Teknoloji Fakültesi

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

KURAL TABANLI E-TİCARET WEB SİTESİ GELİŞTİRME

BİTİRME PROJESİ

2. ARA RAPORU

Bilgisayar Mühendisliği Bölümü

DANIŞMAN

Dr. Öğr. Üyesi Eyüp Emre ÜLKÜ

İSTANBUL, 2025

ÖNSÖZ

Bu proje çalışması fikrinin oluşması ve ortaya çıkmasındaki önerisi ve desteğinden dolayı değerli hocam Dr. Öğr. Üyesi Eyüp Emre ÜLKÜ' ye teşekkür ederim.

İÇİNDEKİLER

Proje henüz tamamlanmadığından içindekiler tablosu Ana Raporda düzenlenecektir.

ÖZET

KURAL TABANLI WEB-SİTESİ OLUŞTURURKEN OPTİMAL FORMÜLASYONLARIN ARAŞTIRILMASI VE UYGULANMASI

Bu proje kapsamında "Kural Tabanlı E-Ticaret Sitesi"nin verilerine bağlı olarak farklı parametrelerle (lokasyon, tıklama sayısı, sık ziyaret edilen kategoriler vs.) müşteriye otomatik öneri yapan sistemin optimal seviyeye ulaşması hedeflenmektedir.

Mart, 2025 **Öğrenciler**

Mehmet Ali GÜLYURDU

Bora KIRARSLAN

GİRİŞ

1.1 - Projenin Amacı ve Önemi

Projede, Java ve Spring Boot teknolojileri kullanılarak modern bir e-ticaret sitesi tasarlanması amaçlanmaktadır. Yapılacak bu web sitesindeki asıl amaç; kullanıcıların davranışlarını takip ederek onları ana sayfada genel olarak ilgilendikleri, almaya meyilli oldukları ürünleri göstermek ve bunun sonucunda satış ihtimalini arttırmaktır.

Günümüzde e-ticaret sektörünün hızla büyümesi ve tüketici alışkanlıklarının dijital ortama kaymasıyla birlikte, kullanıcı deneyimini iyileştiren ve kişiselleştirilmiş alışveriş fırsatları sunan platformlara duyulan ihtiyaç artmıştır. Bu proje kapsamında geliştirilecek sistem, kullanıcıların önceki ziyaretleri, arama geçmişleri, satın alma alışkanlıkları ve diğer etkileşimleri gibi çeşitli veri kaynaklarını analiz ederek, kullanıcılara özel ürün önerileri sunacaktır. Böylece kullanıcıların ilgilendikleri ürünlere daha hızlı ulaşmaları sağlanacak, site üzerinde geçirilen zamanın etkinliği artırılacak ve kullanıcı memnuniyeti yükseltilecektir.

Ayrıca, proje kapsamında geliştirilecek sistem, işletmelerin satış stratejilerini daha verimli hale getirmelerine yardımcı olacaktır. Kullanıcıların satın alma eğilimlerini ve tercihlerini analiz ederek, işletmelerin stok yönetimi, ürün yerleşimi, kampanya ve promosyon planlaması gibi konularda daha bilinçli kararlar almaları mümkün olacaktır. Bu

sayede işletmeler, pazarlama maliyetlerini düşürerek, daha yüksek satış rakamlarına ulaşabilecek ve rekabet avantajı elde edebilecektir.

