



Teknoloji Fakültesi

MARMARA ÜNİVERSİTESİ

TEKNOLOJİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİTİRME PROJESİ

Kural Tabanlı E-Ticaret Sitesi Geliştirme

PROJE YAZARLARI

Bora KIRARSLAN, Mehmet Ali GÜLYURDU

DANIŞMAN

Dr. Öğretim Üyesi Eyüp Emre ÜLKÜ

İSTANBUL, 2025

MARMARA ÜNİVERSİTESİ
TEKNOLOJİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Marmara Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği Öğrencisi Mehmet Ali GÜLYURDU ve Bora KIRARSLAN' ın “Kural Tabanlı E-Ticaret Sitesi Geliştirme” başlıklı bitirme projesi çalışması, 19/06/2025 tarihinde sunulmuş ve jüri üyeleri tarafından başarılı bulunmuştur.

Jüri Üyeleri

Prof. Dr. Şahin UYAYER (Üye)

Marmara Üniversitesi (İMZA)

Dr. Öğr. Üyesi Eyüp Emre ÜLKÜ (Danışman)

Marmara Üniversitesi (İMZA)

Arş. Gör. Duygu KAYAOĞLU (Üye)

Marmara Üniversitesi (İMZA)

İÇİNDEKİLER

ŞEKİL LİSTESİ	i
TABLO LİSTESİ	iii
ÖZET	iv
ABSTRACT	v
1. GİRİŞ	1
1.1. Bitirme Projesinin Amacı	1
1.2. Literatür Özeti.....	2
2. MATERYAL VE YÖNTEM	4
2.1. Kullanılan Teknolojiler.....	4
2.1.1 Html, css ve javascript kullanımı	4
2.1.2 Mysql veri tabanı kullanımı.....	5
2.2. Kullanılan Teknolojilerin Test Edildiği ve Çalıştırıldığı Ortamlar	9
2.3. Dosya Düzeni ve Proje Hiyerarşisi.....	9
2.3.1 Entity sınıfları	13
2.3.2 Repository sınıfları	15
2.3.3 Service sınıfları.....	16
2.3.4 Controller sınıfları	18
2.4. End-Point Tasarımı.....	19
2.5. DTO, Request ve Response Dizinleri.....	21
2.5.1 DTO sınıfları ve açıklamaları	22
2.5.2 Request sınıfı ve açıklaması	22
2.5.3 Response sınıfı ve açıklaması.....	22
2.6. Enum Sınıfları	22
2.7. Security Sınıfları.....	24
2.7.1 JWTUtil sınıfı.....	24

2.7.2	SecurityConfig sınıfı	24
2.7.3	JWTAuthenticationFilter sınıfı.....	25
2.8.	Yapılandırmalar	25
2.8.1	application.properties yapılandırması.....	25
2.8.2	pom.xml yapılandırması	27
2.9.	E-Posta Doğrulama Sistemi.....	28
2.10.	README.md dosyası	29
2.11.	HTML – CSS Tasarımları	29
2.12.	Javascript Kullanımı	37
2.12.1	Ana sayfa (index.html)	37
2.12.2	Giriş sayfası (login.html).....	37
2.12.3	Kayıt sayfası (register.html)	37
2.12.4	Ürünler sayfası (products.html).....	38
2.12.5	Ürün detayları sayfası (productdetails.html)	38
2.12.6	Sepet sayfası (cart.html)	38
2.12.7	Ödeme sayfası (checkout.html)	38
2.12.8	Profil sayfası (profile.html)	38
2.12.9	Sipariş geçmişi (order-history.html).....	39
2.12.10	Sipariş onayı (order-confirmation.html).....	39
2.12.11	E-posta doğrulama (email-verification.html)	39
2.12.12	İndirimli kategoriler (discountCategories.html)	39
3.	BULGULAR VE TARTIŞMA	40
3.1.	Kişiselleştirilmiş İndirim Mantığı	40
3.1.1	Toplanan veriler.....	41
3.1.2	Toplanan verilere atanan ağırlık değerleri.....	41
3.1.3	Kullanıcı etkileşimlerinden skor oluşturma.....	42

3.2.	Renk Seçimlerinin Kullanıcıya Etkisi	44
3.2.1	Turuncu rengin kullanımı	45
3.2.2	Siyah ve beyaz renklerinin kullanımı	45
3.2.3	Yeşil rengin kullanımı	45
3.3.	Güvenli Web Sitesi Deneyimini Sağlama	45
3.4.	Kullanıcı Davranışları ve Ödül Mekanizması	46
3.5.	Kural Tabanlı İndirim Sistemlerinin Sektördeki Yeri	47
3.6.	Sistem Mimarisi ve Teknolojik Altyapısı.....	47
3.7.	Kullanıcının Geçmiş Siparişlerine Bağlı Skor Çarpanı	49
3.8.	Gerçek Hayatta Kullanıcı Aksiyonları ve Satın Alma.....	50
3.8.1	Türkiye’de genel e-ticaret satın alma oranı	51
4.	SONUÇLAR	51
5.	KAYNAKÇA	53

ŞEKİL LİSTESİ

Şekil 2. 1: Mysql workbench üzerindeki şemanın ekran alıntısı	7
Şekil 2. 2: Veri tabanı şeması	Hata! Yer işareti tanımlanmamış.
Şekil 2. 3: UserRepository sınıfına ait ekran görüntüsü	10
Şekil 2. 4: UserService sınıfından ekran görüntüsü	10
Şekil 2. 5: Kullanıcı kaydı yapan servis fonksiyonu	11
Şekil 2. 6: UserController sınıfına ait ekran görüntüsü	11
Şekil 2. 7: Projenin IDE üzerindeki dosya mimarisine ait ekran görüntüsü.....	12
Şekil 2. 8: Entity klasörü içerisinde yer alan "user" sınıfına ait ekran görüntüsü	13
Şekil 2. 9: Yetkilendirme ve doğrulama katmanı	19
Şekil 2. 10: Kullanıcı yönetimi katmanı	19
Şekil 2. 11: Ürün ve kategori katmanı	20
Şekil 2. 12: Alışveriş sepeti katmanı	20
Şekil 2. 13: Kullanıcı aktiviteleri katmanı	21
Şekil 2. 14: Sipariş işlemleri katmanı	21
Şekil 2. 15: InteractionType.java.....	23
Şekil 2. 16: UserMultiplier.java	23
Şekil 2. 17: application.properties dosyasına ait ekran görüntüsü	25
Şekil 2. 18: email-verification.html	30
Şekil 2. 19: index.html.....	31
Şekil 2. 20: login.html	31
Şekil 2. 21: productdetails.html	32
Şekil 2. 22: products.html	33
Şekil 2. 23: profile.html	33
Şekil 2. 24: register.html	34
Şekil 2. 25: order-confirmation.html	35

Şekil 2. 26: checkout.html	36
Şekil 2. 27: discountCategories.html	37
Şekil 3. 1: Ana sayfada görüntülenen indirim bildirimi	40
Şekil 3. 2: Spring Boot'un katmanlı mimarisini temsil eden görsel	47

TABLO LİSTESİ

Tablo 3.1: Belirlenen ağırlıklar.....	43
Tablo 3.2: Renk seçiminin kullanıcıya etkisi.....	44
Tablo 3.3: Geçmiş sipariş tutarına göre skor çarpanları.....	49
Tablo 3.4: Senaryolar ve Satın Alma Olasılıkları Tablosu.....	51

ÖZET

Bu çalışma, modern e-ticaret sektöründe kullanıcı deneyimini iyileştirmek ve müşteri bağlılığını artırmak amacıyla geliştirilen kural tabanlı bir e-ticaret platformunu ele almaktadır. Geleneksel e-ticaret sistemlerinde uygulanan tek tip indirim stratejilerinin yetersizliği ve kişiselleştirme eksikliği, müşteri kazanımı ve sadakatinde önemli bir sorun teşkil etmektedir. Bu problemi çözmek için, Spring Framework altyapısı üzerine inşa edilen ve *MySQL* veritabanı ile desteklenen bir sistem geliştirilmiştir. *JWT* ve *BCrypt* gibi güvenlik çözümlerine backend kısmında yer verilmiştir. Böylece oturum açma işlemleri güvenli ve kolay şekilde gerçekleşirken kullanıcıların şifreleri veri tabanında şifreli biçimde saklanması sağlanmıştır. Spring'in katmanlı yapısı sayesinde siteye ölçeklendirme sağlanmış ve kod karmaşasının önüne geçilmiştir. Sistem, kullanıcı davranışlarını veri tabanında toplar ve servis katmanında bazı hesaplar yapar. Böylece kişiselleştirilmiş indirimler sunan kural tabanlı bir yapı kullanılmış oldu. Frontend tarafında *HTML*, *CSS* ve *JavaScript* teknolojileri kullanılarak kullanıcı dostu bir arayüz tasarlanmıştır. Özellikle Javascript sayesinde müşterinin belli bir kategoride indirim kazanması halinde otomatik olarak ilgili ürünlerde indirimi gerçek zamanlı gösterilmiştir. Geliştirilen platform, kullanıcıların alışveriş alışkanlıklarını ve tercihlerini takip ederek, belirli koşullar altında otomatik olarak devreye giren indirim mekanizmaları sunmaktadır. Örneğin, belirli bir ürün kategorisinde ilgi gösteren ancak satın alma yapmayan kullanıcılara özel teklifler veya sadık müşterilere özel indirimler tanımlanabilmektedir. Sonuç olarak, bu çalışma e-ticaret işletmelerine dinamik ve etkili bir pazarlama aracı sunarak rekabet avantajı sağlamakta ve kullanıcı deneyimini önemli ölçüde iyileştirmektedir.

ABSTRACT

This study addresses the development of a rule-based e-commerce platform designed to enhance user experience and increase customer loyalty in the modern e-commerce industry. The inadequacy of uniform discount strategies and the lack of personalization in traditional e-commerce systems pose significant challenges in customer acquisition and retention. To address this issue, a system was developed using the Spring Framework infrastructure, supported by a MySQL database. Security solutions such as JWT and BCrypt were implemented in the backend to ensure secure and convenient login processes, with user passwords stored in an encrypted format. Thanks to Spring's layered architecture, the platform was built with scalability in mind, reducing code complexity and improving maintainability. The system collects user behavior data in the database and performs calculations in the service layer. As a result, a rule-based structure that offers personalized discounts was implemented. On the frontend, HTML, CSS, and JavaScript technologies were used to design a user-friendly interface. JavaScript, in particular, was utilized to dynamically display discounts in real-time for relevant products when users qualified for specific category-based offers. The developed platform monitors users' shopping habits and preferences, enabling automated discount mechanisms to be triggered under certain conditions. For instance, special offers may be presented to users who show interest in a particular product category but do not make a purchase, or exclusive discounts may be provided to loyal customers. In conclusion, this study provides a dynamic and effective marketing tool for e-commerce businesses, offering a competitive advantage and significantly improving the user experience.

1. GİRİŞ

Geleneksel e-ticaret siteleri, uzun yıllardır dijital alışverişin temelini oluşturmakla birlikte, günümüz rekabetçi pazar ortamında kullanıcı beklentilerini karşılamakta yetersiz kalmaktadır. Bu sitelerin en belirgin sorunlarından biri, tüm kullanıcılara aynı şekilde uygulanan ve kişiselleştirme içermeyen indirim politikalarıdır. Her kullanıcıya benzer kampanyaların sunulması, farklı ilgi alanlarına ve alışveriş alışkanlıklarına sahip bireylerin ihtiyaçlarını göz ardı eder. Kullanıcının önceki alışveriş geçmişi, gezinti davranışları, tıklama alışkanlıkları veya sepette beklettiği ürünler gibi değerli veriler analiz edilmeden yapılan indirim stratejileri, çoğu zaman düşük etkileşim ve satışla sonuçlanır. Bu durum, yalnızca müşteri memnuniyetini azaltmakla kalmaz, aynı zamanda e-ticaret platformlarının gelir potansiyelini de sınırlar. Kişiselleştirilmiş pazarlama stratejilerinin hızla önem kazandığı günümüzde, kullanıcı davranışlarını analiz etmeyen ve buna uygun teklifler sunmayan sistemler, sadık müşteri kitlesi oluşturmakta başarısız olur. Ayrıca, her kullanıcıya aynı indirim sunmak, bazı kullanıcılar için gereksiz teşvikler yaratırken, bazı kullanıcılar için yeterince cazip olmayan teklifler haline gelebilir. Bu bağlamda, veri temelli, kullanıcı odaklı ve dinamik indirim sistemlerinin geliştirilmesi, e-ticaret sektöründe rekabet avantajı sağlayan temel unsurlardan biri haline gelmiştir.

1.1. Bitirme Projesinin Amacı

Günümüzde hızla büyüyen e-ticaret sektöründe, kullanıcıların platforma çekilmesi ve bağlılıklarının artırılması, rekabet avantajı sağlayan temel unsurlardan biridir. Ancak, geleneksel e-ticaret sistemlerinde uygulanan tek tip indirim stratejileri, kullanıcı davranışlarını yeterince dikkate almadığından, müşteri kazanımı ve sadakati açısından sınırlı bir etki yaratmaktadır. Bu durum, kullanıcıların alışveriş deneyimlerinin kişiselleştirilmemesi ve genel indirimlerin etkisiz kalması gibi problemlere yol açmaktadır.

Bu çalışmanın amacı, kullanıcı davranışlarını analiz ederek kişiselleştirilmiş indirimler sunan kural tabanlı bir sistem geliştirmektir. Sistem, kullanıcıların alışveriş alışkanlıklarını ve site üzerindeki etkileşimlerini değerlendirerek, bireysel ihtiyaçlara uygun indirimler tanımlayabilmektedir. Örneğin, belirli bir ürün kategorisine sıkça ilgi gösteren ancak satın alma yapmayan kullanıcılara özel teklifler sunulmakta ya da yüksek harcama yapan sadık müşterilere özel indirimler sağlanmaktadır.

Geliştirilen sistem, makine öğrenmesinden farklı olarak belli parametrelere ve bu parametrelerin değerlendirileceği bir formüle dayandırılmıştır. Formüle etme işlemi diğerlerinden ayrılmaktadır ve parçalı fonksiyona dayandırılmıştır.

1.2. Literatür Özeti

E-ticarete kişiselleştirme ve kullanıcı deneyimini iyileştirme çabaları, özellikle tavsiye sistemlerinin gelişimiyle birlikte önemli bir ivme kazanmıştır. Ricci, Rokach ve Shapira'nın [1] yanı sıra Jannach, Zanker, Felfernig ve Friedrich'in [2] çalışmaları, bu sistemlerin temel prensiplerini, mimarilerini ve e-ticaretteki uygulama alanlarını detaylı biçimde açıklamaktadır. Bu sistemler, kullanıcıların geçmiş davranışları, demografik özellikleri ve benzer kullanıcıların tercihleri doğrultusunda ürün veya hizmet önerileri sunarak alışveriş deneyimini kişiselleştirmeyi hedefler.

Kişiselleştirilmiş indirim stratejileri de bu sistemlerin önemli bir uzantısı olarak değerlendirilmekte olup, kullanıcıların fiyat hassasiyetine dayalı öneriler sunulması bu alandaki etkinliği artırmaktadır [3]. Sun, Guo ve Barnes [4], fiyat duyarlılığı ve öneri sistemleri entegrasyonu üzerine yaptıkları çalışmada, kişiye özel indirimlerin kullanıcı davranışını olumlu yönde etkilediğini göstermiştir. Wang, Li ve Zhang'ın [5] mevsimsel ve zamansal faktörlere dayanan bağlamsal öneri sistemi çalışması, bu tür verilerin stratejik olarak kullanılabileceğini ortaya koymaktadır. Bunun yanı sıra, Adomavicius ve Tuzhilin [6] bağlamsal farkındalığın öneri sistemlerinde kritik bir rol oynadığını savunmuştur.

Yakın dönem literatürde, oyunlaştırma da kişiselleştirilmiş kullanıcı deneyimi sağlamada etkili bir yöntem olarak öne çıkmaktadır. Örneğin, Hamari ve Sarsa [7] ile Xu ve Hamari [8], oyunlaştırmanın kullanıcı etkileşimini artırmadaki etkilerini tartışmışlardır. Özellikle Sadri, Eirinaki ve Varlamis [9], oyunlaştırma temelli öneri sistemlerinin kullanıcı sadakati üzerindeki pozitif etkilerini ortaya koymuştur.

Kullanıcı deneyimini etkileyen estetik faktörler de göz ardı edilmemelidir. Hall ve Hanna [10], web tasarımı ve renk kontrastlarının kullanıcı davranışına etkisini incelerken; Kaya ve Epps [11], Labrecque ve Milne [12], Singh [13] gibi araştırmalar renklerin pazarlama üzerindeki psikolojik etkilerine dikkat çekmiştir. Bu çalışmalar, renklerin ve tasarımın sadece görsellik değil, aynı zamanda kullanıcı algısı ve kararları üzerinde doğrudan etkili olduğunu göstermektedir.

Güvenlik ve performans açısından da sistem altyapısının ölçeklenebilir ve güvenli olması gerekmektedir. Jones, Bradley ve Sakimura tarafından tanımlanan JSON Web Token (JWT) [14] ile OAuth 2.0 [15] gibi teknolojiler, güvenli oturum yönetimi ve yetkilendirme için günümüzde yaygın olarak tercih edilmektedir. Özellikle Chen ve Zhao [16], e-ticaret sistemlerinde kullanıcı güvenliğini sağlamak için JWT avantajlarını belirtmiştir.

