



**KARABÜK ÜNİVERSİTESİ**  
**MÜHENDİSLİK FAKÜLTESİ**  
**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

# **OTOPARK REZERVASYON OTOMASYONU**

**Hazırlayanlar**

Gizem TÜRKÖĞLU

Behiye Kübra YILMAZ

**LİSANS TEZİ**

**Tez Danışmanı**

**Dr.Öğr.Üyesi Ümit ATİLA**

**KARABÜK**

**2020**

*“Bu tezdeki tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu; ayrıca bu kuralların ve ilkelerin gerektirdiği şekilde, bu çalışmadan kaynaklanmayan bütün atıfları yaptığımı beyan ederim.”*

Gizem T RK  LU

Behiye K bra YILMAZ

Gizem TRKLU ve Behiye Kbra YILMAZ tarafından hazırlanan “OTOPARK REZERVASYON OTOMASYONU” başlıklı bu projenin Bitirme Projesi Tezi olarak uygun olduğunu onaylarım.

Dr.r.yesi mit Atila

.....

Bitime Projesi Danışmanı, Bilgisayar Mhendislięi Anabilim Dalı

...../...../2020

Bilgisayar Mhendislięi blm , bu tez ile, Bitirme Projesi Tezini onamıştır

Doç. Dr. İlker TRKER

.....

Blm Başkanı

## **ÖNSÖZ**

Bu çalışmanın yürütülmesi sırasında anlayışını ve desteğini esirgemeyen, öğrencisi olmaktan onur duyduğumuz danışmanımız Dr.Öğr.Üyesi Ümit ATİLA'ya, bu süreçte maddi ve manevi desteklerini hiç esirgemeyen, tercihlerimizde yanımızda olan TÜRKOĞLU ve YILMAZ ailelerine sonsuz teşekkürlerimizi sunuyoruz.

## ÖZET

Teknolojinin hızla gelişmesi beraberinde birçok kolaylık ve yenilik getirmiştir. Bu sayede teknoloji hayatımızın her alanında ihtiyaç duyduğumuz vazgeçilmez bir parça olmuştur. Bu yeniliklerden biri de günlük işlerimizi web sitesi üzerinden online olarak halledebilme kolaylığı sağlamaktır[1]. Günümüzde online olarak yürütemeyeceğimiz iş yok denecek kadar az hale gelmiştir. Gerçekleştirilen bu projede Symfony Framework web sitesinde MVC yapısının sağladığı kolaylık ile açık veya kapalı otoparklarda kullanılabilen bir otopark rezervasyon sistemi tasarlanmıştır. Web sitesine girildiği zaman kullanıcıyı bir üye giriş sayfası karşılamaktadır. Sisteme üye olan her kullanıcının bilgileri alınarak bir profil sayfası oluşturulmaktadır. Bu profilde sisteme üye olan kullanıcının yapmış olduğu rezervasyonları görebilmesi ve istediği zaman rezervasyonunu iptal edebilmesi sağlanmıştır. Projede boş olan park yerleri yeşil, dolu olan park yerleri ise kırmızı renk ile gösterilmiştir. Sisteme üye olan kullanıcı boş olan yerlerden birini seçerek istediği zaman aralığını belirterek rezervasyon işlemini yapabilmektedir. Oluşturulan bu web sitesi sayesinde; park yeri bulmak için kullanıcı zamanından tasarruf sağlar, sistem park alanlarının bir görünümünü sağlar.

***Anahtar Sözcükler:** Symfony Framework, Website*

## **ABSTRACT**

The rapid development of technology has brought with it many conveniences and innovations. In this way, technology has become an indispensable part that we need in every aspect of our lives. One of these innovations provides the convenience to handle our daily business online via the website. Nowadays, the work that we cannot carry out online has become almost non-existent. In this project, a parking reservation system that can be used in open or closed car parks is designed with the convenience of the MVC structure on the Symfony Framework website. When enter the website, a member login page is opened for the user. A profile page is created by obtaining the information of each user who is a member of the system. In this profile, it is provided that the user who is a member of the system can see the reservations made and the user can cancel the reservation at any time. The empty parking spaces in the project are shown in green and the full parking spaces are shown in red. The user who is a member of the system can choose one of these empty places places and the user can make a reservation by specifying the time interval user want. Due to this website, it saves the user time to find a parking space, the system provides a view of the parking spaces.

