



RENT A CAR RAPOR

Hazırlayanlar

İsmail Arif Güleç
Betül Daşçı
Salih Enes Özdel
Şeyma Çıldır
Berke Tangör
Ahmet Akif Kasım

EKİM 2023

Proje Hakkında

Proje Görevleri

Araç Kayıtları

- Sistemde kayıtlı araçların detayları tutulacak, bu detaylar şunları içerecek:
 - Araç plakası
 - Marka
 - Model
 - Yıl
 - Günlük kira bedeli
 - Durumu
(kullanılabilir, kirada, serviste)

Müşteri Yönetimi

- Müşteri bilgileri tutulacak, bu bilgiler şunları içerecek:
 - Ad
 - Soyad
 - E-posta
 - Telefon numarası
 - Kimlik bilgileri
- Müşterilerin kaydı, düzenlenmesi, silinmesi ve listelenmesi mümkün olacak.

Rezervasyonlar

- Müşteriler araçları rezerve edebilecek.
- Rezervasyonlar tarih ve saat bazında tutulacak.
- Müşterilere kiralama başlangıç ve bitiş tarihleri seçme olanağı sunulacak.
- Araçların rezervasyon durumu (boş, rezerve edilmiş, kirada) gösterilecek.

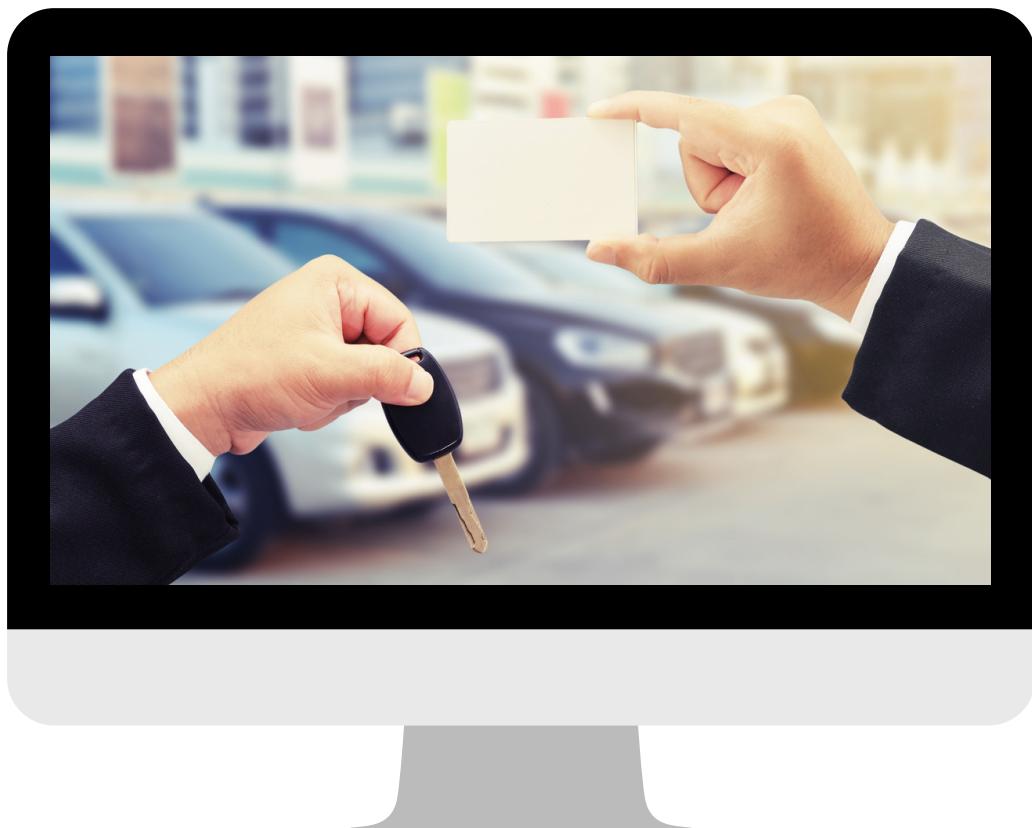
Kiralama İşlemi

- Müşteriler araçları kiralamak için rezervasyon yapabilecek veya anlık kiralamalar yapabilecek.
- Kiralama işlemi sırasında müşteri bilgileri ve araç bilgileri kaydedilecek.
- Kiralama süresine göre ücret hesaplanacak ve ödeme işlemi gerçekleştirilecek.

