



ONDOKUZ MAYIS ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

WEB PROGRAMLAMA FİNAL ÖDEVİ

YAZIKOLİK ELEKTRONİK TİCARET SİTESİ

16060145-Rabia ÖZCAN

16060015-Yahya Furkan KILIÇOĞLU

17060807-Hüseyin MUTLU

17060721-Muhammed Abdurrahman ÖTÜN

HAZİRAN, 2021

İÇİNDEKİLER

I GİRİŞ

1	ÖZET	2
2	AMAÇ	4
3	KAPSAM	5
3.1	Proje Kapsamı	5
3.2	Hedef Kitle	5

II KAVRAMSAL TASARIM

4	GEREK SINİMLER	7
4.1	Müşteri Gereksinimleri	7
4.2	Yönetici Gereksinimleri	7
5	SİSTEM KULLANICILARI TANIMI	9
5.1	Müşteri	9
5.2	Yönetici	9
6	SİSTEM MODELLERİ	10
6.1	ACTOR-GOAL MODEL	10
6.2	USE-CASE	10
6.2.1	Belirlenen Üç Kritik Senaryo İçin Use-Case	11
7	GELİŞTİRME	15

7.1	PROGRAMLAMA DİLİ VE IDE SEÇİMİ	15
7.2	Yazılımın Hiyerarşik Yapısı	15
8	KULLANILAN TEKNOLOJİLER	17
8.1	MAVEN	17
8.1.1	POM Belgesi	17
8.1.2	Bağımlılıklar ve Maven Deposu	18
8.2	HİBERNATE	18
8.3	WEB SERVİS VE AJAX	18
8.3.1	Rest ve RestFull	19
8.3.2	SOAP(Simple Access Protocol)	19
III MANTIKSAL TASARIM		
9	ACTIVITY DIAGRAMS	21
IV KAYNAK KOD VE AÇIKLAMALAR		
10	ARAYÜZ VE TASARIM VE İLGİLİ KODLAR	24
10.1	ANA SAYFA	24
10.1.1	index.jsp	26
10.1.2	product.jsp	35
10.2	SEPET	38
10.3	ÜYE GİRİŞ İŞLEMLERİ	39
10.3.1	Kayıt Ol	39
10.3.2	Giriş	43
10.4	ÖDEME İŞLEMLERİ	46
10.5	YÖNETİCİ PANELİ	51

11	WEB SERVİS	53
11.1	Card.java	55
11.2	CardProcess.java	56
11.3	MyResource.java	58
12	ÜRÜNLERİN YER ALDIĞI VERİTABANI KODLARI VE AÇIKLAMALARI	60
13	SERVLETLAR	61
13.1	AdminServlet	62
13.2	BasketServlet	64
13.3	MainServlet	69
V	SONUÇ	
14	SONUÇLAR	76

ŞEKİLLER LİSTESİ

1	Sistem İçin Use-Case Modeli	11
2	Yazılımın Hiyerarşik Yapısı	16
3	Sisteme Kayıt Olma Diyagramı	21
4	Sisteme Giriş Diyagramı	22
5	Sipariş Oluşturma Diyagramı	22
6	Satın Alma Diyagramı	22
7	Ana Sayfa Arayüzü	25
8	Sepet Arayüzü	38
9	Üye Kayıt Arayüzü	39
10	Üye Giriş Arayüzü	43
11	Ödeme Ekranı Arayüzü	46
12	Admin İçin Yönetim Paneli	52
13	Sipariş Tamamlandıktan Sonra Kullanıcı Arayüzü	54

ÇİZELGELER LİSTESİ

1	Actor-Goal Modeli	10
---	-----------------------------	----

BÖLÜM: I

GİRİŞ

Ö Z E T

Bu proje genel hatları ile kitap satışı yapılan elektronik ticaret sitesidir. Müşteriler için kitaplar kategorize edilmiştir. Müşteriler isterlerse kategoriler arasında seçim yaparak isterlerse de arama çubuğu kullanarak kitapları listeleyebilirler; sisteme kayıt olarak ya da giriş yaparak seçtikleri ürünü sepete ekleyip, sipariş oluşturabilir ya da sepetteki ürünlerini bekletebilirler. Sipariş verdikten sonra müşteri sipariş numarasını ya da geçmişe dönük siparişlerini görüntüleyebilir. Sistem yöneticisi ise sisteme giriş yaptığında toplam satış miktarını ve ciroyu görüntüleyebilirler.

Proje gerçekleştirirken Java programlama dili, maven proje yapısı ve veri tabanı bağlantısı için hibernate tercih edilmiştir. Bu dil ve mimarilerin seçilmesinin sebebi birbirleri ile uyumlu çalışması ve kod karmaşasının ortadan kalkmasıdır.

Anahtar Kelimeler: Web programlama, Web servis, Web Tasarım, Hibernate, Maven, AJAX, Elektronik ticaret, Bootstrap

Görev Dağılımı: Arayüz: Yahya Furkan Kılıçoğlu, Muhammed Abdurrahman Ötün
Web Servis: Hüseyin Mutlu, Yahya Furkan Kılıçoğlu, Rabia Özcan
Veritabanı : Muhammed Abdurrahman Ötün, Hüseyin Mutlu
Ajax Sorguları : Rabia ÖZCAN, Yahya Furkan Kılıçoğlu
Servletlar: Rabia ÖZCAN, Muhammed Abdurrahman Ötün

Github Linki: <https://github.com/huseyinmuttlu/YaziKolikWebProject>

Youtube Video Linki: https://youtu.be/h83_KeEgT-E

AMAÇ

Bu çalışmanın amacı, kitap satışı yapan firmalar için elektronik ortamdada estetik ve işlevsel bir mağaza sunmaktır. Satıcının sektördeki pazar payını yükseltmesi amacı ile rakiplerinden daha çok tercih edilme sebebi olacak şekilde müşteri açısından kolay kullanılabilir görsel bir arayüze sahiptir. Projede daha fazla okuyucuya hitap etmek amacıyla çeşitli kategoriler sunulmuştur. Satıcı içinde işlevsel kullanım amaçlanmış, tasarım sade tutulmuştur. Projede satıcının fiziksel mağazalara kıyasla çok daha masrafsız şekilde süreci yönetmesi, pazarlama fırsatları ve hedef kitle için sonsuz bir alandan faydalanması amaçlanır. Bu proje ile müşteri açısından satıcı açısından 7/24 açık bir kitap satış mağazası hedeflenmektedir.

K A P S A M

3.1 *Proje Kapsamı*

Proje ürünün tanıtımının yapılması, ticari muameleler hesaplarının ödenmesi ile ilgili tüm ticari etkinlikleri kapsamaktadır. Okuyucu ve kitap satışı yapan mağaza arasında şahıssız iletişim kurmakta, taraflar birbirleri ile muhattap olmadan alışveriş yapabilmektedir.

3.2 *Hedef Kitle*

Projenin hedef kitlesi şahıslar, perakende satış noktaları, butik mağazalar, market zincirler, eğitim, eğlence, gibi pek çok sektördür.

Projenin kullanım hedef kitlesi ise şahıslar, perakende satış noktaları, kırtasiyeler, sahaflar ve diğer kitap satıcılarıdır.

BÖLÜM: II

KAVRAMSAL TASARIM

GEREKSİNİMLER

4.1 *Müşteri Gereksinimleri*

- Sisteme kayıt olmalıdır.
- Sisteme giriş yapmalıdır.
- Satın alacağı kitabı seçip, sepete eklemelidir.
- Sepette kaç adet ürün olduğunu görüntülemelidir.
- Seçtiği ürünleri satın almak için ödeme işlemi yapmalıdır.
- Ödeme işleminin başarılı oldu ise sipariş numarası görmelidir.
- Ödeme işlemi başarısız oldu ise hata mesajını görmelidir.
- Geçmişe dönük siparişlerini görüntülemelidir.

4.2 *Yönetici Gereksinimleri*

- Sisteme giriş yapmalıdır.

- Sipariş listemelidir.
- Sipariş detaylarını görmelidir.
- Toplam ciro ve satış adedini görüntülemelidir.

SİSTEM KULLANICILARI TANIMI

Sistem, kitap satışı yapılan bir elektronik ticaret sitesidir. Sistemin iki tip kullanıcısı vardır. İlki ürün satın almak isteyen müşteri, ikincisi ürünleri satan ve takibini yapan sistem sahibi yöneticidir.

5.1 *Müşteri*

Müşteri, sisteme giriş yaparak ya da kayıt olarak farklı kategorilerdeki kitapları sistemden bulabilir ve kredi kartı aracılığı ile satıcıdan bu kitabı satın alabilir. Müşteri siparişinden sonra sisteme giriş yaptığında siparişlerini görüntüleyebilir.

5.2 *Yönetici*

Yönetici, sisteme giriş yaparak, alınan siparişleri listeleyebilir, toplam satış adedini, sipariş detaylarını ve toplam ciroyu görüntüleyebilir.

SİSTEM MODELLERİ

6.1 ACTOR-GOAL MODEL

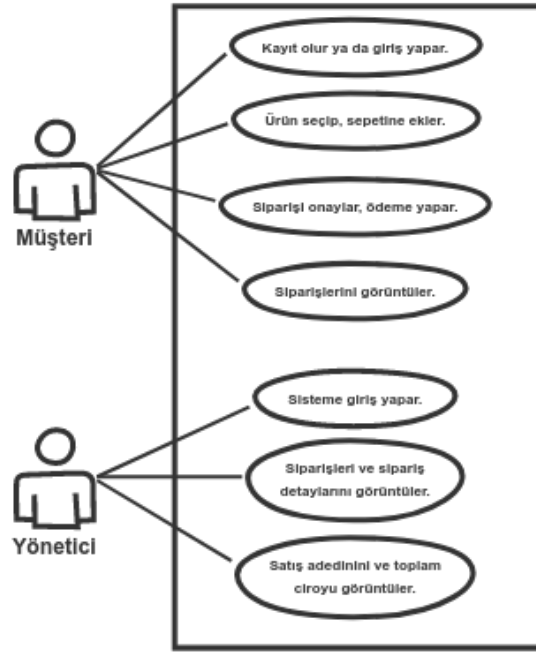
Aktör, sistemi kullanacak kişi veya başka bir sistem iken hedef, aktörün sistemi kullanma amacını belirtir. Çizelge 1’de sol tarafta aktörler, sağ tarafta ise hedefler yer almaktadır.