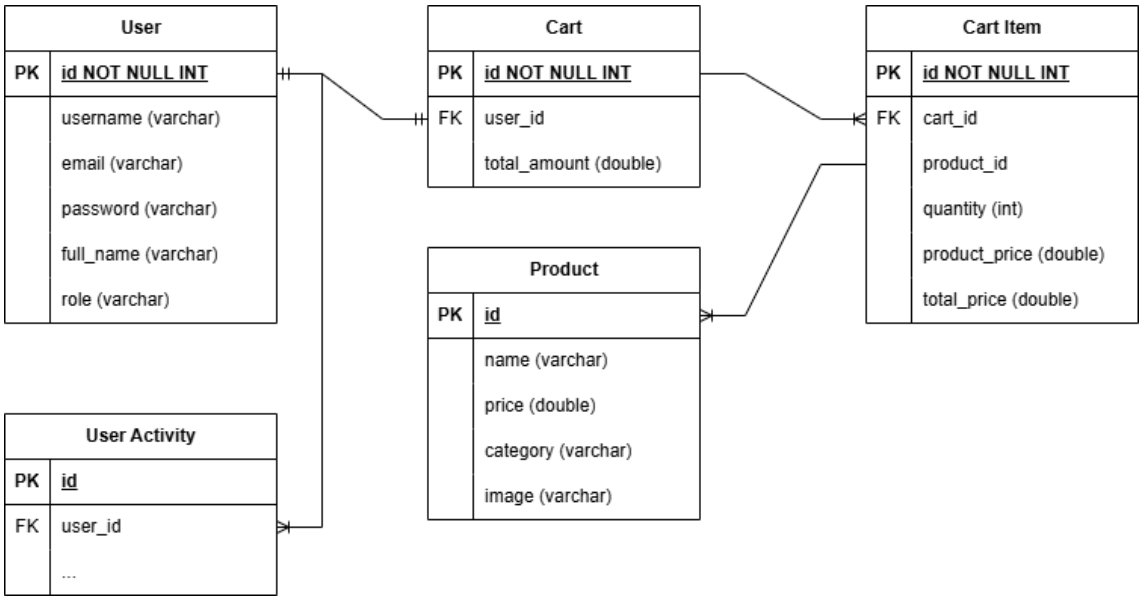
Projenin bir diğer önemli amacı ise, kullanıcı verilerinin güvenliğini ve gizliliğini sağlamaktır. Java ve Spring Boot gibi güvenilir ve yaygın kullanılan teknolojiler sayesinde, kullanıcıların kişisel bilgilerinin korunması ve güvenli ödeme sistemlerinin entegrasyonu mümkün olacaktır. Bu durum, kullanıcıların siteye olan güvenini artıracak ve uzun vadede müşteri sadakatini sağlayacaktır.

Sonuç olarak, bu proje, kullanıcı odaklı, veri temelli ve güvenli bir e-ticaret platformu oluşturarak hem kullanıcıların alışveriş deneyimini iyileştirmeyi hem de işletmelerin satış performansını artırmayı hedeflemektedir. Bu yönüyle proje, sektördeki güncel ihtiyaçlara cevap veren, yenilikçi ve sürdürülebilir bir çözüm sunmayı amaçlamaktadır.

BULGULAR VE TARTIŞMA

2.1- Projede Yapılan Güncellemeler (Ara Rapor - 1)

"Kural Tabanlı" fonksiyonelliğini henüz web-sitesine eklemeden önce web-sitesini Java ve Spring Boot teknolojileri kullanarak sağlam bir temel atmayı öncelikli hedef edindik. Dolayısıyla web-sitesinin back-end kısmına ağırlık vererek ve Entity sınıfları yazarak işe başladık. Bahsedilen Entity sınıfları için aşağıda verilen veritabanı şemasını örnek edinmeye karar verdik.



Basitçe açıklamak gerekirse:

- "User" tablosunu kullanıcıların kişisel bilgilerini ve sitedeki rollerini (standart kullanıcı, admin gibi) içerisine kaydetmek maksadıyla oluşturduk.
- "Cart" tablosunu kullanıcılara ait sepetlerin bilgisini ve sepetin tutarını belirlemek için oluşturduk.
- "Cart Item" tablosunu "Cart" içerisine eklenecek ürünün birim başına fiyatını ve miktarını belirlemek, bundan da yola çıkarak toplam tutarını belirlemek için oluşturduk.
- "Product" tablosunu ürünlerin ismini, birim fiyatını ve hangi kategoriye ait

olduğunu belirlemek (örn. ofis malzemesi, teknolojik eşya) için oluşturduk.
- "User Activity" tablosunu kullanıcıların aktivitelerini takip etmek ve "Kural Tabanlı" fonksiyonunu hayata geçirecek bilgileri tutması maksadıyla oluşturduk. Bu sınıfta henüz detaylı veri satırları oluşturmadık.