Son yıllarda yapılan çalışmalar arasında Zhang ve Liu'nun [17] yapay zekâ destekli öneri sistemlerinin kullanıcı davranışı üzerindeki etkilerine dair çalışması; Lin ve Yeh'in [18] gerçek zamanlı öneri sistemleri geliştirme üzerine çalışması ve García-Mendoza et al.'ın [19] makine öğrenmesi algoritmalarının tavsiye sistemlerine entegrasyonu üzerine çalışmaları öne çıkmaktadır.

2. MATERYAL VE YÖNTEM

E-ticaret sitemizin geliştirilmesinde temel altyapı ve uygulama çatısı olarak Spring Framework tercih edilmiştir. Spring Framework'ün modüler yapısı, projenin farklı katmanlarını (sunum, iş mantığı, veri erişimi) birbirinden bağımsız ve esnek bir şekilde geliştirmemize olanak tanımıştır. Özellikle Spring MVC modülü, kullanıcı arayüzünden gelen isteklerin karşılanması, uygun iş mantığı servislerine yönlendirilmesi ve sonuçların kullanıcıya sunulması süreçlerinde merkezi bir rol oynamaktadır. Bu sayede, kullanıcı kayıt, ürün listeleme, sepet işlemleri ve sipariş yönetimi gibi temel e-ticaret fonksiyonları için RESTful API uç noktaları ve web arayüzleri etkin bir şekilde oluşturulmuştur.

2.1. Kullanılan Teknolojiler

Projemiz için çeşitli teknoloji çözümlerinden faydalanılmıştır. Aşağıda projenin her bir bölümü için kullanılan çözümler yer almaktadır:

Backend: Spring Boot 2.x

Frontend: HTML, CSS ve Javascript ile Thymeleaf temaları

Veri tabanı: MySQL ile JPA/Hibernate

E-posta: E-posta doğrulaması için Spring Mail

Build Aracı: Maven

Güvenlik: Spring Security ve JWT

2.1.1 Html, css ve javascript kullanımı

E-ticaret sitemizin kullanıcı arayüzünün (ön yüzünün) oluşturulması ve kullanıcı deneyiminin zenginleştirilmesi amacıyla temel web teknolojileri olan HTML, CSS ve JavaScript etkin bir şekilde kullanılmıştır.

HTML (HyperText Markup Language), web sayfalarımızın temel iskeletini ve içeriğini yapılandırmak için kullanılmıştır. Ürün listeleme sayfaları, ürün detay sayfaları, kullanıcı profili, alışveriş sepeti ve ödeme adımları gibi tüm sayfaların semantik yapısı *HTML* etiketleri kullanılarak oluşturulmuştur. Bu sayede, içeriğin arama motorları tarafından daha

iyi anlaşılması ve erişilebilirlik standartlarına uygunluk hedeflenmiştir. Formlar, tablolar, başlıklar ve paragraflar gibi temel içerik elemanları *HTML* ile tanımlanmıştır.

CSS (Cascading Style Sheets), *HTML* ile oluşturulan bu yapısal iskeletin görsel sunumunu ve estetiğini sağlamak amacıyla kullanılmıştır. Sitenin genel renk paleti, tipografi, sayfa düzenleri (layout), elemanların konumlandırılması, boşluklar, kenarlıklar ve gölgeler gibi tüm stil tanımlamaları *CSS* ile gerçekleştirilmiştir.

Javascript, indirimleri sepette ve anasayfada ürünlerin normal fiyatının üstü çizilerek dinamik bir şekilde görüntülenmesi sağlanmıştır. JavaScript kullanımı bu projede sınırlı kalmıştır.

2.1.2 Mysql veri tabanı kullanımı

E-ticaret sitemizin tüm dinamik verilerinin kalıcı olarak saklanması, yönetilmesi ve sorgulanması amacıyla ilişkisel veritabanı yönetim sistemi (RDBMS) olarak *Mysql* tercih edilmiştir. *Mysql*'in yaygın kullanımı, güvenilirliği, performansı ve açık kaynak olması bu tercihte önemli rol oynamıştır.

E-ticaret sistemi, veri bütünlüğünü sağlamak, tekrarları ortadan kaldırmak ve sorgu performansını optimize etmek amacıyla veritabanı normalizasyon prensiplerine uygun olarak tasarlanmıştır. Veritabanı şeması, Üçüncü Normal Form (3NF) seviyesine kadar normalize edilmiş ve performans optimizasyonu için gerekli yerlerde stratejik denormalizasyon uygulanmıştır.

2.1.2.1 Temel Varlıklar ve Birinci Normal Form (1NF)

Veritabanı tasarımı, Birinci Normal Form uyumluluğunu tekrarlayan grupları ortadan kaldırarak ve atomik değerler sağlayarak garanti eder:

Kullanıcı Varlığı (user tablosu):

Birincil anahtar: id (otomatik oluşturulan)

Atomik özellikler: username, email, password, full_name, role, age, gender

Tekrarlayan gruplar veya çok değerli özellikler yok

Tekrarları önlemek için username ve email üzerinde benzersiz kısıtlamalar

Kategori Varlığı (category tablosu):

Birincil anahtar: id (otomatik oluşturulan)

Atomik özellikler: name (benzersiz kısıtlama)

Bileşik veya çok değerli özellikler olmadan basit, atomik yapı

Ürün Varlığı (product tablosu):

Birincil anahtar: id (otomatik oluşturulan)

Atomik özellikler: product_name, price, image

Yabancı anahtar: category_id (Category.id'yi referans eder)

Tüm özellikler tek, atomik değerler içerir.

2.1.2.2 İkinci Normal Form (2NF) Uygulaması

Tasarım, tüm anahtar olmayan özelliklerin birincil anahtara tam olarak işlevsel bağımlı olduğundan emin olarak İkinci Normal Form'u sağlar:

Sepet ve Sepet Ögesi Ayrımı:

cart tablosu: Sepet seviyesi bilgileri içerir (id, user_id, total_amount).

cart_item tablosu: Öge seviyesi bilgileri içerir.
(id, cart_id, product_id, quantity, product_price, total_price)

Bu ayrım, sepet ögesi detaylarının hem sepet hem de ürüne bağımlı olacağı kısmi bağımlılıkları ortadan kaldırır.

Kullanıcı Etkileşimi Varlığı:

Birincil anahtar: id (otomatik oluşturulan)

Tüm özellikler (user_id, category_id, product_id, interaction_type, timestamp, value) birincil anahtara tam olarak bağımlıdır. Kısmi bağımlılık yoktur.

2.1.2.3 Üçüncü Normal Form (3NF) Uyumluluğu

Veritabanı, geçişli bağımlılıkları ortadan kaldırarak Üçüncü Normal Form'u sağlar:

Sipariş Geçmişi Varlığı:

Birincil anahtar: id (otomatik oluşturulan).

Bağımlılıklar: order_number, customer_name, delivery_address, order_items, total_amount, order_date, user_id

Ara özellikler aracılığıyla geçişli bağımlılıklar yok.

Favoriler Varlığı:

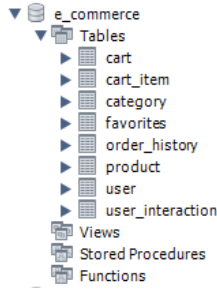
Bileşik ilişkilerle junction table tasarımı

Birincil anahtar: id (otomatik oluşturulan)

Yabancı anahtarlar: user_id, product_id, category_id

Tüm özellikler birincil anahtara doğrudan bağımlı

2.1.2.4 Veri tabanı üzerindeki tablolar ve ilişkiler



Şekil 2. 1: Mysql workbench üzerindeki şemanın ekran alıntısı

Sistemimizde *MySQL*, aşağıdaki temel veri kategorilerini depolamak için kullanılmaktadır:

Şekil 2.1’de belirtilen *e_commerce* veri tabanı şemasına göre, sistemimizde *MySQL* aşağıdaki temel veri kategorilerini ve tablolarını içermektedir:

Kullanıcı Bilgileri (*user* tablosu), üye olan kullanıcıların kimlik bilgileri (örneğin, kullanıcı adı, şifrelenmiş parola), e-posta adresi ve yaş gibi temel verileri ve kullanıcıların e-posta kaydı yapıp yapmadığına dair boolean veri tipinde bir sütun ve *JWT* ile ilgili bilgileri yer almaktadır.

Ürün Kataloğu (*product* ve *category* tabloları), satışa sunulan her bir ürünün detaylarını (ürün ID, adı, açıklaması, fiyatı, görsel yolları vb.) ve ürünlerin sınıflandırıldığı kategorileri (kategori ID, kategori adı vb.) barındırır. *product* tablosu ile ilişkilendirilerek ürünlerin hangi kategoriye ait olduğu belirlenir.

Sepet Bilgileri (*cart* ve *cart_item* tabloları): *cart* tablosu: Her kullanıcının aktif alışveriş sepetini temsil eder. Genellikle kullanıcı ID’si ile ilişkilidir ve sepetin toplam tutarı gibi özet bilgiler içerebilir.

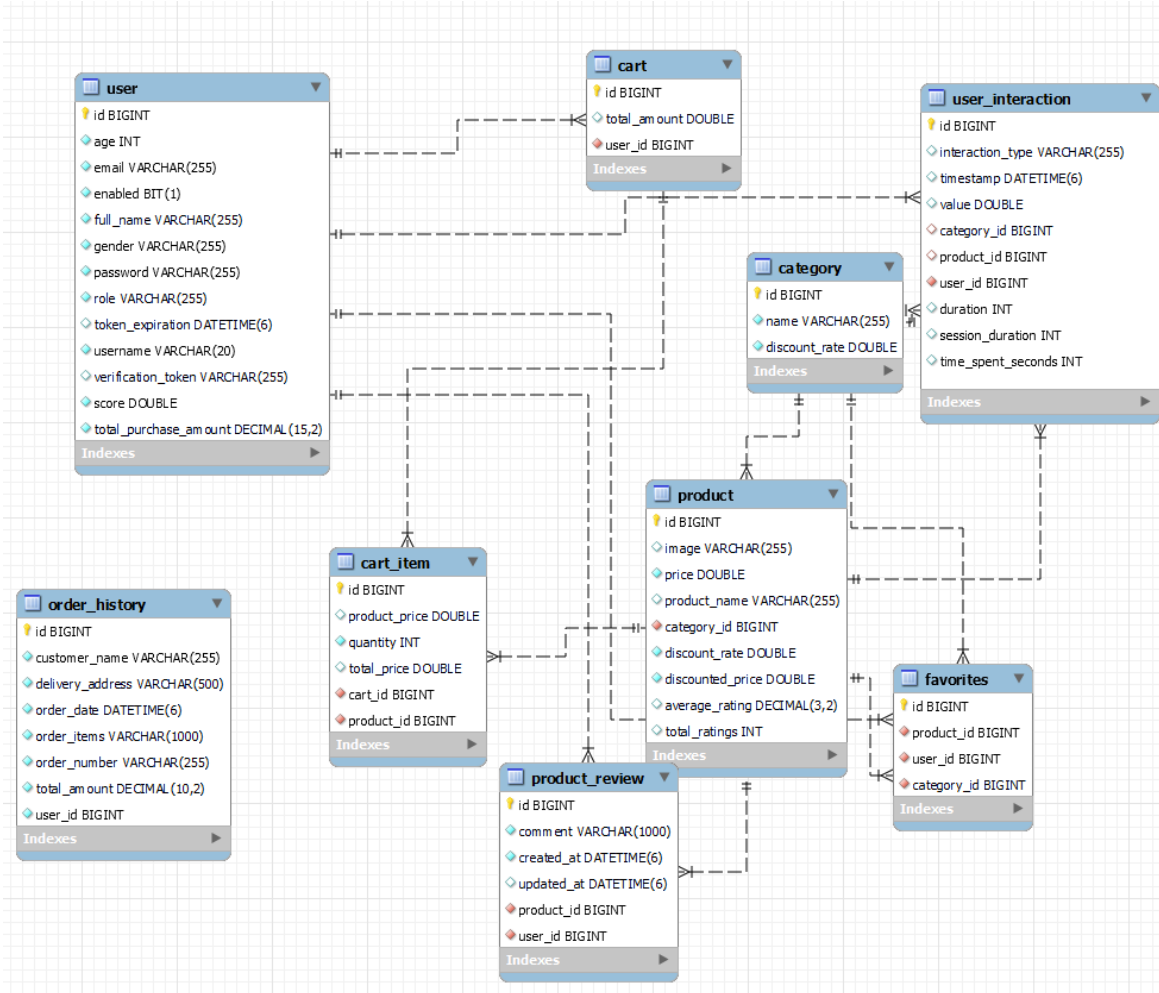
cart_item tablosu: Bir sepette bulunan her bir ürünü ayrı bir kayıt olarak tutar. Hangi sepete ait olduğu (*cart* ID), hangi ürün olduğu (*product* ID), ürün adedi ve o anki fiyatı gibi bilgileri içerir.

Favori Ürünler (favorites tablosu): Kullanıcıların beğendikleri ve daha sonra kolayca erişmek üzere işaretledikleri ürünlerin kaydını tutar. Kullanıcı ID'si ve ürün ID'si arasında bir ilişki kurar.

Kullanıcı Etkileşimleri (user_interaction tablosu):

Kullanıcıların sitedeki önemli hareketlerini ve etkileşimlerini (ürüne tıklama, sepete ekleme ve favorilere ekleme) kaydetmek için kullanılır.

Bu veriler, kullanıcı deneyimini analiz etmek, kişiselleştirilmiş öneriler sunmak ve kural tabanlı indirim sistemini beslemek için değerli bir kaynaktır. Şekil 2.2’de gösterilen şekilde ilişki kurulmuştur.



Şekil 2. 2:Veri tabanı şeması

Backend üzerindeki **Entity** sınıflarının **Hibernate (JPA)** ile otomatik tablo ve veri sütunları oluşturması gibi özelliklerden “2.3.1 Entity Sınıfları” bölümünde bahsedilmiştir.

Kullanıcının parola bilgileri veri tabanına *BCrypt* şifreleme yöntemiyle kaydedilmektedir. Encode şekilde veri tabanında saklanan parola bilgileri gerektiğinde tekrar decode edilerek kullanılmaktadır. Böylece olası veri tabanı bilgi sızıntılarında dahi kullanıcıların parola bilgilerine kesin olarak erişmek mümkün olmaz (Bu durum ancak ilgili parolanın daha önce sızdırılmadığı durumlarda geçerlidir).

2.2. Kullanılan Teknolojilerin Test Edildiği ve Çalıştırıldığı Ortamlar

Sitemizin arka ve ön yüzü için kodlama, test etme ve Maven yardımıyla ayağa kaldırabilmek için IntelliJ IDEA isimli ortamı kullandık. Bu ortam bize kod yazarken kolaylık sağladığı gibi Spring içerisindeki notasyonları da kolayca entegre edebiliyor.

Ayrıca Spring Boot eklentilerini de *pom.xml* dosyasının içinde tutarak rahatça ekleme, silme ve güncelleme işlemleri yapabiliyoruz.

Proje; Chrome, Firefox ve Microsoft Edge tarayıcıları üzerinde ayağa kaldırıldığında sorunsuz çalıştığı gözlemlenmiştir. HTML, CSS ve JS testleri manuel olarak yapılmıştır. Küçük ölçekli projede JUnit gibi birim testlerinin yazılmasına ihtiyaç duyulmamıştır.

Maven ile build edilmiş ve yeni eklenen kütüphaneler ve bağımlılıklar komutlarla test edilmiştir. Örneğin; *maven clean* komutu, önceden kalan derlemeleri temizler ve çakışmaların önüne geçer. *Maven install* komutu ise projeyi derler, test eder ve Maven içerisine kurar. Projede sık sık güncellemeler yapıldığı için *maven clean install* komutu da sürekli kullanılmıştır. Yapılandırmalar ile ilgili detaylara “2.8 Yapılandırmalar” kısmında yer verilmiştir.

Web sitesi yerel ağlarda çalışabilecek şekilde tasarlanmıştır, herhangi bir deploy işlemi yapılmamıştır. Kolaylık olması açısından varsayılan port **8080** kullanılmıştır.

2.3. Dosya Düzeni ve Proje Hiyerarşisi

Sistemde Spring resmi dokümantasyonlarından elde ettiğimiz ve orada önerilen bulgulara göre tasarlanmış bir mimari yer almaktadır. Bu mimarinin amacı, koda ölçeklenebilirlik katarak daha sonra güncellenecek her şeyde geliştiriciye kolaylık sağlamasıdır.

Şekil 2.3'te görüleceği üzere Spring'in temel yapı taşları olan Model, Service ve Controller sınıfları ayrı ayrı klasör içerisinde geliştirilmiştir. Ayrıca, diğer yardımcı sınıfları işlevlerine göre kategorize ederek kod mimarisinin daha rahat anlaşılabilmesi sağlanmıştır.

Model katmanındaki sınıflar projede **Entity** sınıflarına karşılık gelmektedir, veri tabanında her bir sınıf adı tablo adını temsil eder. Model katmanının sorumluluğu, servis katmanında işlenecek verilerin sağlanmasıdır. Hangi verilerin nasıl çekileceği Spring tarafından sağlanan ve **Repository** notasyonlu sınıflarda bulunan **native olmayan sorgular** ile sağlanmaktadır.