Müşteri	Ürünleri listeler, seçim yapar. Seçtiği ürünü satın alır. Geçmişe dönük siparişlerini görüntüler.
Yönetici	Siparişleri, adet ve detay olarak görüntüler. Toplam ciro miktarını görüntüler.

Çizelge 1: Actor-Goal Modeli

6.2 USE-CASE

Sistemin use-case modeli Şekil 1’de yer almaktadır.



Şekil 1: Sistem İçin Use-Case Modeli

6.2.1 Belirlenen Üç Kritik Senaryo İçin Use-Case

Senaryolar müşteri için sisteme kayıt olma ve giriş yapma, ürün sipariş verme ve ödeme yapma, yönetici için ise siparişleri görüntüleyip toplam ciro miktarını görme şeklinde belirlenmiştir.

6.2.1.1 Use-Case 1 Müşterinin Sisteme Kayıt Olması ve Giriş Yapması

- **Adı:** Yazıkolik internet sitesine kayıt olma ve giriş yapma
- **Aktörü:** Müşteri
- **Seviye:** Müşteri hedefi
- **Ön Koşullar:** Müşterinin kitap almak için Yazıkolik internet sitesine girmesi
- **Başarılı Senaryo:**

1. Müşteri internet sitesine girer.
2. Siteye giren müşteri kayıt olma işlemini başlatır.
3. Müşteri kendine ait bilgileri sisteme girer.
4. Sistem müşteriden girdiği bilgileri doğrulamasını ister.
5. Sistem girilen bilgiler aracılığı ile müşteriye kayıt eder.
6. Müşteri kayıt olmak için kullandığı bilgiler ile sisteme giriş yapar.

- **Alternatif yollar:**

1. Sistem dördüncü adımda doğrulama istediğinde müşteri üçüncü adımda hata yaptığını anlar.
 - a) Sistem müşteriye üçüncü adıma yönlendirir.

6.2.1.2 Use-Case 2 Ürün Sipariş Verme ve Ödeme Yapma

- **Adı:** Siteden kitap siparişi verip, ödemesini yapma
- **Aktörü:** Müşteri
- **Seviye:** Müşteri hedefi
- **Ön Koşullar:** Müşterinin siteden kitap seçmesi
- **Başarılı Senaryo:**
 1. Müşteri siteye giriş yapar.
 2. Siteden bir kitap seçip, sepetine ekler.
 3. Sepetine gidip, siparişi onaylar.
 4. Ödeme işlemleri için sisteme kredi kartı bilgilerini girer.

5. Sistem kartı kontrol eder.
6. Ödeme tamamlanır ve müşteri sipariş numarasını görüntüler.

- **Alternatif yollar:**

1. Müşteri dördüncü adımda kredi kartı bilgilerini yanlış girmiştir ve sistem beşinci adımda müşteriye hata mesajı verir.
 - a) Müşteri dördüncü adıma dönüp, doğru kart bilgilerini girer.
2. Müşterinin dördüncü adımda girdiği kartın limiti yetersizdir. Beşinci adımda kullanıcıya "limit yetersiz" mesajı verilir.
 - a) Müşteri dördüncü adıma dönüp, farklı kart bilgileri girer.

6.2.1.3 Use-Case 3 Siparişleri ve Toplam Ciro Miktarını Görme

- **Adı:** Siparişleri listeleme ve toplam ciro miktarını görüntüleme
- **Aktörü:** Yönetici
- **Seviye:** Yönetici hedefi
- **Ön Koşullar:** Müşterilerin siteden kitap siparişi vermesi
- **Başarılı Senaryo:**
 1. Müşteri siteye giriş yapar ve verdiği siparişi onaylar.
 2. Müşteri onayladığı sipariş için ödeme işlemlerini tamamlar.
 3. Sistem siparişi kayıt eder.
 4. Yönetici sisteme giriş yapar.
 5. Yönetici ödemesi tamamlanan siparişleri listeler.

6. Yönetici toplam ciro miktarını görüntüler.

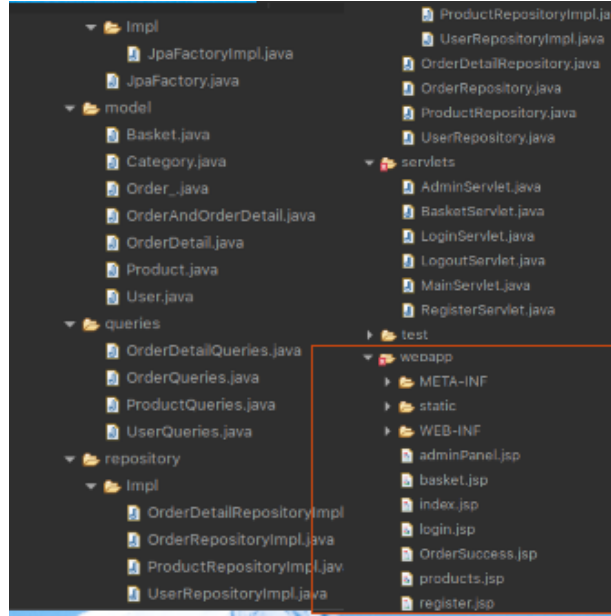
GELİŞTİRME

7.1 PROGRAMLAMA DİLİ VE IDE SEÇİMİ

Arayüz için JSP, fonksiyonlar için Java, veritabanı işlemleri için MySQL kullanılmıştır. IDE olarak Eclipse IDE 2021-06, Apache Tomcat 8 ve yer yer Apache Tomcat 10 tercih edilmiştir.

7.2 Yazılımın Hiyerarşik Yapısı

Yazılımın sahip olduğu hiyerarşik yapı Eclipse 2021-06 programından alınmıştır. Raporun görünümü açısından yanyana gruplanmıştır. Yapı Şekil 2’de görünmektedir.



Şekil 2: Yazılımın Hiyerarşik Yapısı

KULLANILAN TEKNOLOJİLER

8.1 *MAVEN*

Maven, önceleri Java projeleri için kullanılan yapı otomasyon aracı iken zamanla C#, Ruby, Scala ve diğer dillerde yazılmış projeleri yönetmek içinde kullanılabilir hale gelmiştir. Apache tarafından geliştirilmiş olup, Jakarta projesinin bir parçasıdır.

Asıl amacı geliştirme sürecini azaltmak olan maven, sabit bir proje yapısı sağlayarak inşa işlemi, POM(Project Object Model)'a dayanarak bağımlılık güncellemesi yapma ve derleme işlemlerini kolaylaştırır.

8.1.1 *POM Belgesi*

Proje hakkında bilgiler, projenin bağımlılıkları ve projeyi derlemek için gerekli komutlar POM Belgesinde yer alır. Bir proje maven yapısı ile kullanılacak ise kesinlikle bir pom.xml dosyasına sahip olmalıdır.

8.1.2 Bağımlılıklar ve Maven Deposu

Maven projesinde kullanılan JAR dosyaları ya da bir başkasının yazmış olduğu kütüphane, proje için bir bağımlılıktır.

Maven projelerinde kullanılabilecek JAR dosyaları ve kütüphanelerin yer aldığı depodur. <https://mvnrepository.com/> adresi aracılığı ile ulaşılabilen bu depoda yer alan kütüphane ve JAR dosyaları, projenin POM belgesi aracılığı ile projeye eklenmelidir.

8.2 HİBERNATE

Hibernate, ORM(Object Relational Mapping) aracıdır. ORM ise nesne odaklı dillerdeki nesnelerin ilişkisel veri tabanlarındaki kayıtlara nasıl karşılık geldiğini yürüten bir teknolojidir. Hibernate aracılığı ile kod karmaşası olmadan bir nesne veri tabanına kayıt edilip, güncellenip, sorgulama yapılabilir.

8.3 WEB SERVİS VE AJAX

Web servis, platform bağımsız olarak elektronik cihazlar arası iletişimi sağlayan bir yapıdır. Rest veya SOAP şeklinde kullanılabilirler.

8.3.1 *Rest ve RestFull*

Rest, sunucu ve istemci arasındaki iletişimin HTTP protokolü ile yapılmasını sağlayan mimaridir. Tüm bilgiler URI'ler üzerinden sunulur. Restful web servisleri de REST mimarisi temel alınarak gerçekleştirilmiş servislerdir. Amacı sunucu ve istemci arasındaki iletişimi platform bağımsız şekilde kurabilmektir. Restful servis yanıt tipi olarak JSON, HTML, XML gibi bir çok formatı kullanabilir ancak yapısı az yer kapladığı ve bir çok mobil platformda kullanışlı olduğu için JSON daha çok tercih edilir.

8.3.2 *SOAP(Simple Access Protocol)*

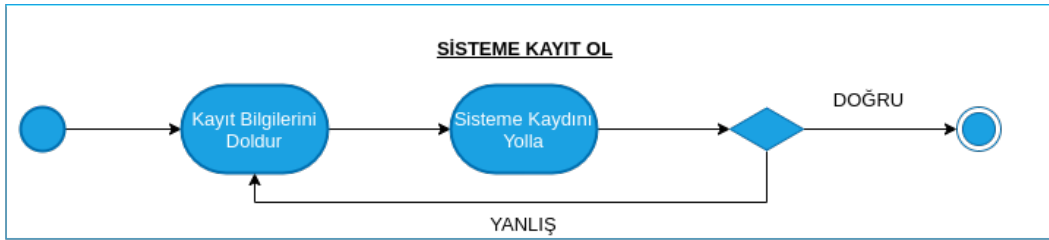
İnternet aracılığı ile bilgi ve mesajları aktarma protokolüdür. SOAP mesajları XML formatında kullanıma mecbur bırakır ve genellikle HTTP protokolü kullanarak gönderir.