Proje Hakkında

Proje Özeti

Hazırlanmış olan Rent A Car Web Sitesi; insanların rahatça araba kiralayabildiği bir platform olarak hazırlanmıştır. Web Sitende; üyelik açma, üyeliğe giriş, tarih aralığı seçip araba kiralama , ödeme gibi fonksiyonlar bulunmaktadır. Web sitemizde arabaların müsaitlik durumu görüntülenebilmektedir. Web Sitesi mobil uyumlu olarak ayarlanmıştır. Web Sitesi hazırlanırken İsmail Arif Güleç'in hazırlamış olduğu CSS kütüphanesi referans olarak kullanılmıştır. Rent A Car Web Sitesi Grup 5 üyeleri tarafından 4 Ekim – 9 Ekim 2023 tarihleri arasında geliştirilmiştir.



Proje İlerlemesi

Birinci Gün / 04.10.2023

İlk gün, bize atanan görev konusuyla ilgili ekip arkadaşları bir araya gelinmiş ve araç kiralama sektörünün dinamiklerini daha iyi anlamak amacıyla benchmark çalışması yapılmıştır. Bu kapsamlı çalışma çerçevesinde, diğer araç kiralama şirketlerinin web siteleri, sunulan ürünler ve fiyat politikaları detaylı bir şekilde incelenmiştir.

Yapılan incelemeler sonucunda, projenin geliştirilmesi ve tasarımın optimize edilmesi için en uygun yol olarak İsmail Arif'in CSS kütüphanesinin kullanılmasına karar verildi.

Görev dağılımı yapılrken, her bir ekip üyesinin projeye en iyi şekilde katkı sağlayabileceği titizlikle değerlendirildi. Bu, her bir üyenin güçlü yönlerinin projenin başarısını maksimize etmek için kullanılmasına olanak tanındı.

Bu başlangıç aşamasında, ekip olarak birlikte çalışmanın ve farklı beceri ve bakış açılarının projenin hızlı gelişimine nasıl katkı sağlayabileceğini vurgulamak istedik. Bu, projenin en etkili şekilde ilerlemesine yardımcı oldu.



İlerleme Aşaması

Proje İlerlemesi

İkinci Gün / 05.10.2023

Perşembe günü iş çıkışında Rent A car Uygulaması yapmaya başlandı. Görev dağılımları yapıldı. Front-end bölümünde web sitesi iskeleti oluşturuldu.

- Header
- Main Body
- Footer

Header içerisinde butonlar tanımlandı. Main Body içerisinde rezervasyon tarih girdisi ve araba kart yapıları oluşturuldu. Footer bölümünde sayfanın haritası, şirketin sosyal medya hesapları ve şirketin bulunduğu konum bilgisi eklendi. Web Sitesinin tasarımları tamamlanmış oldu.

Üçüncü Gün / 06.10.2023

Web sitesinin tasarımı tamamlandıktan sonra butonların aktif çalışması için JS kullanılması gerekiyordu.

Js kullanılarak;

- Müşteri Kaydı, Düzenlenmesi, Silinmesi, Listelenmesi
 - Admin Giriş,
 - Araç Bilgilerinin sisteme eklenmesi,
- İşlemleri yapıldı.



Proje İlerlemesi

Dördüncü Gün 07.10.2023

Ekipçe Rami kütüphanesinde toplanıldı ve JS kullanılarak yapılanlar;

- Giriş ve kayıt olma formlarının ve butonlarının aktifleştirilmesi ,
- Araçların rezervasyon işlemleri, yapıldı.