***Key Words:*** *Symfony Framework, Website*

## İÇİNDEKİLER

### Sayfa

KABUL.....	ii
ÖNSÖZ.....	iii
ÖZET.....	iv
ABSTRACT.....	v
İÇİNDEKİLER.....	vi
ŞEKİLLER DİZİNİ.....	viii

## I. BÖLÜM

1.GİRİŞ.....	9
1.1.LİTERATÜR ÖZETİ.....	10

## II. BÖLÜM

2. MVC YAPISI.....	11
2.1 MVC Yapısı.....	12
2.2 MVC Çalışma Prensibi.....	12
2.3 MVC Kullanımının Avantajları.....	12
2.4 Symfony Framework Yapısı.....	12

## III. BÖLÜM

3. KULLANICI PANELİ.....	14
3.1 Üye Kayıt ve Giriş İşlemleri.....	14
3.1.1 Üye Kayıt.....	14
3.1.2 Üye Giriş.....	15
3.2 Rezervasyon İşlemleri.....	16
3.2.1 Rezervasyon Sayfası.....	16
3.2.2 Rezervasyon Onay.....	17
3.3 Kullanıcı Profili.....	20
3.4 İletişim Sayfası.....	21

#### **IV. BÖLÜM**

4. ADMIN PANELİ.....	22
----------------------	----

#### **V.BÖLÜM**

5.SONUÇ VE DEĞERLENDİRME.....	26
-------------------------------	----

<b>KAYNAKLAR.....</b>	<b>27</b>
-----------------------	-----------

<b>ÖZGEÇMİŞLER.....</b>	<b>28</b>
-------------------------	-----------



## ŞEKİLLER DİZİNİ

### Sayfa

Şekil 2.1 MVC Yapısı.....	11
Şekil 3.1 Üye Kayıt Sayfası.....	14
Şekil 3.2 Registration Controller.....	15
Şekil 3.3 Giriş(Ana) Sayfası.....	16
Şekil 3.4 Rezervasyon Sayfası.....	17
Şekil 3.5 Rezervasyon Onay.....	18
Şekil 3.6 Reservation Controller.....	19
Şekil 3.7 Reservation Controller Devamı.....	20
Şekil 3.8 Message Form .....	21
Şekil 4.1 Message Controller.....	23
Şekil 4.2 Park Controller.....	24
Şekil 4.3 Level Controller.....	25

## **I. BÖLÜM**

### **1.GİRİŞ**

Günümüzde özellikle büyükşehirlerde başta olmak üzere sayıları hızla artan büyük alışveriş merkezleri, terminaller ve havaalanları gün geçtikçe daha fazla tüketici ve yolcu yoğunluğu yaşamaktadır. Bu sebeple karşılaşılan en büyük sorunlardan biri otopark alanlarında oluşan yoğun araç trafiğidir. Hem otopark alanlarının büyük olması hem de fazla araç yoğunluğu sebebi ile insanların uzun zaman boş park yeri ararken gereksiz yakıt harcamalarına ve vakit kaybetmelerine yol açmaktadır. Bunun yanı sıra park yeri arama sırasında yaşanabilecek her türlü sinir ve stres, park yeri arayan insanları olumsuz etkilemektedir.

Bu projenin amacı açık veya kapalı otoparklarda zaman kaybı olmadan araçların yer bulabilmesidir. Projede herhangi bir otopark için Symphony Framework kullanılarak tasarlanan otopark rezervasyon sistemi web sitesi örneği yapılmıştır.

Tasarlanmış olan bu web projesi ile birlikte; otopark, alışveriş merkezleri gibi park ihtiyacının fazla olduğu yerlerde araç sahiplerinin, bu alanlar içerisinde önceden kendine uygun olan park yerini seçip rezervasyonunu yaparak park yeri aranarak harcanan zaman kaybının azaltılması hedeflenmiştir. Aynı zamanda araçların rezervasyon süresi sona erdiğinde yada araç sahibi tarafından rezervasyon iptal edildiğinde, dolu olarak görünen park yerinin tekrar boş olarak gösterilmesi sağlanarak başka bir araç sahibinin aynı yeri rezerve edebilmesine olanak sağlanmıştır.