Aşağıdaki şekilde görüleceği üzere ilgili kullanıcının sorgusu veri tabanındaki 'username' sütunundaki verilerle yapılmaktadır.

```
@Repository 9 usages 1 unknown
public interface UserRepository extends JpaRepository<User, Long> {

    Optional<User> findByUsername(String username); 2 usages 1 unknown
    Optional<User> findByVerificationToken(String token); 1 usage 1 unknown
    boolean existsByEmail(String email); 1 usage 1 unknown
}
```

Şekil 2. 3: UserRepository sınıfına ait ekran görüntüsü

Daha sonra **UserRepository** sınıfından servis katmanında bir nesne oluşturulur ve bu nesne sayesinde istenen veriler servis katmanına taşınır. Bu şekilde **Repository**, **Entity** ve **Service** sınıfları arasında bir iş dağılımı yapılmış ve kod karmaşasının önüne geçilmiş oldu.

```
@Service 1 unknown
public class UserService {

    private final UserRepository userRepository; 13 usages
    private final PasswordEncoder passwordEncoder; 3 usages
}
```

Şekil 2. 4: UserService sınıfından ekran görüntüsü

Örneğin; Şekil 2.5'te gösterilen *userRepository* nesnesinin yardımıyla **yeni bir kullanıcı kaydı** yapılmaktadır. Nesnenin yardımıyla *Dto* nesnesi içerisinde gelen verideki e-posta kontrol edilir ve bu e-postanın aynısıyla daha önceden bir kullanıcı oluşturup oluşturulmadığını bu şekilde kontrol ettik.

```
// Kullanıcı kaydı ve doğrulama tokeni oluşturma
public User registerNewUser(UserDto userDto) {
    if (userRepository.existsByEmail(userDto.getEmail())) {
        throw new RuntimeException("Bu e-posta zaten kayıtlı!");
    }
    User user = new User();
    user.setFullName(userDto.getFullName());
    user.setUsername(userDto.getUsername());
    user.setEmail(userDto.getEmail());
    user.setPassword(passwordEncoder.encode(userDto.getPassword()));
    user.setRole("USER");
    user.setAge(userDto.getAge());
    user.setGender(userDto.getGender());
}
```

Şekil 2. 5: Kullanıcı kaydı yapan servis fonksiyonu

Veri tabanı katmanından servis katmanına sağlanan faydalar bu şekildeydi. Şimdiyse **Controller** sınıflarının **Service** sınıflarıyla olan ilişkisine ve projedeki yerini anlatalım.

Aşağıdaki Şekil 2.6’da *UserController* sınıfında *UserService* sınıfının nesnesi kullanılmıştır. Örneğin; `/userId` olarak tanımlanan end-point üzerinde yapılan işlemde nesnenin *getOneUser* metodu çağırılmıştır. Bu metodun yapacağı işlemler basit veya kompleks fark etmeksizin servis katmanında belirlendiği için **Controller** sınıfındaki kod karmaşası ortadan kalkmış oluyor.

```
@RestController
@RequestMapping("/user")
public class UserController {

    private UserService userService;

    public UserController(UserService userService) { this.userService = userService; }

    @GetMapping
    public List<User> getAllUsers() { return userService.getAllUsers(); }

    @GetMapping("/{userId}")
    public Optional<User> getOneUser(@PathVariable Long userId) { return userService.getOneUser(userId); }

    @PostMapping
    public User createOneUser(@RequestBody User newUser) { return userService.createOneUser(newUser); }

    @PutMapping("/{userId}")
    public User updateOneUser(@PathVariable Long userId, @RequestBody User updatedUser) {
        return userService.updateOneUser(userId, updatedUser);
    }

    @DeleteMapping("/{userId}")
    public void deleteOneUser(@PathVariable Long userId) { userService.deleteOneUser(userId); }
}
```

Şekil 2. 6: UserController sınıfına ait ekran görüntüsü

Böylece bu bölümde sistemde Spring'in sunmuş olduğu katmanlı yapı kullanarak MVC (Model-View-Controller) mimarisine uygun bir site tasarlandığı anlatılmıştır.

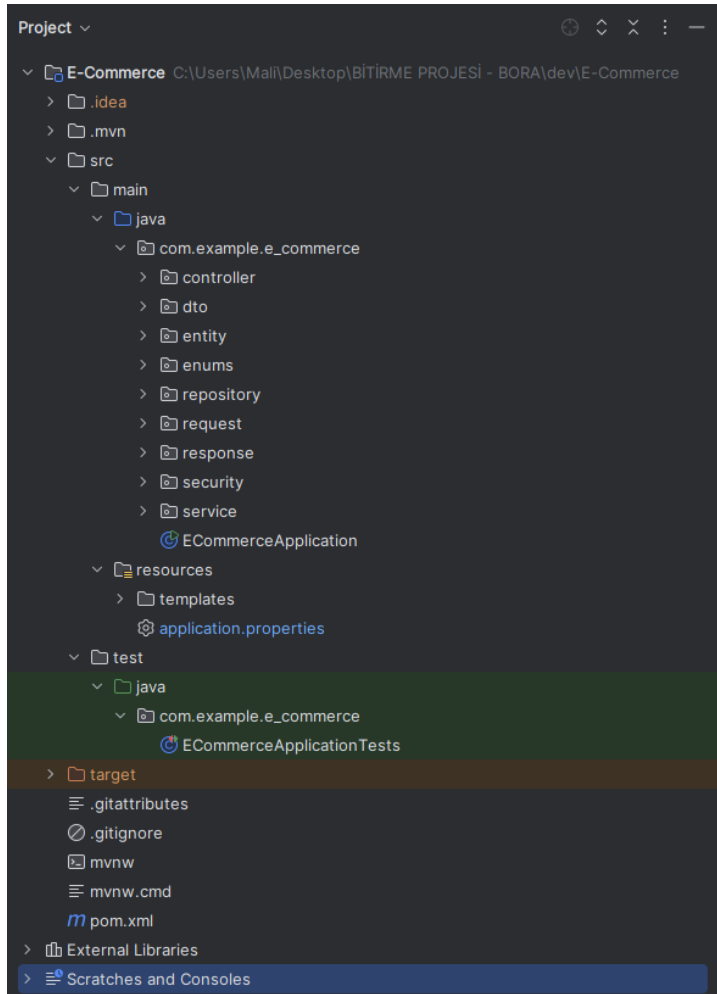
Projenin dosya mimarisinde bahsedilen sınıflar uygun şekilde isimlendirilen dosya dizinlerinin içlerine yerleştirilmiştir. Bu sayede yeni bir sınıf oluşturulacağında aşağıdaki düzen baz alınabilir:

Yeni (Entity) – YeniRepository – YeniService – YeniController

Yukarıdaki düzenle birlikte kolayca yeni özellikler eklenebilir.

Projemizde şu dizinlerle sınıflandırma yapılmış ve dosya hiyerarşisi sağlanmıştır: *controller, dto, entity, enums, repository, request, response, security, service*

Şekil 2.7'de, bahsedilen bu dosya düzeninin IDE ortamındaki görüntüsüne kolayca anlaşılabilmesi açısından yer verilmiştir.



Şekil 2. 7: Projenin IDE üzerindeki dosya mimarisine ait ekran görüntüsü

2.3.1 Entity sınıfları

Entity (veri tabanındaki tablolara denk gelmektedir) sınıflarında, projenin ayağa kaldırılması sırasında ilgili şemanın veri tabanında bulunması halinde otomatik olarak tabloların oluşturulacağı **Hibernate (JPA – Java Persistence API)** teknolojisi kullanılmıştır. Bu teknoloji sayesinde *@Entity* notasyonu ve *@Table* notasyonunun yardımıyla veri tabanında otomatik tablo oluşturulur. *@Id*, *@GeneratedValue* ve *@OneToMany* gibi özelliklerin rahatça implement edilmesine olanak sağlar. Ayrıca **getter-setter** fonksiyonlarının karmaşasından kurtulmak amacıyla **Lombok** kütüphanesinden faydalanılmıştır. Şekil 2.8’te görüleceği üzere belirlenen her nesneye ait ayarlamalar Spring notasyonlarıyla kolayca sağlanmıştır.

Özetle; *@Data* ile Lombok kütüphanesi kullanılmış, *@Entity* ve *@Table* notasyonlarıyla **Hibernate JPA** ile tablo oluşturulması sağlanmış, *@NotBlank* ile ilgili sütunun boş bırakılamayacağı belirtilmiş, *@Id* ile eşsiz bir kimlik sağlanmıştır.

```
@Data
@Entity
@Table(name = "user")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotBlank
    @Size(min = 3, max = 20)
    @Column(nullable = false, unique = true)
    private String username;

    @NotBlank
    @Email
    @Column(nullable = false, unique = true)
    private String email;

    @NotBlank
    @Size(min = 6)
    @Column(nullable = false)
    private String password;

    @NotBlank
    @Column(name = "full_name")
    private String fullName;

    @NotBlank
    @Column(nullable = false)
    private String role;

    @NotNull
    @Column(nullable = false)
    private int age;

    @NotBlank
    @Column(nullable = false)
    private String gender;
}
```

Şekil 2. 8: Entity klasörü içerisinde yer alan "user" sınıfına ait ekran görüntüsü

Aşağıdaki listede kullanılan tüm Entity sınıfları ve neden kullanıldığına dair açıklama metinleri yer almaktadır.

User Tablosu:

Id, kullanıcının veri tabanındaki eşsiz anahtarını ifade eden bir özelliktir. **Username**, kullanıcının adını tutar ve sistemde diğer kullanıcılarla ayırt edilmesini sağlar. **Email**, kullanıcının e-posta adresini saklayan bir alandır. **Password**, kullanıcının şifresini güvenli bir şekilde şifrelenmiş olarak tutar. **FullName**, kullanıcının adını ve soyadını bir arada barındırır. **Role**, kullanıcının sistemde sahip olduğu yetkiyi ve rol bilgisini ifade eder. **Age**, kullanıcının yaş bilgisini depolayan bir özelliktir. Son olarak, **Gender**, kullanıcının cinsiyet bilgisini tutar ve kullanıcıya ait kişisel bilgilerin bir parçasıdır.

UserInteraction Tablosu:

Id, tabloda tutulan kullanıcı aktivite verilerinin eşsiz anahtarını tutan bir özelliktir. **User**, aktiviteyi gerçekleştiren kullanıcıyı ifade eder ve bu kullanıcının kimliğini belirler. **Category**, aktivitenin hangi ürün kategorisinde gerçekleştirildiğini gösterir ve bu sayede aktivitenin kapsamını belirler. **Product**, aktivitenin hangi ürün üzerinde gerçekleştirildiğini ifade eder, böylece spesifik ürün bilgisi sağlanır. **InteractionType**, gerçekleştirilen aktivitenin çeşidini ifade eder ve aktivitenin türünü tanımlar.

Product Tablosu:

Id, ürünün veri tabanındaki eşsiz anahtarını ifade eder ve ürünlerin benzersiz bir şekilde tanımlanmasını sağlar. **ProductName**, ürünün ismini ifade eden bir özelliktir. **Price**, ürünün fiyat bilgisini tutar ve ürünün maliyetini belirtir. **Image**, ürüne ait görselin URL bilgisini saklar ve ürünün görsel temsilini sağlar.

Favorites Tablosu:

Id, favoriye eklenen ürünlerin eşsiz anahtarını tutar ve her favori kaydını benzersiz bir şekilde tanımlar. **User**, ürünü favoriye ekleyen kullanıcının bilgisini ifade eder. **Product**, favoriye eklenen ürünü belirtir ve hangi ürünün favorilere eklendiğini gösterir.

Cart Tablosu:

Id, sepetin veri tabanındaki eşsiz anahtarını tutar ve her bir sepeti benzersiz bir şekilde tanımlar. **User**, sepetin hangi kullanıcıya ait olduğunu belirten bilgiyi içerir. **TotalAmount**,

sepetteki toplam tutarı ifade eder ve sepetin toplam maliyetini gösterir. **CartItems**, sepette yer alan ürünleri tutar ve sepetin içeriğini detaylı bir şekilde listeler.

CartItem Tablosu:

Id, sepet öğesinin veri tabanındaki eşsiz anahtarını ifade eder ve her bir sepet öğesini benzersiz bir şekilde tanımlar. **Cart**, sepet öğesinin ait olduğu sepeti belirtir ve bu öğenin hangi sepete dahil olduğunu gösterir. **Product**, sepet öğesinde bulunan ürünü ifade eder ve hangi ürünün sepete eklendiğini belirtir. **Quantity**, sepet öğesindeki ürünün adet bilgisini tutar ve ürünün kaç tane eklendiğini gösterir. **ProductPrice**, sepet öğesindeki ürünün birim fiyatını içerir. **TotalPrice**, sepet öğesindeki ürünlerin toplam fiyatını tutar ve ürün miktarı ile birim fiyatın çarpımı sonucu oluşan tutarı ifade eder.

OrderHistory Tablosu:

Id, siparişin veri tabanındaki eşsiz anahtarını ifade eder ve her bir siparişi benzersiz bir şekilde tanımlar. **OrderNumber**, sipariş numarasını ifade eder ve siparişe özel bir kimlik sağlar. **CustomerName**, sipariş veren kullanıcının ismini belirtir ve sipariş sahibini tanımlar. **DeliveryAddress**, siparişin teslim edileceği adresi belirtir. **OrderItems**, sipariş verilen öğeleri içerir ve siparişin içeriğini detaylandırır. **TotalAmount**, siparişin toplam tutarını ifade eder ve tüm siparişin maliyetini gösterir. **OrderDate**, siparişin verildiği tarih ve zamanı belirtir. **UserId**, sipariş veren kullanıcının veri tabanındaki eşsiz anahtarını ifade eder ve kullanıcı ile sipariş arasında bağlantı kurar.

Category Tablosu:

Id, kategorinin veri tabanındaki eşsiz anahtarını ifade eder ve her bir kategoriye benzersiz bir şekilde tanımlar. **Name**, kategorinin ismini belirtir ve kategoriye tanımlamak için kullanılır. **Products**, kategori içerisindeki ürünlerin listesini tutar ve bu kategoride yer alan ürünleri detaylı bir şekilde gösterir.

2.3.2 Repository sınıfları

Spring Framework teknolojisinin sağladığı avantajlarla Native Query -elle SQL sorgusu yazma- yazılmasına gerek kalmadan belli söz dizimleriyle veri tabanına erişip CRUD işlemleri yapılabilir. Proje içerisinde yer alan repository sınıfları ve açıklamaları aşağıda verilmiştir.

CartItemRepository: Sepet içerisindeki ürünlerle (CartItem) ilgili veritabanı işlemlerini gerçekleştirir. Sepetteki belirli bir ürünü bulmak veya güncellemek için özel sorgular tanımlanabilir.

CartRepository: Kullanıcıya ait sepetlerin (Cart) veri tabanı işlemlerini yönetir. Kullanıcıya göre sepet bulma gibi işlemler içerir.

CategoryRepository: Ürün kategorileri (Category) ile ilgili temel veri tabanı işlemlerini sağlar. Kategori ekleme, silme, güncelleme ve listeleme işlemlerini içerir.

FavoritesRepository: Kullanıcıların favori ürünleri (Favorites) ile ilgili veritabanı işlemlerini yürütür. Belirli bir kullanıcı ve ürün için favori kaydı bulma, silme veya kullanıcıya ait tüm favori ürünleri listeleme işlemlerini içerir.

ProductRepository: Ürünler (Product) ile ilgili temel veri tabanı işlemlerini sağlar. Ürün ekleme, silme, güncelleme ve arama işlemlerini içerir.

UserInteractionRepository: Kullanıcı etkileşimleri (UserInteraction) ile ilgili veri tabanı işlemlerini gerçekleştirir. Kullanıcının ürünlerle olan etkileşim geçmişini kaydeder ve sorgular.

UserRepository: Kullanıcılar (User) ile ilgili temel veri tabanı işlemlerini yönetir. Kullanıcı ekleme, silme, güncelleme, kullanıcı adı veya e-posta ile arama gibi işlemler içerir.

OrderHistoryRepository: Kullanıcıların sipariş geçmişlerini yöneten bir sınıf olarak tasarlanmıştır. Bu repository, kullanıcı bazlı sipariş geçmişi sorgulama, kullanıcının toplam sipariş sayısını hesaplama ve müşteri ismine göre sipariş geçmişi listeleme fonksiyonlarını içermektedir.

2.3.3 Service sınıfları

Servis sınıflarında enum sınıflarında belirlenen ağırlıklara ve veri tabanından çekilen verilerle beraber hesaplama işlemleri yapılır ve bu veriler Controller sınıfına taşınır. Aşağıda projede kullanılan servis sınıfları ve açıklamalarına yer verilmiştir.

CartItemService: Kullanıcının sepetine ürün ekleme, sepetteki ürün adedini güncelleme, indirim uygulama ve sepetten ürün silme işlemlerini yönetir. Kullanıcı etkileşimlerine göre kategori bazlı indirim hesaplaması yapar.

CartService: Kullanıcıya ait sepetin oluşturulması, var olan sepetin getirilmesi, sepete ürün eklenmesi ve sepetin toplam tutarının güncellenmesi gibi işlemleri yönetir.

CustomUserDetailsService: Spring Security için kullanıcı doğrulama işlemlerini yürütür. Kullanıcı adı ile kullanıcıyı bulur ve güvenlik kontrolleri için gerekli UserDetails nesnesini döndürür.