Beşinci Gün 08.10.2023

4 ekip üyesi Rami kütüphanesinde 2 ekip üyesi online katılım ile toplantıya başlanıldı. Bir yandan raporlama işlemlerine giriş yapıldı. Raporun iskeleti oluşturuldu. Bir yandan da JS kullanarak gruptan istenilen sıradaki görevlere başlanıldı. JS kullanılarak yapılanlar;

- Kiralama işlemleri,
- Ödeme İşlemleri,

tamamlanarak web sitesi hazır hale getirildi.



Proje Rapor Teslimi Ve Sunum Günü

Altıncı Gün 09.10.2023

Rent A Car web sitesi projemiz için kodlama aşaması ve test süreçleri tamamlanmıştır. Kaynak kod ve müşteriye teslim edilecek dokümanlar son kez gözden geçirilerek proje teslimi gerçekleştirılmıştır.

Projede yer alan modüllerin ve proje özetinin açıklandığı; proje geliştirme aşamasında gün gün yapılan geliştirmelerin detaylarıyla proje raporu nihai hale getirilmiş ve Grup 5 üyeleri ile birlikte proje sunumuna çalışılmıştır.

Yedinci Gün 10.10.2023

Web sitesinde yer alan admin paneli, müşteri bilgilerinin tutulduğu müşteri yönetimi, rezervasyon/kiralama işlemleri ve araçların müsaitlik durumuna göre tutulduğu araç kayıtları modülleri tanıtılmış ve sitenin nasıl kullanılacağına ilişkin eğitim verilmiştir.



Proje Zaman Çizelgesi



Detaylı zaman çizelgesi için lütfen
yukarıdaki karekodu okutun!



HTML İskelet Yapısı

CSS Linkleri

```
link rel="stylesheet" href="./assets/css/style.css">
link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.2/css/all.min.css"
integrity="sha512-z3gLpd7yknf1YoNbCzqRKc4qyor8gaKU1qm+nCShxbuBusANI9QpRohGBreCFkKxLhei6S9CQXFebbKuqlg0DA=="
crossorigin="anonymous" referrerpolicy="no-referrer" />
```

Daha önce hazırlanmış olan CSS kütüphanesi link içerisinde çağrılmıştır. Ayrıca navbar ve form yapılarında kullandığımız iconları eklemek için fontawesome kütüphanesini de HTML yapısında kullanıldı..

Navbar

```
<div class="navbar-right p4 bg-red row">
  <ul class="nav-menu col-8">
    <li class="nav-item">
      <a class="nav-link" href="#"> Anasayfa</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#"> Hakkımızda</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#"> Galeri</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#"> İletişim</a>
    </li>
  </ul>
  <ul id="loginSignup" class="nav-menu navbar-right col-4">
    <li id="login" class="nav-item">
      <a id="login-link" class="nav-link" href="#"> Giriş</a>
    </li>
    <li id="signup" class="nav-item">
      <a id="signup-link" class="nav-link" href="#"> Kayıt Ol</a>
    </li>
  </ul>
</div>
```

Navbar kısmında şirketimizn anasayfası, hakkımızda, iletişim ve galeri linkleri bulunmakla beraber yanında kullanıcının giriş yapması için login, kayıt olması için signup linkleri bulunmaktadır.

Rezervasyon Tarih Formu

```
<div class="row rezervasyon">
  <div class="col-6-11 center">
    <h2>Rezervasyon Tarihi</h2>
    <form method="get">
      <label for="startDate">Başlangıç Tarihi:</label>
      <input
        class="bg-gray"
        type="date"
        id="startDate"
        name="startDate"
      >
      <label for="endDate">Bitiş Tarihi:</label>
      <input
        class="bg-gray"
        type="date"
        id="endDate"
        name="endDate"
      >
    </form>
    <button type="submit" id="tarihbtn">Hesapla</button>
  </div>
</div>
```

Rezervasyon tarih formu içerisinde “date” değerleri input etiketi ile alınmak üzere tasarlanmıştır. Ayrıca her bir inputa başlık vermek adına `<label>` etiketleri kullanılmıştır. Form hazırlanırken CSS kütüphanesinde bulunan grid yapısı kullanılmıştır.

