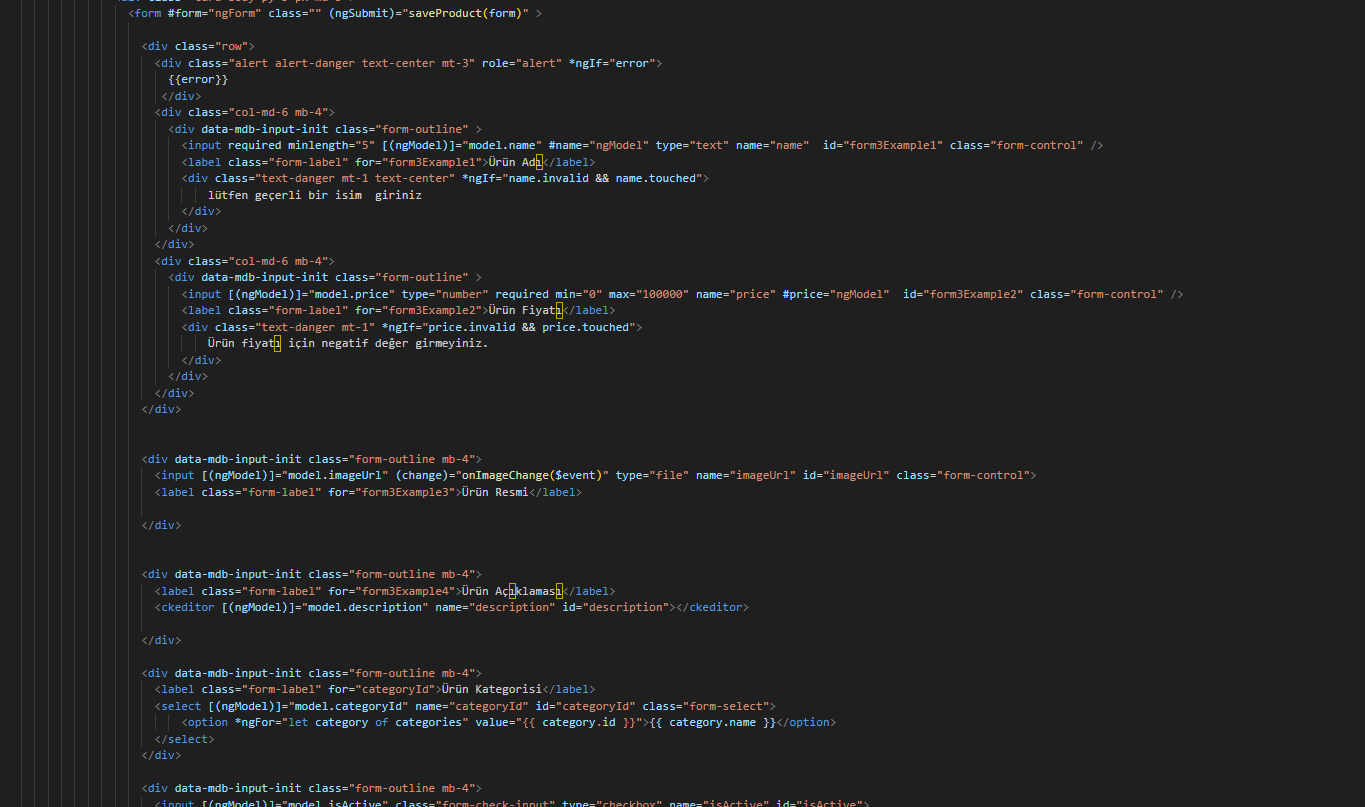
saveProduct yöntemi, form doğrulaması yapar ve yeni ürünü kaydeder. Form geçerli değilse hata mesajı gösterir.

**5.3.12.2 onImageChange Yöntemi**

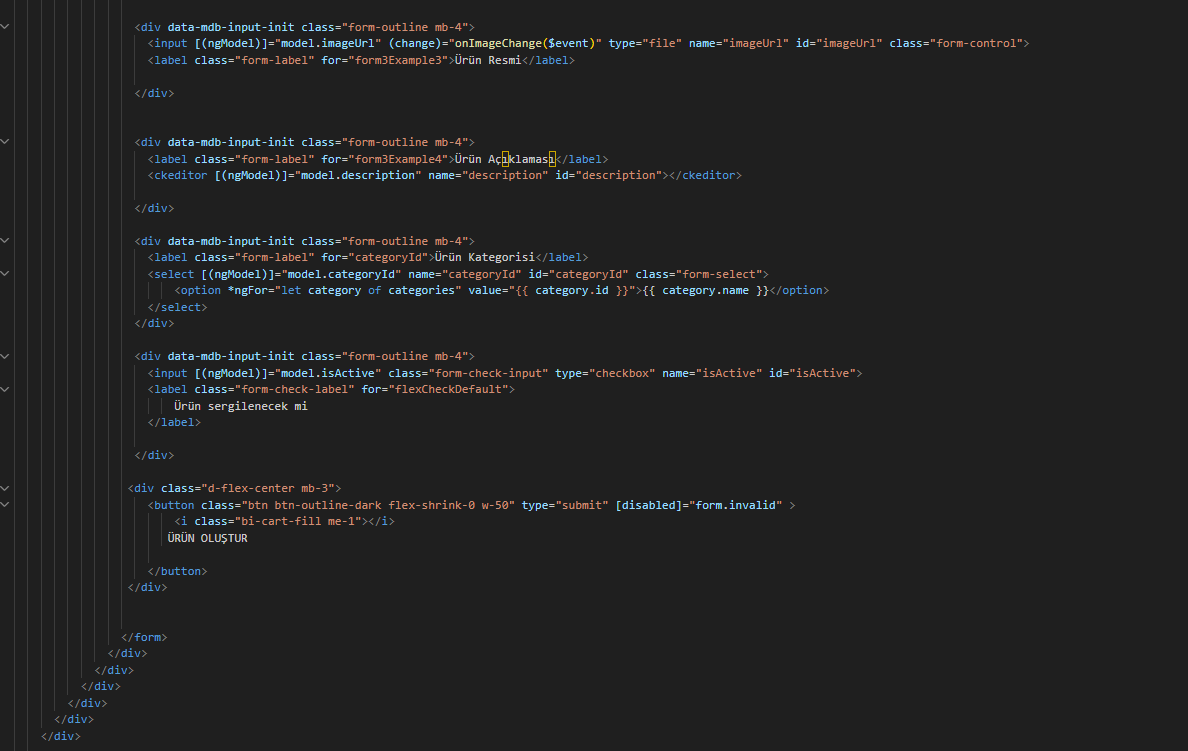
onImageChange yöntemi, resim dosyasını yükler ve HTTP isteği sonucunda dönen URL'yi model.imageUrl değişkenine ata



**Şekil 5.42** pdcreate.component.cs

****

**Şekil 5.43** pdcreate.component.html

****

**Şekil 5.44** pdcreate.component.html

**5.3.13 ProductListComponent**

ProductListComponent, Angular uygulamamızda ürünlerin listesini gösteren bir bileşendir. Bu bileşen, ürünleri kategoriye göre filtreler, sayfalama yapar ve kategori adlarını yükler.

**5.3.13.1 categories Nesnesi ve getCategory Yöntemi**

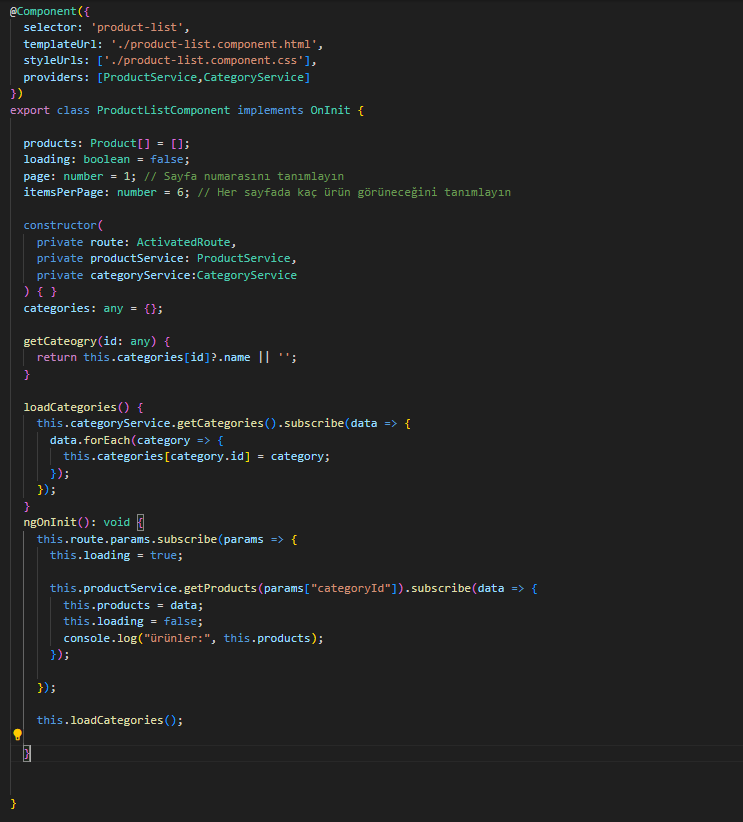
categories nesnesi, kategori bilgilerini saklamak için kullanılır. getCategory yöntemi, kategori ID'sine göre kategori adını döner