EmailService: Kullanıcılara e-posta gönderim işlemlerini (örneğin, hesap doğrulama e-postası) şablon desteğiyle birlikte yönetir.

FavoritesService: Kullanıcının ürünleri favorilere eklemesini sağlar. Favori işlemlerini kaydeder ve kullanıcının favori ürünlerini listeler.

ProductService: Ürünlerin listelenmesi, tekil ürün getirme, ürün oluşturma ve güncelleme gibi temel ürün işlemlerini sağlar.

UserInteractionService: Kullanıcı etkileşimlerini (ürün tıklama, sepete ekleme, favoriye ekleme) kaydeder. Kullanıcı davranışına göre kategori bazlı indirim oranı hesaplar.

UserService: Kullanıcı kaydı, e-posta ile doğrulama, kullanıcı bilgilerini güncelleme, silme ve kullanıcıya ait temel işlemleri yönetir.

OrderHistoryService: Sipariş geçmişi yönetiminden sorumlu servistir. Yeni siparişlerin kaydedilmesi, sipariş detaylarının JSON formatında saklanması, kullanıcıya özel sipariş geçmişi sorgulama ve sipariş sayısı hesaplama gibi işlemleri yürütür. Özellikle sepetteki ürünleri kalıcı sipariş kayıtlarına dönüştürme işlemini gerçekleştirir.

CheckoutService: Ödeme sürecindeki hesaplamaları yöneten servistir. Sepetteki ürünlerin toplam tutarını hesaplama, kargo ücretini belirleme (100 TL üzeri alışverişlerde ücretsiz kargo) ve KDV hesaplama (%18) gibi finansal işlemleri gerçekleştirir. Tüm parasal hesaplamalar BigDecimal kullanılarak hassas bir şekilde yapılmaktadır.

2.3.4 Controller sınıfları

End-point tasarımındaki nesnelerin işlevlerini yönetmeye yarayan sınıflardır. Projede kullanılan sınıflar ve açıklamalarına yer verilmiştir.

VerificationController: Kullanıcıların e-posta doğrulama işlemlerini yönetir. Doğrulama token'ı ile hesap aktivasyonu gibi işlemleri gerçekleştirir.

UserInteractionController: Kullanıcıların ürün ve sepetle olan etkileşimlerini (tıklama, sepete ekleme, favoriye ekleme vb.) API üzerinden kaydeder ve sorgular.

UserController: Kullanıcı ile ilgili temel işlemleri (kullanıcı bilgisi getirme, kullanıcı kaydı, kullanıcı güncelleme ve silme gibi) gerçekleştirir.

ProfileController: Kullanıcı profilinin görüntülenmesi ve güncellenmesi işlemlerini yönetir. Kullanıcıya özel profil sayfası sunar.

IndexController: Ana sayfa ve ürün listeleme işlemlerini yönetir. Kategorilere göre ürünleri listeler, favori ürünleri ve indirim oranlarını kullanıcıya gösterir.

ProductController: Ürünlerle ilgili temel işlemleri (ürün detayını getirme, ürün ekleme, güncelleme ve silme gibi) API üzerinden gerçekleştirir.

FavoriteController: Kullanıcıların favori ürünleri ekleme/çıkarma işlemlerini yönetir. Favori ürünlerin listesini sunar.

CartItemController: Sepet içerisindeki ürünlerin eklenmesi, güncellenmesi ve silinmesi işlemlerini yönetir.

CartController: Kullanıcıya ait sepetin görüntülenmesi, güncellenmesi ve sepet işlemlerinin yönetilmesini sağlar.

AuthController: Kullanıcı kimlik doğrulama (login), kayıt (register) ve oturum işlemlerini yönetir. JWT token oluşturma ve doğrulama işlemlerini de içerir.

CheckoutController: Ödeme sürecini yöneten bir controller'dır. Kullanıcının sepetindeki ürünler için ödeme sayfasını gösterir (*/checkout*), sipariş onayını alır (*/confirm-order*) ve başarılı sipariş sayfasını sunar (*/order-success*). Ödeme sürecinde alt toplam, kargo ücreti,

vergi ve toplam tutarı hesaplar. Kullanıcının teslimat bilgilerini, kart bilgilerini alır ve başarılı bir sipariş durumunda sepeti temizleyerek sipariş onay sayfasına yönlendirir.

OrderHistoryController: Kullanıcının geçmiş siparişlerini görüntülemesini sağlayan bir controller'dır. Tek bir endpoint (*/order-history*) üzerinden çalışır ve giriş yapmış kullanıcının tüm sipariş geçmişini listeler. Her sipariş için detaylı ürün bilgilerini parse eder ve kullanıcıya sunar. Herhangi bir hata durumunda boş bir sipariş listesi göstererek kullanıcıya hata mesajı verir.

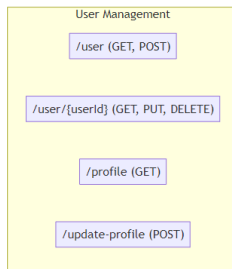
2.4. End-Point Tasarımı

Şekilde yer alan katman kullanıcıların sisteme güvenli bir şekilde giriş yapmasını, kayıt olmasını ve hesap doğrulamalarını sağlar. JWT token yönetimi, oturum kontrolü ve güvenlik protokolleri bu katmanda yönetilir.



Şekil 2. 9: Yetkilendirme ve doğrulama katmanı

Kullanıcı hesap yönetimi, profil bilgileri güncelleme ve kullanıcı verilerinin CRUD operasyonlarını içerir. Kişisel bilgilerin güvenli bir şekilde saklanması ve güncellenmesinden sorumludur.



Şekil 2. 10: Kullanıcı yönetimi katmanı

E-ticaret sisteminin temel ürün yönetimini sağlar. Ürün listeleme, detay görüntüleme, kategori bazlı filtreleme ve ürün CRUD işlemlerini içerir. Ürün kataloğu ve kategori hiyerarşisini yönetir.



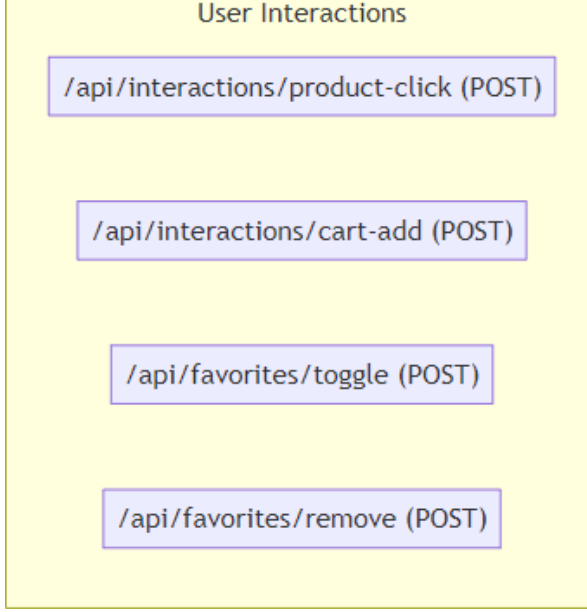
Şekil 2. 11: Ürün ve kategori katmanı

Kullanıcıların alışveriş sepeti işlemlerini yönetir. Sepete ürün ekleme, çıkarma, sepet içeriğini görüntüleme ve ödeme sürecine geçiş gibi temel e-ticaret fonksiyonlarını barındırır.



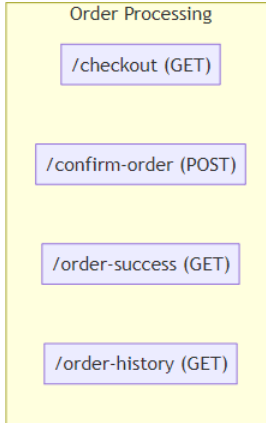
Şekil 2. 12: Alışveriş sepeti katmanı

Kullanıcı davranışlarını ve tercihlerini takip eden analitik katmandır. Favori ürünlerin yönetimi, ürün tıklama ve sepete ekleme gibi kullanıcı etkileşimlerini kaydeder. Bu veriler, kişiselleştirilmiş öneriler ve kullanıcı deneyimini iyileştirmek için kullanılır.



Şekil 2. 13: Kullanıcı aktiviteleri katmanı

Sipariş süreçlerini yöneten end-pointlerin yer aldığı katmandır.



Şekil 2. 14: Sipariş işlemleri katmanı

2.5. DTO, Request ve Response Dizinleri

DTO'lar (Data Transfer Object), projede veri alışverişini daha düzenli ve güvenli şekilde sağlamak için kullanılır. Katmanlar arasında sadece gerekli verilerin taşınmasını sağlar ve veri gizliliği ile kodun okunabilirliğini artırır.

Request sınıfları, kullanıcıdan veya istemciden sunucuya gelen verileri taşımak için kullanılır.

Response sınıfları, sunucudan istemciye gönderilecek yanıt verilerini taşımak için kullanılır. Bu kısımda projede yer alan DTO, Request ve Response sınıflarının açıklamaları detaylarıyla verilmiştir.

2.5.1 DTO sınıfları ve açıklamaları

CartInteractionDto: Kullanıcının sepetle ilgili etkileşimlerini taşımak için kullanılan veri transfer nesnesidir.

LoginRequest: Kullanıcı giriş işlemlerinde, giriş bilgilerini taşımak için kullanılan veri transfer nesnesidir.

UserDto: Kullanıcıya ait temel bilgileri taşımak için kullanılan veri transfer nesnesidir.

UserInteractionDto: Kullanıcının ürün veya kategoriyle olan etkileşimlerini taşımak için kullanılan veri transfer nesnesidir.

2.5.2 Request sınıfı ve açıklaması

CartItemRequest: Kullanıcının sepete ürün ekleme veya güncelleme işlemlerinde, ilgili ürün ve miktar bilgisini sunucuya iletmek için kullanılır.

2.5.3 Response sınıfı ve açıklaması

JwtResponse: Kullanıcıya başarılı bir girişten sonra *JWT* (token) bilgisini sunucudan istemciye iletmek için kullanılır.

2.6. Enum Sınıfları

InteractionType adındaki enum sınıfında indirimde baz alınacak parametreler ve bu parametrelerin ağırlıkları belirlenmiştir. Buradaki kriterler kullanıcı skoru sağlayacak, bu skor kullanıcıya ve kategoriye özel indirimi sağlayacaktır.


```

package com.example.e_commerce.enums;

public enum InteractionType {
    PRODUCT_CLICK(weight:0.0953),    // ürün tıklama
    DETAILS_EXPAND(weight:0.1429),    // ürün detaylarını genişletme
    RATING(weight:0.1905),           // ürün puanlama
    REVIEW_SUBMIT(weight:0.2143),     // ürün yorumu yazma
    CART_ADD(weight:0.2381),          // sepete ekleme
    FAVORITES_ADD(weight:0.2857),     // favorilere ekleme
    PURCHASED(weight:0.3809);         // satın alma

    private final double weight;

    InteractionType(double weight) {
        this.weight = weight;
    }

    public double getWeight() {
        return weight;
    }
}

```

Şekil 2. 15: InteractionType.java

Diğer enum sınıfı *UserMultiplier* içerisinde ise her kullanıcı seviyesinin alacağı seviyeler belirlenmektedir.

```

public enum UserLevelMultiplier {
    STANDART(multiplier:0.15),
    BRONZ(multiplier:0.40),
    GUMUS(multiplier:0.55),
    ALTIN(multiplier:0.80),
    ELMAS(multiplier:1.00);

    private final double multiplier;

    UserLevelMultiplier(double multiplier) {
        this.multiplier = multiplier;
    }

    public double getMultiplier() {
        return multiplier;
    }

    /**
     * Seviye adına göre enum değerini döndürür
     */
    public static UserLevelMultiplier fromLevelName(String levelName) {
        if (levelName == null) {
            return STANDART;
        }

        try {
            return valueOf(levelName.toUpperCase());
        } catch (IllegalArgumentException e) {
            return STANDART;
        }
    }

    /**
     * Satın alma tutarına göre seviyeyi belirler
     */
    public static UserLevelMultiplier fromPurchaseAmount(double amount) {
        if (amount >= 25000) return ELMAS;
        else if (amount >= 10000) return ALTIN;
        else if (amount >= 5000) return GUMUS;
        else if (amount >= 1000) return BRONZ;
        else return STANDART;
    }
}

```

Şekil 2. 16: UserMultiplier.java

2.7. Security Sınıfları

Projede Spring Security teknolojisinin sağlamış olduğu kolaylıklardan faydalanılmıştır. End-pointler üzerinde yetkilendirme sağlama, token yapılandırılmaları, veri şifreleme ve çözme (decode ve encode) gibi işlevselliklerin projeye sağlandığı kısımdır.

JWT mekanizması sayesinde, tamamen kontrolümüzde bulunan tokenin geçerlilik süresi ve imzalama yöntemi belirlenerek, oturum açan kullanıcıların aynı tarayıcı üzerinden tekrar giriş yapmak zorunda kalmadan kimlik doğrulama işlemlerinin güvenli ve kesintisiz bir şekilde sürdürülmesi amaçlanmıştır.

Aşağıda, projede kullanılan *Security* sınıflarının detaylı açıklamaları verilmiştir.

2.7.1 JWTUtil sınıfı

JwtUtil sınıfı, JSON Web Token (JWT) üretimi ve doğrulaması için yardımcı metotlar sunan bir yardımcı (utility) sınıfıdır. Modern web uygulamalarında, kullanıcı kimlik doğrulamasını stateless (durumsuz) şekilde gerçekleştirmek için JWT sıklıkla tercih edilir. Bu sınıf, token üretimi ve doğrulaması gibi temel fonksiyonları merkezi bir noktada toplamayı amaçlar.

Token Oluşturma: *generateToken(String username)* metodu, verilen kullanıcı adıyla ilişkili olarak belirlenen süre (örneğin 1 saat) için geçerli olacak bir *JWT* üretir. Token, kullanıcı adını (subject), oluşturulma ve bitiş tarihini içerir ve güvenli bir anahtar ile imzalanır.

Token'dan Kullanıcı Adı Çekme: *extractUsername(String token)* metodu, verilen *JWT*'den kullanıcı adını (subject) ayırır. Bu, kimlik doğrulama sürecinde kullanılır.

Token Doğrulama: Token'ın bütünlüğünü ve geçerliliğini kontrol eder.

2.7.2 SecurityConfig sınıfı

SecurityConfig sınıfı, Spring Security çerçevesinin uygulama üzerindeki güvenlik yapılandırmasını özelleştirmek için kullanılan bir konfigürasyon sınıfıdır. Uygulamanın hangi uç noktalarına (endpoint) kimlerin erişebileceğini, hangi kimlik doğrulama mekanizmalarının kullanılacağını ve güvenlik filtrelerinin nasıl çalışacağını belirler.

Kullanıcı Detay Servisi: Sistemdeki kullanıcıların yüklenmesi ve kimlik doğrulaması için özel bir CustomUserDetailsService kullanılır.

Şifreleme: Kullanıcı şifrelerinin güvenli şekilde saklanabilmesi için BCrypt algoritması ile şifreleme sağlanır (PasswordEncoder bean'i).

JWT Filtre Entegrasyonu: Her istekte *JWT* doğrulaması yapan özel bir filtre (*JwtAuthenticationFilter*) güvenlik zincirine eklenir.

Yetkilendirme Kuralları: Hangi endpoint'lerin herkese açık, hangilerinin ise kimlik doğrulaması gerektirdiği detaylı şekilde tanımlanır. Örneğin, */register*, */login* gibi uç noktalar herkese açıkken, */api/interactions/*** gibi uç noktalar kimlik doğrulaması gerektirir.

2.7.3 JWTAuthenticationFilter sınıfı

JwtAuthenticationFilter, Spring Security'nin filtre zincirine eklenen ve her gelen *HTTP* isteğinde *JWT*'nin geçerliliğini kontrol eden bir filtre sınıfıdır. Bu filtre, kimlik doğrulama sürecini otomatikleştirir ve her istekte kullanıcının geçerli bir token'a sahip olup olmadığını kontrol eder.

Token Elde Etme: Filtre, öncelikle *HTTP* header'ında "Authorization: Bearer ..." şeklinde bir JWT olup olmadığını kontrol eder. Eğer bulunamazsa, isteğe eklenmiş bir cookie'den *jwtToken* adında bir token arar.

Token Doğrulama ve Kullanıcı Yükleme: Eğer geçerli bir JWT bulunursa, token'dan kullanıcı adı ayrıştırılır ve sistemdeki kullanıcı detayları yüklenir. Token geçerliyse, Spring Security'nin güvenlik bağlamına (*SecurityContext*) bir kimlik doğrulama objesi eklenir.

Filtre Zincirine Devamlılık Sağlama: Kimlik doğrulama tamamlandıktan sonra, isteğin işlenmeye devam etmesini sağlar.