Araç Kart Yapısı

```
<!--ARABA LİSTELE-->
<div id="row" class="row arabaListele">
</div>
```

Araçların tutulacağı alan “row” id’sine sahip bir `<div>` olarak tasarlanmıştır. Gelecek elementler scriptte “createElement” metodu kullanılarak yazılmıştır. Bu `<div>` HTML sayfasına çektirmek için hazırlanmıştır.

Müşteri Kart Yapısı

```
<!--MÜSTERİ LİSTELE-->
<div class="row">
    <table class="col-12 border="">
        <thead>
            <tr>
                <th>İsim</th>
                <th>Soyisim</th>
                <th>Email</th>
                <th>Şifre</th>
                <th>Telefon</th>
                <th>Tc No</th>
                <th>Kiraladığı Araba</th>
                <th>Kiraladığı Zaman</th>
                <th>Settings</th>
            </tr>
        </thead>
        <tbody id="musteriListe">
            <!--Müşterilistsi-->
        </tbody>
    </table>
</div>
```

Müşteri bilgileri HTML içerisinde hazırlanan tablo yapısında tutulacaktır. Burada tablonun başlık kısmı yani <thead> etiketi içerisinde müşteri bilgilerinin hangi formatta olacağı belirtilmiştir. Alt kısımda ise (<tbody>) script dosyasından “createElement” metodu ile müşteriden alınan bilgiler bulunmaktadır.

```
<!--Müşteri Listele Butonu-->
<div class="row">
| | <button id="musteriListesi">Müşteri Listele</button>
</div>
```

Sayfaya yeni kayıt yapan müşteri ve/veya halihazırda sistemde kaydı bulunan müşterinin listelenmesi belirtilen butona basınca sağlanmaktadır.

Müşteri Düzenleme Formu

```
<!-- Düzenleme Formu -->
<div id="duzenleForm" style="display: none;">
<h2>Müşteri Düzenle</h2>
<input type="hidden" id="duzenlenecekMusteriIndex">
<input type="text" id="duzenleIsim" placeholder="İsim">
<input type="text" id="duzenleSoyisim" placeholder="Soyisim">
<input type="email" id="duzenleEmail" placeholder="Email">
<input type="password" id="duzenleSifre" placeholder="Şifre">
<input type="text" id="duzenleTelefon" placeholder="Telefon">
<input type="text" id="duzenleTc" placeholder="TC Kimlik No">
<button type="button" id="duzenleKaydet">Kaydet</button>
<button type="button" id="duzenleIptal">İptal</button>
</div>
```

Müşteri düzenlemek için sistemde kaydı bulunan müşterilerin her bir bilgi içeren girdisini değiştirebilmesi adına `<input>` etiketleri ile gerekli olan değerler alınır. Alınan bu değerler script dosyasında düzenlenmektedir.

Giriş - Signup Formu (Popup)

```
<div id="login-window">
  <form method="get" class="bg-purple">
    <ul class="form-container">
      <li class="form-item btn-bg-pink">
        <div class="form-img ms4 btn-bg-white">
          | <i class="fa-solid fa-user"></i>
        </div>
        <div class="form-input">
          | <input id="yeni_email" type="text" placeholder=" Email" name="Email" />
        </div>
      </li>

      <li class="form-item btn-bg-pink">
        <div class="form-img ms4 btn-bg-white">
          | <i class="fa-solid fa-lock"></i>
        </div>
        <div class="form-input">
          | <input id="yeni_password" type="password" placeholder=" Şifre" name="sifre" />
        </div>
      </li>
    </ul>

    <input
      | class="submit-btn ft-poppins text-3 btn-bg-pink"
      | type="button"
      | value="Submit"
      | id="login-button"
    />
  </form>
</div>
```

Bu kod parçasında giriş formu tasarlanmıştır. Formda da bilgiler input ile alınarak buton ile gönderilmektedir.

