

CodeIgniter4 ile Şehirlerarası Otobüs Firması Web Sitesi Projesi

Intercity Bus Company Website Project with CodeIgniter4

Kaan Toraman, Eda Obuz, Hacer Sueda Efe

Bilişim Sistemleri Mühendisliği
Teknoloji Fakültesi - Kocaeli Üniversitesi
İzmit, Kocaeli /Türkiye

kaan41tor@gmail.com , edaobuz4@gmail.com , suedafe@gmail.com

Canlı sunucu: <http://umuttepeturizm.online/umuttepe/>

Github: <https://github.com/Addicted-to-Chaos/UmuttepeTurizm-Yazlab2-1>

Özet

Bu rapor bir şehirlerarası otobüs firması için geliştirilen web sitesi projesine ait, problem tanımı, yapılan araştırmalar, akış şeması, veritabanı, kullanıcı arayüzü, backend ve karşılaşılan sorunlarla ilgili bilgiler içermektedir. Proje PHP, HTML, CSS ve SQL kullanılarak geliştirilmiştir, veritabanı için MySQL kullanılmış, framework olarak ise CodeIgniter4 kullanılmıştır.

Abstract

This report contains information about the problem definition, research, flow chart, database, user interface, backend and problems encountered for an intercity bus company website project. The project was developed using PHP, HTML, CSS and SQL. MySQL was used for the database, and CodeIgniter4 was used as the framework.

1. Problem Tanımı

Problem tanımı kısaca bir şehirlerarası bir otobüs firması olan Umuttepe Turizm'e bir yolcu bilet satışı web sitesi hazırlanması. Bu web sitesinin CodeIgniter kullanılarak geliştirilmesi istendi. Toplam 4 büyük şehirden 8 saat aralıklarla otobüs seferi kaldırılması gerekti, totalde bir gün içerisinde toplam 36 sefer etmekte. Bilet alımı yaparken otobüsün içine kuşbakışı bakalarak koltuk seçimi yapabilme, kredi kartı ile ödeme, rezervasyon seçeneği, öğrenci indirimi gibi indirimler, alınan biletlere PNR kodu oluşturulması, bunun kullanıcıya email atılması, iptal edilen biletlerin ücretinin kullanıcının bakiyesine eklenmesi ve kullanıcı yeni bilet alırken bu bakiyeyi kullanabilmesi gibi pek çok özellik problem tanımında bulunuyor.

Ayrıca şehirlerarası otobüs yolculukları yoğun iş tempusu ve sık seyahat ihtiyacı olan kişiler tarafından tercih edilen önemli bir ulaşım yöntemi. Ancak, mevcut çevrimiçi otobüs rezervasyon platformlarında kullanıcılar bazen karmaşık arayüzlerle ve yetersiz bilgilerle

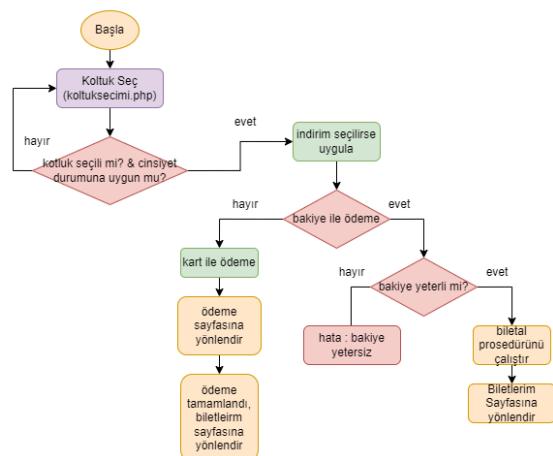
karşılaşabilmektedir. Bu durum seyahat planlamasını ve rezervasyon sürecini zorlaştırarak kullanıcı deneyimini olumsuz etkilemektedir.

Bu geliştirilen siteyle mevcut otobüs rezervasyon sistemlerindeki eksiklikleri ve kullanıcı deneyimindeki zorlukların aşılması amaçlandı. Projenin temel amacı kullanıcıların kolayca otobüs seyahatlerini planlayabilmelerini, uygun seferleri bulabilmelerini, bilet rezervasyonlarını gerçekleştirebilmelerini ve seyahatleriyle ilgili gerekli bilgilere erişebilmelerini sağlayacak kullanıcı dostu bir platform sunmaktadır.

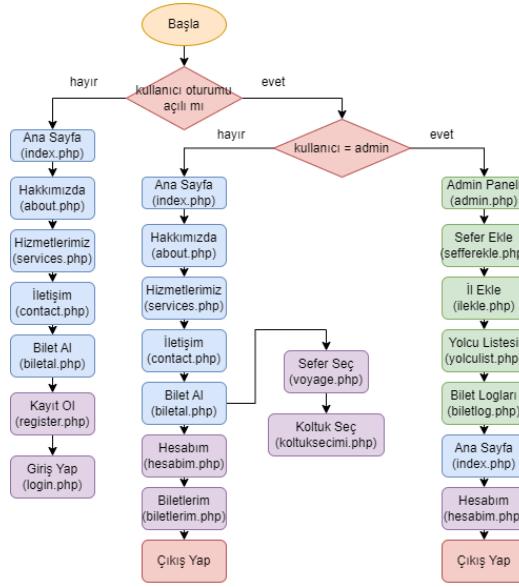
2. Yapılan Araştırmalar

Proje gereklikleri doğrultusunda geliştirmek istediğimiz sistem için hangi araçları ve hangi veritabanı yönetim sistemini kullanabileceğimizle ilgili kapsamlı bir araştırma süreci gerçekleştirildi. CodeIgniter kullanımı zorunlu olduğu için ve daha önce kullanmadığımız bir framework olduğu için sitenin frontendi ile veritabanını bağlama aşamasında bu frameworkle ilgili pek çok araştırma yapıldı.

3. Akış şeması



Şekil 1.1 : Akış Diyagramı - Bilet alma



Şekil 1.2 : Akış Diyagramı - Web Sitesi

4. Kullanıcı Arayüzü (Frontend)

Sitenin frontend tasarımı ve Admin paneli için internetten hazır bootstrap template kullanıldı. Bu template de ana sayfa, hakkımızda, hizmetlerimiz ve ana admin paneli gibi sayfalar bulunmasına karşın ihtiyaç olan diğer sayfalar bulunmamaktaydı, bu yüzden bazı diğer sayfaların tasarımını sıfırdan yaptırdı. Giriş yap kayıt ol, hesabım, biletlerim, sefer seçimi, koltuk seçimi, il ve sefer ekleme gibi sayfalar sıfırdan yapılanlardan bazlılardır.