**5.3.13.2 loadCategories Yöntemi**

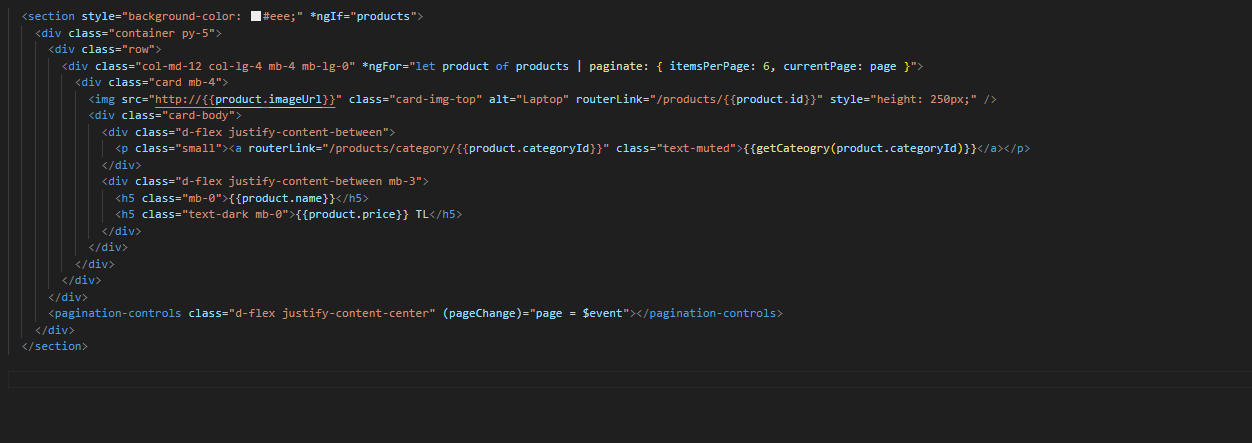
loadCategories yöntemi, kategori bilgilerini yükler ve categories nesnesine atar.

**5.3.13.3 ngOnInit Yöntemi**

ngOnInit yöntemi, bileşen yüklendiğinde çalışır. URL parametrelerine abone olarak kategoriye göre ürünleri yükler ve loadCategories yöntemini çağırarak kategorileri yükle

****

**Şekil 5.45** productlist.component.cs

****

**Şekil 5.46** productlist.component.html

**5.3.14 ProductComponent**

ProductComponent'i oluşturur. Bu bileşen, belirli bir ürünün detaylarını göstermek için kullanılır ve ayrıca ilgili ürün kategorisindeki diğer ürünleri listeler. İşlevselliği şu şekildedir:

categoryName ve categoryId değişkenleri, ürünün kategori adını ve kimliğini saklar.

product değişkeni, görüntülenen ürünün detaylarını tutar.

products dizisi, aynı kategorideki diğer ürünlerin listesini tutar.

isAuthenticated değişkeni, kullanıcının oturum açmış olup olmadığını belirtir.

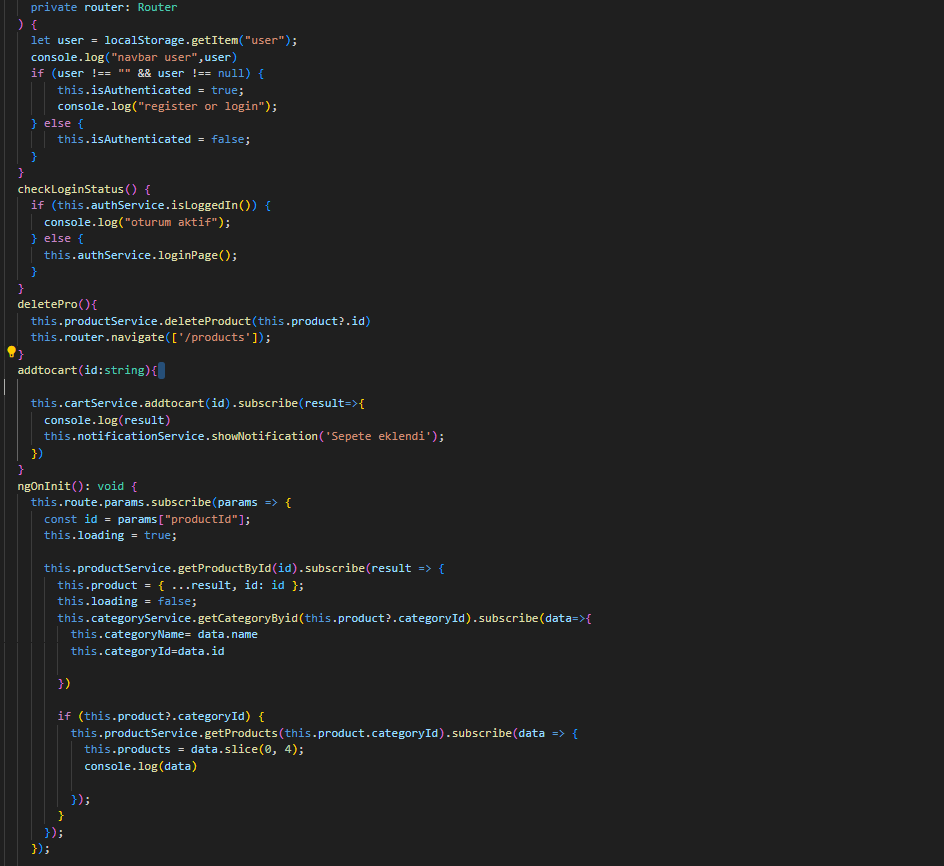
Bileşenin constructor metodu, bileşen oluşturulduğunda çalışır ve kullanıcının oturum durumunu kontrol eder. Eğer kullanıcı oturum açmışsa, isAuthenticated değeri true olarak ayarlanır.

checkLoginStatus metodu, kullanıcının oturum açma durumunu kontrol eder. Eğer oturum açılmışsa, konsola "oturum aktif" yazdırır; aksi takdirde, kullanıcıyı giriş yapma sayfasına yönlendirir.

deletePro metodu, mevcut ürünü siler ve kullanıcıyı ürünler sayfasına yönlendirir.

addtocart metodu, belirli bir ürünü sepete ekler ve bunu kullanıcıya bildirim olarak gösterir.

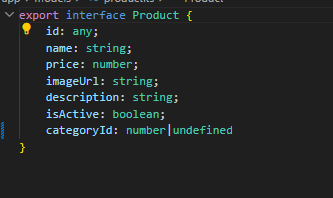
ngOnInit metodu, bileşenin başlangıç durumunu ayarlar. Bu metod, URL'den alınan productId parametresine göre ilgili ürünün detaylarını ve aynı kategorideki diğer ürünleri yükler.

****

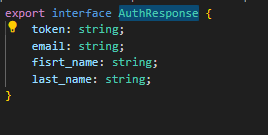
**Şekil 5.47**productlist.component.cs

**5.3.15** **Models Klasörü**

Bu klasör, Angular projelerinde veri yapısını ve veri nesnelerini tanımlamak için kullanılır. Bu klasör içerisinde, projenin kullanacağı veri modelleri veya sınıflar yer alır. Veri modelleri, uygulamanın kullandığı verilerin yapısını ve özelliklerini temsil eder. Örneğin, bir kullanıcı modeli, bir araç modeli veya bir ürün modeli gibi veri nesneleri bu klasör altında tanımlanabilir. Bu modeller, veri alışverişlerinde kullanılır. API çağrılarında kullanılan veri formatını tanımlamak için kullanılabilir ve bileşenler arasında veri paylaşımını kolaylaştırır. Models klasörü, projenin veri katmanını ve veri yönetimini daha organize bir şekilde sağlamak için kullanılır.