Kullanıcı aktivitelerini takip edeceğimiz "User Activity" tablosunun geliştirilme kısmını daha sonraya bıraktık. Çünkü önceliğimizi modern ve hızlı bir web-sitesi kurmaya daha sonra "Kural Tabanlı" fonksiyonunu içine eklemeyi düşündük.

Web-sitenin güvenliği açısından Spring Security sunmuş olduğu imkanlardan faydalandık. Bunu;

- token sistemi (JWT),
- oturum (session) sistemi,
- ilgili kullanıcının site üzerinde erişebileceği end-pointleri belirlemek ve
- şifreleri HASH algoritmalarıyla gizleyip tekrar encode etmek için kullandık.

Site üzerindeki end-point mimarisini henüz tamamlamadık, dolayısıyla bu raporda yer vermedik.

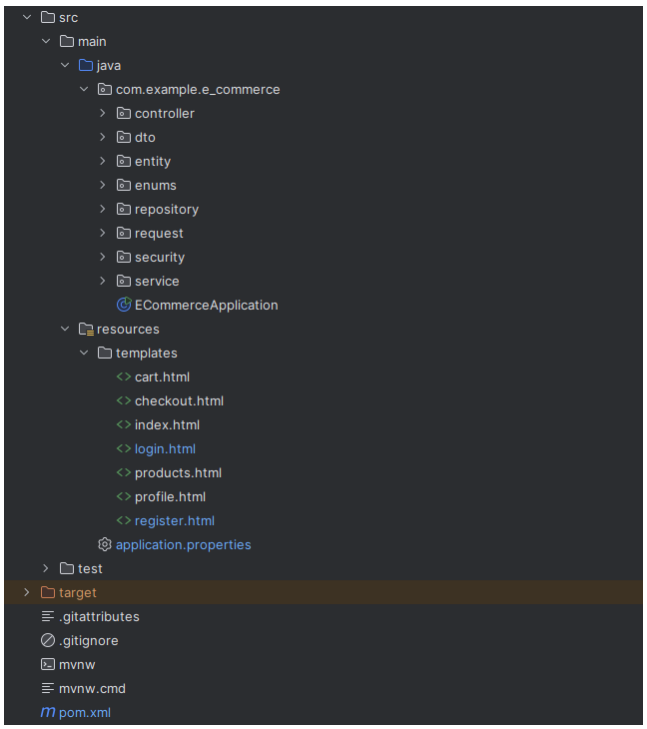
Sitenin veritabanı bağlantısı yapıldı ve kontrol edildi

Web sitesi şu anda ayağa kalkmak için, yerelde 8080 portundan yayın yapıyor ve MySQL veritabanını kullanıyor.

Site, RestAPI üzerinden istekleri kontrol edecek, PostMan üzerinden de back-end'deki işlevsellikler denetlenecek.

2.2- Projede Yapılan Güncellemeler (Ara Rapor - 2)

Proje, back-end kısmında önemli mimari değişiklikler yaparak rahatça modifiye edilebilecek hale getirildi.



Ekran görüntüsünde de görüleceği üzere proje, MVC mimarisine uygun, hiyerarşik bir şekilde tasarlandı.

2.2.1- Kullanılan Teknolojiler ve Araçlar

- Java 17+
- Spring Boot
- Maven
- HTML (Thymeleaf)
- Spring Security
- JPA/Hibernate
- Git (versiyon kontrolü için)

2.2.2- Katmanlı Mimari ve Modülerlik

Projemizde, yazılım mühendisliğinde yaygın olarak kabul gören katmanlı mimari yaklaşımı benimsenmiştir. Bu mimari, uygulamanın fonksiyonel alanlarını mantıksal katmanlara ayırarak, kodun okunabilirliğini, sürdürülebilirliğini ve test edilebilirliğini artırmayı hedefler.

Proje, Spring Framework tarafından kullanılması önerilen katmanlı mimariye uygun şekilde tasarlandı. **Controller – Service – Database (Repository)** katmanlarının her biri olması gerektiği gibi, fonksiyonellikleri birbirine karıştırılmadan kodlandı.