4.1 Ana Sayfa



Şekil-2 index.php



Şekil-3 index.php

Şekil-4 index.php

Şekil-5 index.php

Şekil-6 index.php

4.1.1 Hakkımızda

Şekil-7 about.php

4.1.2 Hizmetlerimiz

Şekil-8 services.php

4.1.3 İletişim

Şekil-9 contact.php

4.1.4 Bilet Bul

Şekil-10 biletal.php

4.1.5 Sefer Seç

Şekil-11 voyage.php

4.1.6 Koltuk Seç

Şekil-12 koltuksec.php

4.2 Giriş Yap - Kayıt Ol

Şekil-13 register.php

Şekil-14 login.php

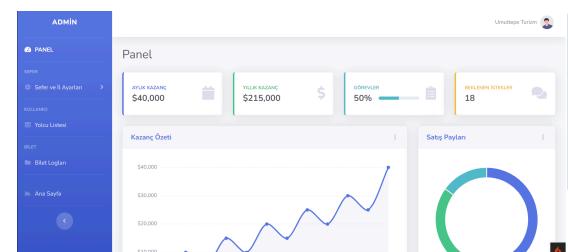
4.3 Hesabım

Şekil-15 hesabim.php

4.3.1 Bilet Bilgilerim

Şekil-16 biletlerim.php

4.4 Admin Paneli



Şekil-17 admin.php

4.4.1 Sefer Ekle

Şekil-18 seferekle.php

4.4.2 İl Ekle

Şekil-19 ilekle.php

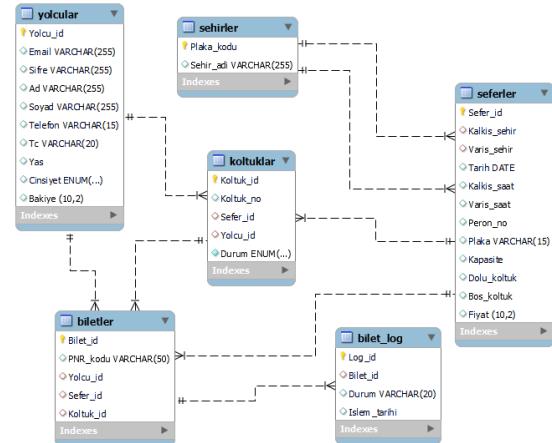
4.4.3 Kullanıcılar

Şekil-20 yolculist.php

5. Veritabanı

Veritabanı yönetim sistemi olarak MySQL kullanılmıştır. Veritabanında temelde 6 adet tablo bulunmaktadır bunlar Yolcular, Şehirler, Seferler, Koltuklar, Biletler ve Bilet_log tablolarıdır. Yolcular tablosunda kullanıcıların gerekli bilgileri tutulmaktadır bunlar isim soyisim, tc kimlik numarası, cinsiyet, telefon numarası, eposta adresidir. Şehirler tablosunda şehrin plaka kodu ve adı tutulmaktadır. Seferler tablosunda seferin kalkış ve varış şehir bilgileri, sefer tarihi, kalkış varış saatleri, peron numarası, otobüsün plakası ve seferin fiyatı bilgileri tutulmaktadır; Kalkış ve varış şehirleri için şehirler tablosundan referans alınmıştır foreign key kullanılarak. Koltuklar tablosunda koltuk numarası, sefer id'si, yolcu_id ve koltuğun durumu tutulmaktadır, koltuk durumu değişkeni enum olarak ayarlanmıştır rezerve, satın alındı veya müsait seçenekleri vardır, yolcu_id ve sefer_id değişkenleri ilgili tablolardan foreign keyle bağlanmıştır. Biletler tablosunda PNR_kodu, yolcu_id, sefer_id, koltuk_id tutulmaktadır, aynı şekilde yolcu sefer ve koltukların id'leri ilgili tablolardan foreign keyle

bağlanmıştır. Son olarak bilet_log tablosu bulunmaktadır bu tablo iptal edilen bilet işlemlerinin de takibinin yapılabilmesi için bir işlem geçmiş tablosu olarak oluşturulmuştur, burada bilet_id durum ve işlem tarihi tutulmaktadır. Yani kullanıcı bilet satın da alsa, rezerve de etse, askıya da alsa tüm durumlar için tarihle beraber ayrı ayrı log kaydi oluşturulmaktadır.



Şekil-21 : veritabanı tabloları

Veritabanında bazı işlemlerin gerçekleştirilmesi için stored procedureler kullanılmıştır. SeferYenile prosedürü prosedüre gönderilen sayı kadar sonraki güne seferleri ekler, yani 2 verilirse şu andan 2 gün sonrasının seferlerini ekler toplam 36 adet olmak üzere. Bu seferlerin hepsinin 25er adet koltuğu da olması gereği için bununla da ilgili KoltuklarıSefereEkle isimli bir prosedür yazılmış olup bu prosedür SeferYenile prosedürü içerisinde her sefer eklendikten sonra çağrılmıştır. Böylece sadece tek prosedür çağrılarak işlem halledilebilmiştir.

```

CREATE PROCEDURE KoltuklarıSefereEkle(IN yeni_sefer_id BIGINT)
BEGIN
    DECLARE koltuk_no BIGINT DEFAULT 1;

    WHILE koltuk_no <= 25 DO
        INSERT INTO Koltuklar (Koltuk_no, Sefer_id, Durum)
        VALUES (koltuk_no, yeni_sefer_id, 'Müsait');
        SET koltuk_no = koltuk_no + 1;
    END WHILE;
END$$
    
```

Şekil-22 : koltuk ekleme prosedürü

```

CREATE PROCEDURE SeferYenile(gun_sayisi INT)
BEGIN
    DECLARE next_date DATE;
    DECLARE sefer_varmi INT;
    DECLARE yeni_sefer_id BIGINT;

    -- Verilen gun sayisi kadar sonrasına
    SET next_date = DATE_ADD(CURDATE(), INTERVAL gun_sayisi DAY);
    -- Yarın için sefer var mı diye kontrol et
    SELECT COUNT(*) INTO sefer_varmi FROM Seferler WHERE Tarih = next_date;

    -- Eğer yarın için sefer yoksa ekle
    IF sefer_varmi = 0 THEN
        -- ANKARA - BURSA
        INSERT INTO Seferler (Kalkis_sehir, Varis_sehir, Tarih, Kalkis_saat, Varis_saat,
        VALUES (6, 16, next_date, '00:00', '08:00', 1, '06A123', 25, 0, 25, 300);
        SET yeni_sefer_id = LAST_INSERT_ID();
        CALL KoltuklariSeferEkle(yeni_sefer_id);
    END IF;

```

Sekil-23 : sefer yenileme prosedürü

Biletal prosedürü bulunmaktadır bu prosedüre yolcu_id sefer_id ve koltuk_no girildiğinde koltuklar tablosunda koltuk durumunu satın alındı yaparak koltuğa satın alan yolcunun_id'sini girmekte, bir pnr_kodu oluşturarak biletler tablosunda bilet oluşturmaktır ve bilet_log'a bu durumu girmektedir. Ayrıca biletin rezerve eden kullanıcıyla satın almak isteyen kullanıcının id'si aynı ise satın alma işlemine izin vermektedir.

```

-- Rezervasyon yapan yolcu aynı, satın alabilir
UPDATE Koltuklar
SET Durum = 'Satın Alındı',
    Yolcu_id = p_yolcu_id
WHERE Sefer_id = p_sefer_id AND Koltuk_no = p_koltuk_no;

UPDATE Biletler
SET PNR_kodu = pnr_kodu
WHERE Yolcu_id = p_yolcu_id AND Sefer_id = p_sefer_id AND Koltuk_id = koltuk_id_val;

SELECT Bilet_id FROM Biletler WHERE Yolcu_id = p_yolcu_id AND Sefer_id = p_sefer_id AND Koltuk_no = p_koltuk_no;

-- Bilet Log'a kaydet
INSERT INTO Bilet_Log (Bilet_id, Durum, Islem_tarihi)
VALUES (bilet_id_val, 'Satın Alındı', NOW());

```

Sekil-24 : bilet al prosedürünün bir bölümü

Aynı şekilde rezerve prosedürü bulunmaktadır. Rezerve işlemleri sefere 2 gün kalaya kadar geçerli olabileceği için rezerve edebilmek için sefere 2 günden fazla zaman bulunmalıdır bu kontroller prosedür içerisinde yapılarak duruma göre rezervasyon işlemine izin verilmektedir.

```

-- Koltuğun durumunu kontrol et
SELECT Durum INTO koltuk_durumu_val
FROM Koltuklar
WHERE Sefer_id = p_sefer_id AND Koltuk_no = p_koltuk_no;

-- Seferin tarihini kontrol et
SELECT Tarih INTO sefer_tarihi
FROM Seferler
WHERE Sefer_id = p_sefer_id;

-- Koltuğun durumuna ve sefer tarihine göre işlem yap
IF koltuk_durumu_val = 'Musait' AND DATEDIFF(sefer_tarihi, CURDATE()) > 2 THEN
    -- Koltuk müsait ve sefere 2 günde fazla süre var

    INSERT INTO Biletler (PNR_kodu, Yolcu_id, Sefer_id, Koltuk_id)
    VALUES (NULL, p_yolcu_id, p_sefer_id, (SELECT Koltuk_id FROM Koltuklar WHERE Sefer_id = p_sefer_id />
    SELECT LAST_INSERT_ID()) INTO bilet_id_val;

    UPDATE Koltuklar
    SET Durum = 'Reserve', Yolcu_id = p_yolcu_id
    WHERE Sefer_id = p_sefer_id AND Koltuk_no = p_koltuk_no;

    -- Bilet Log'u kaydet
    INSERT INTO Bilet_Log (Bilet_id, Durum, Islem_tarihi)
    VALUES (bilet_id_val, 'Rezervasyon İptali Başarıyla Gerçekleştirildi', NOW());