## 2. LİTERATÜR ÖZETİ

Yusnita Rahayu and Fariza N. Mustapa tarafından hazırlanan “A Secure Parking Reservation System Using GSM Technology” isimli çalışmada veritabanından otopark durum kontrolü ve otopark giriş çıkışlarına şifre sistemi koyarak güvenlik odaklı yaklaşımları konu almışlardır.

Yoğun araç trafiğinden kaynaklanan gürültü ve hava kirliliğini üzerinde rezervasyon sisteminin etkileri ile ilgili de çalışmalar yapılmıştır. Mei-Ting TSAI ve Chih-Peng CHU tarafından yapılan “Evaluating Parking Reservation Policy in Urban Areas: An Environmental Perspective” adlı çalışmada rezervasyon sistemiyle park işlemi gerçekleştirildiğinde hava ve gürültü kirliliği oranlarındaki olumlu anlamda değişimlere değinmişlerdir.

Hongwei Wang, “A Reservation-based Smart Parking System” isimli çalışmasıyla rezervasyon sisteminin, sürücülerin park yeri arama süresinin önemli ölçüde azalmasına yardımcı olabileceği düşüncesine değinmiştir.

Otopark kullanımı artarken sürücüler için hizmet kalitesi de artırılmalıdır Bu konuyla ilgili Nikolaos Doulamis, Eftychios Protopapadakis ve Lambros Lambrinos tarafından hazırlanan “Improving Service Quality for Parking Lot Users Using Intelligent Parking Reservation Policies” adlı çalışmalarında, aralıklı planlama ilkelerini kullanan bir optimizasyon stratejisine dayanan, akıllı bir park rezervasyonu yönetim mimarisi önerisinde bulunmuşlardır.

## II. BÖLÜM

### 2. MVC YAPISI

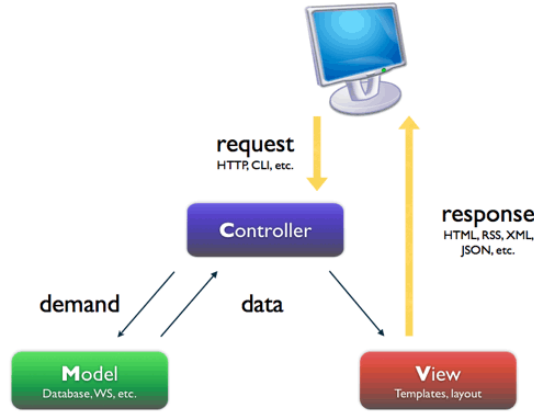
#### 2.1 MVC Yapısı

MVC (Model-View-Controller), yazılan uygulamanın iş mantığı ile kullanıcı arayüzünü birbirinden ayırıştıran, uygulamanın farklı amaçlara hizmet eden kısımlarının birbirine girmesini engelleyen yazılım mimarisidir[3].

**Model:** Model, proje içerisinde kullanılacak olan nesnelerin oluşturulduğu kısımdır. Günlük hayattaki somut nesnelerin, bilgisayar ortamında modellenmesi anlamına gelir[2].

**View:** Proje tamamlandığında kullanıcının gördüğü arayüzdür. Bu bir web sayfası, masaüstü uygulaması arayüzü veya mobil bir tasarım olabilir. Projenin yapısına göre bu tasarım farklı şekillerde oluşturulabilir[2].

**Controller:** Projedeki tüm işlemlerin (veritabanı işlemleri, hesaplamalar, veri aktarımı v.b) yapıldığı kontrol bölümüdür. Controller ayrıca model ve view arasındaki veri akışını da kontrol eder[2].



Şekil 2.1 MVC Yapısı

Şekil 2.1’de MVC yapısı gösterilmektedir.

## 2.2 MVC Çalışma Prensibi

Bir MVC projesinde kullanıcı, tarayıcı üzerinden (View) sayfaya istek yaptığında bu istek Controller'a iletilir. Controller isteği gerçekleştirmek üzere model ve bağlantılı bileşenleri ile gerekli sınıf ve metotları çağırır. Elde ettiği sonuçları View'e göndererek sayfanın görüntülenmesini sağlar[2].

## 2.3 MVC Kullanımının Avantajları

**Proje geliştirme süresini kısaltması:** Projede ki katmanlar birbirinden farklı olduğu için geliştiriciler eş zamanlı çalışarak kodlama yapabilirler. Ayrıca bu katmanların birleştirilmesi işlemi oldukça kolay gerçekleştirilir[2].