****

***Şekil 5.48*** *Product.ts*

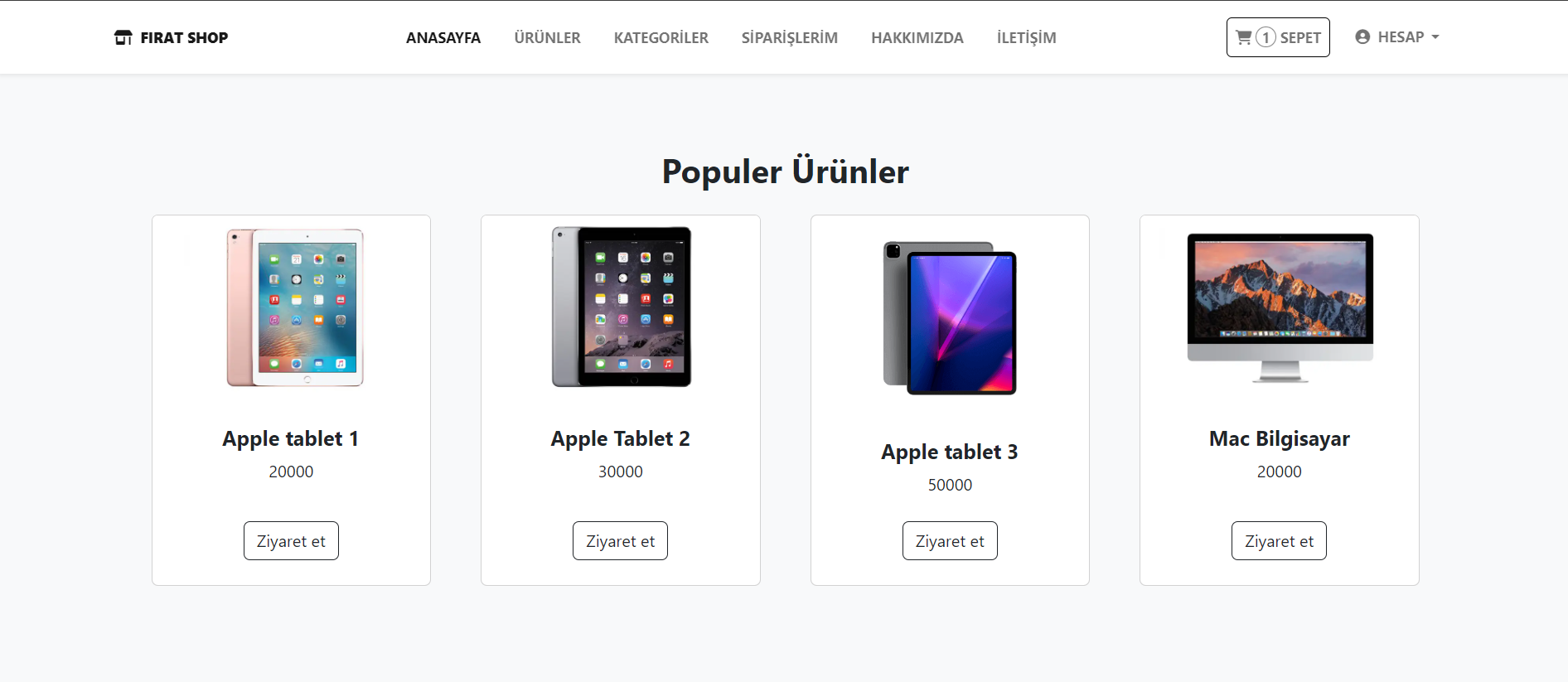
****

***Şekil 5.49***AuthResponse.ts

**5.4 Ekranlar**

****

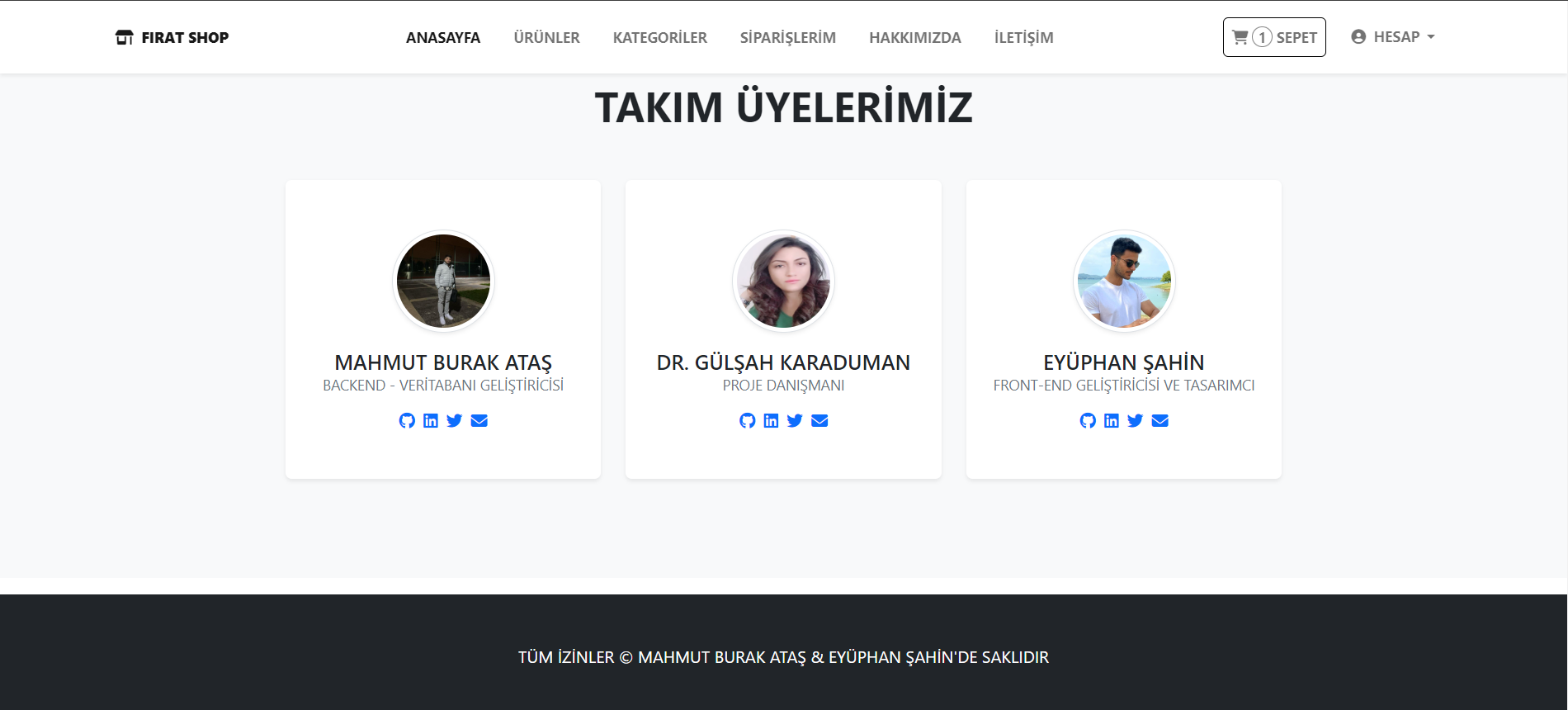