BÖLÜM: III

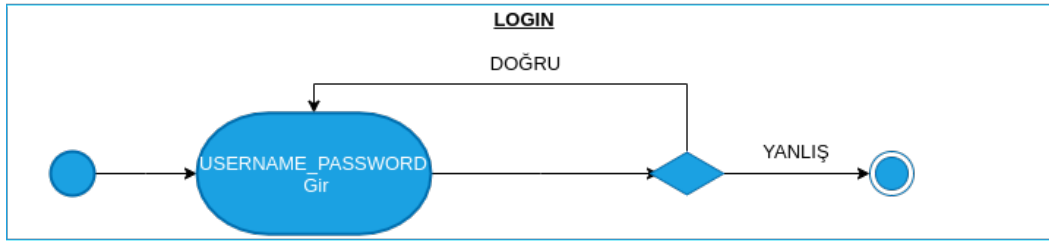
MANTIKSAL TASARIM

ACTIVITY DIAGRAMS

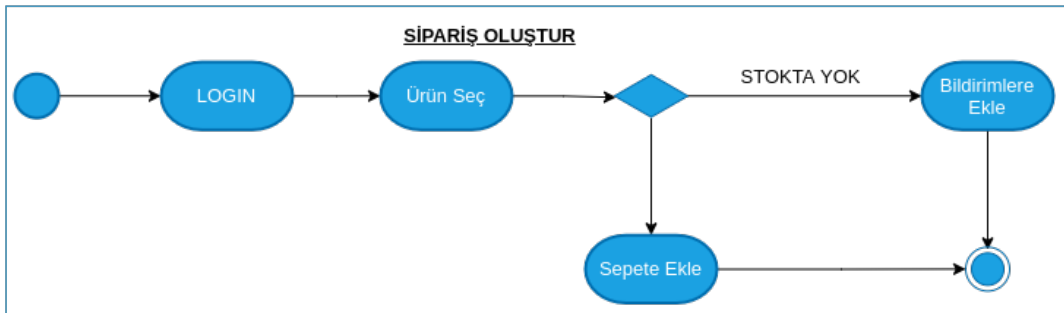
Sisteme müşteri ilk önce kayıt olmalıdır. Kayıt olmak için Şekil 3'deki adımları takip etmelidir. Kayıt olduktan sonraki zamanlarda sisteme giriş yapmak için Şekil 4'deki adımları, sipariş oluşturmak için Şekil 5'deki adımları takip etmelidir. Satın alma işlemi için sisteme giriş yapmış olmalı ve Şekil 6'daki adımları takip etmelidir.



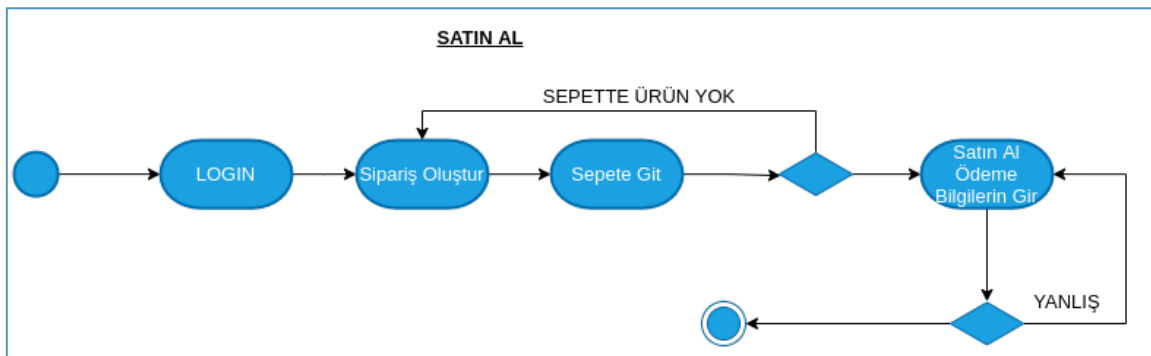
Şekil 3: Sisteme Kayıt Olma Diyagramı



Şekil 4: Sisteme Giriş Diyagramı



Şekil 5: Sipariş Oluşturma Diyagramı



Şekil 6: Satın Alma Diyagramı

BÖLÜM: IV


KAYNAK KOD VE AÇIKLAMALAR

ARAYÜZ VE TASARIM VE İLGİLİ KODLAR

Bu bölümde sayfa yerleşimine göre, internet sitemizin ara yüzü, ardından o arayüze ait kodlar ve açıklamaları yer almaktadır.


10.1 ANA SAYFA

Ana sayfanın kodlar ve ardından Şekil 7’da internet sitemizin ana sayfa görüntüsü yer almaktadır. Ana sayfamızın script kodları arasında yer alan `console.log()` şeklindeki komutlar ekranda veri göstermekten ya da back-endde işe yaramaktan ziyade kodlamayı yaparken, istediğimiz çıktıyı alıp almadığımızı konsol üzerinden görebilmek için yazdığımız komutlardır.



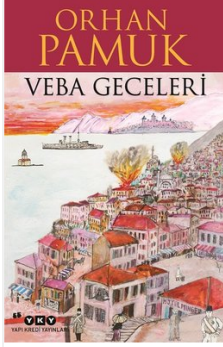
[Üye Ol](#)
[Giriş Yap](#)
[Sepet](#)

Edebiyat
Eğitim
Felsefe
Bilim
Yazılım



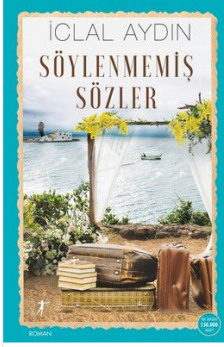
Balıkçı ve Oğlu
Zülfü Livaneli
İnkılap Kitabevi
19.6

Ekle - 1 +



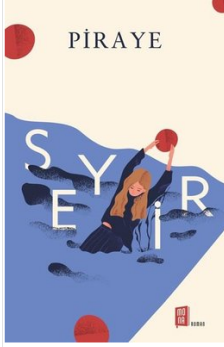
Veba Geceleri
Orhan Pamuk
Yapı Kredi Yayınları
29.25

Ekle - 1 +



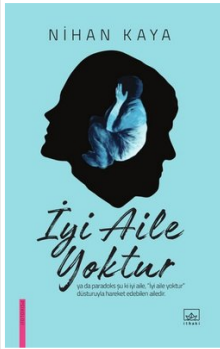
Söylenmemiş Sözler
İclal Aydın
Artemis Yayınları
27.3

Ekle - 1 +



Seyir
Piraye
Mona
28.6

Ekle - 1 +



İyi Aile Yoktur
Nihan Kaya
İthaki Yayınları
19.25


Ekle - 1 +

Kategoriler

- Edebiyat
- Eğitim
- Felsefe
- Bilim
- Yazılım

İletişim

Email: furkan.kilicoglu@bil.omu.edu.tr
Email: abdurrahman.otun@bil.omu.edu.tr
Email: rabia.ozcan@bil.omu.edu.tr
Email: huseyin.mutlu@bil.omu.edu.tr



Daha büyük haritayı görüntüle

Ondokuz Mayıs Üniversitesi

Adres: Atakum/Samsun

© Copyright 2021 - Bu Ödevin Tüm Hakları Saklıdır.

Şekil 7: Ana Sayfa Arayüzü

10.1.1 *index.jsp*

```
1 <%@ page import="javax.servlet.http.HttpSession"%>
2 <%@ page import="java.util.ArrayList,com.yazikolik.model.Product"%>
3 <%@ page language="java" contentType="text/html; charset=UTF-8"
4     pageEncoding="UTF-8"%>
5 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
6 <!DOCTYPE html>
7 <html lang="tr">
8 <head>
9     <meta charset="UTF-8">
10    <meta http-equiv="X-UA-Compatible" content="IE=edge">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0"
12        >
13    <meta name="description" content="Kitap Satis Sitesi">
14    <meta name="keywords" content="kitap ,okumak ,egitim ,roman ,edebiyat">
15    <meta name="author" content="Rabia OZCAN, Yahya Furkan KILICOGLU,
16        Muhammed Abdurrahman OTUN, Huseyin MUTLU">
17    <!--CSS bagimliliklari-->
18    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/
19        bootstrap.min.css" rel="stylesheet" >
20    <!--Javascript Bagimliliklari-->
21    <script> <%@ include file = "static/js/fontawesome-icons.js" %></
22        script>
23    <title>YaziKolik</title>
24    <script src="https://code.jquery.com/jquery-3.6.0.slim.min.js"></
25        script>
26    <style type="text/css">
27    <%@ include file = "static/css/style.css" %>
```



```
23     </style>
24     <style>
25     .google-maps {
26         position: relative;
27         padding-bottom: 75%;
28         height: 0;
29         overflow: hidden;
30     }
31     .google-maps iframe {
32         position: absolute;
33         top: 0;
34         left: 0;
35         width: 100% !important;
36         height: 100% !important;
37     }
38 </style>
39 </head>
40 <body>
41 <nav class="navbar" style="background-color: #F1EBF5;">
42     <div class="container-fluid py-2 row text-center">
43         <div class="col-md-2">
44             <a class="navbar-brand ms-2" href="#">
45                 <!-- Logo boyutu width ve height ile ayarlanabilir -->
46                 
48             </a>
49         </div>
50         <div class="col-md-7 my-4">
51             <form class="d-flex w-100">
```

```

51     <input class="form-control" type="search" placeholder="Kitap veya
        Yazar Adı" aria-label="Search" id="searchInput">
52     <button class="btn btn-outline-primary ms-2 d-flex" id="searchBtn
        " type="submit"><i class="fas fa-search pt-1 pe-1"></i>Ara</
        button>
53 </form>
54 </div>
55 <div class="col-md-3">
56     <div class="btn-group-vertical me-2">
57         <c:choose>
58             <c:when test="{sessionScope.user != null}">
59                 <a href="LogoutServlet" class="btn btn-outline-primary mx-1">
60                     <i class="fas fa-sign-out-alt"></i><span class="px-2"> İkiş Yap
                        </span>
61                 </a>
62             </c:when>
63             <c:otherwise>
64                 <a href="register" class="btn btn-outline-primary btn-sm">
65                     <i class="fas fa-user-plus"></i><span class="px-2">Üye Ol</span
                        >
66                 </a>
67                 <a href="login" class="btn btn-outline-primary mt-2 btn-sm">
68                     <i class="fas fa-user-check"></i><span class="px-2">Giriş Yap</
                        span>
69                 </a>
70             </c:otherwise>
71         </c:choose>