```

Sekil-25 : rezerve iptal prosedürünün bir bölümü

Tam tersi işlem için ise Askiya_al prosedürü çalıştırılarak istenen bilet_id'si verildiğinde koltuğun durumunu müsaite çevirerek bilet_log'a bu durumu girmekte, kullanıcının hesap bakiyesine bilet fiyatını iade etmekte ve biletler tablosunda biletin koltuk_id kısmını null yaparak bilet iptalini gerçekleştirmektedir. Bilet iptalleri 1 gün kalaya kadar gerçekleştirilebileceği için bu kontrolden prosedür içerisinde yapılmaktadır.

```

-- Biletin iptali
IF kalan_gun >= 1 THEN
    -- Biletin fiyatını ve yolcunun bakiyesini al
    SELECT Fiyat INTO bilet_fiyat
    FROM Seferler
    WHERE Sefer_id = p_sefer_id;

    SELECT Bakiye INTO yolcu_bakiye
    FROM Yolcular
    WHERE Yolcu_id = p_yolcu_id;

    -- İptal edilen (askiya alınan) biletin ücretini müsteri bakiyesine ekle
    UPDATE Yolcular
    SET Bakiye = Bakiye + bilet_fiyat
    WHERE Yolcu_id = p_yolcu_id;

    -- Bilet durumunu askiya al
    UPDATE Biletler SET koltuk_id = NULL
    WHERE Bilet_id = p_bilet_id;

    -- Koltuk durumunu güncelle
    UPDATE Koltuklar
    SET Durum = 'Musait', Yolcu_id = NULL
    WHERE Koltuk_id = p_koltuk_id;

```

Sekil-26 : askiya al prosedürünün bir bölümü

Rezervasyonİptal prosedürü ise sefere 2 gün kalmış olmasına rağmen bilet hala satın alınmamışsa rezervasyonu iptal etmektedir.

```

-- Rezervasyonun iptal edileceği tarihi bul
SELECT S.Sefer_id, B.Bilet_id, K.Yolcu_id, K.Koltuk_id
INTO seferIdVar, biletIdVar, yolcuIdVar, koltukIdVar
FROM Biletler B
JOIN Koltuklar K ON B.Koltuk_id = K.Koltuk_id
JOIN Seferler S ON B.Sefer_id = S.Sefer_id
WHERE K.Durum = 'Rezerve'
AND S.Tarih - CURDATE() <= 2;

-- Eğer bulunduğu rezervasyonu iptal et ve koltuğu müsait duruma getir
IF seferIdVar IS NOT NULL THEN
    UPDATE Koltuklar
    SET Durum = 'Musait'
    WHERE Koltuk_id = koltukIdVar;

    DELETE FROM Biletler
    WHERE Bilet_id = biletIdVar;

    -- Log tablosuna işlemi kaydet
    INSERT INTO Bilet_Log (Bilet_id, Durum, Islem_tarihi)
    VALUES (biletIdVar, 'Rezervasyon İptali Başarıyla Gerçekleştirildi', NOW());

    SELECT 'Rezervasyon iptali başarıyla gerçekleştirildi.' AS Result;
ELSE

```

Sekil-27 : rezerve iptal prosedürünün bir bölümü

Bu prosedürün günlük olarak çalıştırılması için event oluşturulmuştur, bu event sayesinde sistem her gün bu prosedürü çağrımaktır ve sefere 2 günden az kalan rezervasyonlar iptal edilmektedir.

```
-- HER GÜN OTOMATİK ÇALIŞIP 2 GÜNDEN AZ KALAN REZERVASYONLARI IPTAL EDECEK
DELIMITER //

CREATE EVENT IF NOT EXISTS RezervasyonIptalEvent
ON SCHEDULE
EVERY 1 DAY
STARTS CURRENT_DATE
COMMENT 'Bu olay, rezervasyon iptallerini kontrol etmek için her gün çalışır.'
DO
BEGIN
    CALL RezervasyonIptal();
END //

DELIMITER ;
```

Şekil-28 : rezerve iptal eventi

6. Backend

6.1 Codeigniter4 Kullanımı

Projede, CodeIgniter 4 framework kullanılarak hızlı ve etkili bir web uygulaması geliştirildi. Başlangıçta, framework kurulumu gerçekleştirilip gerekli yapılandırmalar yapıldı. CodeIgniter 4, MVC mimarisine dayanır; bu nedenle modeller, app/Models dizininde, sayfalar app/Views içinde ve denetleyiciler app/Controllers dizininde gruplandırıldı. Rotalar, işlevselliklerine göre app/Config/Routes.php dosyasında tanımlandı ve yönlendirildi.

CodeIgniter'in sunduğu otomatik yükleme, hata işleme ve veritabanı erişimi gibi özellikler, proje backendini yönetmeyi oldukça kolaylaştırdı. Bu özellikler, geliştiricilerin daha hızlı ve daha güvenilir bir şekilde kod yazmalarını sağlayarak geliştirme sürecini optimize etti.

Ayrıca, CodeIgniter'in sunduğu geniş kapsamlı belgeler ve topluluk desteği, geliştiricilerin sorunlarını çözmelerine ve yeni özellikler eklemelerine yardımcı oldu. Bu sayede, proje hem sağlam bir temel üzerine inşa edildi hem de gelecekteki genişletmeler için uygun bir zemin oluşturuldu.

6.2 Veritabanı Bağlantısı

CodeIgniter ile MySQL veritabanı arasındaki bağlantıyı kurmak için codeigniter dosyası içerisinde app -> config -> database.php kodunda default olarak ayarlanmış olan veritabanı bilgilerini kullanılabilecek olan yeni veritabanı bilgileriyle değiştirmek gereklidir. Localde çalıştırırken hostname kısmına localhost, username kısmına root yazılmalı, password boş bırakılmalı, database kısmına kullanılan veritabanının adı yazılmalı. MySQL port olarak 3306 kullandığı için MySQL kullanılacaksa port da buna uygun olarak değiştirilmeli.

```
/*
public array $default = [
    'DSN'          => '',
    'hostname'     => 'localhost',
    'username'     => 'root',
    'password'     => '',
    'database'     => 'umuttepe',
    'DBDriver'     => 'MySQLi',
    'DBPrefix'     => '',
    'pConnect'     => false,
    'DBDebug'      => true,
    'charset'      => 'utf8',
    'DBCollat'     => 'utf8_general_ci',
    'swapPre'      => '',
    'encrypt'      => false,
    'compress'     => false,
    'strictOn'     => false,
    'failover'     => [],
    'port'         => 3306,
    'numberNative' => false,
];
```

Şekil-29 : veritabanı bağlantısı database.php

Benzer şekilde değişiklik yapılması gereken bir diğer kısım ise direkt CodeIgniter dosyasının içerisindeki env isimli dosya .env uzantılı olan değil direkt ismi env olan değiştirilmeli. Database başlığı altındaki kısım aynı database.php de değiştirilen şekilde değiştirilmeli.

```
#-----
# DATABASE
#-----

# database.default.hostname = localhost
# database.default.database = umuttepe
# database.default.username = root
# database.default.password =
# database.default.DBDriver = MySQLi
# database.default.DBPrefix =
# database.default.port = 3306

# database.tests.hostname = localhost
# database.tests.database = mydatabase
# database.tests.username = root
# database.tests.password =
# database.tests.DBDriver = MySQLi
# database.tests.DBPrefix =
# database.tests.port = 3306
```