2.8. Yapılandırmalar

2.8.1 application.properties yapılandırması

Şekil 2. 17: application.properties dosyasına ait ekran görüntüsü

Projemizdeki bazı temel yapılandırmalar için *application.properties* dosyasının içeriğinde bazı değişiklikler ve eklemeler yapıldı.

spring.application.name=e-commerce: Uygulamanın adını tanımlar. Bu isim, Spring Boot'un bazı bileşenlerinde ve loglarda uygulamayı tanımlamak için kullanılır.

spring.jpa.hibernate.ddl-auto=update: Hibernate'in veri tabanı şeması üzerinde otomatik olarak hangi işlemleri yapacağını belirler. *Update* ise var olan tablo ve sütunları günceller, eksik olanları ekler, ancak mevcut verileri silmez. Geliştirme ortamı için uygundur.

spring.datasource.url=jdbc:mysql://localhost:3306/e_commerce: Uygulamanın bağlanacağı *MySQL* veri tabanının adresini belirtir. *Localhost*, veri tabanı sunucusunun adresi (yerel makine kullanılmıştır). **3306**, *MySQL*'in varsayılan portu, **e_commerce** ise kullanılacak veri tabanının adıdır

spring.datasource.username=root: Veri tabanına bağlanmak için kullanılacak kullanıcı adı. Genellikle *MySQL*'in varsayılan yönetici hesabıdır.

spring.datasource.password=Mali71693135: Veri tabanına bağlanmak için kullanılacak paroladır.

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver: *MySQL* veri tabanı için kullanılacak *JDBC* sürücüsünün tam sınıf adıdır. *MySQL* 8 için *com.mysql.cj.jdbc.Driver* kullanılmıştır.

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect: Hibernate'in, SQL sorgularını *MySQL* 8 sürümüne uygun şekilde oluşturmasını sağlar. Veri tabanı türüne göre optimize edilmiş SQL üretir.

spring.mail.host=smtp.gmail.com: E-posta göndermek için kullanılacak *SMTP* sunucusunun adresidir. Burada Gmail'in *SMTP* sunucusu kullanılmıştır.

spring.mail.port=587: *SMTP* sunucusunun portu. Gmail için TLS ile güvenli bağlantı kurmak amacıyla genellikle 587 kullanılır. Bu yüzden sebeple projede 587 portu kullanılmıştır.

spring.mail.username=***@gmail.com:** E-posta göndermek için kullanılacak Gmail adresi. (Gizlenmiştir)

spring.mail.password=**:** *SMTP* üzerinden e-posta gönderebilmek için kullanılacak şifre. (Gizlenmiştir)

spring.mail.properties.mail.smtp.auth=true: SMTP sunucusuna bağlanırken kimlik doğrulamanın (authentication) gerekip gerekmediğini belirtir. Gmail için gereklidir.

spring.mail.properties.mail.smtp.starttls.enable=true: SMTP bağlantısında TLS (Transport Layer Security) kullanılmasını sağlar. Böylece e-posta iletimi sırasında veri şifrelenir.

2.8.2 pom.xml yapılandırması

pom.xml dosyası, Java tabanlı projelerde kullanılan Maven yapılandırma dosyasıdır. Spring Boot projelerinde *pom.xml*, projenin bağımlılıklarını (kütüphanelerini), eklentilerini (plugin'lerini) ve proje ile ilgili temel yapılandırma ayarlarını tanımlar. Bu dosya sayesinde gerekli kütüphaneler otomatik olarak indirilir ve proje derleme, test etme gibi işlemler kolayca yönetilir.

Bizim projemizde ise *pom.xml* aşağıda listelenen ve açıklamaları verilen özellikler için kullanılmıştır:

io.jsonwebtoken:jjwt-api: JWT (JSON Web Token) oluşturma ve doğrulama işlemleri için temel API'leri sağlar.

io.jsonwebtoken:jjwt-impl: JWT işlemlerinin arka planda çalışmasını sağlayan uygulama bileşenlerini içerir.

io.jsonwebtoken:jjwt-jackson: JWT işlemlerinde JSON dönüşümleri için Jackson kütüphanesini kullanır.

org.springframework.boot:spring-boot-starter-mail: E-posta gönderimi için gerekli Spring Boot yapılandırmalarını içerir.

org.springframework.boot:spring-boot-starter-data-jpa: Veritabanı işlemleri için Spring Data JPA desteği sağlar.

org.springframework.boot:spring-boot-starter-security: Uygulamaya güvenlik, kimlik doğrulama ve yetkilendirme özellikleri ekler.

org.springframework.boot:spring-boot-starter-thymeleaf: Web sayfaları için Thymeleaf şablon motorunu projeye ekler.

org.springframework.boot:spring-boot-starter-validation: Veri doğrulama işlemleri için gerekli bileşenleri sağlar.

org.springframework.boot:spring-boot-starter-web: Web uygulaması geliştirmek için temel Spring Boot bileşenlerini ekler.

org.thymeleaf.extras:thymeleaf-extras-springsecurity6: Thymeleaf şablonlarında Spring Security entegrasyonu sağlar.

org.springframework.boot:spring-boot-devtools: Geliştirme sırasında otomatik yeniden başlatma ve kolaylıklar sunar.

com.mysql:mysql-connector-j: MySQL veritabanına bağlantı için gereken JDBC sürücüsüdür.

org.projectlombok:lombok: Kodu kısaltmaya ve sadeleştirmeye yarayan anotasyonlar sunar.

org.springframework.boot:spring-boot-starter-test: Testler için gerekli Spring Boot ve JUnit bileşenlerini içerir.

org.springframework.security:spring-security-test: Spring Security ile ilgili test işlemleri için kullanılır.

2.9. E-Posta Doğrulama Sistemi

Projemizde kullanıcı kayıt işlemi yapılırken düzensiz hesap açma işlemini önlemek ve doğrulanmış hesapların giriş yapabilmesi için posta doğrulama sistemi kullanılmıştır.

EmailService.java dosyası içerisindeki detaylara aşağıda yer verilmiştir.

EmailService.java: Projemizde kullanıcıya e-posta göndermek için kullanılan bir servis sınıfıdır. Bu sınıfta, *JavaMailSender* ve *TemplateEngine* (Thymeleaf) bağımlılıkları kullanılmıştır. *JavaMailSender*, uygulamanın e-posta gönderebilmesi için gerekli olan bileşendir. *TemplateEngine* ise gönderilecek e-postanın içeriğini dinamik olarak oluşturmak için kullanılmıştır.

Sınıfın yapıcı metodunda (constructor) *JavaMailSender* ve *TemplateEngine* nesneleri enjekte edilir, böylece bu bileşenler sınıf içerisinde kullanılabilir olur. Sınıfın temel

fonksiyonu *sendVerificationEmail* isimli metottur. Bu metot, bir *User* nesnesi ve *siteUrl* parametrelerini alır. User nesnesinden kullanıcının e-posta adresi, adı ve doğrulama token'ı alınmıştır. Gönderilecek e-postanın alıcısı, gönderen adresi, gönderen adı ve konusu belirlenir. Kullanıcıya özel doğrulama bağlantısı oluşturulur. Thymeleaf'in *Context* nesnesiyle şablona gönderilecek değişkenler hazırlanır. Bu değişkenler genellikle kullanıcının adı ve doğrulama bağlantısıdır. *TemplateEngine*'in *process* metodu ile email-verification adındaki şablon dosyasından, kişiye özel HTML içerik üretilmiştir.

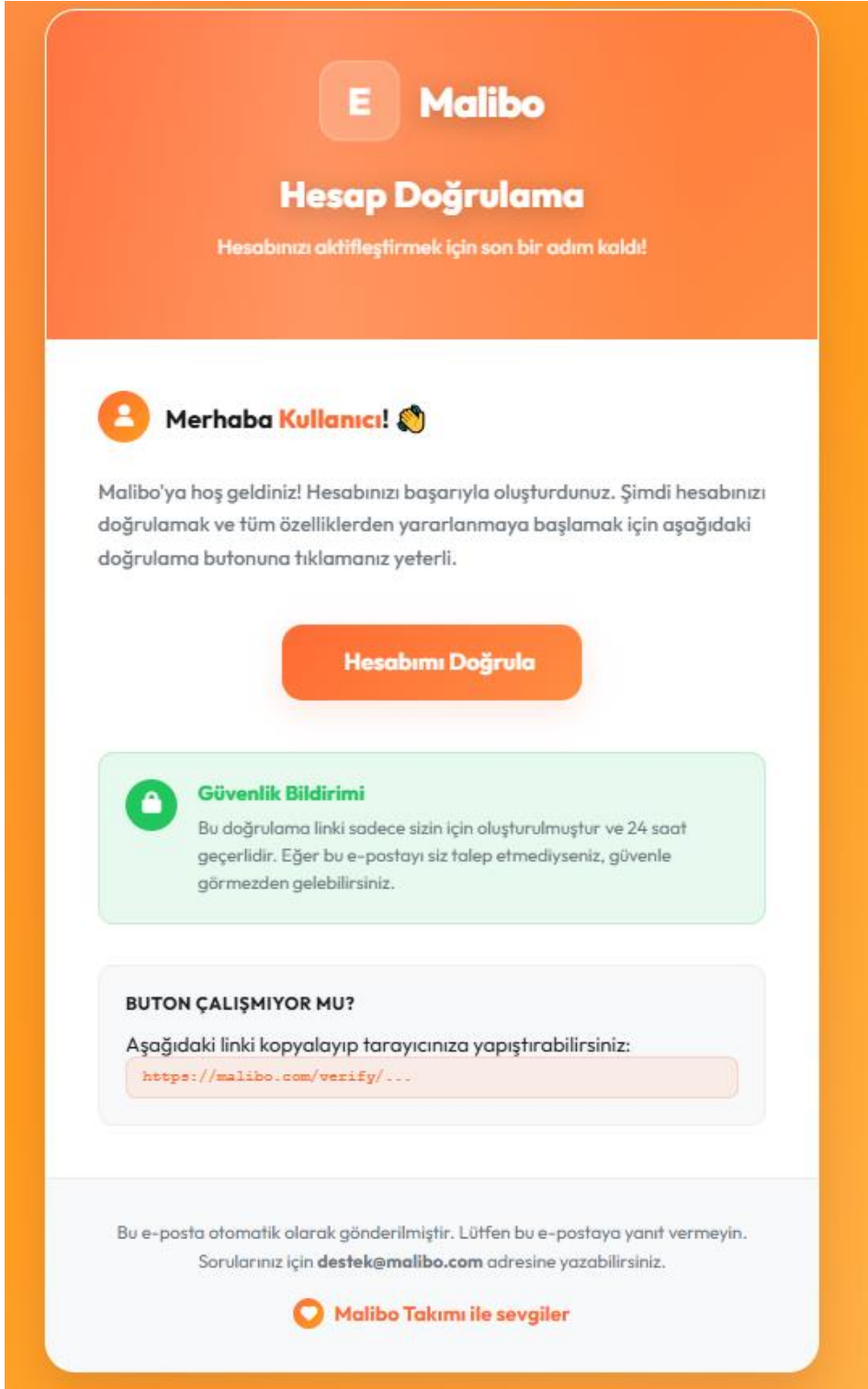
2.10. README.md dosyası

Projemizin README.md dosyası bulunmaktadır. Dosya içeriğinde; kullanılan tüm sınıfların açıklamaları, endpoint tasarımları ve projenin yerel sunucuda nasıl çalıştırılabileceğini adım adım anlattık. Projemizin öne çıkan özelliklerine de burada yer verilmiştir. Buradaki dokümantasyon İngilizce hazırlanmıştır.

2.11. HTML – CSS Tasarımları

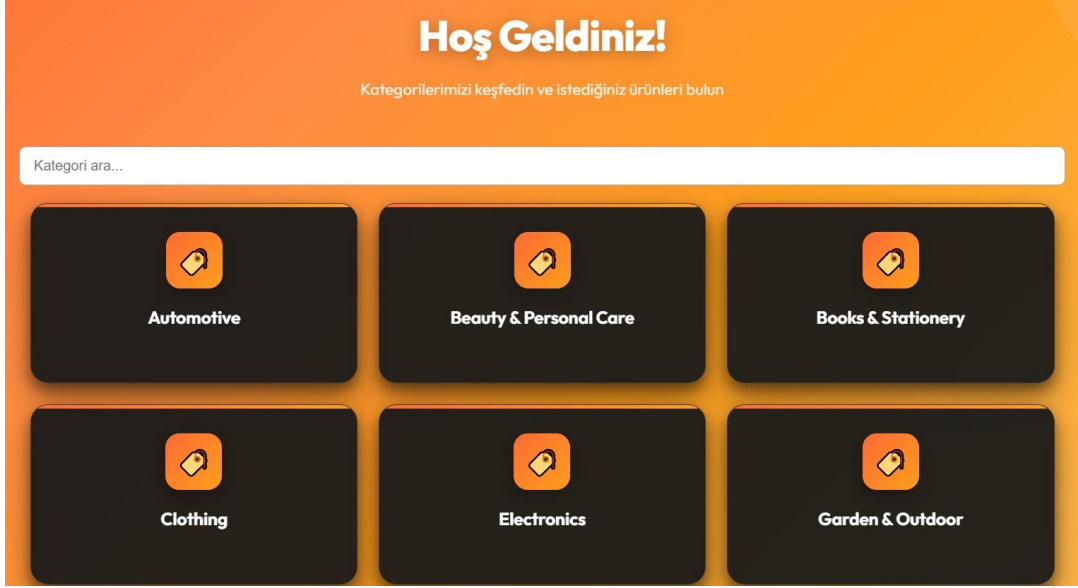
Bu kısımda projenin ön yüzünde (front-end) kullanılan HTML dosyalarının ön izlemeleri ve açıklamaları yazılmıştır. Projede birçok farklı HTML dosyası kullanılmıştır. HTML dosyalarının içerisinde *CSS* ve *Javascript* kodları yer almaktadır.

Sonraki görsel, e-posta doğrulama aşamasında kullanıcıya gönderilen e-postaya aittir.



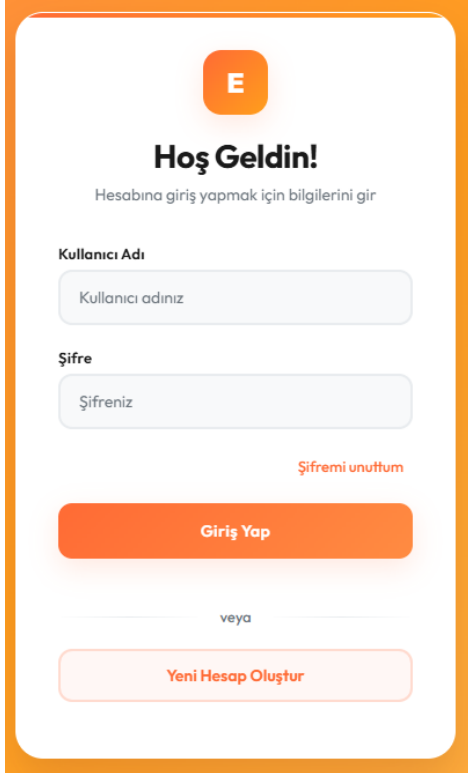
Şekil 2. 18: email-verification.html

Sonraki görsel, ana sayfada görüntülenen; kategori arama kutucuğu, tıklanabilen ve ilgili kategorilere yönlendiren butonların yer aldığı sayfaya aittir.



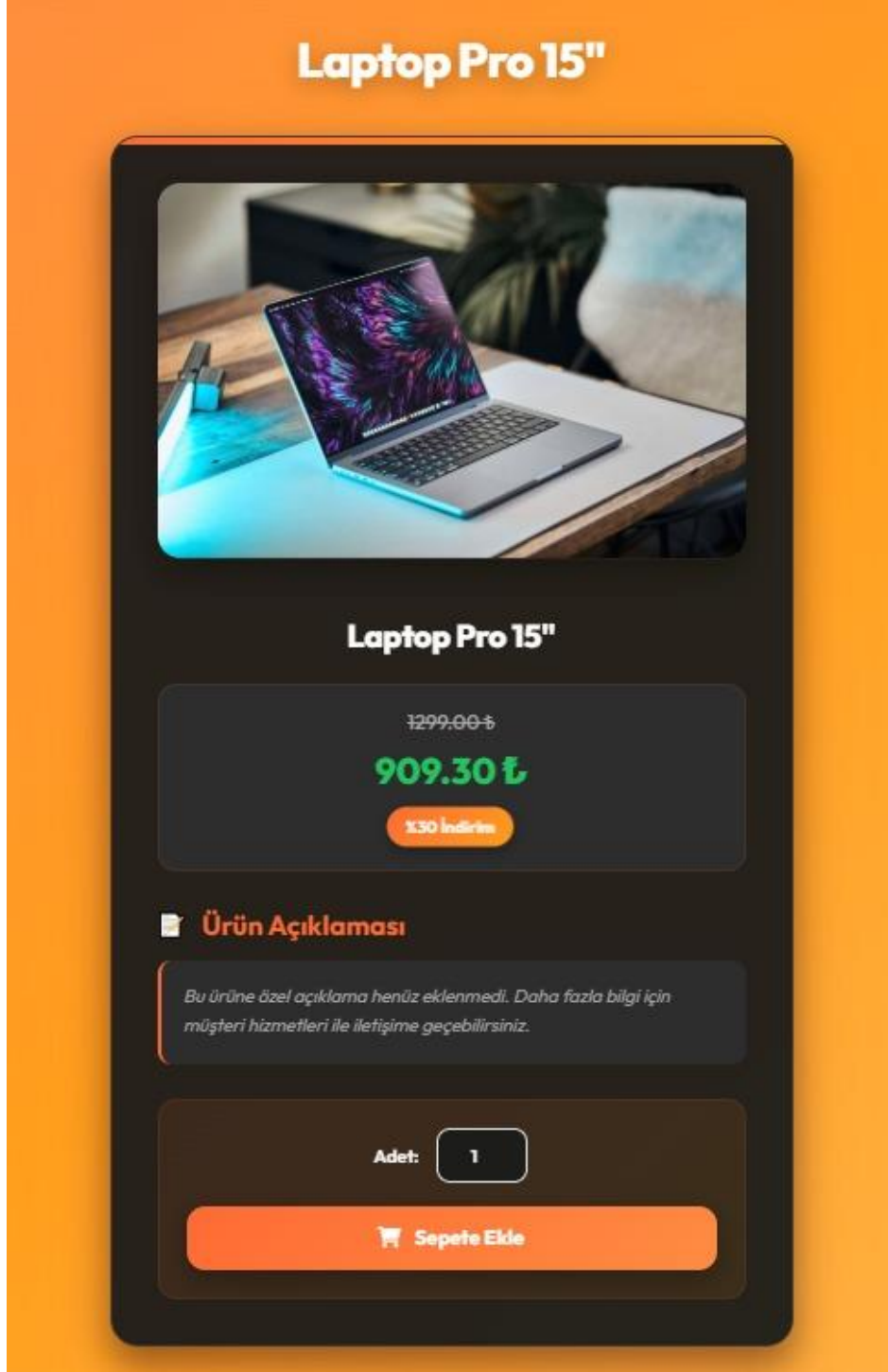
Şekil 2. 19: index.html

Sonraki görsel, kullanıcıların web sitesine erişebilmesi için tasarlanan giriş sayfaya aittir.