```

```

72     <a href="basket" class="btn btn-outline-danger btn-sm mt-2"><span
        id="basketCounter" class="badge bg-warning rounded-pill mx-1
        ">0</span><i
73         class="fas fa-shopping-cart"></i><span class="px-2">Sepet </
        span></a>
74     </div>
75 </div>
76 </div>
77 </nav>
78
79 <!-- Kategori gösterim yeri-->
80 <div class="container-fluid" style="background-color: #4887B0;">
81     <div class="text-center py-3">
82         <a class="nav-1 rounded-3 mx-1 px-2 category-item" href="edebiyat.
            jsp">Edebiyat </a>
83         <a class="nav-2 rounded-3 mx-1 px-2 category-item" href="egitim.jsp
            ">Eğitim </a>
84         <a class="nav-3 rounded-3 mx-1 px-2 category-item" href="felsefe.
            jsp">Felsefe </a>
85         <a class="nav-4 rounded-3 mx-1 px-2 category-item" href="bilim.jsp"
            >Bilim </a>
86         <a class="nav-5 rounded-3 mx-1 px-2 category-item" href="yazilim.
            jsp">Yazı lım</a>
87     </div>
88 </div>
89
90 <section id="productjsp">
91 <%@ include file="products.jsp" %>
92 </section>

```

93

```
94 <footer class="footer mt-auto" style="background-color: #F1EBF5;">
```

```
95     <div class="container-fluid">
```

```
96         <div class="row row-cols-3 text-center">
```

```
97             <div class="col">
```

```
98                 <div class="my-1"><a class="footer-link" href="#"><b>
```

```
                    Kategoriler </b></a></div>
```

```
99                 <div class="my-1"><a class="footer-link" href="
```

```
                    product.jsp">Edebiyat </a></div>
```

```
100                <div class="my-1"><a class="footer-link" href="
```

```
                    edebiyat.jsp">Eğitim </a></div>
```

```
101                <div class="my-1"><a class="footer-link" href="egitim
```

```
                    .jsp">Felsefe </a></div>
```

```
102                <div class="my-1"><a class="footer-link" href="
```

```
                    felsefe.jsp">Bilim </a></div>
```

```
103                <div class="my-1"><a class="footer-link" href="bilim.
```

```
                    jsp">Yazılım </a></div>
```

```
104            </div>
```

```
105        <div class="col">
```

```
106            <div class="my-1"><a class="footer-link" href="#"><b>
```

```
                İletişim </b></a></div>
```

```
107            <div class="my-1"><a class="footer-link" href="mailto:
```

```
                furkan.kilicoglu@bil.omu.edu.tr">Email: furkan.
```

```
                kilicoglu@bil.omu.edu.tr </a></div>
```

```
108            <div class="my-1"><a class="footer-link" href="mailto:
```

```
                abdurrahman.otun@bil.omu.edu.tr">Email:
```

```
                abdurrahman.otun@bil.omu.edu.tr </a></div>
```

```

109      <div class="my-1"><a class="footer-link" href="mailto:
      :rabia.ozcan@bil.omu.edu.tr">Email: rabia.
      ozcan@bil.omu.edu.tr </a></div>
110      <div class="my-1"><a class="footer-link" href="mailto:
      :huseyin.mutlu@bil.omu.edu.tr">Email: huseyin.
      mutlu@bil.omu.edu.tr </a></div>
111  </div>
112  <div class="col">
113  <div class="google-maps">
114      <iframe src="https://www.google.com/maps/embed?pb=!1m18
      !1m12!1m3!1d11978.07358235022!2d36.18899684401959!3
      d41.36282063753321!2m3!1f0!2f0!3f0!3m2!1i1024!2i768
      !4f13.1!3m3!1m2!1s0x40887eff9abfffe9%3
      A0x96d2f4886017389e!2
      zNDHCsDIxJzUxLjgiTiAzNsKwMTEnMDQuMCJF!5e0!3m2!1str
      !2str!4v1624139653817!5m2!1str!2str" width="2,66"
      height="2" style="border:0;"></iframe>
115
116  </div>
117  <div class="my-1"><a class="footer-link" href="https://
      www.google.com/maps/place/41%C2%B021'51.8%22N+36%C2%
      B011'04.0%22E/@41.3644013,36.1669385,14z/data=!4m6!3
      m5!1s0x40887eff9abfffe9:0x96d2f4886017389e!7e2!8m2!3
      d41.3644013!4d36.184448">Adres: Atakum/Samsun</a></
      div>
118  </div>
119  </div>

```

```
120         <div class="border-top container text-center py-2 text-muted"
            > Copyright 2021 – Bu Ödevin Tüm Hakları Saklıdır.</div>
            >
121     </div>
122 </footer>
123
124 <script type="text/javascript"
125     src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/
            jquery.min.js">
126 </script>
127
128 <script>
129 $(function () {
130     $("#searchBtn").click(function () {
131         var searchInput = $("#searchInput").val();
132         console.log(searchInput);
133         document.getElementById("productjsp").innerHTML="";
134         $.post("/YaziKolik/index", {action:"search",searchInput:
            searchInput }, function (data) {
135             $("#productjsp").html(data);
136         });
137
138
139     });
140     $(".btn-decrease").click(function () {
141         var currentVal = parseInt($(this).next().val());
142         if(currentVal > 1){
143             currentVal -= 1;
144             $(this).next().val(currentVal);
```

```
145         }
146     });
147     $(".btn-increase").click(function() {
148         var currentVal = parseInt($(this).prev().val());
149         if(currentVal < 50){
150             currentVal += 1;
151             $(this).prev().val(currentVal);
152         }
153     });
154 });
155
156 $(''.cartAdd').click(function() {
157     var deger;
158     var values = [];
159     for ( var i =0; i<4;i++){
160         deger=$(this).parent().prev().children("p:nth("+i+")").text();
161         if(i==3){
162             var price = parseFloat(deger);
163             var quantity = parseInt($(this).next().next().val());
164             var cost = price * quantity;
165             values.push(cost.toFixed(3));
166             continue;
167         }
168         values.push(deger);
169     }
170     values.push(quantity);
171     console.log(values[0]);
172     console.log(values[4]);
```

```

173     $.post("/YaziKolik/index", {action:"addBasket",productTitle:values
        [0],productQuantity:values[4]}, function (basketCounter) {
174         document.getElementById("basketCounter").innerHTML=basketCounter;
175     });
176
177     console.log(values); //değerleri alıp almadığını görmek istedim.
178     $('toast').toast('show');
179     //values dizisini ajax komutları ile servletdaki sessiona yolla
180 });
181
182 </script>
183 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/js/
        bootstrap.bundle.min.js"></script>
184 </body>
185 </html>

```

Yukarıda verilen kodda 25 ve 37 satırları arası, sayfanın altında yer alan footer bölgesindeki harita için stil kodlarıdır. 41 ve 77 satırları arasında yer alan `$sessionScope.user != null` tümcesi kullanıcı girişi yapıldığını kontrol ederek navbarda "Çıkış Yap" butonunu gösterir. Kullanıcı giriş yapmadı ise navbarda "Üye Ol" ve "Giriş Yap" butonları bulunur. Bu butonların hemen altında "Sepet" butonu bulunur ve sepetin içerisindeki ürün adedini de gösterir.

84 ve 92 satırları arasında YazıKolik sitemizde bulunan kategoriler listelenir. 90 ve 92 satırları arasında product.jsp sayfası include edilir ve ürünler sergilenir. 94 ve 122 arasındaki satırlar sitenin footer alanıdır. 128 ve 136 satırları arasında yer alan script kodu product.jsp idli jsp sayfasındaki productları siler. Ajax kullanarak post isteği yapar. Post metodundan gelen bilgiye göre sorgu yapılır. print writer nesnesi kullanılarak üretilen html kodu yine product.jsp idli section alanının içerisinde jquery yardımıyla eklenir.

140 ve 146 satırları arasında yer alan script kodu btn-decrease sınıfı ile tanımlanmış butona tıklandığında çalışmaya başlar. Tıklanan butonun yer aldığı satırın sonraki satırının değerini alıp, currentVal değişkenine atar. currentVal değeri birden büyük ise azalt butonuna basılınca ürün adedi azaltılır ve mevcut elementin sonraki elementinin değeri yeni adet sayısı ile güncellenir. 147 ve 154 satırları ise 140 ve 146 arasında yer alan satırların mantıksal olarak tersine çalışır. 140-146 satırları azalt butonu için, 147-154 satırları arası arttır butonu için yazılmıştır.

156 ve 180 satırları arasında yer alan script kodları "Sepete Ekle" butonu için çalışır. Sepete ekle butonlarının sınıfı cartAdd şeklinde tanımlanmıştır. Bu sınıftan bir butona tıklandığında, tıklanan butonun içinde yer aldığı divin önceki divine gidip, orada yer alan childların textini çeker. Bunu bir for döngüsü ile yapar ve aldığı her değeri values isimli diziye atar. Üçüncü childa gelindiğinde, alınan değer floata çevrilip, price değişkenine atanır. Adet sayısı ise mevcut butonun iki sonraki satırından çekilip quantity değişkenine atılır. Sepette toplam fiyatı gösterebilmek içinde adet ve fiyat çarpılarak cost değişkenine atılır ve bu değişken diziye eklenir iken duyarlılığı da üçe ayarlanır. Alınan bu değerler Ajax komutu ile sitenin main pageine gönderilir. Ayrıca yine mevcut butona tıklandığında ekranda "Ürün sepete eklendi" şeklinde tost mesaj görüntülenir.