**Yeniden kullanılabilirlik:** Özellikle web projelerinde sunucu taraflı yazılan kodlar sadece o kontroller tarafından kullanılmaktadır. MVC ile yazılan kodlar ise kontrollerden tamamen bağımsızdır ve farklı projelerde kolaylıkla kullanılabilirler[2].

**Performans:** Özellikle web projelerinde ViewState kavramı birçok verinin tutulması gereken durumlarda performans düşüşü meydana getirmektedir. MVC projelerinde Viewstate bulunmadığından performans kaybı meydana gelmez[2].

## 2.4 Symfony Framework Yapısı

Kendi sitesinde tanımlandığı gibi, Symfony, bir PHP bileşenleri seti, bir Web Uygulama çerçevesi, bir Felsefe ve bir Topluluk'tan oluşan ve hep birlikte uyum içinde çalışan bir sistemdir[6]. PHP 5 kurulu Unix, Mac OS ve Windows platformlarında çalışabilmektedir. Yahoo! geliştiricileri tarafından da desteklenmekte

ve sosyal imleme platformu del.icio.us'un da altyapısını oluřturmaktadır[4]. Framework kullanmak bir gereklilik deęildir. Daha kaliteli ve hızlı uygulamalar geliřtirmek iin kullanabileceęimiz bir ara olarak kabul edilmektedir. İř kurallarına tam olarak uyan, yapılandırılmıř srdrlebilir ve ykseltilebilir bir uygulama geliřtirdięinizden emin olarak alıřmanızı saęlamaktadır. Framework geliřtiricilerin, modlleri tekrar tekrar kullanmalarını saęlayarak, Framwork'e baęlı kalmadan, dięer alıřma alanlarına odaklanmaları iin zaman kazandırmaktadır[6].

### III. BÖLÜM

#### 3.KULLANICI PANELİ

Kullanıcı paneli oluşturmak için hazır bir template seçildi.

##### 3.1 Üye Kayıt ve Giriş İşlemleri

###### 3.1.1 Üye Kayıt

Kullanıcının üye olabilmesi için gerekli bilgileri girebileceği Şekil 3.1’deki kayıt sayfası yapılmıştır. Bu kayıt sayfası Şekil 3.2’deki kodlar ile oluşturulmuştur.

The image shows a user registration form titled "REGISTER" in a teal box. Below the title, there are four input fields labeled "Name", "Surname", "Email", and "Password". At the bottom, there is a teal button with a lock icon and the text "SAVE". The form is centered between two vertical gray bars.

Şekil 3.1 Üye Kayıt Sayfası

```

/**
 * @Route("/register", name="app_register")
 */
public function register(Request $request, UserPasswordEncoderInterface
{
    $user = new User();
    $form = $this->createForm( type: UserType::class, $user);
    $form->handleRequest($request);

    if ($form->isSubmitted() && $form->isValid()) {
        // encode the plain password

        $user->setPassword(
            $passwordEncoder->encodePassword(
                $user,
                $form->get('password')->getData()
            )
        );
        $entityManager = $this->getDoctrine()->getManager();
        $entityManager->persist($user);
        $entityManager->flush();
        return $guardHandler->authenticateUserAndHandleSuccess(
            $user,
            $request,
            $authenticator,
            providerKey: 'main' // firewall name in security.yaml
        );
    }
    return $this->render( view: 'registration/adminregister.html.twig', [
        'registrationForm' => $form->createView(),
    ]);
}

```


**Şekil 3.2** Registration Controller


### 3.1.2 Üye Giriş

Kullanıcı kayıt olduktan sonra siteye giriş yapılabilmesi için bir üye giriş sayfası tasarlanmıştır.

Aynı zamanda bu tasarlanan sayfa sitenin ana sayfasıdır(Şekil 3.3).





 Sign in

Not logged in

Please sign in

Email

Password

Sign in

Don't have an account yet?

[Create an account](#)

**Şekil 3.3** Giriş(Ana) Sayfası

## 3.2 Rezervasyon İşlemleri

### 3.2.1 Rezervasyon Sayfası

Siteye giriş yapan kullanıcıyı uygun park yeri seçebileceği bir sayfa karşılamaktadır. Rezervasyon sayfası oluşturulması için park ve level adında iki controller oluşturuldu. Park kontrolünden otoparklara ait numara bilgisi, level kontrollünden ise otoparka ait harf bilgisi çekilmiştir. Ardından kod bölümünde iç içe for döngüsü kullanılarak rezervasyon sayfasına matris şeklinde bir otopark görseli oluşturuldu.