**Şekil 5.50** Anasayfa

****

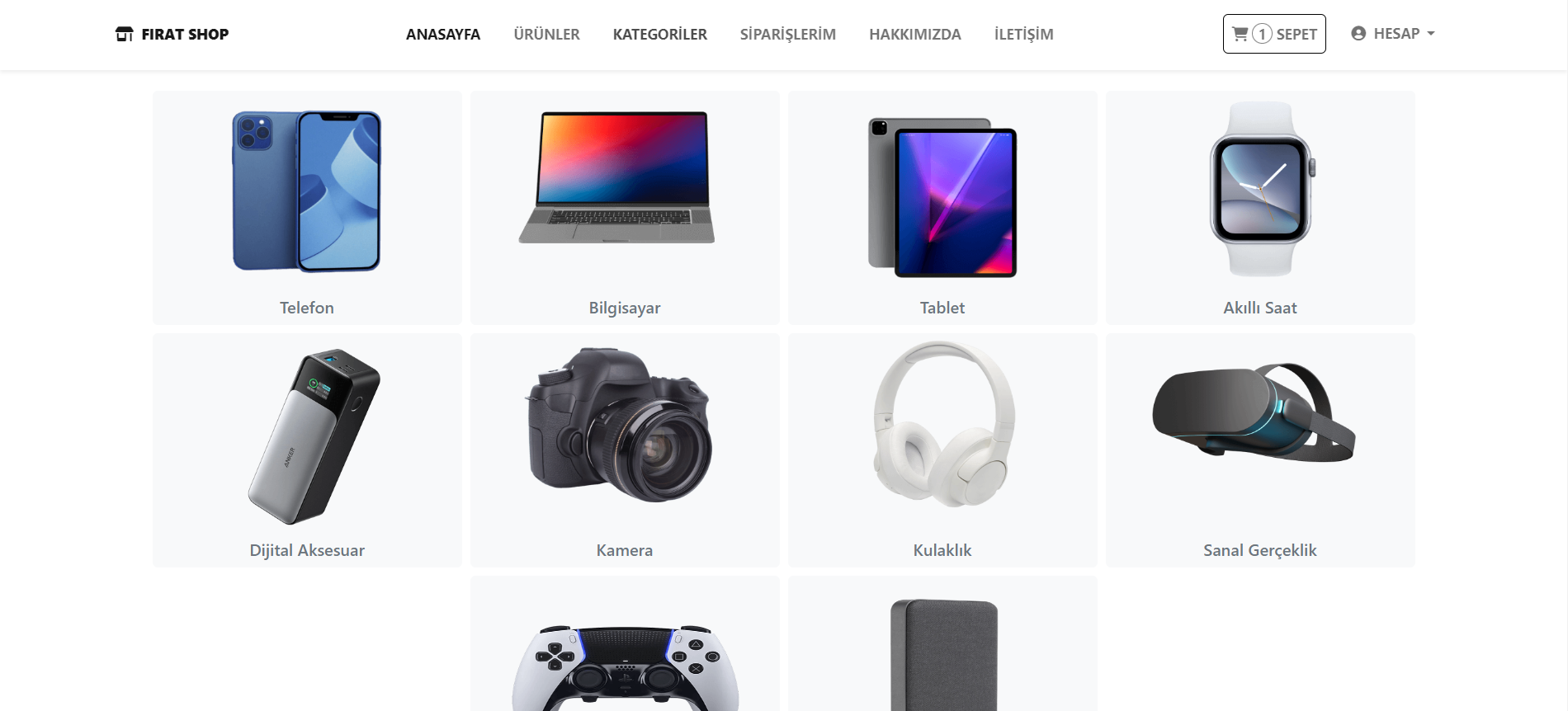
**Şekil 5.51** Popüler ürünler

****

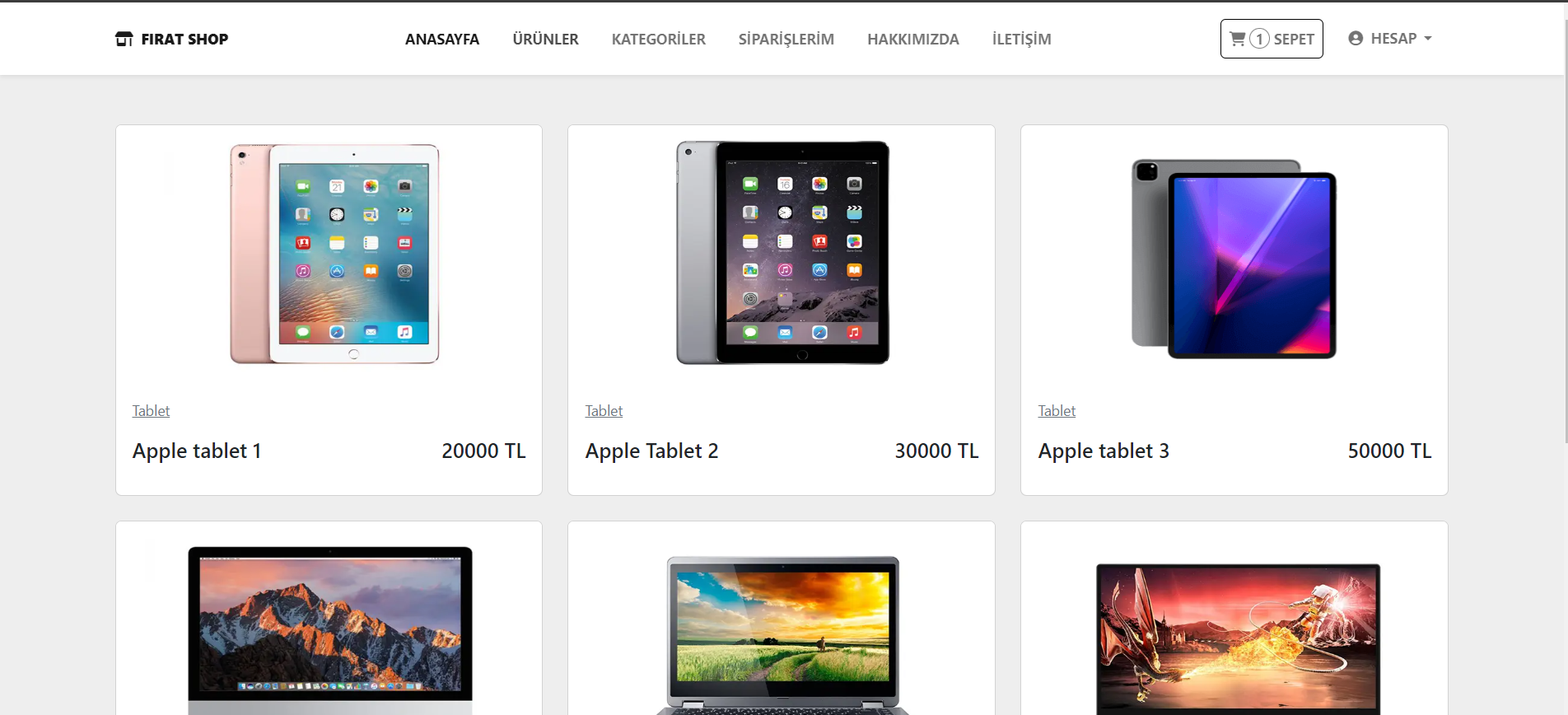
**Şekil 5.52** Haber Bülteni

****

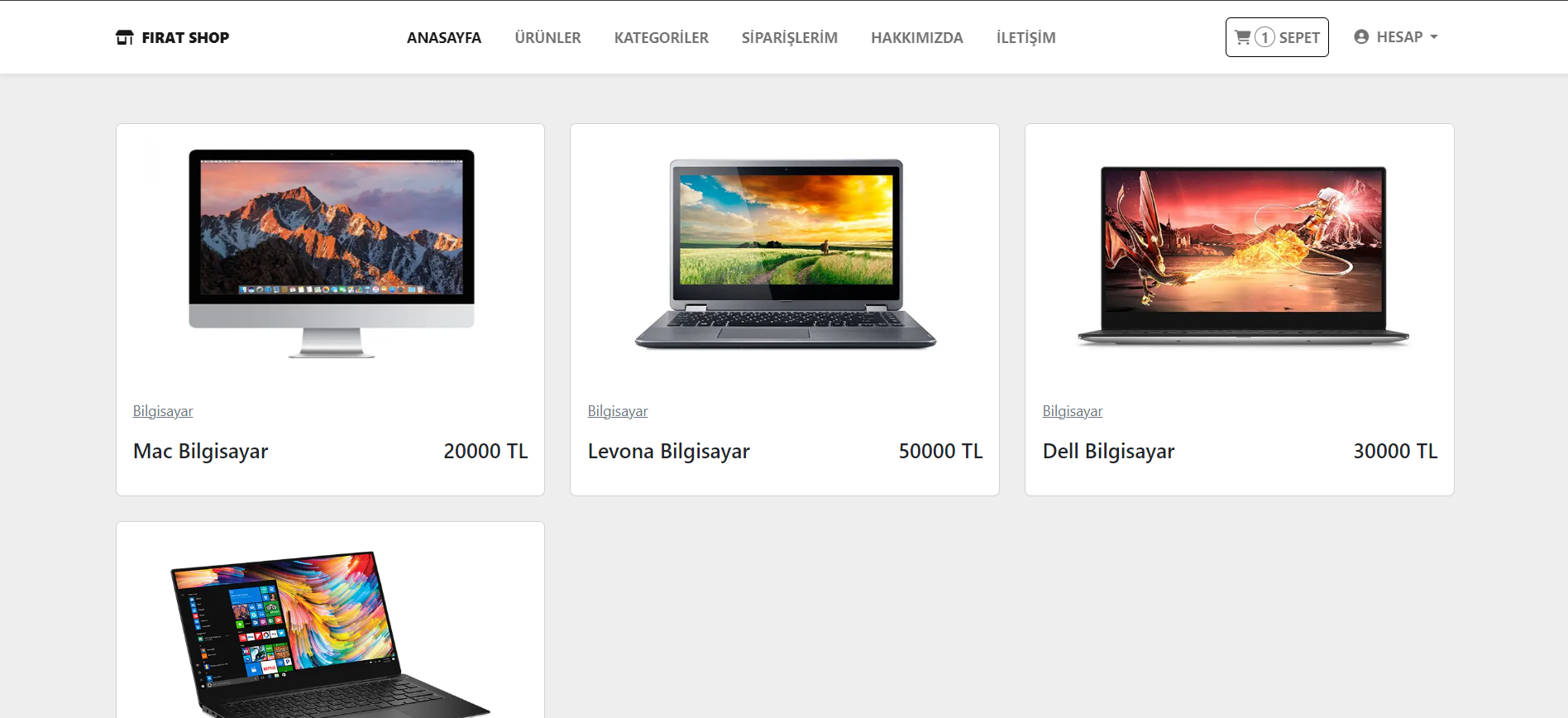