10.1.2 *product.jsp*

```

1 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
2 <%@ page import="java.util.ArrayList,com.yazikolik.model.Product"%>
3 <div class="container my-2">
4     <div class="row row-cols-2 row-cols-md-3 row-cols-lg-4">
5         <c:forEach items="${productList}" var="product">

```

```

6      <!--ürün başlangıcı-->
7
8      <div class="col">
9
10     <div class="card text-center mb-4 shadow-items py-2">
11
12         <div class="pb-3">
13
14             
15
16         </div>
17
18         <div class="border-top border-bottom pt-1 mb-2">
19
20             <p class="product-attr">{product.getTitle()}</p>
21
22             <p class="product-attr">{product.getAuthor()}</p>
23
24             <p class="product-attr">{product.getPublisher()}</p>
25
26             >
27
28             <p class="product-attr">{product.getPrice()}</p>
29
30         </div>
31
32         <div>
33
34             <button class="me-3 btn btn-outline-success cartAdd"
35
36                 type="submit"><i class="fas fa-shopping-cart pe-2
37
38                 "></i>Ekle</button>
39
40
41             <button class="btn rounded-pill btn-sm btn-outline-
42
43                 danger btn-decrease" type="button"><i class="fas
44
45                 fa-minus"></i></button>
46
47
48             <input type="number" value="1" style="width:25px;
49
50                 border:none;padding-left:6px;" class="mx-1">
51
52             <button class="btn rounded-pill btn-sm btn-outline-
53
54                 success btn-increase" type="button"><i class="fas
55
56                 fa-plus"></i></button>
57
58
59         </div>
60
61     </div>
62
63 </div>
64
65 </div>
66
67 <!--ürün bitişi-->

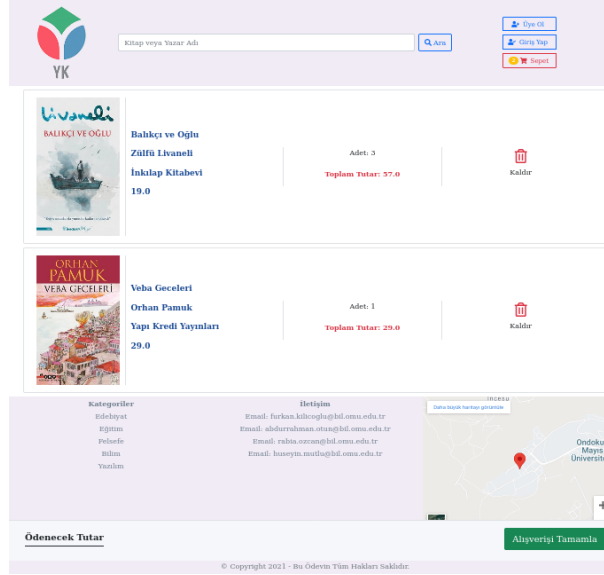
```

```
27     </c:forEach>
28
29 </div>
30
31 <!--Toast Start-->
32 <div class="fixed-top top-0 end-0">
33 <div class="toast position-absolute top-0 end-0 align-items-center
    text-white bg-success" role="alert" aria-live="assertive" aria-
    atomic="true" data-bs-delay="1500">
34 <div class="toast-header">
35     <i class="fas fa-check pe-3"></i>
36     <strong class="me-auto">Urun sepete eklendi</strong>
37     <button type="button" class="btn-close" data-bs-dismiss="toast"
        aria-label="Close"></button>
38 </div>
39 </div>
40 </div>
41 <!--Toast Finish-->
```

product.jsp, ana sayfa için referans oluşturmaktadır. Ürünler isim, yazar, yayıncı ve fiyat şeklinde listelenmektedir. Her ürünün altında arttırma, azaltma ve sepete ekleme butonu bulunmaktadır. Sepete ekle butonuna tıklandığında çalışacak tost mesaj komutları da burada yer almaktadır.

10.2 SEPET

Müşteri ürün ekledikten sonra sepetinin görünümü Şekil 8’de yer almaktadır.



Şekil 8: Sepet Arayüzü

10.3 ÜYE GİRİŞ İŞLEMLERİ

Eğer müşteri sisteme kayıtlı ise Şekil 4’de yer alan arayüz ile sisteme giriş yapılabilir iken, sisteme kayıtlı olmayan müşteriler Şekil 9 ’de yer alan arayüz ile sisteme kayıt olabilirler. Kayıt olma arayüzünün kodu 10.3.1 bölümünde, giriş yapma arayüzünün kodu ise 10.3.2 bölümünde anlatılmıştır.

10.3.1 Kayıt Ol

Şekil 9: Üye Kayıt Arayüzü

```

1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
4 <!DOCTYPE html>
5 <html lang="tr">

```

```

6 <head>
7     <meta charset="UTF-8">
8     <meta http-equiv="X-UA-Compatible" content="IE=edge">
9     <meta name="viewport" content="width=device-width, initial-scale=1.0"
10     >
11     <meta name="description" content="Kitap Satış Sitesi">
12     <meta name="keywords" content="kitap ,okumak ,eğitim ,roman ,edebiyat">
13     <meta name="author" content="Rabia ÖZCAN, Yahya Furkan KILI O LU ,
14     Muhammed Abdurrahman ÖTÜN, Hüseyin MUTLU">
15     <!--CSS bağımlilikları-->
16     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/
17     bootstrap.min.css" rel="stylesheet" >
18     <!--Javascript Bağımlilikları-->
19     <script> <%@ include file = "static/js/fontawesome-icons.js" %></
20     script>
21     <title>YaziKolik</title>
22     <style type="text/css">
23     <%@ include file = "static/css/style.css" %>
24     </style>
25 </head>
26 <body>
27     <div class="container border p-3 my-3" id="registerform">
28     <form method="post" action="" >
29         <div class="mb-3 btn btn-primary w-100" style="cursor:auto">
30             YazıKolik Üyelik Formu
31         </div>
32         <div class="mb-3">
33             <label for="isim" class="form-label">Ad</label>

```

```
30      <input type="text" class="form-control" id="isim" name="
      firstName" required>
31    </div>
32    <div class="mb-3">
33      <label for="soyisim" class="form-label">Soyad</label>
34      <input type="text" class="form-control" id="soyisim" name="
      lastName" required>
35    </div>
36    <div class="mb-3">
37      <label for="emailadres" class="form-label">E-Mail Adresi</label>
      >
38      <input type="email" class="form-control" id="emailadres" name="
      email" aria-describedby="emailHelp" required>
39    </div>
40    <div class="mb-3">
41      <label for="telno" class="form-label">Cep Telefonu</label>
42      <input type="text" class="form-control" id="telno" name="
      phoneNumber" required>
43    </div>
44
45    <div class="mb-3">
46      <label class="form-label" for="adres">Adres</label>
47      <textarea class="form-control" id="adres" name="address"
      required></textarea>
48    </div>
49    <div class="mb-3">
50      <label for="sifre" class="form-label">Şifre</label>
51      <input type="password" class="form-control" id="sifre" name="
      password">
```

```
52     </div>
53     <div class="mb-3">
54         <a href="" class="float-end pt-1" style="text-decoration: none;
55             ">Zaten Üye Misin?</a>
56     </div>
57     <button type="submit" class="btn btn-success w-25">Kayıt Ol</
58         button>
59
60 </form>
61 </div>
62
63 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/js/
64     bootstrap.bundle.min.js"></script>
65
66 </body>
67 </html>
```

Kayıt olmak için bir form tanımlanmış, bu form aracılığı ile kullanıcıdan ad, soyad, e-mail adresi, cep telefonu, adres ve şifre bilgileri alınmaktadır. Alınan bu bilgiler post methodu ile servlete gönderilmektedir.

10.3.2 Giriş

Kullanıcı sisteme giriş yapmak istediğinde Şekil 10'da yer alan arayüz ile karşılaşmaktadır.

The image shows a user login form titled 'Üye Giriş'. It contains three input fields: 'E-Mail', 'Şifre', and a 'Giriş Yap' button. Below the button is a small link that says 'Hemen Kayıt Ol'.

Şekil 10: Üye Giriş Arayüzü

```

1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
4 <!DOCTYPE html>
5 <html lang="tr">
6 <head>
7     <meta charset="UTF-8">
8     <meta http-equiv="X-UA-Compatible" content="IE=edge">
9     <meta name="viewport" content="width=device-width, initial-scale=1.0"
10     >
11     <meta name="description" content="Kitap Satış Sitesi">
12     <meta name="keywords" content="kitap ,okumak ,eğitim ,roman ,edebiyat">
13     <meta name="author" content="Rabia ÖZCAN, Yahya Furkan KILI O LU ,
14         Muhammed Abdurrahman ÖTÜN, Hüseyin MUTLU">
15     <!--CSS bağımlilikleri-->
16     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/
17         bootstrap.min.css" rel="stylesheet" >
18     <!--Javascript Bağımlilikleri-->

```

```
16     <script> <%@ include file = "static/js/fontawesome-icons.js" %></
      script>
17     <title>YaziKolik </title>
18     <style type="text/css">
19     <%@ include file = "static/css/style.css" %>
20     </style>
21 </head>
22 <body>
23
24     <div class="container border p-3 my-3 text-center w-50" id="
      loginform">
25         <form action="login" method="post" >
26             <div class="mb-3 bg-secondary py-2 rounded-3" style="color:
              white;">
27                 <h1>Üye Girişi </h1>
28
29             </div>
30             <div class="mb-3">
31                 <hr>
32                 <label for="email" class="form-label">E-Mail </label>
33                 <input type="email" class="form-control" id="email" name=
              "email" required>
34             </div>
35             <div class="mb-3">
36                 <label for="sifre" class="form-label">Şifre </label>
37                 <input type="password" class="form-control" id="sifre"
              name="password" required>
38             </div>
39             <div class="mb-3">
```

```
40         <button type="submit" class="btn btn-outline-success w
           -100">Giriş Yap</button>
41     </div>
42     <div class="mb-3">
43         <a href="" style="text-decoration: none; color: teal" class
           ="float-start mt-2"><small>Hemen Kayıt Ol</small></a>
44         <div class="clearfix"></div>
45     </div>
46 </form>
47 </div>
48 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/js/
           bootstrap.bundle.min.js"></script>
49 </body>
50 </html>
```

Üyelerin sisteme giriş yapabilmesi için post metodu ile bir form tanımlanmıştır. Bu form aldığı bilgileri kontrol edip, cevap döndermesi için servlete gönderir.