Şekil-30 : veritabanı bağlantısı, env dosyası

6.3 Kayıt ol-Giriş Yap

6.3.1 Kayıt ol

Web sitesine kayıtları ve girişleri kontrol etmek amacıyla oluşturulmuş formlardır. Kullanıcının siteye ilk olarak kayıt olması gerekmektedir. Kayıt ol ekranındaki isterleri doldurduktan sonra bu veriler veri tabanına aktarılmaktadır. Kullanıcı 'Kayıt Ol' butonuna bastığında '`<form action="<?php echo site_url('/kayitOl'); ?>" method="post">`' işlemi gerçekleşir ve verileri göndermek için ilgili fonksiyon çağırılır. Şekil-31'de görüldüğü üzere ilk olarak veritabanındaki hangi tabloya veri göndereceğimize karar vermek için bu model kod dosyasını oluşturmak gerekmektedir.

```

<?php namespace App\Models;
use CodeIgniter\Model;

11 references | 0 implementations
class UsersModel extends Model{
    18 references
    protected $table='yolcular';
    7 references
    protected $allowedFields=['Yolcu_id','Email','Sifre','Ad',
    'Soyad','Telefon','Tc','Yas','Cinsiyet','Bakiye'];
}

```

Şekil-31

Şekil-32'de görülen kodlar kullanıcının girdiği verileri form metodu içindeki değişken isimleri ile veritabanındaki ilgili tablodaki değişken isimlerini birbirine eşleyerek '\$this-> request->getVar()' metodu ile veritabanına gönderir. Ayrıca veritabanına bu verileri gönderirken oluşturduğumuz model dosyasını çağrırmak gerekmektedir.

```

$data = [
    'Email' => $this->request->getVar('Email'),
    'Sifre' => $this->request->getVar('Sifre'),
    'Ad' => $this->request->getVar('Ad'),
    'Soyad' => $this->request->getVar('Soyad'),
    'Telefon' => $this->request->getVar('Telefon'),
    'Tc' => $this->request->getVar('Tc'),
    'Yas' => $yas,
    'Cinsiyet' => $this->request->getVar('Cinsiyet')
];

$model = new UsersModel();
$model->insert($data);

```

Şekil-32

Şekil-33'deki kod ile yapılacak işlerin hangi rotada ilerleyeceğini belirleyen metodlar bulunmaktadır. Bu metodlar sayesinde web sayfalarında yönlendirmeler sağlanabilmektedir. Bu şekilde verilen 'kayıtol' ile Şekil-32'deki kod çağrılmaktadır. 'register' ile de Şekil-34'deki fonksiyon çağrılmaktadır. Bu şekilde kayıt olma işlemi kesintisiz bir şekilde gerçekleştirilmektedir.

```

<?php

use CodeIgniter\Router\RouteCollection;
/**
 * @var RouteCollection $routes
 */
$routes->get('/', 'Home::index');
$routes->post('/kayitol', 'User::kayitol');
$routes->get('/register', 'Home::register');

```

Şekil-33

Şekil-4'deki kod kullanıcı isteklerini karşılamak için farklı sayfaları döndürür.

```

0 references | 0 overrides
public function register(){
    $session=session();
    if( $session->has('user') ){
        return view('hesabim');
    }else{
        return view('register');
    }
}

```

Şekil-34

6.3.2 Giriş Yap

Kayıt ol işlemi başarılı bir şekilde gerçekleştirilirse kullanıcı giriş yap işlemi ile siteyi kullanabilmektedir. 'Giriş Yap' sayfasında kullanıcı 'Email' ve 'Şifre' verileri giriş sağlayabilmektedir. Şekil-35'de görülen form metodu cağızır ve girilen verileri post işlemi ile Şekil-6'daki 'girisYap' fonksiyonuna gönderilir.

```

<?><?php $session=session(); echo $session->getflashdata('login'); ?></?>
<form action="=php echo site_url('/girisYap');?" method="post">
    <h2>Giriş Yap</h2>
    <label for="Email">E-mail:</label>
    <input type="email" id="Email" name="Email" required>

    <label for="Sifre">Şifre:</label>
    <input type="password" id="Sifre" name="Sifre" required>

    <div class="form-row">
        <button type="submit" class="btn btn-primary" name="kaydet">Giriş Yap</button>
    </div>

    <p>Hesabınız yok mu? <a href="register" class="register-link">Kayıt Ol</a></p>
</form>

```

Şekil-35

Bu fonksiyonda da kayıt oluda kullanılan model kullanılmıştır. Kayıt olunca gönderilen verilen bu fonksiyonda çağrılarak veritabanına kayıtlı olma durumu kontrol olmaktadır. İki farklı giriş yapılmaktadır. Biri admin girişi diğeri normal kullanıcıların erişimine açık kullanıcı girişidir. Şekil-37'de giriş yapıldıktan sonra girilen verilerin kontrolü sağlanıyor. Eğer veriler admin verileri ile eşleşiyor ise admin sayfasına yönlendirme yapılmaktadır. Eğer normal kullanıcı ise anasayfa kısmasına yönlendirme yapılmaktadır.

```

0 references | 0 overrides
public function girisYap() {
    $model = new UsersModel();
    $result = $model->where('Email', $this->request->getVar('Email'))
                ->where('Sifre', $this->request->getVar('Sifre'))
                ->first();

    $session = session();

```

Şekil-36

```

if ($result) {
    if ($result['Email'] == 'admin@gmail.com' && $result['sifre'] == '123') {
        // Admin giriş yapıldığında
        $session->set('user', [
            'Volcu_id' => $result['Volcu_id'],
            'Email' => $result['Email'],
            'Sifre' => $result['Sifre'],
            'Ad' => $result['Ad'],
            'Soyad' => $result['Soyad'],
            'Telefon' => $result['Telefon'],
            'Tc' => $result['Tc'],
            'Yas' => $result['Yas'],
            'Cinsiyet' => $result['Cinsiyet'],
            'Bakiye' => $result['Bakiye']
        ]);
        return redirect()->to('admin'); // Admin sayfasına yönlendir
    } else {
        // Normal kullanıcı giriş yapıldığında
        $session->set('user', [
            'Volcu_id' => $result['Volcu_id'],
            'Email' => $result['Email'],
            'Sifre' => $result['Sifre'],
            'Ad' => $result['Ad'],
            'Soyad' => $result['Soyad'],
            'Telefon' => $result['Telefon'],
            'Tc' => $result['Tc'],
            'Yas' => $result['Yas'],
            'Cinsiyet' => $result['Cinsiyet'],
            'Bakiye' => $result['Bakiye']
        ]);
        return view('index', ['user' => $result]);
    }
} else {
    return view('login');
}

```

Şekil-37

Aynı şekilde ‘kayıt ol’da bulunan Şekil-33 ve 34’deki kodlar giriş yap içinde yazilarak kullanıcı isteklerini karşılamak için farklı sayfaları döndürme işlemi sağlanmaktadır.

6.4 Admin Hesabı

6.4.1 Sefer Ekle

Admin hesabı ile giriş yapıldığında admin web sitesine yeni seferler ekleyebilmektedir. Sefer ekle sayfasındaki isterleri doldurduğunda Şekil-38’deki form çalışmaktadır. Bu form ilk olarak tüm isterler girilmiş mi diye ‘bilgiKontrol()’ fonksiyon ile kontrol sağlamaktadır. Eğer tüm isterler doldurulmuş ise girilen verileri ‘ekleSefer’ fonksiyonuna göndermektedir. Sefer ekle tablosu şehir bilgilerini Şekil-39’daki modelden çekmektedir.

```

<form method="post" action="=php echo site_url('/ekleSefer'); ?" onsubmit="return bilgikontrol();">
<div class="row">
<div class="container">
    <div class="form-wrapper">
        <div class="form-column">
            <label for="seferKalkisSehir">Kalkış Şehri:</label>
            <?php
                use App\Models\UserModelSehirler;
                $sehirModel = new UserModelSehirler();
                $sehirler = $sehirModel->findAll();
            >
        </div>

```

Şekil-8

```

class UserModelSehirler extends Model{
    18 references
    protected $table='sehirler';
    7 references
    protected $allowedFields=['Sehir_adi', 'Plaka_kodu'];
}

```

Şekil-39

```

function bilgiKontrol() {
    var seferKalkisSehir = document.getElementById('seferKalkisSehir').value;
    var seferVarisSehir = document.getElementById('seferVarisSehir').value;

    var seferDate = document.getElementById('seferDate').value;
    var kalkissaat = document.getElementById('departure-time').value;
    var varisSaat = document.getElementById('arrival-time').value;
    var seferPeron = document.getElementById('seferPeron').value;
    var seferPlaka = document.getElementById('seferPlaka').value;
    var seferFiyat= document.getElementById('seferFiyat').value;
}