**Şekil 4.53** Takım üyelerimiz

****

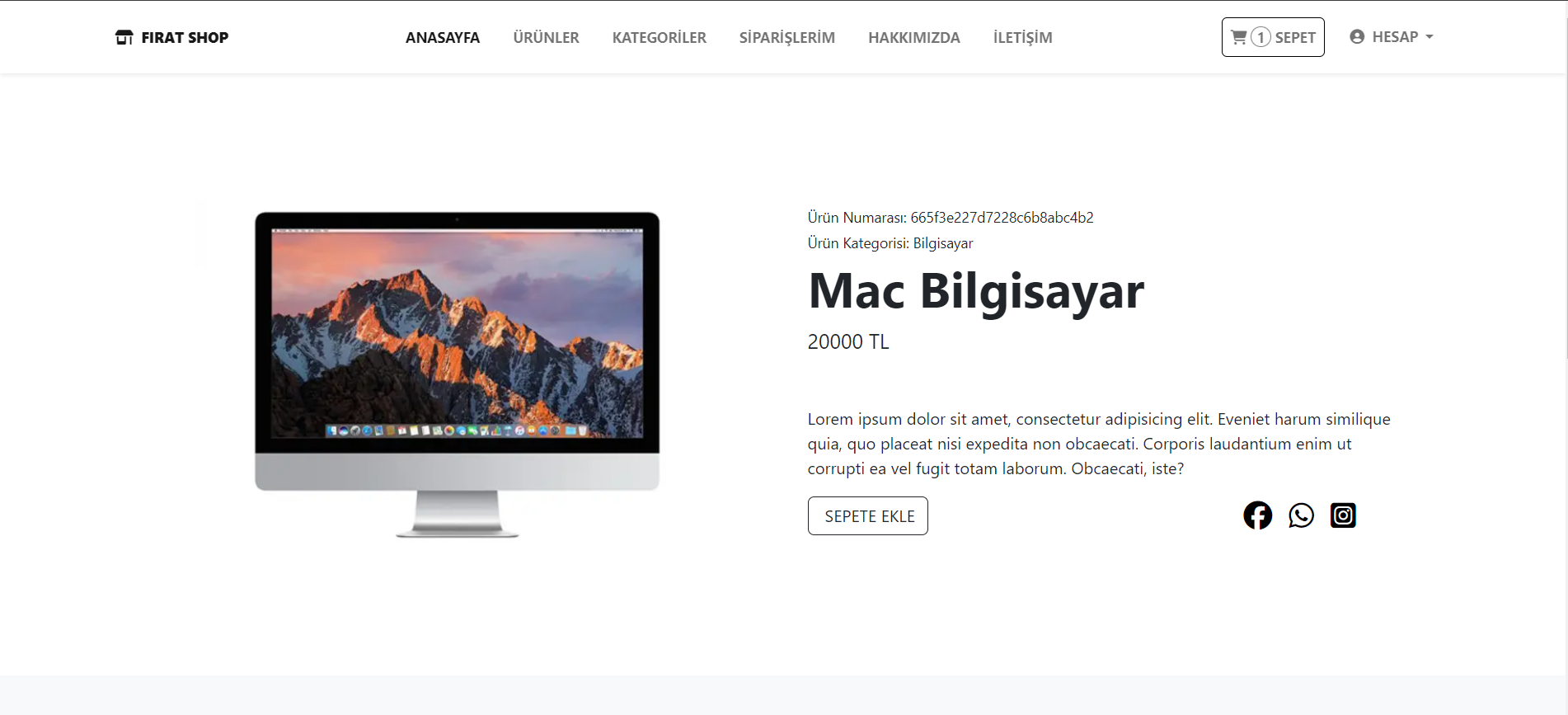
**Şekil 5.54** Kategoriler

****

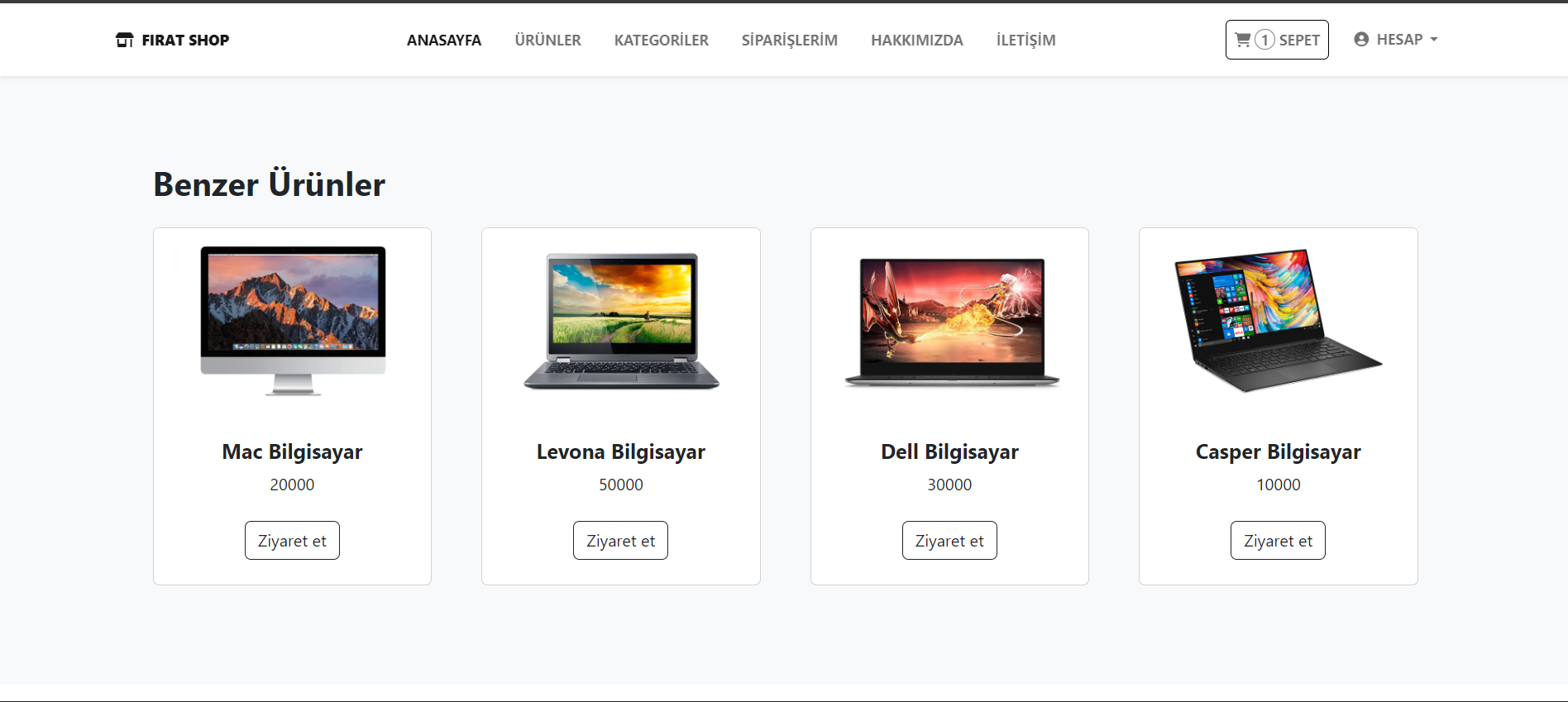
**Şekil 5.55** Tüm ürünler

****

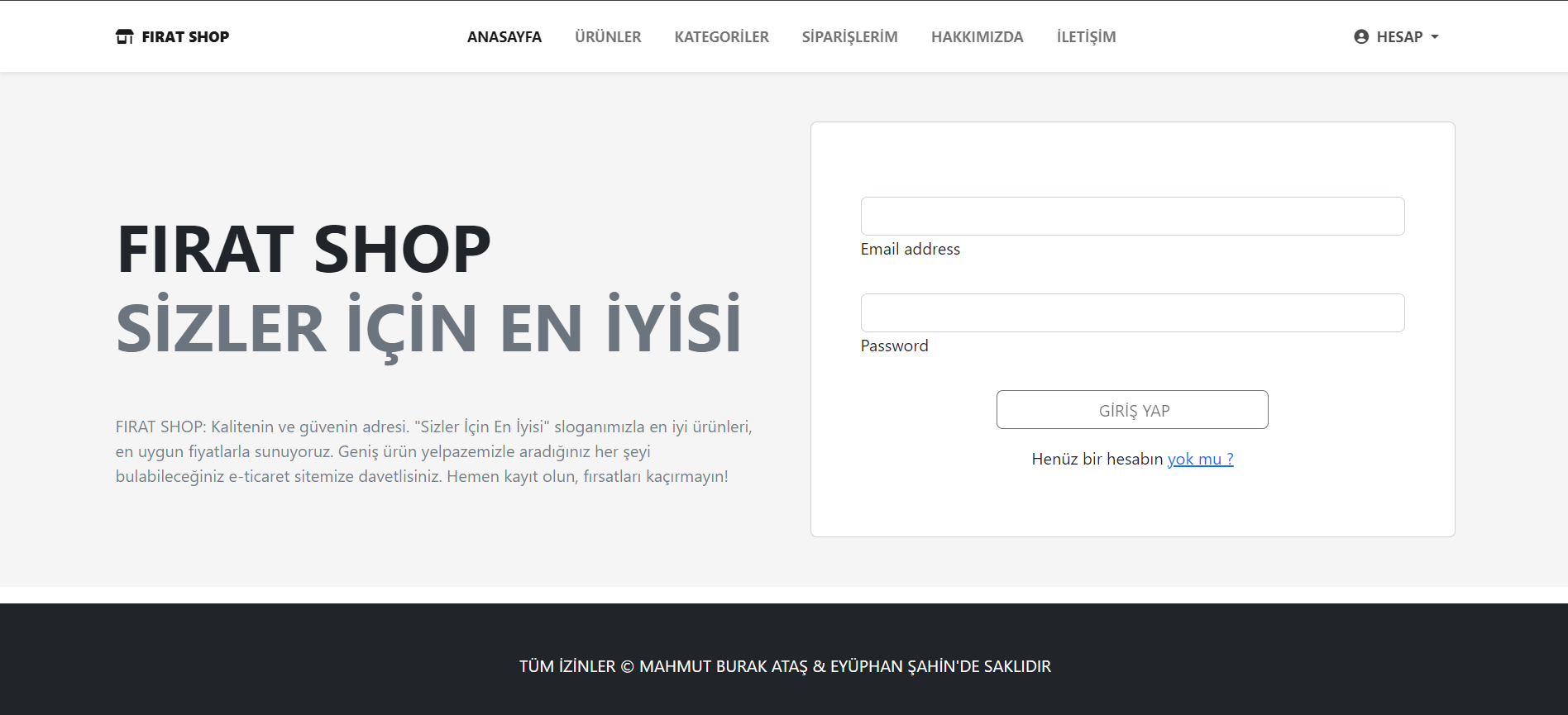