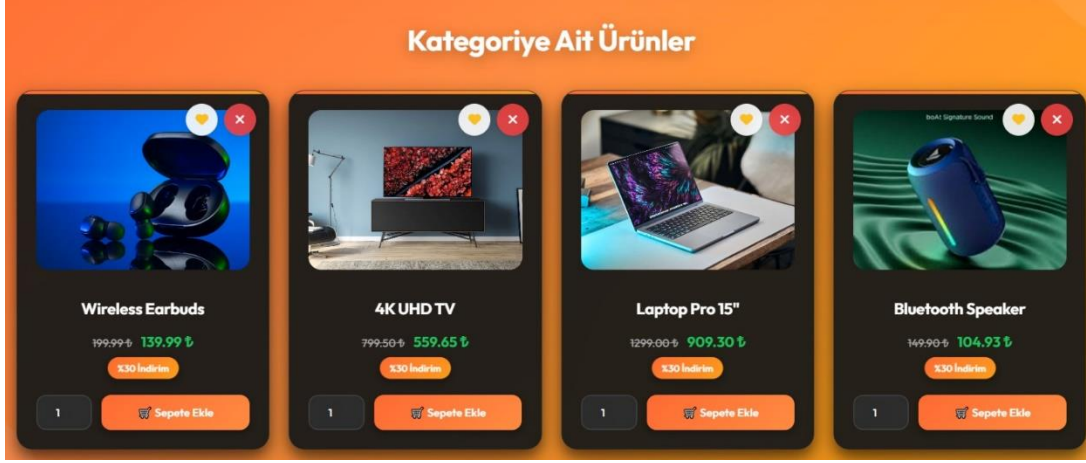
Şekil 2. 20: login.html

Sonraki görsel, herhangi bir ürüne tıklandığında detaylarını gösteren sayfaya aittir.



Şekil 2. 21: productdetails.html

Sonraki görsel, kullanıcının bulunduğu kategoriye ait ürünlerin yer aldığı sayfaya aittir.



Şekil 2. 22: products.html

Sonraki görsel, mevcut kullanıcıların bilgilerini değiştirmesine olanak sağlayan sayfaya aittir.

Profil Bilgileri

Profil Ayarları
Hesap bilgilerinizi güncelleyebilirsiniz

Kişisel Bilgiler

AD SOYAD: Adınızı ve soyadınızı girin

KULLANICI ADI: Kullanıcı adınızı girin

İletişim Bilgileri

E-POSTA ADRESİ: E-posta adresinizi girin

Güvenlik

ŞİFRE: [Gizli]

Profil Güncelle

Şekil 2. 23: profile.html

Sonraki görsel, yeni kullanıcı kaydı için tasarlanan sayfaya aittir.

E

Hesap Oluştur

Yeni hesabını oluşturmak için bilgilerini gir

Ad ve Soyad

Adınız ve soyadınız

Kullanıcı Adı

Benzersiz kullanıcı adı

E-posta Adresi

ornek@email.com

Cinsiyet

Yaş

Seçiniz

Yaşınız

Şifre

Güçlü bir şifre oluşturun

Hesap Oluştur

Hesap oluşturarak **Kullanım Şartları**'nı ve **Gizlilik Politikası**'nı kabul etmiş olursunuz.

Zaten hesabınız var mı?

Giriş Yap

Şekil 2. 24: register.html

Sonraki görsel, sipariş tamamlandıktan sonra sipariş detaylarını içeren sayfaya aittir.

Siparişiniz Alındı!

Teşekkür ederiz. Siparişiniz başarıyla oluşturuldu.

Sipariş No: #ORD-2024-001

Sipariş Tarihi: 15 Haziran 2024 14:30

Müşteri Bilgileri

AD SOYAD
John Doe

E-POSTA
john@example.com

Teslimat Adresi

ADRES
Örnek Mahalle, Örnek Sokak No:123

ŞEHİR / POSTA KODU
İstanbul / 34000

Sipariş Detayları

Ürünler

Ürün Adı

Ad

₺99.99

Ara Toplam

₺299.97

Kargo

₺15.00

Ücretsiz

KDV (%18)

₺53.99

Toplam Tutar

₺368.96

Ödeme Bilgileri

ÖDEME YÖNTEMİ
Kredi Kartı (**** * 1234)


ÖDEME DURUMU
Ödeme Alındı


Alışverişe Devam Et

Siparişi Yazdır


Şekil 2. 25: order-confirmation.html

Sonraki görsel; iletişim bilgileri, ödeme bilgileri ve teslimat adresi bilgilerinin kullanıcıdan talep edildiği ve siparişin oluşturma işlemini gerçekleştiren sayfaya aittir.


 **Ödeme İşlemi**


 **Sipariş Özeti**

Ara Toplam	₺32.00
Kargo	₺15.00
KDV (%18)	₺05.76
Toplam	₺52.76


 **İletişim Bilgileri**

E-POSTA ADRESİ


 E-posta adresinizi girin

 **Teslimat Adresi**


AD SOYAD

 Adınızı ve soyadınızı girin


ADRES


 Sokak, mahalle, bina no

ŞEHİR


 Şehir

POSTA KODU


 Posta kodu

 **Ödeme Bilgileri**


KART NUMARASI

 1234 5678 9012 3456


KART SAHİBİ


 Kart üzerindeki isim


SON KULLANMA

 MM/YY

CVV

 123

 Ödeme bilgileriniz SSL ile güvence altındadır

 **Sipariş Tamamla**

Sonraki görsel, indirim kazanmış bir kullanıcının hangi kategoride yüzde kaç indirim kazandığını gösteren sayfaya aittir.



Şekil 2. 27: discountCategories.html

2.12. Javascript Kullanımı

Projede *.html* uzantılı dosyaların içerisinde tasarım için kullanılan HTML ve CSS'in yanında Javascript kodları da bulunmaktadır. Buradaki Javascript kodları, animasyondan form validasyonuna varıncaya kadar geniş bir yelpazede kullanılmıştır.

2.12.1 Ana sayfa (index.html)

sBu sayfadaki JavaScript kodları öncelikle *logout()* fonksiyonu ile kullanıcıyı güvenli şekilde sistemden çıkarma işlemini gerçekleştirir. Kategori kartları üzerinde hover efektleri ve animasyonlar ile kullanıcı deneyimini zenginleştirir. Kategoriler arasında gerçek zamanlı arama özelliği sunar ve logo animasyonu ile görsel geri bildirim sağlar. Ayrıca erişilebilirlik için Enter ve Space tuşları ile klavye navigasyonu desteği bulunmaktadır.

2.12.2 Giriş sayfası (login.html)

Login sayfasının JavaScript kodları form validasyonu yaparak kullanıcı adı ve şifre kontrollerini gerçekleştirir. Sunucuya Ajax ile *async fetch* kullanımı ile giriş istekleri gönderir. Buton durumu değişikliği ve yükleme animasyonu ile kullanıcıya görsel geri bildirim sunar. Input alanlarında focus ve blur efektleri ile etkileşimi artırır. Başarısız giriş durumlarında hata yönetimi yaparak kullanıcıyı bilgilendirir.

2.12.3 Kayıt sayfası (register.html)

Kayıt sayfasında JavaScript kodları kayıt formunun işlenmesini sağlar. Yükleme durumunda buton durumu yönetimi yaparak kullanıcıya işlem hakkında bilgi verir. Kullanıcı deneyimi

için input animasyonları ve görsel efektler ekler. Logo ile etkileşimli animasyon özelliği bulunur.

2.12.4 Ürünler sayfası (products.html)

Products sayfası en kapsamlı JavaScript kodlarına sahiptir. *handleProductClick()* fonksiyonu ile ürün tıklama etkileşimlerini kaydeder ve *handleAdd()* ile sepete ürün ekleme işlemini gerçekleştirir. *toggleFavorite()* ve *removeFavorite()* fonksiyonları ile favori ürün ekleme ve çıkarma işlemlerini yönetir. *showNotification()* ile kullanıcıya bildirimler gösterir, *logout()* ile çıkış yapma işlemi sağlar ve klavye navigasyonu ile erişilebilirlik desteği sunar.

2.12.5 Ürün detayları sayfası (productdetails.html)

Ürün detay sayfasında *handleAdd()* fonksiyonu ile detay sayfasından sepete ekleme işlemi gerçekleştirilir. Miktar girişi için 1-99 arası validasyon kontrolü yapar. Ürün resmi üzerinde zoom efekti sağlar ve *logout()* ile çıkış yapma işlemi bulunur. Escape tuşu ile geri gitme özelliği klavye kısayolu olarak sunulmuştur.

2.12.6 Sepet sayfası (cart.html)

Sepet sayfasının JavaScript kodları *handleRemove()* fonksiyonu ile sepetten ürün çıkarma işlemini yönetir. Animasyonlu ürün kaldırma ile görsel geri bildirim sağlar. Profil veri çekme işlemi ile kullanıcı bilgilerini alır ve sayfa yükleme sırasında giriş animasyonları ekler. *logout()* fonksiyonu ile çıkış yapma özelliği mevcuttur.

2.12.7 Ödeme sayfası (checkout.html)

Checkout sayfası kart numarası formatını otomatik olarak 4'lü gruplar halinde düzenler. Tarih girişini MM/YY formatında kontrol eder ve CVV alanında sadece rakam girişine izin verir. Tüm alanların dolu olması için form validasyonu yapar. Profil verilerini çekerek formu otomatik doldurma özelliği sunar ve boş sepet ile ödeme yapılmasını engeller.

2.12.8 Profil sayfası (profile.html)

Profil sayfasında JavaScript kodları profil bilgilerini güncelleme işlemini yönetir. Kullanıcı adının ilk harfini profil avatarında gösterir. Input focus efektleri ve hover animasyonları ile

etkileşimi artırır. Mevcut kullanıcı bilgilerini profil veri çekme işlemi ile alır ve *logout()* fonksiyonu ile çıkış yapma özelliği bulunur.

2.12.9 Sipariş geçmişi (order-history.html)

Bu sayfada minimal JavaScript kullanımı mevcuttur. Sadece temel navigasyon için *history.back()* fonksiyonu ile geri gitme özelliği bulunmaktadır. Çoğunlukla statik içerik sunduğu için kompleks JavaScript işlemleri gerektirmemiştir.

2.12.10 Sipariş onayı (order-confirmation.html)

Sipariş onay sayfası başarı animasyonu ile sipariş onay sürecini görselleştirir. Opsiyonel olarak confetti efekti ile kutlama animasyonu sunar. Sayfanın en üstüne otomatik scroll yaparak kullanıcı deneyimini iyileştirir. *window.print()* fonksiyonu ile siparişi yazdırma özelliği sağlar.

2.12.11 E-posta doğrulama (email-verification.html)

E-posta doğrulama sayfası doğrulama linkini panoya kopyalama özelliği sunar. Doğrulama butonuna yapılan tıklamaları takip eder ve progressive loading animasyonları ile sayfa yükleme deneyimini zenginleştirir.

2.12.12 İndirimli kategoriler (discountCategories.html)

İndirimli kategoriler sayfası *logout()* fonksiyonu ile çıkış yapma ve *showNotification()* ile bildirim sistemi sağlar. Kategori tıklama etkileşimi ve Enter/Space tuşları ile klavye navigasyonu desteği bulunur. Logo animasyonu ile etkileşimli görsel öğeler ekler.

Tüm sayfalarda ortak kullanılan JavaScript özellikleri modern Ajax istekleri için *fetch()* API, DOM etkileşimleri için Event Listeners, asenkron işlemler için *Async/Await*, hata yönetimi için Error Handling, yükleme durumu gösterimi için Loading States, CSS ile koordineli animasyonlar, görsel geri bildirimler için User Feedback ve klavye navigasyonu desteği ile Accessibility özelliklerini içerir.

3. BULGULAR VE TARTIŞMA

Bu bölümde, geliştirilen e-ticaret sisteminde uygulanan kişiselleştirilmiş indirim mantığı, kullanıcı aktivitelerinden elde edilen verilerin işlenişi ve bu verilerin enum sınıfında ağırlıklandırılarak indirim hesaplamalarına yansıtılması ele alınacaktır. Kullanıcıların ürüne tıklama, sepete ekleme ve favoriye ekleme gibi aktiviteleri sistem tarafından kaydedilmiş; bu aktiviteler, belirlenen ağırlıklara göre enum yapısında sınıflandırılarak, her kullanıcıya özel indirim oranları oluşturulmuştur. Güvenlik tarafında ise Spring Security ile yetkilendirme ve erişim kontrolü sağlanırken, JWT tabanlı oturum yönetimi sayesinde kullanıcıların kolay ve güvenli bir şekilde sisteme erişimi mümkün kılınmıştır. Ayrıca, kullanıcı deneyimini artırmak amacıyla ön yüzde seçilen renk paletinin etkisi de bu bölümde değerlendirilecektir.

3.1. Kişiselleştirilmiş İndirim Mantığı

Projede enum üzerinde belirlenen metriklerin puanlandırması sonucu belli bir eşiği geçen kullanıcılar ilgili kategoride indirimler kazanmaktadır. Kullanıcı indirim kazandığında ana sayfada şekildeki gibi bir pop-up ile karşılaşır:



Şekil 3. 1: Ana sayfada görüntülenen indirim bildirimi

3.1.1 Toplanan veriler

Kullanıcıdan toplanan veriler olarak “ürüne tıklama”, “sepete ekleme” ve “favorilere ekleme” gibi metriklerinin seçilmesinin temel nedeni, bu etkileşimlerin kullanıcıların satın alma niyetini ve ilgi düzeyini en iyi şekilde yansıtan davranışlar olmasıdır. Literatürde, kullanıcıların bir ürüne tıklaması, o ürüne olan ilk ilgiyi; sepete eklemesi, satın alma niyetinin güçlendiğini; favorilere eklemesi ise gelecekteki potansiyel satın alma davranışını gösteren önemli bir gösterge olarak kabul edilmektedir [20], [21]. Bu üç metrik, kullanıcı davranışlarının analizinde ve kişiselleştirilmiş öneri ya da indirim sistemlerinde yaygın olarak kullanılmaktadır. Örneğin, Jannach ve arkadaşları [22], ürün tıklamalarının ve sepete ekleme işlemlerinin satın alma olasılığını öngörmeye yüksek korelasyon gösterdiğini belirtmiştir. Benzer şekilde, Linden ve arkadaşları [23], favorilere ekleme gibi işlemlerin, kullanıcıların ürünlere olan uzun vadeli ilgisini anlamada önemli bir rol oynadığını vurgulamıştır. Bu nedenle, çalışmamızda kişiselleştirilmiş indirimlerin hesaplanmasında bu üç temel kullanıcı aktivitesi dikkate alınmıştır.

3.1.2 Toplanan verilere atanan değerler

Kullanıcı aktivitelerine atanan ağırlık değerleri, bu aktivitelerin satın alma davranışı üzerindeki etkisini yansıtacak şekilde belirlenmiştir. Literatürde, kullanıcıların ürüne tıklama, sepete ekleme ve favorilere ekleme gibi etkileşimlerinin satın alma olasılığı ile farklı derecelerde ilişkili olduğu gösterilmiştir.

Ürüne tıklama, genellikle kullanıcının ürüne karşı ilk ilgisini temsil eder ve bu nedenle ağırlığı diğerlerine göre daha düşüktür [20]. Sepete ekleme işlemi, kullanıcının satın alma niyetinin ciddi şekilde güçlendiğini gösterir ve bu nedenle daha yüksek bir ağırlık ile değerlendirilir [22]. Favorilere ekleme ise, kullanıcının ürüne olan uzun vadeli ilgisini ve potansiyel gelecekteki satın alma davranışını yansıttığı için, bazı çalışmalarda sepete eklemeden bile daha yüksek bir etkiyle ilişkilendirilmiştir [23].