1. Controller Katmanı

Controller (denetleyici) katmanı, istemciden (kullanıcıdan) gelen HTTP isteklerini karşılayan ve uygun servis katmanı metodlarını çağıran uç noktaları (endpoint) barındırır. Bu katmanda, yalnızca yönlendirme ve veri aktarımı işlemleri gerçekleştirilmekte olup, herhangi bir iş mantığına (business logic) yer verilmemiştir. Böylece, "Separation of Concerns" prensibine sadık kalınmıştır.

2. Service Katmanı

Service (hizmet) katmanı, uygulamanın temel iş mantığının (business logic) yazıldığı yerdir. Controller'dan gelen talepler burada işlenir, gerekli formüller ve kurallar uygulanır. Bu katmanda, veri erişim işlemleri doğrudan yapılmaz; bunun yerine **repository** katmanına başvurulur. Böylece kodun yeniden kullanılabilirliği ve test edilebilirliği artırılmış olur.

3. Repository Katmanı

Repository (depo) katmanı, uygulamanın veri erişiminden sorumludur. Veritabanı ile etkileşim bu katmanda gerçekleşir. Spring Data JPA kullanılarak, CRUD (Create, Read, Update, Delete) gibi temel veri tabanı işlemleri kolayca yönetilebilmekte ve Native Query yazmaktan bizi kurtarmaktadır.

4. Entity ve DTO Katmanları

Entity sınıfları, veritabanı tablolarını temsil ederken; DTO (Data Transfer Object) sınıfları,

veri transferini daha sağlıklı şekilde yapar. Çünkü DTO kullanırken Entity içerisindeki tüm veri sütunlarını çağırmamız gerekmez, bunu biz seçebiliriz.

5. Security Katmanı

Spring Security kullanılarak, uygulamanın kimlik doğrulama ve yetkilendirme süreçleri güvenli bir şekilde yönetilmektedir. Kullanıcıların sisteme giriş ve kayıt işlemleri güvenli bir şekilde gerçekleştirilmektedir. JWT ile desteklenmiştir.

6. Request ve Enums Katmanları

Request paketinde, genellikle kullanıcıdan alınan verileri modelleyen sınıflar yer almakta; enums paketinde ise kural tabanlı özelliğine ön ayak olacak veriler ve formül katsayıları yer alacaktır. (örneğin tıklama sayısı, sepete ekleme veya çıkarma davranışları vb.)

2.2.3- Kaynaklar ve Şablonlar (Front-end)

Projenin kaynak dosyaları (resources/templates), kullanıcıya sunulan arayüzleri oluşturan HTML/CSS şablonlarını içermektedir. Her bir sayfa (ör. login, register, products, cart vb.) MVC mimarisine uygun olarak ayrı ayrı tasarlanmıştır.

2.2.4- Konfigürasyon ve Bağımlılıklar

“application.properties” dosyası, uygulamanın temel yapılandırmalarını (veritabanı bağlantısı, port ayarları, güvenlikle ilgili yapılandırmaları vb.) içermektedir. Maven ve pom.xml dosyası ile proje bağımlılıkları (dependency) yönetilmekte, bu sayede site ayağa kaldırılırken gereken özellikler çalıştırılabilmektedir.

“application.properties” dosyasının güncel hali aşağıdaki gibidir:

```
# application.properties
# JWT Configuration
jwt.secret=mySecretKey12345678901234567890123456789012345678901234567890
jwt.expiration=86400000

# Database Configuration
spring.datasource.url=jdbc:mysql://localhost:3306/ecommerce
spring.datasource.username=root
spring.datasource.password=Mali71693135
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# JPA Configuration
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect

# Server Configuration
server.port=8080
```

Görüldüğü üzere burada; JWT konfigürasyonları, veri tabanı bağlantısı ve giriş bilgileri, JPA ile ilgili ayarlar ve port ayarı yapılmıştır.