10.4 ÖDEME İŞLEMLERİ

Müşteriler sepetteki ürünlerini satın almak istediklerinde Şekil 11 ile karşılaşır. Kredi kartı bilgilerini girip, ödeme yap butonuna bastığında müşterinin kredi kartı bilgileri web servise gider, web serviste kart bilgileri kontrol edilir. Eğer bütün bilgiler doğru ve limit yeterli ise web servis cevabı servlete gönderir. Servlet aldığı cevaba göre kullanıcıya sipariş numrasını gösterir ya da kredi kartı bilgilerinde hata olduğuna dair bir mesaj görüntülenir.

Şekil 11: Ödeme Ekranı Arayüzü

```

1 <!--Modal Yapısı Başlangıcı-->
2 <div class="modal fade " id="exampleModal" tabindex="-1" aria-labelledby=
  "exampleModalLabel" aria-hidden="true">

```

```
3 <div class="modal-dialog">
4   <div class="modal-content">
5     <div class="modal-header">
6       <h5 class="modal-title" id="exampleModalLabel">Kredi Kartı
          Bilgileri </h5>
7       <button type="button" class="btn-close" data-bs-dismiss="modal"
          aria-label="Close"></button>
8     </div>
9     <div class="modal-body">
10      <form action="" method="POST" id="payment-form">
11        <div class="mb-3">
12          <label for="credit-card-owner" class="form-label">Kart
              Sahibinin Adı ve Soyadı </label>
13          <input type="text" class="form-control" id="credit-card-owner"
              placeholder="Ad Soyad" name="ownerName">
14        </div>
15        <div class="mb-3">
16          <label for="credit-card" class="form-label">Kredi Kartı
              Numarası </label>
17          <input type="text" class="form-control" id="credit-card"
              placeholder="Kart Numarası" maxlength="19" name="
              cardNumber">
18        </div>
19        <div class="row row-cols-2 mb-3">
20          <div class="col">
21            <label for="tarih" class="form-label">Ay ve Yıl </label>
22            <input type="text" class="form-control me-3" maxlength="5"
              placeholder="AA/YY" id="tarih" name="cardDate">
23          </div>
```

```

24     <div class="col">
25         <label for="cvv" class="form-label">CVV Kodu</label>
26         <input type="text" class="form-control" maxlength="3"
           placeholder="CVV" id="cvv" name="cardCvv">
27     </div>
28 </div>
29 </div>
30 <div class="py-3 px-4 border-top">
31     <div class="float-start">50,65</div>
32     <div><button type="submit" class="btn btn-outline-success float-
           end" id="doPay">Ödeme Yap</button></div>
33 </form>
34 </div>
35 </div>
36 </div>
37 </div>
38 <!--Modal Bitişi-->

```

Bu kodlar sepet kodlarının içerisinde yer almaktadır. Kredi kartı bilgilerini form ile alıp post metodu ile web servise gönderir. İlgili jquery kodu ise :

```

1 <script>
2 $( "#doPay" ). click ( function () {
3     $. ajax ( {
4         url : ' ',
5         type : "POST" ,
6         data : $( "#payment-form" ). serialize () ,
7         success : function ( result ) {
8             alert ( result );
9         },

```

```
10         error: function(result) {
11             alert(result);}
12         });
13     });
14
15     $("#credit-card,#cvv").keypress(function (e) {
16         if ((e.which < 48 || e.which > 57) && (e.which !== 8) && (e.
17             which !== 0)) {
18             return false;
19         }
20         if (this.value.length === 4 || this.value.length === 9 || this.
21             value.length === 14) {
22             this.value = this.value += ' ';
23         }
24         return true;
25     });
26
27     $("#tarih").keypress(function (e) {
28         if(this.selectionStart==0 && e.which > 49 ){
29             return false;
30         }
31         if ((e.which < 48 || e.which > 57) && (e.which !== 8) && (e.
32             which !== 0)) {
33             return false;
34         }
35         if (this.value.length === 2) {
36             this.value = this.value += '/';
37         }
```

```
36         return true;  
37     });
```

10.5 YÖNETİCİ PANELİ

Admin için yönetim paneli Şekil 12’de yer almaktadır.



Tüm Siparisler

Fatura No	Siparis veren	Siparis Tarihi	Tutar
5101	muhammed	2021-06-20]	48.0
5101	Ktap Adi	Adet	Toplam Tutar
	Balıkçı ve Oğlu	1	19.0
	Ktap Adi	Adet	Toplam Tutar
	Veba Geceleri	1	29.0
5102	muhammed	2021-06-20]	47.0
5102	Ktap Adi	Adet	Toplam Tutar
	Seyir	1	28.0
	Ktap Adi	Adet	Toplam Tutar
	İyi Aile Yoktur	1	19.0
5201	muhammed	2021-06-20]	56.0
5201	Ktap Adi	Adet	Toplam Tutar
	Veba Geceleri	1	29.0
	Ktap Adi	Adet	Toplam Tutar
	Söylenmemiş Sözler	1	27.0

Kategoriler

- Edebiyat
- Eğitim
- Felsefe
- Bilim
- Yazılım


İletişim

Email: furkan.kilicoglu@bil.omu.edu.tr

Email: abdurrahman.otun@bil.omu.edu.tr

Email: rabia.ozcan@bil.omu.edu.tr

Email: huseyin.mutlu@bil.omu.edu.tr



Daha büyük haritayı görüntüle


Adres: Atakum/Samsun

© Copyright 2021 - Bu Ödevin Tüm Hakları Saklıdır.

Şekil 12: Admin İçin Yönetim Paneli

WEB SERVİS

Kullanıcı kredi kartı bilgilerini doldurup, siparişini tamamladıktan sonra karşılaştığı arayüz Şekil 13’de yer almaktadır. Kullanıcı siparişini tamamlayabilmesi için sisteme giriş yapmış olmalıdır. Sipariş tamamlandıktan sonra sipariş numarası, tutar ve tarihi görüntülenmektedir. Daha sonra kullanıcı ister ise logoya tıklayarak ana sayfaya dönebilmektedir.



Ara

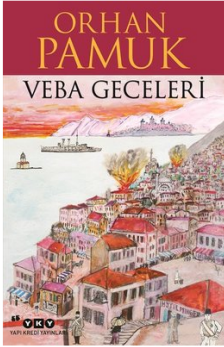
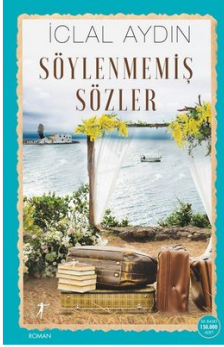
Çıkış Yap

Sepet

Sipariş No: 5201

Sipariş Tutarı: 56.0

Sipariş Tarihi: Sun Jun 20 19:54:51 TRT 2021

	Veba Geceleri	1	29.0
	Söylenmemiş Sözler	1	27.0

Kategoriler

- Edebiyat
- Eğitim
- Felsefe
- Bilim
- Yazılım


İletişim

Email: furkan.kilicoglu@bil.omu.edu.tr

Email: abdurrahman.otun@bil.omu.edu.tr

Email: rabia.ozcan@bil.omu.edu.tr

Email: huseyin.mutlu@bil.omu.edu.tr



Daha büyük haritayı görüntüle

Kıyeye kısayolları Harita verileri ©2021 Kullanım Şartları Harita hatası bildirin

Adres: Atakum/Samsun

© Copyright 2021 - Bu Ödevin Tüm Hakları Saklıdır.

Şekil 13: Sipariş Tamamlandıktan Sonra Kullanıcı Arayüzü

11.1 *Card.java*

```
1
2
3 public class Card {
4     private String cardOwnerName;
5     private String cardNumber;
6     private String cardMonth;
7     private String cardYear;
8     private String cardCvv;
9     public Card(String cardOwnerName, String cardNumber, String cardMonth,
10                String cardYear, String cardCvv) {
11         this.cardOwnerName = cardOwnerName;
12         this.cardNumber = cardNumber;
13         this.cardMonth = cardMonth;
14         this.cardYear = cardYear;
15         this.cardCvv = cardCvv;
16     }
17     public String getCardOwnerName() {
18         return cardOwnerName;
19     }
20     public void setCardOwnerName(String cardOwnerName) {
21         this.cardOwnerName = cardOwnerName;
22     }
23     public String getCardNumber() {
24         return cardNumber;
25     }
26     public void setCardNumber(String cardNumber) {
27         this.cardNumber = cardNumber;
```

```
27     }  
28     public String getCardMonth() {  
29         return cardMonth;  
30     }  
31     public void setCardMonth(String cardMonth) {  
32         this.cardMonth = cardMonth;  
33     }  
34     public String getCardYear() {  
35         return cardYear;  
36     }  
37     public void setCardYear(String cardYear) {  
38         this.cardYear = cardYear;  
39     }  
40     public String getCardCvv() {  
41         return cardCvv;  
42     }  
43     public void setCardCvv(String cardCvv) {  
44         this.cardCvv = cardCvv;  
45     }  
46 }
```

Bu sınıfta kart nesnesinin sahip olacağı özellikler, ve getter setter metotları tanımlanmıştır.