```

Şekil-40

Sefer ekle butonuna basıldıktan sonra girilen veriler Şekil-42’deki ‘eklesefer’ fonksiyonu ile veritabanına gönderilmektedir. Veritabanında da bu verilerin hangi tabloya ekleneceği Şekil-41’deki modelde belirlenmektedir. Sefer ekleme işlemi içinde aynı şekilde ‘kayıt ol’da bulunan Şekil-33 ve 34’deki gibi kodlar yazılarak kullanıcı isteklerini karşılamak için farklı sayfaları döndürme işlemi sağlanmaktadır.

```

<?php namespace App\Models;
use CodeIgniter\Model;

8 references 0 implementations
class UserModelSeferler extends Model{
    18 references
    protected $table='seferler';
    7 references
    protected $allowedFields=['sefer_id','Kalkis_sehir','Varis_sehir','Tarih','Kalkis_saat',
    'Varis_saat','Peron_no','Plaka','Kapasite','Dolu_koltuk','Bos_koltuk','Fiyat'];
}

```

Şekil-41

```

public function eklesefer()
{
    $kalkisSehir = $this->request->getVar('seferKalkisSehir');
    $varisSehir = $this->request->getVar('seferVarisSehir');
    $seferTarih = $this->request->getVar('seferDate');
    $kalkissaat=$this->request->getVar('departure-time');
    $varisSaat = $this->request->getVar('arrival-time');
    $seferPeron = $this->request->getVar('seferPeron');
    $seferPlaka = $this->request->getVar('seferPlaka');
    #$seferKapasite = $this->request->getVar('seferKapasite');
    $seferFiyat = $this->request->getVar('seferFiyat');

    $model = new UserModelSeferler();

    $updateData = [
        'Kalkis_sehir' => $kalkisSehir,
        'Varis_sehir' => $varisSehir,
        'Tarih' => $seferTarih,
        'Kalkis_saat' => $kalkissaat,
        'Varis_saat' => $varisSaat,
        'Peron_no' => $seferPeron,
        'Plaka' => $seferPlaka,
        'Fiyat' => $seferFiyat,
        'Dolu_koltuk'=>0
    ];
}

```

Şekil-42

6.4.1 Şehir Ekle

Web sitesine sefer yapılacak farklı şehir eklemek için oluşturulmuş sayfadır.

```

<?php namespace App\Controllers;
class SehirController extends Controller{
    18 references
    protected $model;
    7 references
    protected $view;
}

```

Şekil-43

Admin sadece eklemek istediği ilin kodunu girmesi yeterlidir. İl koduna ait il adı ‘updateCityName’ fonksiyonu ile otomatik olarak belirtilmektedir.

```
function updateCityName() {
    var plaka_kodu = document.getElementById("Plaka_kodu").value;
    var sehir_adi_input = document.getElementById("Sehir_adi");
    var plaka_il = [
        "01": "Adana",
        "02": "Adiyaman",
        "03": "Afyonkarahisar",
        "04": "Ağrı",
        "05": "Amasya",
        "06": "Ankara",
        "07": "Antalya",
        "08": "Artvin",
        "09": "Aydin",
        "10": "Balikesir",
        "11": "Bilecik",
    ];
}
```

Sekil-44

Plaka kodu seçili il ekle butonuna basıldığında Şekil-45’deki ‘ilekle’ fonksiyonu çağrılmaktadır ve eklenen ili Şekil-39’daki model ile ‘sehirler’ tablosuna ekleme sağlanmaktadır.

```
public function ilekle()
{
    $Plaka_kodu = $this->request->getVar('Plaka_kodu');
    $sehir_adi = $this->request->getVar('Sehir_adi');

    // Plaka kodu seçilmemişse hata göster
    if(empty($Plaka_kodu)) {
        return redirect()->to(site_url('/ilekle'))->with('message', 'Lütfen bir plaka kodu seçin.');
    }

    $model = new UserModelBiletiler();

    // Veritabanında belieli bir plaka kodunun kaydının olup olmadığını kontrol et
    $existingRecord = $model->where('Plaka_kodu', $Plaka_kodu)->first();

    if($existingRecord) {
        // Eğer plaka kodu zaten kayıtlı ise kullanıcıya bilgi ver
        return redirect()->to(site_url('/ilekle'))->with('message', 'Bu plaka kodu zaten kayıtlı.');
    }

    // Plaka kodu kayıtlı değilse ekleme işlemine devam et
    $updateData = [
        'Plaka_kodu' => $Plaka_kodu,
        'Sehir_adi' => $sehir_adi
    ];

    // Oturumu güncelle
    $model->insert($updateData);
    return view('ilekle');
}
```

Sekil-45

6.4.1 Yolcu Listesi

Bu sayfada admin siteye kayıtlı tüm yolcuları görmektedir. Ancak bu listede yolcuların sadece ad, soyad, e-posta, telefon, yaş ve cinsiyet bilgileri gösterilmektedir. Bu tablodaki veriler Şekil-31’de bulunan modelden çekilmektedir.

```
<?php
use App\Models\UsersModel;
$yolcularModel = new UsersModel();
$yolcular = $yolcularModel->findAll();
if(empty($yolcular)){
    echo '<p style="color:red; font-size: 30px;><b>Aktif Sefer bulunmamaktadır.</b></p>';
}
else{
    $counter = 1;
    foreach ($yolcular as $yolcu) {
        echo '<tr>';
        echo '<td>' . $counter . '</td>';
        echo '<td>' . $yolcu['Ad'] . '</td>';
        echo '<td>' . $yolcu['Soyad'] . '</td>';
        echo '<td>' . $yolcu['Email'] . '</td>';
        echo '<td>' . $yolcu['Telefon'] . '</td>';
        echo '<td>' . $yolcu['Yas'] . '</td>';
        echo '<td>' . $yolcu['Cinsiyet'] . '</td>';
        $counter++;
    }
?>
```

Sekil-46

6.4.1 Bilet Bilgileri

Bu sayfada alınmış veya rezerve edilmiş bilet bilgileri bulunmaktadır.

```
use App\Models\UserModelBiletilog;
use App\Models\UserModelBiletiler;

$biletilogModel = new UserModelBiletilog();
$biletilog = $biletilogModel->findAll();

$biletilerModel = new UserModelBiletiler();
$biletiler = $biletilerModel->findAll();

if(empty($biletilog)){
    echo '<p style="color:red; font-size: 30px;><b>Aktif Sefer bulunmamaktadır.</b></p>';
}
else {
    $counter = 1;
    foreach ($biletilog as $biletilog) {
        $biletil = $biletil[$counter - 1];
        echo '<tr>';
        echo '<td>' . $biletilog['Bilet_id'] . '</td>';
        echo '<td>' . $biletil['PNR_kodu'] . '</td>';
        echo '<td>' . $biletil['Yolcu_id'] . '</td>';
        echo '<td>' . $biletil['Koltuk_id'] . '</td>';
        echo '<td>' . $biletilog['Durum'] . '</td>';
        echo '<td>' . $biletilog['Islem_tarihi'] . '</td>';
        echo '</tr>';
        $counter++;
    }
}
```