2.2.5- Versiyon Kontrol Sistemi - Git

Projede iş bölümü; front-end, back-end ve test aşaması olarak ayrıldığından ve ekip çalışması yapıldığından proje GitHub adresine yüklenmiştir. **master** ve **dev** branchleri oluşturulmuştur. Projede yapılan değişiklikler önce **dev** branch üzerinde commit ve push işlemleri yapılarak ve ekip iletişimi sağlanarak test ediliyor. Daha sonra uygun görülürse **master** branch üzerinde değişiklikler yapılıyor. Ayrıca projenin herhangi bir bilgisayarda nasıl ayağa kaldırılacağı hakkında bitime yakın bir rehber hazırlanıp README.md dosyasında bu talimatlara yer verilecektir.

2.2.6- Projedeki Lombok Kütüphanesinin Kullanımı Hakkında

Lombok sayesinde, özellikle veri modellerinde sıkça ihtiyaç duyulan getter, setter, constructor gibi metotlar otomatik olarak oluşturulmuş, böylece kodun daha sade ve okunabilir olması sağlanmıştır. Ayrıca, tekrarlayan kodların azaltılması ile geliştirme sürecinin hızlandırılması amaçlanmıştır.

2.2.7- Kullanıcı Davranışlarının Kural Tabanlı Modellenmesi ve Literatür İlişkisi – Projede Kullanılan Enum Seti

Literatürde, kullanıcı davranışlarının öneri sistemlerine entegrasyonu üzerine çeşitli araştırmalar yapılmıştır. Ricci ve ark. (2015) tarafından, öneri sistemlerinde kişiselleştirmenin başarısında kullanıcı tıklama ve görüntüleme davranışlarının kritik öneme sahip olduğu ortaya konmuştur [1]. Benzer şekilde, Jannach ve ark. (2016) tarafından, kullanıcıların sepete ekleme ve favorilere alma gibi mikro etkileşimlerinin ürün önerilerinin doğruluğunu artırmada etkili olduğu gösterilmiştir [2]. Ayrıca, kullanıcıların geçmiş satın alma davranışları ve kategori bazlı ilgi düzeylerinin, öneri sistemlerinin uzun vadeli kişiselleştirme performansını güçlendirdiği belirtilmiştir (Zhang et al., 2019) [3].

Hazırlanan enum setinde, literatürde önerilen bu yaklaşımlar dikkate alınarak; **ürün tıklama**, **görüntüleme süresi**, **sepet ve favori etkileşimleri**, **kategori ve marka ilgisi**, **fiyat hassasiyeti**, **mevsimsel ilgi** ve **satın alma geçmişi** gibi birden çok faktör ile kullanıcı davranışlarının izlenmesi sağlanmıştır. Her bir parametrenin, kullanıcıların platformdaki tercih ve eğilimlerinin daha isabetli biçimde tespit edilmesine olanak tanıdığı görülmektedir. Son yıllarda gerçekleştirilen akademik çalışmalarda (Sun et al., 2021; Wang et al., 2022), kullanıcıların **fiyat hassasiyeti** ve **mevsimsel tercihlerinin** de öneri algoritmalarında dikkat edilmesinin, satış oranlarını önemli ölçüde artırdığı gösterilmiştir [4] [5].

Bu parametrelerin sistemde ayrı ayrı izlenmesi ve literatürdeki yaklaşımlardan esinlenilmekle birlikte özgün bir formülasyonun geliştirilmiştir. Böylece hem kullanıcı memnuniyetini hem de ticari başarıyı artıracak şekilde bir öneri sistemi oluşturulmuştur.

İLGİLİ BÖLÜMÜN KAYNAKÇASI

1. Ricci, F., Rokach, L., & Shapira, B. (2015). Recommender Systems Handbook. Springer.
2. Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2016). Recommender Systems: An Introduction. Cambridge University Press.

3. Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep Learning based Recommender System: A Survey and New Perspectives. ACM Computing Surveys, 52(1), 1-38.
4. Sun, Q., Guo, G., & Barnes, S. J. (2021). Incorporating User Price Sensitivity into Recommendation. Information & Management, 58(2), 103-120.
5. Wang, Y., Li, X., & Zhang, Y. (2022). Seasonal Product Recommendation in E-Commerce. Expert Systems with Applications, 196, 116529.