Bu bağlamda oranlar, Tablo 3.1’de gösterildiği şekilde atanmıştır. Bu oranlar, kullanıcıların satın alma yolculuğundaki davranışlarının göreceli önemini ve satın almaya olan yakınlıklarını temsil etmektedir. Benzer ağırlıklandırma yaklaşımları, öneri sistemlerinde ve kişiselleştirilmiş pazarlama stratejilerinde de yaygın olarak kullanılmaktadır [22], [23].

Kullanıcıların eylemleri, davranışsal değerlerine göre normalize edilmiş sabit ağırlıklarla temsil edilir. Bu ağırlıklar aşağıdaki gibi belirlenmiştir:

Tablo 3.1: Aktivite değeri

Aktivite Türü	Değişken Adı	Aktivite Skoru
Ürüne Tıklama	PRODUCT_CLICK	0.0953
Ürün Detayı Görüntüleme	DETAILS_EXPAND	0.1429
Ürün Değerlendirmesi	RATING	0.1905
Yorum Yapma	REVIEW_SUBMIT	0.2143
Sepete Ekleme	CART_ADD	0.2381
Favorilere Ekleme	FAVORITES_ADD	0.2857
Satın Alma	PURCHASED	0.3809

3.1.3 Kullanıcı etkileşimlerinden skor oluşturma

E-ticaret platformlarında kullanıcıların gerçekleştirdiği aktivitelerin (ör. ürün tıklama, sepete ekleme, favorilere ekleme) belirli ağırlıklarla puanlanarak bir skor oluşturulması, kullanıcı segmentasyonu ve kişiselleştirilmiş kampanya yönetimi açısından yaygın ve etkili bir yöntemdir [20], [22]. Bu yöntemle, her bir kullanıcının belirli bir kategoriye olan ilgisi nicel olarak ölçülebilir hale gelir.

“Materyal ve Yöntem” bölümünde detayları verilen şekilde, kullanıcının farklı etkileşimlerinin ağırlıkları toplanarak bir toplam skor elde edilmektedir. Bu skor,

kullanıcının ilgili kategoriye olan toplam etkileşimini ve potansiyel değerini yansıtır. Toplanan skurun belirli aralıklara bölünerek indirim seviyelerine dönüştürülmesi, kullanıcıların sadakatini ödüllendirmek ve onları daha fazla alışverişe teşvik etmek için etkili bir stratejidir [23].

Bu yaklaşım, kullanıcıların geçmiş davranışlarına göre kişiselleştirilmiş indirimler sunulmasına olanak tanır. Örneğin, düşük skorlar için indirim uygulanmazken, skor arttıkça indirim oranı da artar. Böylece, platform hem kullanıcı bağlılığını artırır hem de kullanıcıların alışveriş davranışlarını daha iyi analiz edebilir [22], [23].

Kullanıcı Skorunun Hesaplanması

Belirli bir k kategorisi için bir kullanıcının skoru aşağıdaki formül ile hesaplanır:

$$S_{u,k} = \sum_{i=1}^n w_i$$

Burada:

kullanıcının k kategorisindeki toplam etkileşim skoru,

i. etkileşimin ağırlığı,

o kategori altında kullanıcı tarafından gerçekleştirilen etkileşim sayısı parametre olarak alınmıştır.

Hiç satın almamış kullanıcı için:

$$D_{u,k} = \begin{cases} 10\%, & \text{eğer } S_{u,k} > 5 \text{ ve } P_{u,k} = 0 \\ 0\%, & \text{diğer durumlarda} \end{cases}$$

Satın alım yapan kullanıcılar için:

$$f(s) = \begin{cases} 0.20 & \text{eğer } 9 < s < 17 \\ 0.30 & \text{eğer } 17 \leq s < 25 \\ 0.40 & \text{eğer } 25 \leq s < 33 \\ 0.50 & \text{eğer } s > 33 \end{cases}$$

formülleri uygulanmaktadır.

3.2. Renk Seçimlerinin Kullanıcıya Etkisi

E-ticaret sitelerinde kullanılan renklerin kullanıcı deneyimi üzerindeki etkisi uzun süredir pazarlama ve insan-bilgisayar etkileşimi alanında araştırma konusu olmuştur. Labrecque ve Milne'nin [40] çalışmasında, doğru renk kombinasyonlarının kullanıldığı e-ticaret sitelerinde **kullanıcıların %85'inin renklerin satın alma kararlarını doğrudan etkilediğini** belirttiği tespit edilmiştir. Aynı çalışmada, **marka tanınırlığının doğru renk seçimiyle %80 oranında artırılabilirdiği**, ayrıca renk uyumunun kullanıcı güvenini %65 oranında pozitif etkilediği ortaya konmuştur. Aşağıdaki örnek tabloda, renk uyumu yüksek olan e-ticaret siteleri ile düşük olanlar karşılaştırıldığında, **ortalama dönüşüm oranlarının %21'e karşılık %13** olduğu görülmektedir. Bu bulgular, kullanıcı arayüzünde kullanılan renk paletinin yalnızca estetik değil, aynı zamanda **dönüşüm oranları, sadakat ve satış performansı** açısından da kritik olduğunu göstermektedir.

Tablo 3.2: Renk seçiminin kullanıcıya etkisi

Renk Uyumu	Ortalama Dönüşüm Oranı	Kullanıcı Güveni	Marka Hatırlanabilirliği
Yüksek (Uygun renk)	%21	%65	%80
Düşük (Uyumsuz renk)	%13	%32	%45

Bu bağlamda, geliştirilen e-ticaret platformlarının kullanıcı arayüzü tasarımında renk psikolojisi dikkate alınmalı; özellikle arka plan, buton ve vurgu renkleri, kullanıcı

davranışlarını olumlu yönde etkileyecek biçimde seçilmelidir. Renk seçiminin hedef kitleye göre özelleştirilmesi, kişiselleştirme stratejilerine destek sağlayan tamamlayıcı bir unsur olarak değerlendirilebilir.

3.2.1 Turuncu rengin kullanımı

Sitenin genelinde arka plan olarak turuncu tonlarının seçilmesi, sıcaklık, enerji ve samimiyet hissi yaratır. Turuncu renk, kullanıcıların dikkatini çekmede etkilidir ve alışveriş ortamlarında pozitif duygular uyandırarak kullanıcıların sitede daha fazla vakit geçirmesine katkı sağlar. Ayrıca, turuncu güven ve hareketlilikle ilişkilendirilir; bu da kullanıcıların aksiyon almaya teşvik edilmesi açısından önemlidir [24].

3.2.2 Siyah ve beyaz renklerinin kullanımı

Beyaz ve siyah renkler, arayüzde denge ve kontrast oluşturmak için kullanılmıştır. Beyaz, temiz ve ferah bir his vererek içeriklerin öne çıkmasını sağlar. Siyah ise metinlerde ve önemli vurgularda kullanılarak okunabilirliği artırır ve profesyonel bir görünüm kazandırır. Bu iki yardımcı renk, ana renk olan turuncunun etkisini güçlendirir ve arayüzde görsel hiyerarşi oluşturur [25].

3.2.3 Yeşil rengin kullanımı

Sipariş tamamlandı sayfasında yeşil temanın kullanılması, başarı ve olumlu sonuç hissi uyandırır. Yeşil renk, psikolojik olarak güven, onay ve rahatlama ile ilişkilendirilir. Kullanıcıya işlemin başarılı olduğu hızlı ve net bir şekilde iletilmiş olur [26].

3.3. Güvenli Web Sitesi Deneyimini Sağlama

Sitenin güvenlik altyapısında, modern ve güvenli bir kimlik doğrulama ile yetkilendirme mekanizması oluşturmak amacıyla Spring Security kütüphanesinden yararlanılmıştır. Bu kütüphane sayesinde, uygulamanın tüm giriş noktaları üzerinde kapsamlı bir kontrol sağlanmakta ve yalnızca yetkili kullanıcıların belirli kaynaklara erişimine izin verilmektedir. Kimlik doğrulama işlemlerinde ise JSON Web Token (JWT) tabanlı bir sistem tercih edilmiştir. Kullanıcı sisteme başarılı bir şekilde giriş yaptıktan sonra, ona özel olarak oluşturulan bir JWT token'ı üretilmekte ve bu token, sonraki tüm isteklerde kimlik doğrulaması için kullanılmaktadır. Bu yöntem sayesinde, sunucu tarafında herhangi bir oturum bilgisi tutulmasına gerek kalmamakta, yani oturum yönetimi tamamen “stateless” bir

şekilde gerçekleştirilmektedir. Böylece hem performans artışı sağlanmakta hem de uygulamanın ölçeklenebilirliği kolaylaşmaktadır.

Kullanıcıların şifreleri ise güvenliğin en hassas noktalarından biri olarak ele alınmış ve bu amaçla *BCrypt* algoritması ile şifrelenmektedir [27]. *BCrypt*, günümüzde şifre saklama konusunda en çok önerilen algoritmalarından biri olup, brute-force saldırılarına karşı yüksek direnç göstermektedir. Sonuç olarak, hem kimlik doğrulama hem de veri güvenliği açısından güncel standartlara uygun, güvenli ve esnek bir altyapı oluşturulmuştur.

3.4. Kullanıcı Davranışları ve Ödül Mekanizması

Şekil 2.6'daki gibi görüleceği ve yorumlanabileceği üzere; kullanıcı, satın alım yapmadan da indirime ulaşabilir. Sistemin nasıl çalıştığını öğrenen ve bunu lehine çevirecek şekilde sürekli tıklama aktivitesi gibi skoru arttıran eylemlerde bulunan kullanıcılar mutlaka olacaktır. Bu başta istenmeyen bir durum olarak gözükse de sitedeki etkileşimi arttırmak, kullanıcıların bu basit aktivitelerle bile indirim sağlayabildiğini görünce bunu diğer insanlarla paylaşarak daha fazla kişiye deyim yerindeyse beyindeki ödül mekanizmasını manipüle ederek ulaşabiliriz. Nitekim, dijital platformlarda kullanıcıların ödül beklentisiyle tekrarlı ve basit aksiyonlar gerçekleştirdiği, bu davranışların ise hem bireysel motivasyonu hem de sosyal yayılımı artırdığı çeşitli çalışmalarda gösterilmiştir [28][29]. Özellikle oyunlaştırma (gamification) ve anlık ödüllendirme mekanizmalarının, kullanıcıların platformda daha fazla vakit geçirmesine ve deneyimlerini çevreleriyle paylaşmalarına yol açtığı bilinmektedir [30]. Benzer şekilde, **Temu** gibi büyük e-ticaret platformları da kullanıcı etkileşimini ve platformda geçirilen süreyi artırmak için ödül ve oyunlaştırma mekanizmalarını aktif olarak kullanmaktadır. Bu platformlarda, kullanıcılar uygulamaya her giriş yaptıklarında, belirli görevleri tamamladıklarında ya da arkadaşlarını davet ettiklerinde çeşitli indirimler, kuponlar veya sanal ödüller kazanabilmektedir. Örneğin, günlük giriş ödülleri, “çarkı çevir” gibi şans oyunları, alışveriş puanları veya arkadaş davetiyle elde edilen ekstra avantajlar, kullanıcıların platforma tekrar tekrar dönmesini ve bu avantajları sosyal çevreleriyle paylaşmasını teşvik etmektedir. Ayrıca, kullanıcıların alışveriş yapmasalar bile uygulama içerisinde aktif kalmaları sağlanarak, hem platformun algoritmalarına daha fazla veri sağlanmakta hem de kullanıcıların alışveriş yapma ihtimali zamanla artırılmaktadır. Bu tür stratejiler, hem bireysel motivasyonu hem de viral yayılımı destekleyerek, platformun kullanıcı tabanını hızlı bir şekilde büyütmesine yardımcı olmaktadır.

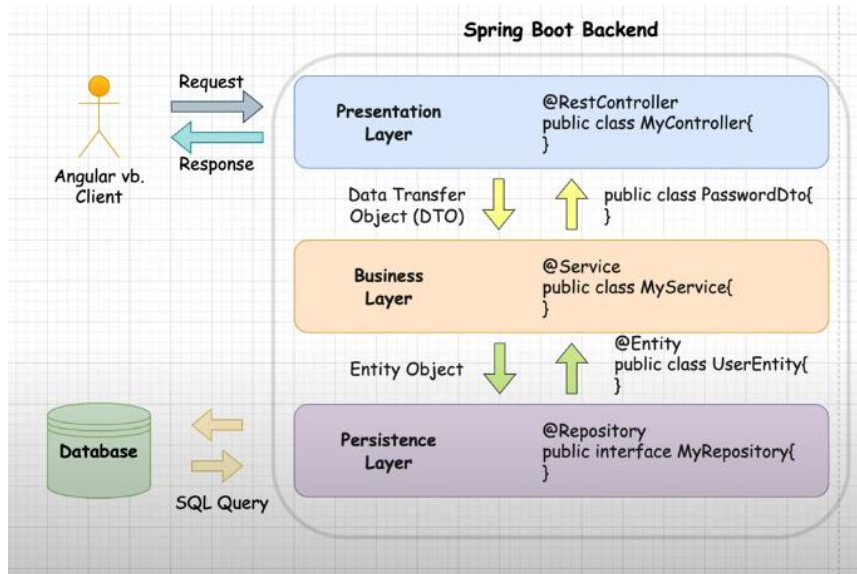
3.5. Kural Tabanlı İndirim Sistemlerinin Sektördeki Yeri

Kural tabanlı indirim sistemleri, birçok farklı e-ticaret sitesi tarafından kullanılmaktadır. Bu projede özellikle **Trendyol GO** örnek alınarak geliştirme yapılmıştır.

Kural tabanlı indirim sistemlerinin e-ticaret projelerine entegrasyonu, müşteri sadakati oluşturma, satış hacmini artırma, rekabet avantajı sağlama ve davranışsal veri toplama gibi çok yönlü faydalar sunmaktadır. Özellikle Trendyol Go gibi yenilikçi stratejiler, müşterilerin tekrar alışveriş yapma olasılığını artırarak uzun vadeli müşteri bağlılığı oluşturur [35]. Araştırmalar, artan indirim kuponlarının tüketicilerin harcama miktarını yükselttiğini ve toplam satış hacmine doğrudan katkı sağladığını göstermektedir [36]. Ayrıca, bu tür sistemler sayesinde toplanan davranışsal veriler, müşteri segmentasyonu ve kişiselleştirilmiş kampanya yönetimi açısından önemli analiz fırsatları sunar [37]. E-ticaret sektöründeki yoğun rekabet ortamında, yenilikçi indirim uygulamaları sayesinde rakiplerden ayrılmak ve pazarda farklılaşmak mümkün hale gelmektedir [38]. Kısa vadede indirimlerin kâr marjını azaltabileceği düşünülse de, sadık müşterilerin edinilmesiyle uzun vadede müşteri başına gelirden artış sağlanmakta ve müşteri yaşam boyu değeri yükselmektedir [39].

3.6. Sistem Mimarisi ve Teknolojik Altyapısı

Web sitesinin ölçeklenebilirliği ve sonradan kolayca geliştirilebilirliği açısından belli bir dosya düzeni ve Spring'in sunmuş olduğu katmanlı yapı bu projede titizlikle uygulanmıştır.



Şekil 3. 2: Spring Boot'un katmanlı mimarisini temsil eden görsel

Şekil 3.1’de görüldüğü gibi, Spring bize temiz ve düzenli kod yazma olanağı sağlıyor. Proje yukarıdaki mimariye uygun tasarlanmıştır.

E-ticaret platformumuzda Spring Boot çok katmanlı mimarisini tercih etmemizin temel nedeni, bu mimarinin sağladığı modülerlik, bakım kolaylığı, ölçeklenebilirlik ve yüksek performans gibi avantajlardır. Modülerlik, uygulamanın presentation (sunum), business (iş mantığı) ve persistence (veri erişim) gibi katmanlara ayrılmasına olanak tanır; bu yapı sayesinde her bir katman bağımsız biçimde geliştirilebilir, test edilebilir ve gerektiğinde yeniden kullanılabilir [31]. Böylece, örneğin kullanıcı arayüzünde yapılan bir değişiklik, iş mantığı veya veri erişim katmanını etkilemeden uygulanabilir. Bu ayrım, büyük ekiplerle çalışırken kodun yönetimini ve sürdürülmesini kolaylaştırır.

Spring Boot’un sunduğu yerleşik Tomcat sunucusu ve mikroservis mimarisi desteği, uygulamanın farklı bileşenlerinin bağımsız olarak ölçeklenmesini ve dağıtılmasını mümkün kılar [32]. Özellikle e-ticaret platformlarında trafik yoğunluğu dönemsel olarak büyük değişiklikler gösterebildiğinden, bu ölçeklenebilirlik özelliği sistemin performansını ve kullanılabilirliğini artırır. Ayrıca, Spring Boot’un minimal konfigürasyon gerektiren yapısı, geliştiricilerin karmaşık yapılandırmalarla zaman kaybetmeden hızlı bir şekilde uygulama geliştirmesine olanak tanır [33].

Güvenlik açısından da Spring Boot, Spring Security entegrasyonu ile güçlü bir koruma katmanı sunar. JWT (JSON Web Token) tabanlı kimlik doğrulama ve rol bazlı erişim kontrolü gibi gelişmiş güvenlik özellikleri, hem kullanıcı verilerinin gizliliğini hem de ödeme işlemlerinin güvenliğini sağlar [34]. Bu sayede, platformumuzda hem kullanıcı deneyimi hem de veri güvenliği üst düzeyde tutulmaktadır. Ayrıca, Spring Boot’un geniş topluluk desteği ve sürekli güncellenen ekosistemi, yeni güvenlik tehditlerine karşı hızlı çözümler geliştirilmesini kolaylaştırır.