**Şekil 5.56** Kategoriye göre ürünler

****

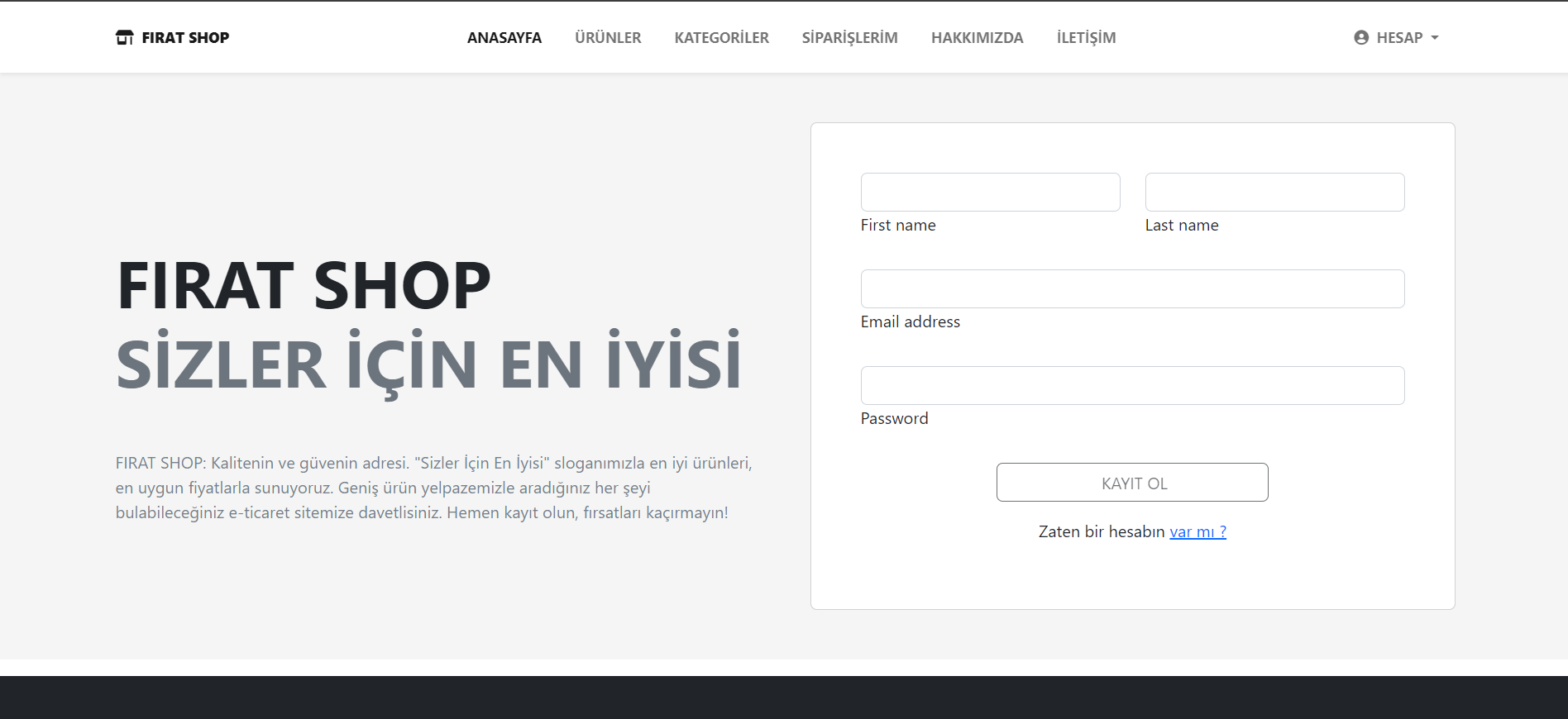
**Şekil 5.57** Ürün detay sayfası

****

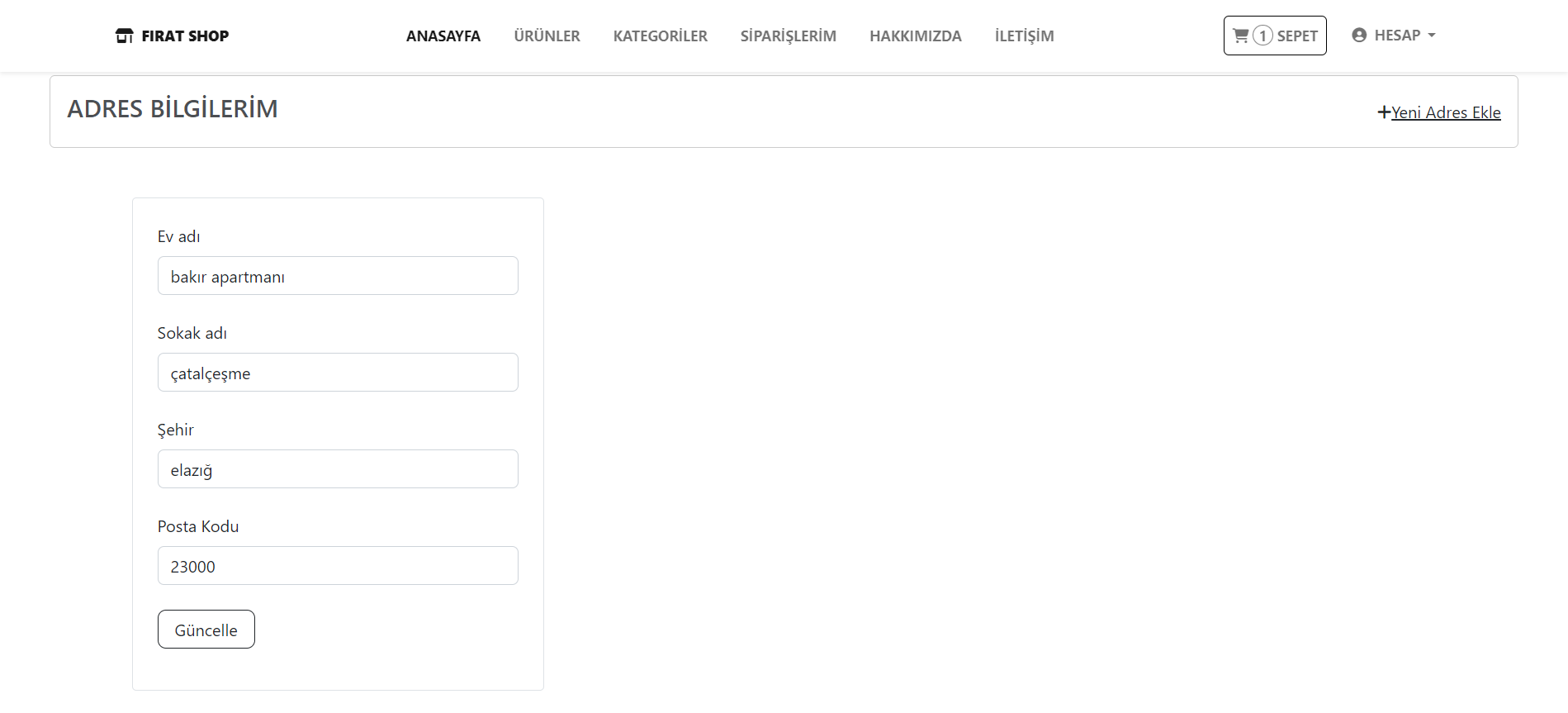
**Şekil 5.58** Ürüne benzer ürünler sayfası