Sekil-47

6.5 Bilet Al

Bu sayfada kullanıcı bilet alma işlemine başlayabilir. Tek-yön veya gidiş-dönüş olarak mı bilet alacağını seçtiğten sonra nereden nereye gideceği şehir bilgilerini girdikten sonra tarih bilgisi seçip seferleri göster butonuna basarak işlemi gerçekleştirmektedir. Şekil-48’deki kod isterler doldurulduktan sonra kullanıcayı hangi kısma yönlendirileceğini göstermektedir. Tabloya şehir bilgileri ‘sehirler’ tablosundan çekilmektedir. Eğer admin öncede siteye sefer eklemiş se geçerli seferler gösterilmektedir. Ancak kullanıcıların istediği konumlara ve tarihlerde herhangi bir sefer bilgisi yoksa ekranда sefer bulunmadığına dair bilgi verilmektedir. Şekil-49’daki kod ile kullanıcıların girdiği verilere göre geçerli sefer kontrolü sağlanmaktadır.

```
<?php action='<?php echo site_url('/seferler'); ?>' method='post'
<div class="form-group">
    <div class="form-checkbox">
        <label for="gidisdonus">
            <input type="radio" id="gidisdonus" name="flight-type" value="gidisdonus" />
            <span>Gidiş - Dönüş</span>
        </label>
        <label for="tekyon">
            <input type="radio" id="tekyon" name="flight-type" value="tekyon" />
            <span>Tek Yön</span>
        </label>
    </div>
</div>
<div class="row">
<div class="col-md-6">
    <div class="form-group">
        <span class="form-label" style="color: #57a6a6;">Nereden:</span>
        <?php
use App\Models\UserModelSehirler;
$sehirModel = new UserModelSehirler();
$sehirler = $sehirModel->findAll();
        >>
        <select class="form-control" id="sehirNereden" name="sehirNereden" required>
            <option value="" selected disabled>Sehir Seçiniz</option>
            <?php
                foreach ($sehirler as $sehir) {
                    echo '<option value="' . $sehir['Plaka_kodu'] . '">' . $sehir['Sehir_adi'] . '</option>';
                }
            </?php
        </select>
    </div>
</div>
<div class="col-md-6">
    <div class="form-group">
        <span class="form-label" style="color: #57a6a6;">Nereden:</span>
        <?php
use App\Models\UserModelSehirler;
$sehirModel = new UserModelSehirler();
$sehirler = $sehirModel->findAll();
        >>
        <select class="form-control" id="sehirNereden" name="sehirNereden" required>
            <option value="" selected disabled>Sehir Seçiniz</option>
            <?php
                foreach ($sehirler as $sehir) {
                    echo '<option value="' . $sehir['Plaka_kodu'] . '">' . $sehir['Sehir_adi'] . '</option>';
                }
            </?php
        </select>
    </div>
</div>
</div>
```

Sekil-48

```

public function seferler()
{
    $model = new UserModel();
    $results = $model->where('Kalkis_sehir', $this->request->getVar('sehirNereden'))
        ->where('Varis_sehir', $this->request->getVar('sehirNereye'))
        ->findAll();
    $seferler = [];
    foreach ($results as $result) {
        $sefer = [
            'Sefer_id' => $result['sefer_id'],
            'Kalkis_sehir' => $result['Kalkis_sehir'],
            'Varis_sehir' => $result['Varis_sehir'],
            'Tarih' => $result['Tarih'],
            'Kalkis_saat' => $result['Kalkis_saat'],
            'Varis_saat' => $result['Varis_saat'],
            'Peron_no' => $result['Peron_no'],
            'Plaka' => $result['Plaka'],
            'Kapasite'=> $result['Kapasite'],
            'dolu_koltuk'=> $result['dolu_koltuk'],
            'bos_koltuk'=> $result['bos_koltuk'],
            'Fiyat'=> $result['Fiyat']
        ];
        $seferler[] = $sefer;
    }
    $data['seferler'] = $seferler;
    return view('voyage', $data);
}

```

Şekil-49

6.6 Sefer Seçimi

Bu sayfada kullanıcı girişlerine göre geçerli seferler listelenmektedir. Şekil-50'deki kod sefer listesi listelenmesini sağlamaktadır. İstenen seferin koltuk seç butonuna basarak koltuk seçim ekranına geçiş sağlanabilmektedir.

```

<?php else >
    <?php foreach ($seferler as $sefer): ?>
        <?php
            $seferTarihi = strtotime($sefer['Tarih']);
            $bugunTarihi = strtotime(date("Y-m-d"));
            if ($bugunTarihi < $seferTarihi): ?>
                <div class="service-content-inner d-flex align-items-center bg-white border border-primary rounded p-4 ps-0">
                    <img alt="Service icon p-4">
                    <div class="service-content">
                        <p>Sefer Tarihi: <span>Sefer Koltuk Bireni</span></p>
                        <div>Kalkış Şehri : </div><div>Varis Şehir : </div>
                        <div>Sefer Tarih : </div><div>Sefer Saati : </div>
                        <div>Sefer : </div><div>Sefer ID : </div>
                    </div>
                </div>
                <div class="d-flex align-items-center" style="margin-left: auto; margin-right: 20px;">
                    <div>Fiyat bilgisi </div>
                    <div class="me-3" style="margin-bottom: 35px; font-size: 26px; font-weight: bold;">
                        <?php echo $sefer['Fiyat'] ?>
                    </div>
                    <div>Koltuk Seç </div>
                </div>
            </?php endif ?>
        </div>
    <?php endforeach ?>

```

Şekil-50

6.7 Koltuk Seçimi

Koltuk seçimi için kullanıcıya birden çok seçenek vermiştir. Kişi biletini öğrenci ise öğrenci indiriminden yararlanarak, 65 yaş üstü ise bunun indiriminden, memur ise memur indiriminden, güvenlik gücü ise ücretsiz satın alabilmektedir. Satın alınan yanında ise kullanıcı istediği koltuğu bir müddet boyunca bir miktar ücret ödeyerek rezerve edebilmektedir.

6.7.1 Bakiye ile Öde

Eğer kullanıcı bakiye ile ödeme seçeneğini seçmiş ise ilk olarak kullanıcının bakiye bilgisi Şekil-51'deki kod ile kontrol edilir. Eğer bakiye miktarı bilet ücreti için yeterli ise Şekil-52'deki kod ile ödeme gerçekleştirilmektedir ve bakiye miktarı güncellenmektedir. Kullanıcı bakiye ile ödeme yaparak koltuk rezerve de edebilmektedir.

```

function kontrolEt() {
    var secilenFiyat = document.getElementById("biletFiyat").value;
    var bakiye = parseInt(document.getElementById("bakiye").value);

    if (bakiye < secilenFiyat) {
        document.getElementById("bakiyeodebtn").disabled = true;
        document.getElementById("bakiyeyetersiz").style.display = "block";
    } else {
        document.getElementById("bakiyeodebtn").disabled = false;
        document.getElementById("bakiyeyetersiz").style.display = "none";
    }
}

document.addEventListener("DOMContentLoaded", kontrolEt);
document.getElementById("biletFiyat").addEventListener("change", kontrolEt);

```

Şekil-51

```

$model = new UserModel();
$yanit=$model->where('Yolcu_id',$yolcuId)->first();
$bakiye=$yanit['Bakiye']-$biletFiyat;
$updateData=[
    'Bakiye'=>$bakiye
];

$model->where('Yolcu_id', $yolcuId)->set($updateData)->update();

if($biletFiyat>=300){
    $durum='Satın Alındı';
}
else{
    $durum='Rezerve';
}
$koltukModel=new UserModel(koltuklar());
$koltukData=[
    'yolcu_id'=>$yolcuId,
    'Durum'=>$durum
];
$koltukModel->where('Koltuk_no',$koltukId)->where('Sefer_id',$seferId)->set($koltukData)->update();

echo "Ödeme Yapıldı Yönünlendiriliyorsunuz...";sleep(2);
return view ('odendi', $data);

```

Şekil-52

6.7.2 Kart ile Öde

Kullanıcı bilet ve rezerve ücretini kart ile ödeme yapmak isterse kart ile ödeme butonuna basın kart bilgilerini girmesi gereken bir sayfa ekrana çıkmaktadır. 'kartodememe' fonksiyonu ile hangi yolcunun hangi koltuğu ve sefer için ödeme yapacağı kart ile ödeme sayfasına aktarılmaktadır. Kullanıcı kart bilgilerini girerek gerekli ücret ödenebilmektedir.

```

public function kartodememe()
{
    $yolcuId = $this->request->getVar('yolcuu');
    $seferId = $this->request->getVar('seferr');
    $koltukId = $this->request->getVar('seat');
    $pnrKodu = generatePNR(6);
    $biletFiyat = $this->request->getVar('biletFiyat');

    $data = [
        'yolcuId' => $yolcuId,
        'seferId' => $seferId,
        'koltukId' => $koltukId,
        'pnrKodu' => $pnrKodu,
        'biletFiyat' =>$biletFiyat
    ];
    return view('payment', $data);
}