11.2 CardProcess.java

```
1 package com.service.web;  
2  
3 import java.util.*;
```

```
4 public class CardProcess {
5     Card card1 = new Card("RABİA ÖZCAN", "1234 5678 1234 5678", "03", "22", "
        333");
6     Card card2 = new Card("YAHYA FURKAN KILICOGLU", "4213 5419 4571 1624",
        "06", "26", "457");
7     Card card1 = new Card("MUHAMMED OTUN", "1324 4219 7315 9462", "04", "28"
        , "164");
8     Card card1 = new Card("HUSEYİN MUTLU", "1249 4875 1674 7461", "02", "27"
        , "421");
9     private List<Card> cardList = new ArrayList<>();
10
11     public void addItemList(Card card) {
12         cardList.add(card);
13     }
14     public boolean cardCompare(Card user) {
15         addItemList(card1);
16         addItemList(card2);
17         addItemList(card3);
18         addItemList(card4);
19         for(Card card : cardList) {
20             if(card.getCardOwnerName().equals(user.getCardOwnerName()) &&
21                 card.getCardNumber().equals(user.getCardNumber()) && card.
22                     getCardMonth().equals(user.getCardMonth()) &&
23                     card.getCardYear().equals(user.getCardYear()) && card.getCardCvv
24                         ().equals(user.getCardCvv())) {
25                 return true;
26             }
27         }
28     }
29     return false;
```



```
13         if(compare.cardCompare(user) == true) {
14             // veri tabanına siparişi kaydet sipariş no oluştur ve geri dönd
                ür.
15             String message = "Ödeme Başarılı Şekilde Gerçekleşti";
16             return message;
17         }
18         String message = "Ödeme Başarısız";
19         return message;
20
21     }
22 }
```

MyResource sınıfı incelediğimizde @PATH ile sınıfın çalışacağı adres belirlenmiştir. @PATH kullanımı @path ifadesi sınıflar için kullanabileceği gibi metotlar içinde kullanılabilir. @Produces ile çıktının türünü belirledik. @FormParam Kullanıcıdan html form parametrelerini almamıza yaramaktadır. CardProcess sınıfından compare adlı nesne türettik. Monthandyear string dizisi oluşturur ve cart tarihindeki / işaretini kaldırır. Card sınıfından user üretir ve card bilgilerini yapıcı metot sayesinde doldurur. Eğer compare nesnesi içerisindeki cardCompare sınıfındaki kullanıcı verirken kart bilgileriyle doğruysa ödeme başarılı mesajını geri döndürür. Eğer değilse başarısız bilgisini geri döndürür. Bunların hepsi yapıcı bir metot içerisinde kullanılmıştır.

ÜRÜNLERİN YER ALDIĞI VERİTABANI KODLARI VE AÇIKLAMALARI

Bu sınıflar içerisinde sorgularımızı static final string olarak tanımlamaktayız. Repository ve RepositoryImpl paketleri bu paketler içerisindeki sınıflar ile interfacer oluşturulmuştur. Bu interfacer üzerinden implements yapılmıştır. Implements sınıflarında veri tabanı CRUD işlemleri gerçekleştirilmektedir. JpaFactory ve JpaFactoryImpl sınıfları ile jpa içerisinde bulunan entityManager ve entitiyTransection sınıflarını implement ediyoruz.

SERVLETLAR

Web sayfaları eskiden durağan bir yapıya sahipti, kullanıcıdan bilgi alarak bu bilgiler doğrultusunda işlemler yapamıyordu. Zamanla dinamik web sayfasına duyulan istekler artınca kullanıcıdan bilgiler alıp çalışan dinamik web sayfaları üretilmeye başlandı.

Java programlama dili ile dinamik web sayfaları üretmek için bir çok yol vardır, bunlardan biri olan Servlet teknolojisini inceleyeceğiz.

Servlet Nasıl Çalışır ?

Servlet uygulamaları web sunucu (Server) üzerinde çalışır. Kullanıcıdan aldığı verilere göre sonuçlar üreten Java sınıflarıdır. Bu veriler üzerinde gereken işlemler yapıldıktan sonra önceden tasarlanmış biçimlere uyarlanarak kullanıcıya geri döndürülür.

Aşamalar halinde söylemek istersek Servlet:

Kullanıcıdan Server'a istek (request) gönderilir. Server'da bu isteğe ait servlet belirlenir. istek ve tüm bilgi bu servlet'e gönderilir. Servlet bu gelen bilgileri alır ve oluşturulması gereken sonuçları üretir. Genellikle bu sonuçlar HTML sayfası şeklindedir. Servlet oluşturduğu sonucu web sunucusuna gönderir. Web sunucusu Servlet'ten aldığı sonucu (response), isteği yapan kullanıcıya gönderir. Hizmet yöntemi, istemciden gelen istek türüne göre doGet veya doPost'u çağırır, eğer alma isteği gelirse doPost çağırılırsa doGet çağırılır. Sunucu bir sonucu

uygulaması için her istek aldığı anda, sunucu yeni bir iş parçacığı oluşturur ve hizmeti çağırır.

Service () yöntemi, HTTP istek türünü (GET, POST, PUT, DELETE, vb.) Kontrol eder ve uygun şekilde doGet, doPost, doPut, doDelete, vb. Yöntemlerini çağırır.

13.1 AdminServlet

```
1 package com.yazikolik.servlets;
2
3 import java.io.IOException;
4
5 import javax.servlet.RequestDispatcher;
6 import javax.servlet.ServletException;
7 import javax.servlet.annotation.WebServlet;
8 import javax.servlet.http.HttpServlet;
9 import javax.servlet.http.HttpServletRequest;
10 import javax.servlet.http.HttpServletResponse;
11 import com.yazikolik.model.*;
12 import java.util.ArrayList;
13 import com.yazikolik.repository.OrderDetailRepository;
14 import com.yazikolik.repository.OrderRepository;
15 import com.yazikolik.repository.impl.OrderDetailRepositoryImpl;
16 import com.yazikolik.repository.impl.OrderRepositoryImpl;
17
18 /**
19  * Servlet implementation class AdminServlet
20  */
21 @WebServlet("/AdminServlet")
22 public class AdminServlet extends HttpServlet {
```

```

23     private static final long serialVersionUID = 1L;
24
25     /**
26      * @see HttpServlet#HttpServlet()
27      */
28     public AdminServlet() {
29         super();
30         // TODO Auto-generated constructor stub
31     }
32
33     /**
34      * @see HttpServlet#doGet(HttpServletRequest request,
35       *                          HttpServletResponse response)
36      */
37     protected void doGet(HttpServletRequest request, HttpServletResponse
38                          response) throws ServletException, IOException {
39         // TODO Auto-generated method stub
40         RequestDispatcher rd = request.getRequestDispatcher("adminPanel.jsp
41                                                                ");
42         OrderRepository orderRepository = new OrderRepositoryImpl();
43         OrderDetailRepository orderDetailRepository = new
44             OrderDetailRepositoryImpl();
45         ArrayList<Order_> orders = orderRepository.getAllOrders();
46         ArrayList<OrderAndOrderDetail> orderAndOrderDetails = new ArrayList
47             <OrderAndOrderDetail>();
48         for (Order_ order_ : orders) {
49             ArrayList<OrderDetail> orderDetailList = orderDetailRepository.
50                 findByOrder(order_);

```

```

45         orderAndOrderDetails.add(new OrderAndOrderDetail(order_ ,
                                orderDetailList));
46     }
47
48     request.setAttribute("orderAndOrderDetails", orderAndOrderDetails);
49
50     rd.forward(request , response);
51 }
52
53 /**
54  * @see HttpServlet#doPost(HttpServletRequest request ,
55     HttpServletResponse response)
56  */
57 protected void doPost(HttpServletRequest request , HttpServletResponse
58     response) throws ServletException , IOException {
59     // TODO Auto-generated method stub
60     doGet(request , response);
61 }

```

13.2 BasketServlet

```

1 package com.yazikolik.servlets;
2
3 import java.io.IOException;
4 import java.util.ArrayList;
5

```

```

6 import java.util.Date;
7 import javax.servlet.RequestDispatcher;
8 import javax.servlet.ServletException;
9 import javax.servlet.annotation.WebServlet;
10 import javax.servlet.http.HttpServlet;
11 import javax.servlet.http.HttpServletRequest;
12 import javax.servlet.http.HttpServletResponse;
13 import javax.servlet.http.HttpSession;
14
15 import com.yazikolik.model.Basket;
16 import com.yazikolik.model.Order_;
17 import com.yazikolik.model.User;
18 import com.yazikolik.model.Product;
19 import com.yazikolik.model.OrderDetail;
20 import com.yazikolik.repository.OrderDetailRepository;
21 import com.yazikolik.repository.OrderRepository;
22 import com.yazikolik.repository.ProductRepository;
23 import com.yazikolik.repository.impl.OrderDetailRepositoryImpl;
24 import com.yazikolik.repository.impl.OrderRepositoryImpl;
25 import com.yazikolik.repository.impl.ProductRepositoryImpl;
26
27 /**
28  * Servlet implementation class BasketServlet
29  */
30 @WebServlet("/BasketServlet")
31 public class BasketServlet extends HttpServlet {
32     private static final long serialVersionUID = 1L;
33     ProductRepository productRepository = new ProductRepositoryImpl();
34     OrderRepository orderRepository = new OrderRepositoryImpl();

```

```

35 OrderDetailRepository orderDetailRepository = new
    OrderDetailRepositoryImpl();
36 ArrayList<Basket> basket = new ArrayList<Basket>();
37 /**
38  * @see HttpServlet#HttpServlet()
39  */
40 public BasketServlet() {
41     super();
42     // TODO Auto-generated constructor stub
43 }
44
45 /**
46  * @see HttpServlet#doGet(HttpServletRequest request,
    HttpServletResponse response)
47  */
48 protected void doGet(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {
49     // TODO Auto-generated method stub
50     HttpSession session = request.getSession();
51     ArrayList<Basket> basket = (ArrayList<Basket>)session.getAttribute(
        "basket");
52     if (request.getParameter("value") != null) {
53         String stringValue = request.getParameter("value");
54         int intValue = Integer.valueOf(stringValue);
55         for (Basket item : basket) {
56             if (item.getProduct().getProductID() == intValue) {
57                 basket.remove(item);
58                 break;
59             }