Tüm bu avantajlar, e-ticaret platformumuzun kullanıcı deneyimini iyileştirirken, yazılım geliştirme ve bakım süreçlerini de daha verimli ve sürdürülebilir hale getirmektedir. Sonuç olarak, Spring Boot çok katmanlı mimarisinin sağladığı bu esneklik ve güç, modern e-ticaret uygulamalarının ihtiyaçlarını karşılamak için oldukça uygundur.

3.7. Kullanıcının Geçmiş Siparişlerine Bağlı Skor Çarpanı

Kullanıcının yapmış olduğu aktivitelerin bir sonuca varması bizim için elzemdir. Bu yüzden yapılan tüm aktiviteler belli bir skor oluştursa da bu skor ancak satın alım gerçekleştiyse anlam ifade edecek seviyeye gelmelidir. Aşağıdaki tabloda, projedeki kullanıcı seviyeleri, bu seviyeye gelmek için gereken sipariş tutarı ve skor çarpanları yer almaktadır. Tabloya göre kullanıcıların etkileşimlerinin daha değerli olabilmesi için geçmişte belli bir tutarın üstünde alışveriş yapmaları gerekmektedir. “Profil” sayfasındaki *progress bar* bu işlemi bir nevi oyunlaştırarak ve interaktif hale getirerek kullanıcının satın alma ihtimalini arttırdığını düşünmekteyiz.

Tablo 3.3: Geçmiş sipariş tutarına göre skor çarpanları

	Standart	Bronz	Gümüş	Altın	Elmas
Tutar Eşiği	0 TL	1000 TL	5000 TL	10000 TL	25000 TL
Skor Çarpanı	0,15	0,40	0,55	0,80	1,00

Daha sonra yukarıdaki tabloya uygun şekilde aşağıdaki formül uygulanabilecektir:

Nihai İndirim Skoru:

$$\Phi(S, L) = S \times \mu(f(P))$$

- Φ : Nihai indirim skoru
- S : Kullanıcı aktivite skoru
- $\mu(f(P))$: Seviye çarpanı

İndirim Yüzdesi Hesaplama Formülü

$$\delta(\Phi) = \{$$

$$0, \quad \text{if } \Phi < 1.0$$

$$\min(\Phi \times 0.1, 30.0), \text{ if } \Phi \geq 1.0 \}$$

- Alt Sınır: $\delta(\Phi) \geq 0$

- Üst Sınır: $\delta(\Phi) \leq 30.0$
- Aktivasyon Eşiği: $\Phi_{\min} = 1.0$

Örnek Hesaplama (Elmas kullanıcı için):

$$P = 30,000 \text{ TL} \rightarrow f(P) = \text{ELMAS} \rightarrow \mu = 1.00$$

$$S = 15 \text{ puan}$$

$$\Phi = 15 \times 1.00 = 15.0$$

$$\delta = \min(15.0 \times 0.1, 30.0) = \min(1.5, 30.0) = 1.5\%$$

3.8. Gerçek Hayatta Kullanıcı Aksiyonları ve Satın Alma

Sadece ürüne tıklayan bir kullanıcının satın alma oranı genellikle çok düşüktür. Ortalama olarak bu oran %1-2 civarındadır. Yani 100 kullanıcıdan sadece 1-2'si ürünü satın alır.[41]

Ürünü sepete ekleyen kullanıcının satın alma oranı %10-30 arasında değişmektedir. Yani sepete ekleyen her 10 kişiden yaklaşık 1-3'ü satın alma işlemini tamamlar.[41]

Yorum okuyan veya yazan kullanıcının satın alma oranı, sadece tıklayanlara kıyasla daha yüksektir. Yorum okuyan kullanıcıların satın alma oranı %20-25 civarındadır.[42]

Favorilere ekleyen kullanıcılar genellikle ürüne ilgi duyan, ancak hemen satın almayan gruptadır. Bu kullanıcıların satın alma oranı %15-20 civarındadır.[41]

Ürünü tıklayan, detayına bakan, yorum okuyan, sepete ekleyen ve favorilere ekleyen kullanıcıların satın alma oranı %50'ye kadar çıkabilmektedir. Özellikle mobil uygulamalarda ve sadakat programı üyelerinde bu oran daha da yükselebilir.[41]

Tablo 3.4: Senaryolar ve Satın Alma Olasılıkları Tablosu

Senaryo	Satın Alma Olasılığı (%)
Sadece ürün tıklama	1-2
Sepete ekleme	10-30
Yorum okuma/yazma	20-25
Favorilere ekleme	15-20
Sepete ekleyip, yorum okuyup, favorilere ekleme	30-50

3.8.1 Türkiye’de genel e-ticaret satın alma oranı

2024 yılında Türkiye’de internet kullanan bireylerin e-ticaret üzerinden alışveriş yapma oranı %51,7 olarak açıklanmıştır. Ancak bu oran, tüm internet kullanıcılarını kapsar. Gerçek dönüşüm oranları yukarıdaki gibi davranışa göre değişir.[43]

4. SONUÇLAR

Bu çalışma kapsamında, modern e-ticaret platformlarının karşılaştığı temel zorluklardan biri olan tek tip pazarlama stratejilerinin yetersizliği ve kişiselleştirme eksikliğini ele almak amacıyla, kullanıcı davranışlarını analiz ederek dinamik ve kişiselleştirilmiş indirimler sunabilen kural tabanlı bir e-ticaret sistemi başarıyla geliştirilmiş ve detaylı bir şekilde incelenmiştir. Geliştirilen sistem, kullanıcıların platform üzerindeki etkileşimlerini (ürüne tıklama, sepete ekleme ve favorilere ekleme gibi) temel alarak, her bir kullanıcıya özel ve anlamlı indirimler tanımlama yeteneğine sahip bir altyapı sunmaktadır. Bu yaklaşım, kullanıcı deneyimini zenginleştirmenin yanı sıra müşteri bağlılığını artırma ve dolayısıyla işletmeler için rekabet avantajı sağlama potansiyelini ortaya koymuştur.

Elde edilen bulgular, kullanıcı etkileşimlerine atanan ağırlık değerleri aracılığıyla oluşturulan kullanıcı skorlarının, kişiselleştirilmiş indirim stratejilerinin temelini oluşturmada etkin bir yöntem olduğunu göstermiştir. Bu skrolama mekanizması, kullanıcıların belirli ürün kategorilerine olan ilgi düzeylerini nicel olarak ifade etmekte ve bu sayede platformun, kullanıcıların satın alma niyetlerine daha duyarlı ve proaktif bir şekilde yanıt vermesini sağlamaktadır. Sistem, Spring Framework (özellikle Spring Boot ve

Spring MVC) ve MySQL veritabanı gibi çağdaş teknolojiler kullanılarak inşa edilmiş; bu teknolojik tercihler, projenin modüler, ölçeklenebilir ve bakımı kolay bir mimariye sahip olmasını mümkün kılmıştır. Özellikle Spring Boot'un katmanlı mimarisi, sunum, iş mantığı ve veri erişim katmanlarının birbirinden bağımsız geliştirilmesine olanak tanıyarak sistemin esnekliğini ve geliştirme süreçlerinin verimliliğini artırmıştır.

Güvenlik, geliştirilen e-ticaret platformunun temel önceliklerinden biri olarak ele alınmış; Spring Security kütüphanesi aracılığıyla kapsamlı yetkilendirme ve erişim kontrol mekanizmaları tesis edilmiştir. Kullanıcı kimlik doğrulama süreçlerinde JSON Web Token (JWT) tabanlı durumsuz (stateless) oturum yönetimi benimsenmiş, bu sayede kullanıcıların güvenli ve kesintisiz bir erişim deneyimi yaşaması sağlanmıştır. Kullanıcı şifrelerinin BCrypt algoritması ile şifrelenerek saklanması, veri güvenliği standartlarının en üst düzeyde karşılanmasına katkıda bulunmuştur. Ayrıca, kullanıcı arayüzünde tercih edilen renk paletinin (enerji ve dikkati temsil eden turuncu, denge ve okunabilirlik sağlayan siyah ve beyaz, onay ve başarıyı simgeleyen yeşil) kullanıcı deneyimi üzerinde olumlu etkiler yarattığı ve site içi etkileşimi teşvik ettiği gözlemlenmiştir.

Çalışmanın önemli bir sonucu da, geliştirilen ödül mekanizmasının, kullanıcıların doğrudan bir satın alma işlemi gerçekleştirmeseler dahi platform üzerindeki etkileşimleri aracılığıyla indirim kazanabilmelerine olanak tanımasıdır. Bu durumun, başlangıçta istenmeyen bir yan etki gibi görünse de, kullanıcıların platformda daha fazla vakit geçirmesini teşvik ettiği, etkileşim oranlarını artırdığı ve hatta bu deneyimlerin sosyal çevrelerle paylaşılması yoluyla organik bir kullanıcı kazanımına katkı sağlayabileceği öngörülmektedir. Bu yaklaşım, Temu gibi sektördeki öncü platformların uyguladığı oyunlaştırma ve anlık ödüllendirme stratejileriyle benzerlikler taşımakta ve kullanıcı motivasyonunu artırmada önemli bir potansiyel barındırmaktadır.

Netice itibarıyla, bu tez çalışması kapsamında tasarlanan ve hayata geçirilen kural tabanlı e-ticaret sistemi, hem teorik altyapısı hem de pratik uygulamasıyla, e-ticaret işletmelerinin müşteri sadakatini pekiştirmelerine, satış hacimlerini artırmalarına ve kişiselleştirilmiş pazarlama stratejileriyle rekabette öne çıkmalarına olanak tanıyan yenilikçi bir çözüm sunmaktadır. Elde edilen sonuçlar, kullanıcı davranışlarının derinlemesine analizine dayanan ve dinamik olarak uyum sağlayabilen indirim sistemlerinin, günümüzün rekabetçi ve hızla değişen e-ticaret ortamında işletmeler için kritik bir başarı faktörü olduğunu bir kez daha teyit etmektedir. Bu çalışma, gelecekteki benzer sistemlerin geliştirilmesi için bir temel

teşkil etmekte ve e-ticarete kişiselleştirmenin sınırlarını genişletme potansiyeli taşımaktadır.

5. KAYNAKÇA

- [1] F. Ricci, L. Rokach, and B. Shapira, *Recommender Systems Handbook*, 2nd ed. Springer, 2020.
- [2] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, *Recommender Systems: An Introduction*. Cambridge University Press, 2011.
- [3] A. Sharma and N. Aggarwal, “Personalized discounting in recommender systems: A review,” *ACM Computing Surveys*, vol. 54, no. 6, pp. 1–30, 2021.
- [4] Y. Sun, Y. Guo, and S. Barnes, “Exploring the value of personalized pricing in e-commerce,” *Electronic Commerce Research and Applications*, vol. 43, p. 101007, 2021.
- [5] W. Wang, Y. Li, and M. Zhang, “Context-aware recommender systems with temporal data,” *Information Sciences*, vol. 585, pp. 620–639, 2022.
- [6] G. Adomavicius and A. Tuzhilin, “Context-aware recommender systems,” *AI Magazine*, vol. 42, no. 3, pp. 40–53, 2021.
- [7] J. Hamari and H. Sarsa, “Does gamification work? – A literature review of empirical studies,” in *Proc. 47th Hawaii Int. Conf. Syst. Sci. (HICSS)*, 2020, pp. 3025–3034.
- [8] B. Xu and J. Hamari, “The role of gamification in recommender systems,” *Int. J. Inf. Manage.*, vol. 60, p. 102331, 2021.
- [9] F. Sadri, M. Eirinaki, and I. Varlamis, “Gamified recommender systems: A survey,” *User Modeling and User-Adapted Interaction*, vol. 32, pp. 1–27, 2022.
- [10] R. H. Hall and P. Hanna, “The impact of web page text–background colour combinations on readability, retention, aesthetics, and behavioural intention,” *Int. J. Hum. Comput. Interact.*, vol. 19, no. 2, pp. 221–233, 2004.
- [11] N. Kaya and H. H. Epps, “Relationship between color and emotion: A study of college students,” *Color Research & Application*, vol. 29, no. 4, pp. 394–399, 2004.
- [12] L. I. Labrecque and G. R. Milne, “Exciting red and competent blue: The importance of color in marketing,” *J. Acad. Market. Sci.*, vol. 40, pp. 711–727, 2012.

- [13] S. Singh, "Impact of color on marketing," *Management Decision*, vol. 44, no. 6, pp. 783–789, 2006.
- [14] M. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)," *RFC 7519*, May 2015. [Online]. Available: <https://tools.ietf.org/html/rfc7519>
- [15] D. Hardt, A. Nadalin, and E. Byrne, "The OAuth 2.0 Authorization Framework," *RFC 6749*, Oct. 2012. [Online]. Available: <https://tools.ietf.org/html/rfc6749>
- [16] X. Chen and Y. Zhao, "Secure session management in e-commerce platforms using JWT and OAuth2," *J. Cybersecurity and Privacy*, vol. 1, no. 2, pp. 145–162, 2021.
- [17] Y. Zhang and J. Liu, "AI-based recommender systems: Challenges and opportunities," *IEEE Access*, vol. 11, pp. 6253–6266, 2023.
- [18] C. Lin and Y. Yeh, "Real-time recommendation system for e-commerce using streaming data," *Expert Syst. Appl.*, vol. 158, p. 113567, 2020.
- [19] A. García-Mendoza, J. R. Velasco-Mata, and R. A. Luján-Mora, "Machine learning models in e-commerce recommender systems: A systematic review," *Appl. Sci.*, vol. 11, no. 18, p. 8456, 2021.
- [20] D. Jannach, L. Lerche, F. Gedikli, and G. Bonnini, "What recommenders recommend: an analysis of accuracy, popularity, and sales diversity effects," in *RecSys '13: Proceedings of the 7th ACM Conference on Recommender Systems*, 2013, pp. 25–32.
- [21] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, Jan.–Feb. 2003.
- [22] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *UAI '09: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 2009, pp. 452–461.
- [23] M. Quadrana, P. Cremonesi, and D. Jannach, "Sequence-aware recommender systems," *ACM Computing Surveys*, vol. 51, no. 4, pp. 1–36, Jul. 2018.
- [24] S. E. Palmer and K. B. Schloss, "An ecological valence theory of human color preference," *Proceedings of the National Academy of Sciences*, vol. 107, no. 19, pp. 8877–8882, 2010.

- [25] W. Lidwell, K. Holden, and J. Butler, *Universal Principles of Design*. Rockport Publishers, 2010.
- [26] R. Hall, "Color Theory for Designers," *Smashing Magazine*, 2011.
- [27] N. Provos and D. Mazieres, "A future-adaptable password scheme," in *Proc. USENIX Annual Technical Conference*, 1999.
- [28] B. F. Skinner, *Science and Human Behavior*. New York, NY, USA: Macmillan, 1953.
- [29] J. Hamari, J. Koivisto, and H. Sarsa, "Does gamification work? – A literature review of empirical studies on gamification," in *Proc. 47th Hawaii International Conference on System Sciences*, 2014.
- [30] G. Zichermann and C. Cunningham, *Gamification by Design: Implementing Game Mechanics in Web and Mobile Apps*. O'Reilly Media, 2011.
- [31] M. Fowler, *Patterns of Enterprise Application Architecture*, Addison-Wesley, 2002.
- [32] Pivotal Software, Inc., "Spring Boot Reference Documentation," [Online]. Available: <https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>
- [33] C. Walls, *"Spring in Action," 6th Edition*, Manning Publications, 2022.
- [34] R. Winch, "Spring Security Reference," [Online]. Available: <https://docs.spring.io/spring-security/reference/>
- [35] W. J. Reinartz and V. Kumar, "The Impact of Customer Relationship Characteristics on Profitable Lifetime Duration," *Journal of Marketing*, vol. 67, no. 1, pp. 77–99, 2002.
- [36] S. Chen, H. Marmorstein, M. Tsiros, and A. R. Rao, "When More Is Less: The Impact of Base Value Neglect on Consumer Preferences for Bonus Packs over Price Discounts," *Journal of Marketing*, vol. 76, no. 4, pp. 64–77, 2012.
- [37] M. Wedel and P. K. Kannan, "Marketing Analytics for Data-Rich Environments," *Journal of Marketing*, vol. 80, no. 6, pp. 97–121, 2016.
- [38] M. E. Porter, *Competitive Advantage: Creating and Sustaining Superior Performance*, New York: Free Press, 1985.

- [39] S. Gupta and D. R. Lehmann, “Customers as Assets,” *Journal of Interactive Marketing*, vol. 17, no. 1, pp. 9–24, 2003.
- [40] L. I. Labrecque and G. R. Milne, “Exciting red and competent blue: The importance of color in marketing,” *Journal of the Academy of Marketing Science*, vol. 40, pp. 711–727, 2012.
- [41] Ticimax, “2024 Yılı Türkiye E-Ticaret Verileri ve İstatistikleri: Kapsamlı Analiz,” 2024.
- [42] Yeşilyurt Gazete, “Türkiye’de İnternet Kullanım İstatistikleri 2024: E-Ticaret ve Kamu Hizmetleri Trendleri,” 2024.
- [43] TÜİK, “Hanehalkı Bilişim Teknolojileri (BT) Kullanım Araştırması, 2024,” 2024.