Kullanıcı bu görsel üzerinden uygun yeri seçerek giriş çıkış saatlerini ve aracın plakasını girebileceği Şekil 3.4'deki sayfaya yönlendirilecektir.

## D-2

Check-in (date and hour):

gg.aa.yyyy --:--

Check-out (date and hour):

gg.aa.yyyy --:--

Number Plate:

RESERVE

Şekil 3.4 Rezervasyon Sayfası

### 3.2.2 Rezervasyon Onay

Şekil 3.5'de gösterilen sayfada kullanıcının girdiği giriş çıkış saatlerine göre hesaplanan ücret bilgisi gösterilmiştir. Kullanıcının ödeme işlemi için kullanacağı kart bilgilerinin girilmesi gerekmektedir. Son olarak kullanıcının onayı ile rezervasyon işlemi gerçekleştirilmektedir.

## D-2

Price: 30 TL

Checkin: 11-06-2020 17:15

Checkout: 11-06-2020 20:15

Number Plate: 06BKY06

## PAYMENT

Owner
Card Number
Submit

**Şekil 3.5** Rezervasyon Onay

```

/**
 * @Route("/reservations/{pid}/{sid}", name="user_reservations", methods={"GET", "POST"})
 */
public function reservations(Request $request, ReservationRepository $reservationRepository,
                             LevelRepository $levelRepository, $pid, $sid,
                             ParkRepository $parkRepository): Response
{
    $giris=$_REQUEST["checkin"];
    $cikis=$_REQUEST["checkout"];
    $checkin=Date( format: "Y-m-d H:i:s",strtotime($giris)); // tarih alınıyor
    $checkout=Date( format: "Y-m-d H:i:s",strtotime($cikis));

    $checkinl=strtotime($checkin); //tarihin numeric karşılığı
    $checkoutl=strtotime($checkout);
    $diff = round( val: ($checkoutl-$checkinl)/3600); //tarih saat cinsinden hesaplanıyor

    $park=$parkRepository->findOneBy(['id'=>$pid]);
    $levels=$levelRepository->findOneBy(['slot'=>$sid]);

    $data["total"]=$diff*$park->getNumber();
    $data["checkin"]=$checkin;
    $data["checkout"]=$checkout;

    $reservationn = new Reservationn();
    $form = $this->createForm( type: ReservationnType::class, $reservationn);
    $form->handleRequest($request);
    $submittedToken = $request->request->get( key: 'token');
}

```

**Şekil 3.6** Resevation Controller

Bu rezervasyon işlemi için Şekil 3.6 ve Şekil 3.7 ‘deki kodlar kullanılmıştır.

```

if ($form->isSubmitted()) {
    if($this->isCsrfTokenValid( id: 'form-reservationn',$submittedToken)) {
        $entityManager = $this->getDoctrine()->getManager();

        $checkin=date_create_from_format( format: "Y-m-d H:i:s",$giris );
        $checkout=date_create_from_format( format: "Y-m-d H:i:s",$cikis);
        // $reservationn->setDays($diff);
        $checkinl=strtotime($checkin); //tarihin numeric karşılığı
        $checkoutl=strtotime($checkout);
        $diff = round( val: ($checkoutl-$checkinl)/3600); //tarih saat cinsinden hesaplanıyor

        $data["total"]=$diff*$park->getNumber();
        $reservationn->setTotal($data["total"]);
        dump($data);
        die();
        $reservationn->setCheckin(new \DateTime(),$checkin);
        $reservationn->setCheckout(new \DateTime(),$checkout);
        $reservationn->setStatus('New');
        $reservationn->setIp($_SERVER['REMOTE_ADDR']);
        $reservationn->setParkid($pid);
        $reservationn->setLevelid($sid);
        $user=$this->getUser();
        $reservationn->setUserid($user->getId());
        $reservationn->setCreatedAt(new \DateTime()); // Get now datetime

        $entityManager->persist($reservationn);
        $entityManager->flush();

        return $this->redirectToRoute( route: 'user_booking');
    }
}

```

Şekil 3.7 Reservation Controller Devamı

### 3.3 Kullanıcı Profili

Kullanıcı profilinde, kullanıcının ad, soyad ve email gibi kişisel bilgilerinin yer aldığı ‘profile information’, yapmış olduğu rezervasyon işlemlerini görüp yönetebileceği ‘my booking’ ve siteden çıkış yapabilmesi için ‘logout’ sekmeleri oluşturuldu.