****

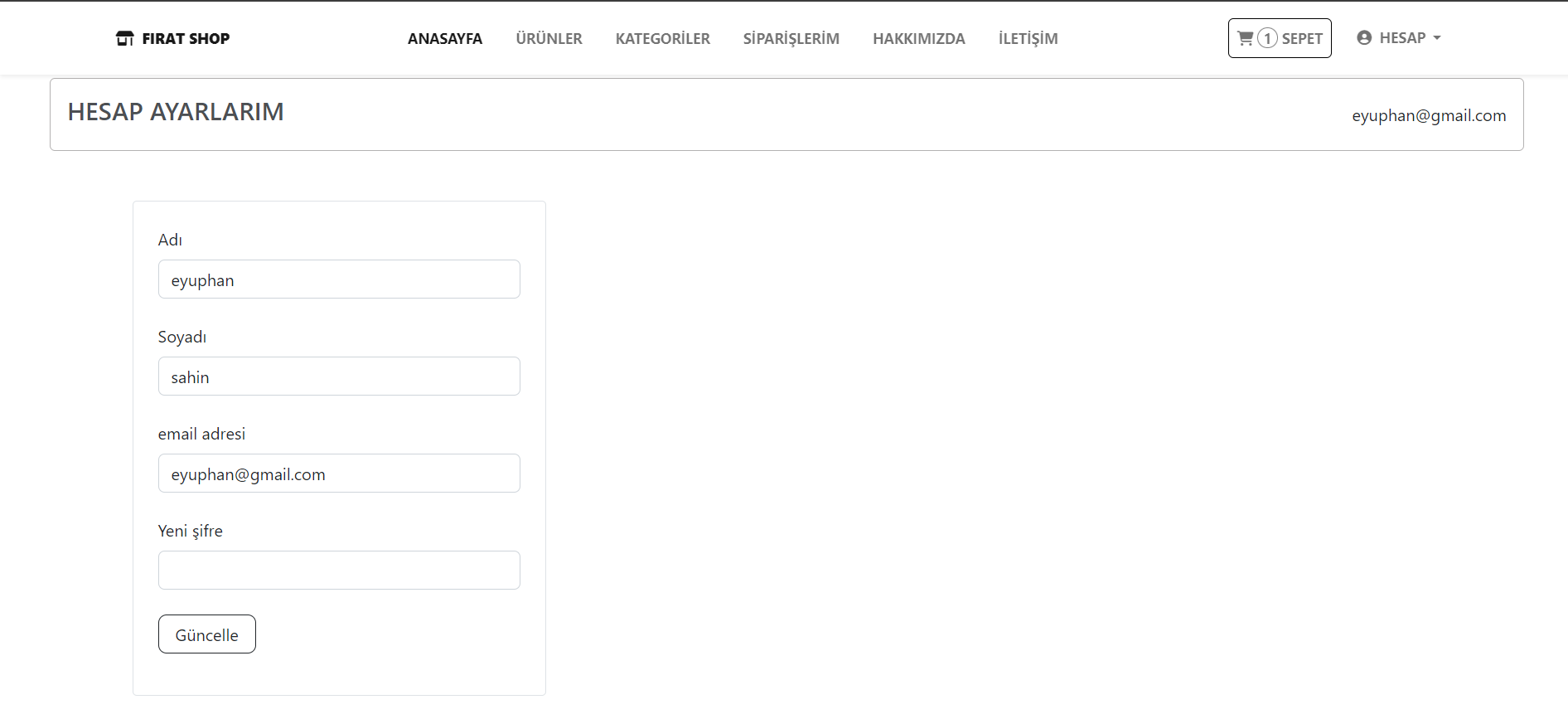
**Şekil 5.59** Giriş yap sayfası

****

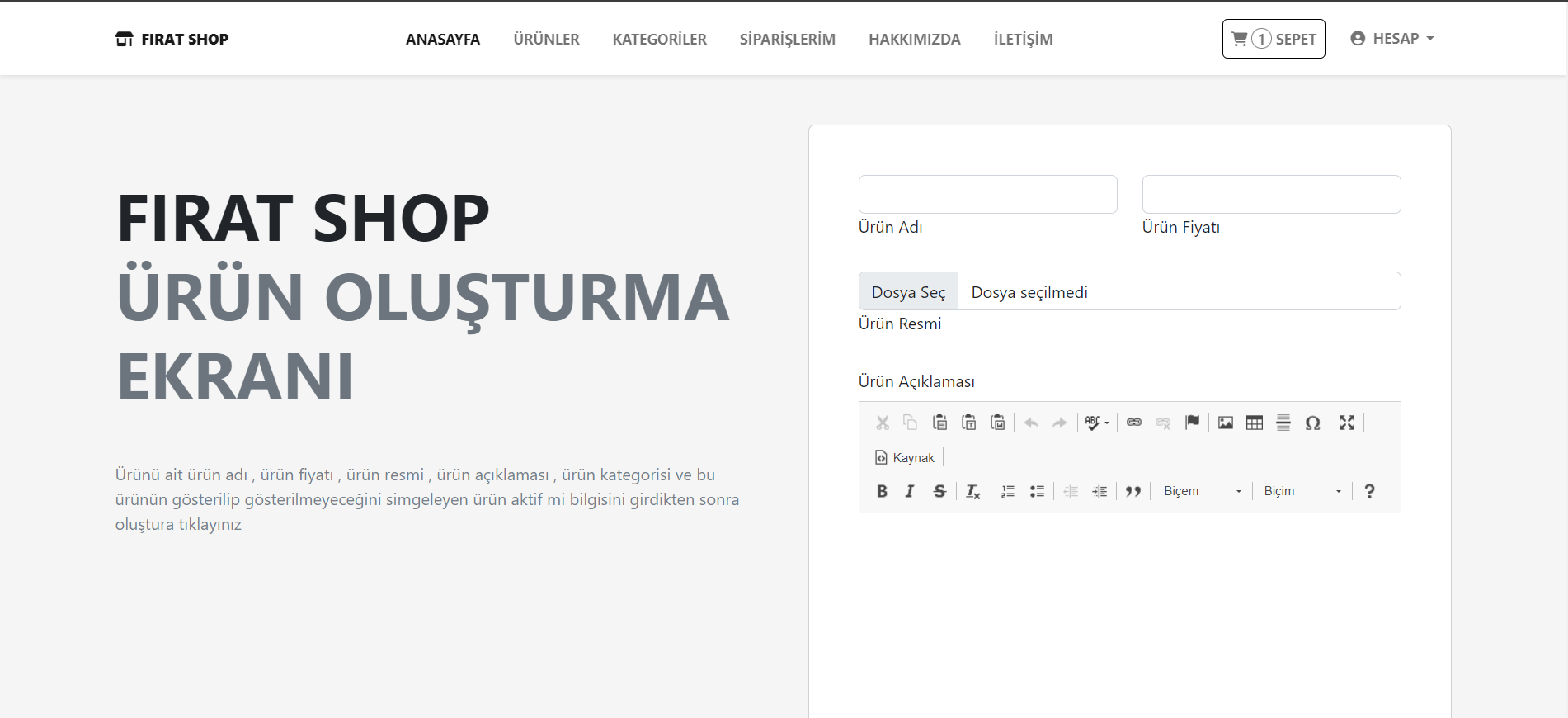
**Şekil 5.60** Kayıt ol sayfası

****

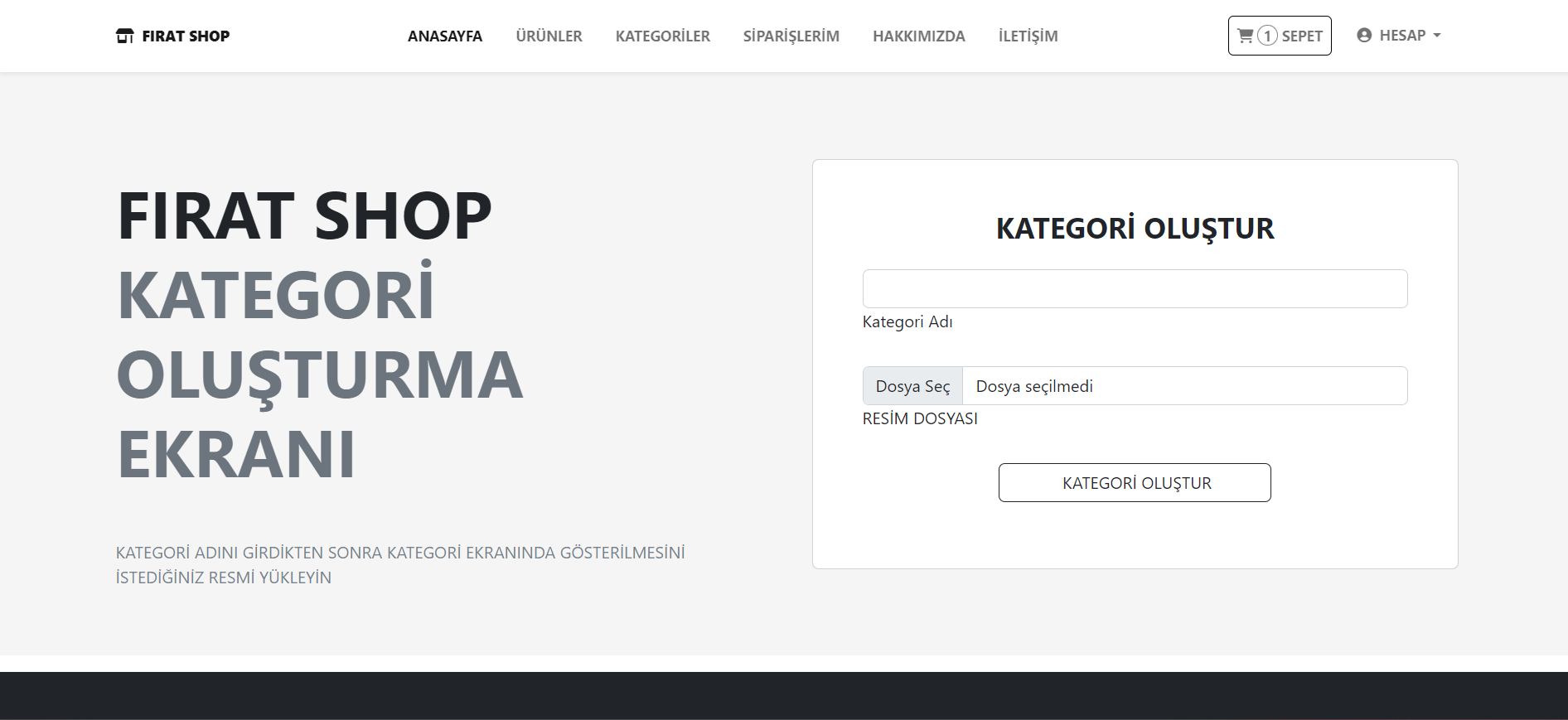
**Şekil 5.61** Adres bilgileri sayfası



**Şekil 5.62** Hesap ayaraları sayfası



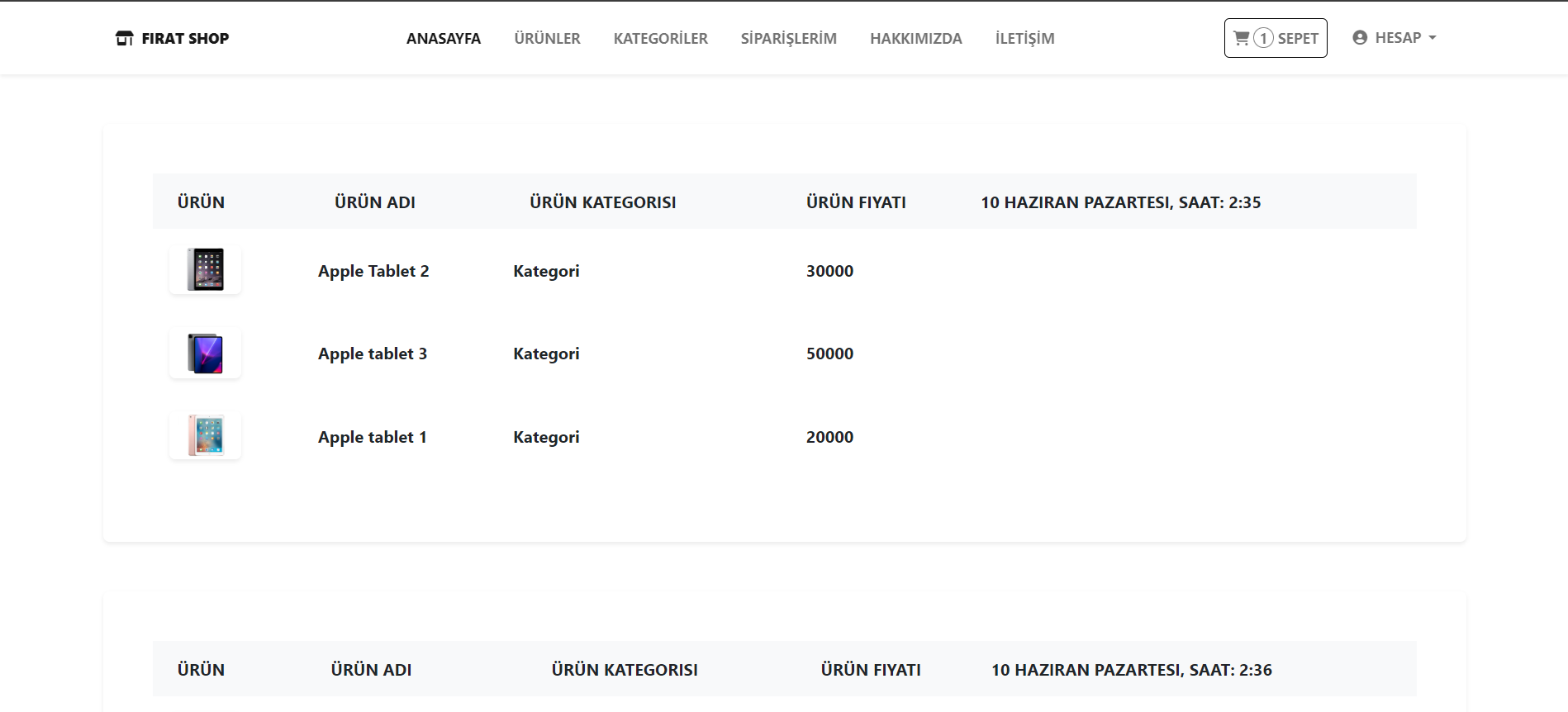
**Şekil 5.63** Ürün oluşturma sayfası



**Şekil 5.64** Kategori oluşturma sayfası



**Şekil 5.65** Sipariş onay sayfası



**Şekil 5.66** Siparişlerim sayfası

**6. SONUÇ**

Bu çalışma kapsamında, Go, Angular ve MongoDB teknolojileri kullanılarak geliştirilen Fırat Shop e-ticaret sitesinin tasarımı ve uygulanabilirliği detaylı bir şekilde ele alınmıştır. Proje, e-ticaret sektöründe kullanıcı memnuniyetini artırmayı ve kullanıcı dostu bir alışveriş deneyimi sunmayı hedeflemiştir.

Go'nun yüksek performanslı ve verimli yapısı, sunucu tarafı işlemlerinin hızlı ve güvenilir bir şekilde gerçekleştirilmesini sağlamıştır. Angular'ın modern ve ölçeklenebilir framework yapısı, kullanıcı arayüzünün dinamik ve duyarlı (responsive) olmasını sağlamıştır. MongoDB'nin esnek ve yüksek performanslı veritabanı yönetimi ise verilerin etkin bir şekilde depolanması ve yönetilmesine olanak tanımıştır. Bu teknolojilerin uyum içinde kullanılması, Fırat Shop'un kullanıcılarına kesintisiz ve sorunsuz bir alışveriş deneyimi sunmasını sağlamıştır.

Fırat Shop, kullanıcıların ürünleri kolayca filtrelemesine, detaylarını görüntülemesine ve güvenli bir şekilde satın alma işlemlerini gerçekleştirmesine olanak tanıyan bir platform olarak tasarlanmıştır. Ayrıca, entegre edilen güvenli ödeme sistemleri sayesinde kullanıcıların ödeme sürecini sorunsuz bir şekilde tamamlaması sağlanmıştır.