2.2.8- Renk Seçimlerinin E-Ticaret Sitelerinde Kullanıcı Deneyimine Etkisi

E-ticaret sitelerinin front-end tarafında renk seçimlerinin kullanıcı davranışları üzerinde önemli bir rol oynadığı çeşitli araştırmalarla ortaya konmuştur (Singh, 2006; Labrecque & Milne, 2012). Bu projede, kullanıcıların dikkatini çekmek ve satın alma eğilimini artırmak amacıyla renk seçimine özel bir önem verilmiştir.

Örneğin, ana sayfa ve ürün detay sayfalarında turuncu renk tercih edilmiştir. Turuncunun, sıcak ve enerjik bir renk olması nedeniyle kullanıcıda **güven ve hareketlilik** duygusu uyandırdığı, dolayısıyla "hemen satın al" gibi çağrı butonlarında sıklıkla kullanıldığı literatürde belirtilmiştir (Labrecque & Milne, 2012). Ayrıca, turuncunun **aciliyet hissi** yaratarak kullanıcıların karar verme süreçlerini hızlandırdığı da vurgulanmaktadır (Singh, 2006).

Site genelinde arka plan rengi olarak beyaz kullanılmıştır. Beyaz rengin, **sadelik ve temizlik algısı** oluşturduğu, bu sayede ürünlerin daha ön plana çıkmasını sağladığı çeşitli çalışmalarla gösterilmiştir (Hall & Hanna, 2004). Bu nedenle, ürün görsellerinin ve metinlerin okunabilirliğini artırmak için beyaz arka plan tercih edilmiştir.

Ayrıca, **güven unsuru** oluşturmak amacıyla ödeme ve kullanıcı üyelik formlarında mavi tonları kullanılmıştır. Mavi rengin, güven ve profesyonellik hissi verdiği, özellikle finansal işlemlerde kullanıcıların kendilerini daha rahat hissetmelerine katkı sağladığı araştırmalarda ifade edilmiştir (Kaya & Epps, 2004).

Sonuç olarak, renk seçiminin kullanıcı deneyimi ve satın alma davranışları üzerindeki etkisi göz önünde bulundurularak, e-ticaret sitesinin ön yüz tasarımında **turuncu, beyaz ve mavi** renkler kullanılmıştır. Bu seçimlerin, literatürde belirtilen psikolojik ve davranışsal etkiler dikkate alınarak yapıldığı ifade edilebilir.

İLGİLİ BÖLÜMÜN KAYNAKÇASI

- Hall, R. H., & Hanna, P. (2004). The impact of web page text-background color combinations on readability, retention, aesthetics, and behavioral intention. *Behaviour & Information Technology*, 23(3), 183–195.
- Kaya, N., & Epps, H. H. (2004). Relationship between color and emotion: A study of college students. *College Student Journal*, 38(3), 396.
- Labrecque, L. I., & Milne, G. R. (2012). Exciting red and competent blue: The importance of color in marketing. *Journal of the Academy of Marketing Science*, 40(5), 711-727.

- Singh, S. (2006). Impact of color on marketing. *Management Decision*, 44(6), 783-789.

2.2.9- JWT ile Kullanıcı Kimlik Doğrulama Süreçlerinde Sağlanan Kolaylıklar

JWT (JSON Web Token) teknolojisinin, e-ticaret sitelerinde kullanıcı oturum yönetiminde sağladığı kolaylıklar çeşitli araştırmalarda ele alınmıştır (Jones et al., 2016; Nadalin et al., 2015). Kullanıcıların siteye giriş yaptıktan sonra kimlik bilgilerinin güvenli ve pratik bir şekilde doğrulanabilmesi amacıyla JWT kullanılmaktadır.

Öncelikle, JWT ile kimlik doğrulama işlemi gerçekleştirildiğinde, kullanıcı adı ve şifre gibi bilgiler sunucu tarafından doğrulanmakta ve ardından kullanıcıya bir JWT token'ı iletilmektedir. Bu token, kullanıcının tarayıcısında saklanmakta ve sonraki her istekte sunucuya gönderilmektedir. Böylece, her işlemde tekrar tekrar kullanıcı adı ve şifre girilmesine gerek kalmamaktadır. Bu durum, **kullanıcı deneyimini önemli ölçüde kolaylaştırmaktadır** (Nadalin et al., 2015).