### 3.4 İletişim Sayfası

Gerekli olduğunda kullanıcının adminine ulaşabilmesi için iletişim kurabileceği ‘contact information’ ve ‘contact form’ bulunmaktadır. ‘Contact form’ üzerinden kullanıcı adminine mesaj gönderebilmektedir.

```
public function buildForm(FormBuilderInterface $builder,
{
    $builder
        ->add( child: 'name')
        ->add( child: 'email')
        ->add( child: 'subject')
        ->add( child: 'message')
        ->add( child: 'status', type: ChoiceType::class, [
            'choices' => [
                'Read' => 'Read',
                'New' => 'New',
                'Answered' => 'Answered'],
        ])
        ->add( child: 'ip')
        ->add( child: 'note')
        ->add( child: 'created_at')
        ->add( child: 'updated_at')
    ;
}
```

Şekil 3.8 Message Form

## IV.BÖLÜM

### 4. ADMIN PANELİ

Admin paneli, adminin siteyi yönetebilmesi için yapılmış bir sayfadır. Bu panelde dört sekme oluşturuldu.

**Form Messages:** Hangi kullanıcının hangi mesajı gönderdiği, mesajın gönderilme saati gibi bilgiler bulunmaktadır. (Şekil 4.1)

**Park:** Otoparkın daha önce eklenmiş olan numaralarını getirilmekte ve yeni numaralar eklenebilmektedir. (Şekil 4.2)

**Level:** Otoparkın daha önce eklenmiş olan harflerini getirilmekte ve yeni harfler eklenebilmektedir.(Şekil 4.3)

**Users:** Admin buradan siteye üye olan kullanıcıların bilgilerini görüntüleyip yönetebilmektedir.

**Setting:** Otopark şirketinin bilgilerinin girilip düzenlenebildiği bölümdür.

```

* @Route("/", name="admin_messages_index", methods={"GET"})
*/
public function index(MessagesRepository $messagesRepository): Response
{
    return $this->render( view: 'admin/messages/index.html.twig', [
        'messages' => $messagesRepository->findBy([], ['id'=>'DESC']),
    ]);
}

/**
* @Route("/new", name="admin_messages_new", methods={"GET","POST"})
*/
public function new(Request $request): Response
{
    $message = new Messages();
    $form = $this->createForm( type: MessageType::class, $message);
    $form->handleRequest($request);

    if ($form->isSubmitted() && $form->isValid()) {
        $entityManager = $this->getDoctrine()->getManager();

        $entityManager->persist($message);
        $entityManager->flush();

        return $this->redirectToRoute( route: 'admin_messages_index');
    }

    return $this->render( view: 'admin/messages/new.html.twig', [
        'message' => $message,
        'form' => $form->createView(),
    ]);
}

```

**Şekil 4.1** Message Controller



```

/ **
 * @Route("/", name="admin_park_index", methods={"GET"})
 */
public function index(ParkRepository $parkRepository): Response
{
    return $this->render( view: 'admin/park/index.html.twig', [
        'parks' => $parkRepository->findAll(),
    ]);
}

/ **
 * @Route("/new", name="admin_park_new", methods={"GET", "POST"})
 */
public function new(Request $request): Response
{
    $park = new Park();
    $form = $this->createForm( type: ParkType::class, $park);
    $form->handleRequest($request);

    if ($form->isSubmitted() && $form->isValid()) {
        $entityManager = $this->getDoctrine()->getManager();
        $entityManager->persist($park);
        $entityManager->flush();

        return $this->redirectToRoute( route: 'admin_park_index');
    }

    return $this->render( view: 'admin/park/new.html.twig', [
        'park' => $park,
        'form' => $form->createView(),
    ]);
}