```



```

60         }

61

62     }

63     request.setAttribute("basketList", basket);

64     RequestDispatcher rd = request.getRequestDispatcher("basket.jsp");

65     rd.forward(request, response);

66 }

67

68 /**

69  * @see HttpServlet#doPost(HttpServletRequest request,

        HttpServletResponse response)

70  */

71 protected void doPost(HttpServletRequest request, HttpServletResponse

        response) throws ServletException, IOException {

72     // TODO Auto-generated method stub

73     HttpSession session = request.getSession();

74     ArrayList<Basket> basket = (ArrayList<Basket>)session.getAttribute(

        "basket");

75     User user = (User)session.getAttribute("user");

76     if(user == null) {

77         response.sendRedirect(request.getContextPath() + "/login");

78     } else if ((String)request.getParameter("ownerName") != "") {

79         float totalAmount=0;

80         Order_ order = new Order_(new Date(), user, totalAmount);

81         ArrayList<OrderDetail> orderDetails = new ArrayList<OrderDetail>

            >();

82         for (Basket item : basket) {

83             orderDetails.add(new OrderDetail(order, item.getProduct(),

                item.getQuantity()));

```

```

84         totalAmount+=item.getProduct().getPrice()*item.getQuantity();
85     }
86     order.setPaymentAmount(totalAmount);
87     orderRepository.save(order);
88     order = orderRepository.lastOrder();
89     for (OrderDetail orderDetail : orderDetails) {
90         orderDetailRepository.save(orderDetail);
91     }
92
93     RequestDispatcher rd = request.getRequestDispatcher("
           OrderSuccess.jsp");
94     request.setAttribute("order", order);
95     request.setAttribute("orderDetails", orderDetails);
96     session.setAttribute("basket", null);
97     rd.forward(request, response);
98 } else {
99     request.setAttribute("error", "KartBilgileri Yanlis.");
100    RequestDispatcher rd = request.getRequestDispatcher("
           OrderSuccess.jsp");
101    rd.forward(request, response);
102 }
103
104    // baketlisti don ve icerigini orderdetails nesneleri icerisine
           yerlestir siparis numarisi goster
105 }
106
107 }

```

13.3 *MainServlet*

```
1 package com.yazikolik.servlets;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5 import java.util.ArrayList;
6
7 import javax.servlet.RequestDispatcher;
8 import javax.servlet.ServletException;
9 import javax.servlet.annotation.WebServlet;
10 import javax.servlet.http.HttpServlet;
11 import javax.servlet.http.HttpServletRequest;
12 import javax.servlet.http.HttpServletResponse;
13 import javax.servlet.http.HttpSession;
14
15 import com.yazikolik.model.Product;
16 import com.yazikolik.model.Basket;
17 import com.yazikolik.repository.ProductRepository;
18 import com.yazikolik.repository.impl.ProductRepositoryImpl;
19
20 /**
21  * Servlet implementation class MainServlet
22  */
23 @WebServlet("/MainServlet")
24 public class MainServlet extends HttpServlet {
25     private static final long serialVersionUID = 1L;
26     ProductRepository productRepository = new ProductRepositoryImpl();
27     ArrayList<Basket> basket = new ArrayList<Basket>();
```

```

28
29  /**
30   * @see HttpServlet#HttpServlet()
31   */
32  public MainServlet() {
33      super();
34      // TODO Auto-generated constructor stub
35  }
36
37  /**
38   * @see HttpServlet#doGet(HttpServletRequest request,
39      HttpServletResponse response)
40   */
41  protected void doGet(HttpServletRequest request, HttpServletResponse
42      response) throws ServletException, IOException {
43      // TODO Auto-generated method stub
44      HttpSession session = request.getSession();
45      if (session.getAttribute("basket") != null) {
46          ArrayList<Basket> basket = (ArrayList<Basket>)session.
47              getAttribute("basket");
48          session.setAttribute("basket", basket);
49      } else {
50          session.setAttribute("basket", basket);
51      }
52      ArrayList<Product> productList = productRepository.getAllProducts()
53          ;
54      request.setAttribute("productList", productList);
55      request.setAttribute("basket", basket);
56      RequestDispatcher rd = request.getRequestDispatcher("index.jsp");

```

```

53         rd.forward(request, response);
54     }
55
56     /**
57     * @see HttpServlet#doPost(HttpServletRequest request,
58         HttpServletResponse response)
59     */
59     protected void doPost(HttpServletRequest request, HttpServletResponse
60         response) throws ServletException, IOException {
61         // TODO Auto-generated method stub
62         response.setContentType("text/html");
63         PrintWriter out = response.getWriter();
64         if(request.getParameter("action").equals("search")) {
65             String searchInput = (String)request.getParameter("searchInput")
66                 ;
67             ArrayList<Product> productlar = productRepository.findByTitle(
68                 searchInput);
69             out.write("<div class=\"container my-2\">\n"
70                 + "            <div class=\"row row-cols-2 row-cols-md-3 row-
71                 cols-lg-4\">");
72             for (Product product : productlar) {
73                 out.write("<div class=\"col\">\n"
74                 + "                    <div class=\"card text-center mb-4 shadow-
75                 items py-2\">\n"
76                 + "                        <div class=\"pb-3\">");
77                 out.write(" <img src=\"" + product.getImageUrl() + " \">");
78                 out.write("</div>\n"
79                 + "                            <div class=\"border-top border-bottom
80                 pt-1 mb-2\">");

```

```

75     out.write("<p class=\"product-attr\">"+product.getTitle()+"</p>"
76         );
77     out.write("<p class=\"product-attr\">"+product.getAuthor()+"</p>"
78         );
79     out.write("<p class=\"product-attr\">"+product.getPublisher()+"
80         </p>");
81     out.write("<p class=\"product-attr\">"+product.getPrice()+"</p>"
82         );
83     out.write("</div>");
84     out.write("    <div>\n"
85         + "        <button class=\"me-3 btn btn-
86             outline-success cartAdd\" type=\"submit\"><i class=\"
87                 fas fa-shopping-cart pe-2\"></i>Ekle</button>\n"
88         + "        <button class=\"btn rounded-pill
89             btn-sm btn-outline-danger btn-decrease\" type=\"button
90                 \"><i class=\"fas fa-minus\"></i></button>\n"
91         + "        <input type=\"number\" value=\"1\"
92             style=\"width:25px;border:none;padding-left:6px;\"
93             class=\"mx-1\">\n"
94         + "        <button class=\"btn rounded-pill
95             btn-sm btn-outline-success btn-increase\" type=\"
96                 button\"><i class=\"fas fa-plus\"></i></button>\n"
97         + "    </div>\n"
98         + "    </div>\n"
99         + "    </div>");
100 }
101 out.write("    </div>\n"
102     + "    </div>\n"
103     + "\n"

```

```

92         + "        <div class=\"fixed-top top-0 end-0\">\n"
93         + "        <div class=\"toast position-absolute top-0 end-0 align
          -items-center text-white bg-success\" role=\"alert\" aria
          -live=\"assertive\" aria-atomic=\"true\" data-bs-delay
          =\"1500\">\n"
94         + "            <div class=\"toast-header\">\n"
95         + "                <i class=\"fas fa-check pe-3\"></i>\n"
96         + "                <strong class=\"me-auto\">Urun sepete eklendi </
          strong>\n"
97         + "                <button type=\"button\" class=\"btn-close\" data
          -bs-dismiss=\"toast\" aria-label=\"Close\"></button>\n"
98         + "            </div>\n"
99         + "        </div>\n"
100        + "    </div>");
101    }
102    else if (request.getParameter("action").equals("addBasket")) {
103        HttpSession session = request.getSession();
104        Product product = productRepository.findProductByTitle((String)
          request.getParameter("productTitle"));
105        String stringQuantity = request.getParameter("productQuantity");
106        int intQuantity = Integer.valueOf(stringQuantity);
107        ArrayList<Basket> basket = (ArrayList<Basket>)session.
          getAttribute("basket");
108        basket.add(new Basket(product, intQuantity));
109        session.setAttribute("basket", basket);
110        String basketCounter = String.valueOf(basket.size());
111        out.write(basketCounter);
112
113    }

```

114

115 }

116 }

BÖLÜM: V

SONUÇ

SONUÇLAR

Bu çalışmada kitap satışı yapılan elektronik ticaret sitelerine alternatif bir sistem geliştirilmesi hedeflenmiştir. Bu doğrultuda kullanılacak teknolojiler ve mimari seçildikten sonra projenin back-end ve front-end kısmı gerçekleştirildi. Java programlama dili, maven, hibernate, jersey web servis kullanılarak proje dinamik bir şekilde çalışır hale getirildi. Diğer alternatiflerinden farklı olarak karmaşık yapıdan uzak, sade ve göz yormayan bir arayüz gerçekleştirildi. Proje gerçekleştirirken tasarım kısmında Bootstrap kütüphanesinde epeyce faydalanıldı. Gerekli yerlerde jsp, html ve css kodları ile iyileştirmeler sağlandı. Fonksiyonel yapıda servlet kodlarında, arayüz ile iletişim kurmak için JSP kullanıldı. Sistem Apache Tomcat Server üzerinde test edildi ve çalışırılığı doğrulandı.

Kodların tamamı raporda yoktur. Githubdan çekebilirsiniz. Kodlar için temel anlamda anlatım yapılmıştır.

Görev dağılımı aramızda karışık şekilde gerçekleşti. İlk başta belirlediğimiz görevlerin tamamen dışına çıkıp, bir birimize yardım ederken, bir birimizin görevlerini de yapmış bulunduk. Bu sebeple net bir görev dağılımı ortaya çıkmadı.