Ayrıca, JWT'nin taşınabilir ve standartlara uygun yapısı sayesinde, kullanıcıların farklı cihazlardan veya tarayıcılardan kolayca giriş yapabilmesi mümkün kılınmıştır. Token'ın doğrulanması için merkezi bir oturum yönetimine ihtiyaç duyulmadığından, sistemin ölçeklenebilirliği artırılmıştır (Jones et al., 2016). Böylece, **yoğun alışveriş dönemlerinde veya kampanya zamanlarında dahi kullanıcı oturumlarının hızlı ve güvenli bir şekilde yönetilebilmesi** sağlanmıştır.

JWT'nin bir diğer avantajı ise, token üzerinde kullanıcıya ait ek bilgiler (örneğin, kullanıcı rolü, yetki düzeyi) taşınabildiğinden, kullanıcının yetkili olduğu sayfalara yönlendirilmesi ve **kişiselleştirilmiş içerik sunulması kolaylaştırılmıştır**. Bu sayede, **kullanıcıya özel bir alışveriş deneyimi sunulması** mümkün olmuştur (Nadalin et al., 2015).

İLGİLİ BÖLÜMÜN KAYNAKÇASI

- Jones, M., Bradley, J., & Sakimura, N. (2016). JSON Web Token (JWT). IETF RFC 7519.
- Nadalin, A., Byrne, B., & Hardt, D. (2015). OAuth 2.0 and JSON Web Token (JWT) as a Token Format. OAuth Working Group.

2.2.10- Projenin End-point Tasarımı

Projede REST API ve MVC yaklaşımları kullanılmıştır. Şu ana kadar kullanılan end-pointlerin ilgili HTTP metodu, Request Body değişkenleri ve Response değişkenleri aşağıdaki tabloda ilgili Controller sınıfının altında ayrı ayrı verilmiştir.

- **AuthController - /api/auth**

Method	Endpoint	Açıklama	Request Body	Response
POST	/api/auth/login	Kullanıcı girişi	email, password	token, user, message
POST	/api/auth/register	Kullanıcı kaydı	UserDto	token, user, message
POST	/api/auth/logout	Kullanıcı çıkışı	Header: Authorization	message
GET	/api/auth/validate	Token doğrulama	Header: Authorization	valid, user

- **CartController - /cart**

Method	Endpoint	Açıklama	Parametreler	Response
GET	/cart/{userId}	Kullanıcı sepetini getir	userId	Cart object
POST	/cart	Sepete ürün ekle	userId, productId, quantity	CartItem
GET	/cart	Sepet sayfasını göster	Principal	cart.html

- **CartItemController - /cart-item**

Method	Endpoint	Açıklama	Parametreler	Response
POST	/cart-item/add	Sepete ürün ekle	productId, quantity, Principal	Redirect to /cart

- **IndexController - /**

Method	Endpoint	Açıklama	Parametreler	Response
GET	/index	Ana sayfa	-	index.html + categories
GET	/products	Kategoriye göre ürünler	category	products.html

Method	Endpoint	Açıklama	Parametreler	Response
GET	/checkout	Ödeme sayfası	-	checkout.html

- **ProductController - /product**

Method	Endpoint	Açıklama	Request Body	Response
GET	/product/{productId}	Ürün detayı	-	Optional<Product>
POST	/product	Yeni ürün oluştur	Product	Product
PUT	/product/{productId}	Ürün güncelle	Product	Product
DELETE	/product/{productId}	Ürün sil	-	void

- **ProfileController - /profile**

Method	Endpoint	Açıklama	Parametreler	Response
GET	/profile	Profil sayfası	UserDetails	profile.html
POST	/update-profile	Profil güncelle	User, Principal	Redirect to /profile

- **UserController - /user**

Method	Endpoint	Açıklama	Request Body	Response
GET	/user	Tüm kullanıcılar	-	List<User>
GET	/user/{userId}	Kullanıcı detayı	-	Optional<User>
POST	/user	Yeni kullanıcı	User	User
PUT	/user/{userId}	Kullanıcı güncelle	User	User
DELETE	/user/{userId}	Kullanıcı sil	-	void