```

Şekil 4.2 Park Controller

```

/ **
 * @Route("/", name="admin_level_index", methods={"GET"})
 */
public function index(LevelRepository $levelRepository): Response
{
    return $this->render( view: 'admin/level/index.html.twig', [
        'levels' => $levelRepository->findAll(),
    ]);
}

/ **
 * @Route("/new", name="admin_level_new", methods={"GET", "POST"})
 */
public function new(Request $request): Response
{
    $level = new Level();
    $form = $this->createForm( type: LevelType::class, $level);
    $form->handleRequest($request);

    if ($form->isSubmitted() && $form->isValid()) {
        $entityManager = $this->getDoctrine()->getManager();
        $entityManager->persist($level);
        $entityManager->flush();

        return $this->redirectToRoute( route: 'admin_level_index');
    }

    return $this->render( view: 'admin/level/new.html.twig', [
        'level' => $level,
        'form' => $form->createView(),
    ]);
}

```

**Şekil 4.3** Level Controller

## **V.BÖLÜM**

### **SONUÇ VE DEĞERLENDİRME**

Günümüzde otoparklara olan ihtiyaç arttıkça, otopark rezervasyon otomasyonu, gelişen bir ihtiyaç halini almıştır[5]. Otopark rezervasyon otomasyonundaki amaç; kapalı veya açık otoparklarda araçların daha kolay bir şekilde zaman kaybı yaşamadan hızlı bir şekilde park edebilmektedir. Hem otopark işletmecilerinin müşterilerine daha iyi hizmet verebilmeleri hem de müşterilerin daha kolay park yeri bulabilmeleri sağlanmıştır. Otopark rezervasyon otomasyonu ilk açıldığında kullanıcıyı üye giriş/kayıt sayfası karşılamaktadır. Eğer kullanıcı daha önce üye olmadıysa öncelikle üye olması gerekmektedir, zaten üye olmuş bir kullanıcı ise direkt rezervasyon sayfası açılmaktadır. Bu sayfada kullanıcılar uygun olan park yerlerinden birini seçebilmeleri sağlanmaktadır. Park yeri seçimi yapıldıktan sonra kullanıcı, rezervasyon için giriş çıkış zaman aralığını ve plakasını girebilmektedir. Ardından bir sonraki sayfada kullanıcının seçmiş olduğu park yeri, giriş çıkış zaman aralığı, ve bu zaman aralığına göre hesaplanan ücret bilgisi gösterilmektedir. Hemen ardından kullanıcının kart bilgileri girmesi istenmekte ve böylece rezervasyon işlemi tamamlanabilmektedir. Bu proje ile araç sahiplerinin daha rahat ve kolay bir şekilde park edebilmeleri hedeflenmiştir.

## KAYNAKLAR

- [1]Akıllı Otopark Otomasyonu-Fatih YALÇINKAYA,Zekeriya KIRMIZIGÜL-KTÜ-2014
- [2] İnternet: <https://twww.kodlamamerkezi.com/asp-net/mvc-model-view-controller-nedir/>
- [3] İnternet: <https://www.blog.koddit.com/yazilim/mvc-nedir-gercek-orneklerle-mvc-nedir-anlayalim>
- [4] İnternet: <https://tr.wikipedia.org/wiki/Symfony>
- [5] İnternet: <https://www.teta.com.tr/otopark-otomasyon-sistemleri>
- [6] İnternet: [https://www.bilgigunlugum.net/webprog/symfony/symfony\\_giris](https://www.bilgigunlugum.net/webprog/symfony/symfony_giris)

## **ÖZGEÇMİŞLER**

Gizem TÜRKOĞLU Zonguldak'ta doğdu; ilk ve orta öğrenimini aynı şehirde tamamladı; Oktay ve Olcay Yurtbay Anadolu Lisesi'nden mezun olduktan sonra 2014 yılında Karabük Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü' ne girdi.

### **ADRES BİLGİLERİ**

Adres : Yeni Mah. Belediye Sitesi B-1 Blok Kat:4 Çaycuma/ZONGULDAK

Tel : 05372723346

e-posta : gizemtrkgl@gmail.com

**(Devamı)**

## **ÖZGEÇMİŞLER**

Behiye Kübra YILMAZ 1995'te Ankara'da doğdu; ilk ve orta öğrenimini aynı şehirde tamamladı; Etimesgut Lisesi'nden mezun olduktan sonra 2014 yılında Karabük Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği (İngilizce) Bölümü' ne girdi.

### **ADRES BİLGİLERİ**

Adres : Topçu Mah. 1501.Sok. Alinteri Apt. No:28/17 Elvankent/ANKARA

Tel : 05548561186

e-posta : behiyekubrayilmaz@gmail.com