```

Şekil-53

6.8 Müşteri Girişи

6.8.1 Bilgileri Güncelle

Kullanıcı veya admin profil bilgilerini yenilemek isterse 'Kullanıcı Bilgilerim' ekranından bilgilerini değiştirmektedir. Tüm isterleri boş bırakmamak üzere

bilgi değişimi sağlanabilmektedir. Tüm isterlerin doluluk durumu Şekil-55'deki 'bilgiKontrol()' fonksiyonu ile kontrol edilmektedir. Kullanıcı 'Bilgileri Güncelle' butonuna basarsa Şekil-56'daki 'bilgileriGuncelle' fonksiyonu çağrılmaktadır. Bu fonksiyon sayesinde kullanıcı bilgileri veritabanında güncellenerek kaydedilmektedir.

```
<div class="card">
    <form action="php echo base_url('/bilgileriGuncelle'); ?" method="post" onsubmit="return bilgikontrol()">
        <div class="form-section">
            <h2>Hesap Bilgileri</h2>
            <br><br>
            <div class="form-group">
                <div class="form-row">
```

Sekil-54

```
function bilgikontrol() {
    var newId = document.getElementById('newAdd').value;
    var newSoyad = document.getElementById('newSoyad').value;
    var newVas = document.getElementById('newVas').value;
    var newPosta = document.getElementById('newEmail').value;
    var newC = document.getElementById('newC').value;
    var newTelefon = document.getElementById('newTelefon').value;
    var newCinsiyet = document.getElementById('newCinsiyet').value;

    var errorMessages = "";

    if (newId === "" || newSoyad === "" || newPosta === "" || newC === "" || newTelefon === "") {
        errorMessages += "Lütfen alanları boş bırakmayınız.<br>";
    }
```

Sekil-55

```
public function bilgileriGuncelle()
{
    $session = session();
    $userData = $session->get('user');
    if (!$userData) {
        return redirect()->to('login');
    }
    $newId = $this->request->getVar('newAdd');
    $newSoyad = $this->request->getVar('newSoyad');
    $newVas = $this->request->getVar('newVas');
    $newCinsiyet = $this->request->getVar('newCinsiyet');
    $newEmail = $this->request->getVar('newEmail');
    $newC = $this->request->getVar('newC');
    $newTelefon = $this->request->getVar('newTelefon');

    $userData['Ad'] = $newId;
    $userData['Soyad'] = $newSoyad;
    $userData['Vas'] = $newVas;
    $userData['Cinsiyet'] = $newCinsiyet;
    $userData['Email'] = $newEmail;
    $userData['C'] = $newC;
    $userData['Telefon'] = $newTelefon;

    $model = new UserModel();

    $updateData = [
        'Ad' => $newId,
        'Soyad' => $newSoyad,
        'Vas' => $newVas,
        'Cinsiyet' => $newCinsiyet,
        'Email' => $newEmail,
        'C' => $newC,
        'Telefon' => $newTelefon
    ];
}
```

Sekil-56

6.8.2 Şifre Değiştir

Kullanıcı veya admin sterseler şifre değişikliği yapabilmektedir. Hesabım sayfasında bulunan şifre değiştir bölümü şifre güncellemeyi sağlamaktadır. 'Şifre Değiştir' butonuna basıldığında Şekil-57'deki form çalışmaktadır. Bu formda Şekil-58'deki 'sifreKontrol()' fonksiyonu ile kullanıcının eski şifresinin doğru girip girmediği ayrıca yeni şifresini iki kez girerek eşleşme durumu kontrol edilmektedir. Kullanıcının girdiği veriler Şekil-59'daki fonksiyona gönderilerek veritabanındaki eski şifre kontrolü sağlanır ve eğer doğrusa yeni şifre veritabanına güncelleme sağlanmaktadır.

```
<form action="php echo base_url('/sifreDegistir'); ?" method="post" onsubmit="return sifrekontrol()">
    <div class="form-section">
        <h2>Şifre Bilgileri</h2>
        <br><br>
        <div class="form-group">
            <div class="form-row">
                <label for="Password">Eski Şifre</label>
                <input type="password" id="oldPass" name="oldPass">
            </div>
            <div class="form-row">
                <label for="Password">Yeni Şifre</label>
                <input type="password" id="newPass" name="newPass">
            </div>
            <div class="form-row">
                <label for="Password">Yeni Şifre Yeniden:</label>
                <input type="password" id="reNewPass" name="reNewPass">
            </div>
```

Sekil-57

```
function sifrekontrol()
{
    var phOldPass = "<?php echo isset($session->user['sifre']) ? $session->user['sifre'] : '' ; ?>";
    var oldPass = document.getElementById('oldPass').value;
    var newPass = document.getElementById('newPass').value;
    var reNewPass = document.getElementById('reNewPass').value;
    var errorMessages = "";

    if (phOldPass != oldPass) {
        errorMessages += "Eski şifreniz yanlış girilmiştir.<br>";
    }

    if (oldPass === "" || newPass === "" || reNewPass === "") {
        errorMessages += "Lütfen tüm şifre alanlarını doldurunuz.<br>";
    }

    if (newPass != reNewPass) {
        errorMessages += "Yeni şifreler birbirile uyusmamaktadır.<br>";
    }

    var errorContainer = document.getElementById("errorMessage2");
    errorContainer.innerHTML = errorMessages;

    if (errorMessages !== "") {
        return false;
    } else {
        return true;
    }
}
```

Sekil-58

```
public function sifreDegistir()
{
    $session = session();
    $userData = $session->get('user');

    if (!$userData) {
        return redirect()->to('login');
    }

    $currentPassword = $this->request->getVar('oldPass');
    $newPassword = $this->request->getVar('newPass');
    $confirmPassword = $this->request->getVar('reNewPass');

    if ($userData['Sifre'] !== $currentPassword) {
        return redirect()->back()->with('error', 'Mevcut şifreniz yanlış.');
    }

    if ($newPassword !== $confirmPassword) {
        return redirect()->back()->with('error', 'Yeni şifreler eşleşmiyor.');
    }

    $model = new UserModel();
    $updateData = [
        'Sifre' => $newPassword
    ];
    $model->where('Yolcu_id', $userData['Yolcu_id'])->>set($updateData)->update();
    $userData['Sifre'] = $newPassword;
    $session->set('user', $userData);
    $successMessage1 = 'Şifreniz başarıyla güncellendi.';
    return view('hesabim', ['successMessage' => $successMessage1]);
}
```

Sekil-59

6.8.3 Bilet Bilgileri

Bu sayfada kullanıcının daha önceden satın almış ya da rezerve etmiş olduğu bilet bilgileri bulunmaktadır. Bu sayfada bilet bilgisi için gösterilen bilgiler: kalkış-varış şehir isimleri, PNR kodu, sefer tarihi, saat bilgisi, peron no, koltuk no, bilet durumu(satin alındı/rezerve), QR kod şeklinde edildir. Kullanıcı satın almış olduğu biletin sefer kalkış zamanına 1 gün kalaya kadar iptal etme hakkı bulunmaktadır. Aynı şekilde rezerve ettiği bilet içinde

sefere 2 gün kala iptal etme hakkı bulunmaktadır. Şekil-60 bilet bilgilerini ekranı yazdırın kod bulunmaktadırktadır.

```
</php>
<?php
use App\Models\UserBiletiBiletingView;
$biletingViewModel=new UserBiletiBiletingView();
$biletingView=$biletingViewModel->where('Yolu_Ad',$session->user['Ad'])->where('Yolu_Soyad',$session->user['Soyad'])->findAll();
if(empty($biletingView)){
    echo '<div><p style="color:red;">sayıtlı bilet bulunamadı!</p></div>';
}
else{
    if( !empty($message) ) {
        echo '<p id="errorMessage" style="color: green;">' . $message . '</p>';
    }
    foreach ($biletingView as $bilet) {
        echo '<div class="item">';
        echo '<form action="#" method="post">' . site_url('/askiyaval') . '</form>';
        echo '<div class="row">';
        echo '<div class="col-6">' . $bilet['Kalkis_sehir'] . '</div>' . $bilet['Varis_sehir'] . '</div>';
        echo '<div class="col-6">' . $bilet['Tarih'] . '</div>';
        echo '<div class="col-6">' . $bilet['Ucret'] . '</div>';
        echo '<div class="col-6">' . $bilet['Durum'] . '</div>';
        echo '</div>';
        echo '</div>';
    }
}
```