Bu çalışma, Go, Angular ve MongoDB teknolojilerinin e-ticaret sektöründe başarılı bir şekilde uygulanabileceğini ve işletmelere önemli katma değer sağlayabileceğini ortaya koymuştur. Araştırma sonuçları, e-ticaret işletmelerine rekabetçi bir avantaj elde etmek ve müşteri memnuniyetini artırmak için en iyi uygulamalar ve stratejiler konusunda değerli bilgiler sunmaktadır.

Sonuç olarak, Fırat Shop, modern teknolojiler kullanılarak geliştirilmiş ve kullanıcıların ihtiyaçlarını en iyi şekilde karşılayacak şekilde tasarlanmış bir e-ticaret platformu olarak sektörde öne çıkmaktadır. Bu projenin, diğer e-ticaret işletmeleri için de örnek teşkil edeceği ve sektöre katkı sağlayacağı öngörülmektedir.

KAYNAKÇA:

[1]https://www.javatpoint.com/what-is-object-oriented-programming

[2]https://en.wikipedia.org/wiki/Go\_(programming\_language)

[3]Sakimura N.;Identity P.; Bradley J.; Jones M. "JSON Web Token (JWT)".Internet Engineering Task Force.2015.6-11.

[4] Mumbaikar S.;Padiya P.; Web Services Based On SOAP and REST Principles.2013.1-4.

[5]Sarısakal M. ; Uysal M.; "WEB TEKNOLOJILERINDEKI HIZLI GELISMELERIN VE WEB PROGRAMLAMA ARAÇLARININ INCELENMESI ", 2001-2002, 3-8

[6]Karagöz E; Tecim V; "Engelsiz Bİlişim", Kriter,2021,140-150

[7]Sarısakal M. ; Uysal M.; "WEB TEKNOLOJILERINDEKI HIZLI GELISMELERIN VE WEBPROGRAMLAMA ARAÇLARININ INCELENMESI ", 2001-2002, 3-8

[8]Huda M. ; "Belajar CRUD Menggunakan PHP dan MySQL", AE Publishing , 2020, 28-70

[9]Wohlgethan E. ; Supporting Web Development Decisions by Comparing Three Major JavaScript Frameworks: Angular, React and Vue.js.2018.35-54.