Şekil-60

6.8.3.1 Rezervasyon - Bilet İptal Et

Şekil-61 rezerve edilmiş biletin iptalini sağlayarak iptal edilecek biletin veritabanından silmektedir. Aynı şekilde kullanıcı aldığı biletin iptalini de bu şekilde sağlayabilmektedir.

```
public function rezervasyoniptal(){
    $db = db_connect();
    $biletid = $this->request->getVar('biletid');
    $bakiye = $this->request->getVar('bakiye');
    $procedureName = "Askiya_AI";
    $query = "CALL $procedureName(?)";
    $result = $this->db->query($query, [$biletid]);
    if ($result)
    {
        $messageIptal='Bilet iptal edilmiştir. Parametrik bakiyenize eklenmiştir.';
    }
    else
    {
        $error = $this->db->error();
        $messageIptal = "Hata oluştu: " . $error['message'];
    }
    return view('bileterim',[ 'message' => $messageIptal]);
}
```

Şekil-61

6.9 QR Bağlantısı

Projede, bilet PNR kodlarının QR kodlarına dönüştürülmesi için endroid/qr-code kütüphanesi kullanılmıştır. Bu kütüphane, yazı halinde verilen PNR bilet kodunu .svg uzantılı QR koduna dönüştürmektedir. QR kodları, bilgilerin hızlı ve güvenli bir şekilde taranılmasını sağlayarak bilet bilgilerinin mobil cihazlarda kolayca erişilebilir olmasını sağlar. Bu yaklaşım, kullanıcıların fiziksel biletleri taşımak yerine dijital bir biçimde bilet bilgilerine erişebilmelerini sağlayarak kullanıcı deneyimini artırır. Bu sayede, seyahat edenlerin bilet bilgilerine her zaman ve her yerde kolayca ulaşmaları mümkün olur.

```
$qr_Bilgi='PNR Kodu: '.$bilet['PNR_kodu']. "\r\n" . 'Saat: '.$bilet['Kalkis_Saat']. "\r\n".
$qr_code=QrCode::create($qr_Bilgi)
->setSize(100);
$writer=new SvgWriter;
$result=$writer->write($qr_code);

header("Content-Type:".$result->getMimeType());
echo '<div id="qrCodeContainer">';
echo $result->getString();
echo '</div>';
echo '<input type="hidden" name="biletid" id="biletid" value="'.$bilet['Bilet_id'].'">' ;
echo '<input type="submit" value="Bileti İptali Et">';


```

Şekil-62

Dönüştürülmüş qr kod tarandığı zaman, PNR bilgisi, yolcu ismi gibi bilgiler vermektedir.

6.10 E-mail Gönderimi (Kayıt Olan kullanıcılar için)

Kullanıcı kayıt ol ekranında kayıt ol butonuna tıklandığında kayıt olunan mail adresine kayıt olunduguuna dair bir mail gelmektedir. Şekil-63 mail gönderilmesini sağlamaktadır.

```
if(!empty($_POST['kaydet'])) {
    $ad = $_POST['Ad'];
    $soyad = $_POST['Soyad'];
    $Email = $_POST['Email'];
    $toEmail = 'umuttepeturizm@outlook.com';

    $mailHeaders =
    "Vn Merhaba " . $ad . " ." . $soyad . ", Vn kaydınız başarıyla yapılmıştır. Eğer kayıt işlemini
    yapmışsınız siz de lütfen bizimle irtibata geçiniz. Vn Saygıları, Umuttepe Turizm";
    if(mail($toEmail, $ad, $mailHeaders)) {
        $message = "Kaydınız başarıyla yapılmıştır. Lütfen e-postanızın kontrol ediniz.";
    }
}
```

Şekil-63

7. Canlı Sunucu

Proje, bir canlı sunucuya CPanel hosting hizmeti kullanılarak alınmıştır. Bu hizmet, kullanıcılarla aylık ücretlendirme modeliyle sunulmaktadır.

CPanel yönetim paneline başarılı bir şekilde giriş yaptıktan sonra, dosya yönetici aracı projenin ana dizininde bulunan bütün dosyalar "public_html" adlı dizine yüklenir. Bu adım, web sitesinin ana içeriğinin sunucuya yerleştirilmesini sağlar.

Veritabanı gereksinimlerini karşılamak için, cPanel içerisinde phpMyAdmin aracı kullanılarak ".sql" uzantılı veritabanı dosyası sunucuya yüklenir. Bu dosya, web uygulamasının veritabanı yapısını içerir ve phpMyAdmin üzerinden sunucuya aktarılır.

Dosya ve veritabanı yükleme işlemlerinin tamamlanmasının ardından, raporda belirtilen adımlar doğrultusunda "6.2" numaralı kısımda bulunan veritabanına bağlantı kurulur. Bu adım, web uygulamasının veritabanıyla iletişim kurmasını sağlayacak bağlantı bilgilerinin doğru şekilde yapılandırılmasını içerir.

Tüm bu adımların başarıyla tamamlanmasının ardından, proje canlı sunucuda erişime açılır ve web sitesi kullanıcılar tarafından ziyaret edilebilir hale gelir.

8. Karşılaşılan Problemler

Codeigniter 4, hala geliştirilmekte olan bir framework olduğundan, kaynak bulma konusunda Codeigniter 3'e kıyasla zorluklar yaşadık. Özellikle kendi web sitesinde bulunan dokümantasyonlar yeterli olmadı ve araştırma sürecimizi oldukça baltaladı. Bu durum, belirli işlevleri gerçekleştirmek için gereken bilgilerin eksiksliğiyle ve bazen de karmaşıklıkla mücadele etmemimize neden oldu. Aynı şekilde composerin doğrulmuş olduğu sürüm sorunları da projenin çalışma sürecinde zorluklar çıkardı.

Google haritalar ve ödeme apileri ekstra ücretlendirmeler veya banka hesapları istediği için projemize dahil edemedik.

Proje isterleri arasında projemiz canlı sunucuda çalışmadığı müddetçe değerlendirmeye alınmayacağı hakkında bir ister olduğu için ve hiçbir bedava hosting hizmeti projemizi çalıştırılmaya yetmediği için cpanel hosting ve domain almak zorunda kaldık.

9. Sonuç

Belirlenen problem tanımı ve amaç doğrultusunda CodeIgniter4 kullanılarak Umuttepe Turizm için web sitesi oluşturulmuştur. Projenin temel amacı olan kullanıcıların otobüs seyahatlerini planlamaları, uygun seferlere erişebilmeleri, bilet alımlarını ve iptallerini kolayca gerçekleştirmeleri ve seyahatleriyle ilgili gerekli bilgilere ulaşabilmelerini sağlayacak kullanıcı dostu bir web sitesi geliştirme amacına ulaşılmıştır.

Geliştirilen platform kullanıcıların seyahatlerini hızlı ve kolay bir şekilde planlamalarına olanak tanırken, kullanıcı dostu bir arayüz sunarak karmaşaklığını en aza indirdi. Kullanıcılar otobüsün içine kuşbakışı bakarak koltuk seçimlerini yapabilme, kredi kartı ile ödeme, rezervasyon seçeneklerini kullanabilme ve öğrenci indirimleri gibi avantajlardan yararlanabilmektedir.

Ayrıca, PNR kodu oluşturarak kullanıcılarla biletlerine kolayca erişebilmeleri için e-posta yoluyla bilgilendirme sağlanmaktadır. İptal edilen biletlerin ücreti, kullanıcıların hesap bakiyelerine otomatik olarak eklenmekte ve kullanıcılar bu bakiyeyi yeni bilet alımlarında kullanabilmektedirler.

Sonuç olarak, geliştirilen platform, kullanıcı dostu arayüzü, güvenli ödeme seçenekleri ve kapsamlı rezervasyon imkanları ile hedefine ulaşmıştır.

Referanslar

- <https://themewagon.com/themes/travela/> [21.03.24]
- https://www.codeigniter.com/user_guide/intro/index.html [21.03.24]
- <https://github.com/android qr-code> [21.03.24]