Ödeme Formu (Popup)

```
<div class="payment-form" id="odeme-window">
    <h2>Ödeme Yap</h2>
    <form method="get" class="odeme-form">
        <ul class="form-container">
            <li class="form-item btn-bg-pink">
                <div class="form-input">
                    <input
                        id="cardNumber"
                        type="text"
                        placeholder=" Kart Numarasını Girin"
                        name="Email"
                    >
                </div>
            </li>
            <li class="form-item btn-bg-pink">
                <div class="form-input">
                    <input
                        id="expirationDate"
                        type="password"
                        placeholder=" MM/YY"
                        name="sifre"
                    >
                </div>
            </li>
            <li class="form-item btn-bg-pink">
                <div class="form-input">
                    <input
                        id="cvv"
                        type="password"
                        placeholder=" CVV"
                        name="sifre"
                    >
                </div>
            </li>
        </ul>
        <input
            class="submit-btn ft-poppins text-3 btn-bg-pink"
            type="button"
            value="Submit"
            id="odeme-button"
        >
    </form>
</div>
```

Bu kod parçasında ödeme ekranı HTML iskeleti hazırlanmıştır. CSS kütüphanesi içerisinde form yapıları kullanılmıştır.

JavaScript

```
class System {
    constructor() {
        this.araba = [];
        this.musteri = [];
    }
    addAraba(item){
        this.araba.push(item);
    }
    addMusteri(item){
        this.musteri.push(item)
    }
    getMyAraba(){
        return this.araba;
    }
    getMyMusteri(){
        return this.musteri;
    }
}
```

Yukarıdaki modülde “System” sınıfı oluşturulmuştur. Sınıfın amacı arabaları ve müşterileri yönetmektir.

addAraba(item) ve addMusteri(item) metotları, arabaları ve müşterileri eklemek için kullanılır.

getMyAraba() ve getMyMusteri() metotları, araba ve müşteri dizisini döndürür ve sonrasında almak için kullanılır.

Bu modül, arabaları ve müşterileri saklar ve bu verilere erişim sağlar.

```
23  class Araba {
24    constructor(plaka, marka, model, yil, kiraBedeli, durum, resim) {
25      this.plaka = plaka;
26      this.marka = marka;
27      this.model = model;
28      this.yil = yil;
29      this.kiraBedeli = kiraBedeli;
30      this.resim = resim;
31      this.durum = durum;
32      this.arabalar = [];
33    }
34    addAraba(item) {
35      this.arabalar.push(item);
36    }
37    getMyAraba() {
38      return this.arabalar;
39    }
40    updateAraba(marka, yeniDurum) {
41      const arabaIndex = this.arabalar.findIndex(
42        (araba) => araba.marka === marka
43      );
44      if (arabaIndex !== -1) {
45        this.arabalar[arabaIndex].durum = yeniDurum;
46      }
47    }
48 }
```

Araba Modülü:

Araba sınıfı, araba nesnelerini oluşturmak için kullanılır. Araba nesneleri plaka, marka, model, yıl, kira bedeli, durum ve resim bilgilerini içerir. Bu modül, araba nesnelerini oluşturur.

```

49  class Musteri {
50    constructor(isim, soyisim, email, sifre, telefon, tc) {
51      this.isim = isim;
52      this.soyisim = soyisim;
53      this.email = email;
54      this.sifre = sifre;
55      this.telefon = telefon;
56      this.tc = tc;
57      this.kiraladigiAraba = "Yok";
58      this.kiraladigiTarih = "-----";
59      this.musteriler = [];
60    }
61    addMusteri(item) {
62      this.musteriler.push(item);
63    }
64    getMyMusteri() {
65      return this.musteriler;
66    }
67    updateMusteri(isim, yeniKiraladigiAraba, yeniKiraladigiTarih) {
68      const musteriIndex = this.musteriler.findIndex(
69        (musteri) => musteri.isim === isim
70      );
71      if (musteriIndex !== -1) {
72        this.musteriler[musteriIndex].kiraladigiAraba = yeniKiraladigiAraba;
73        this.musteriler[musteriIndex].kiraladigiTarih = yeniKiraladigiTarih;
74      }
75    }
76    deleteMusteri(isim) {
77      const index = this.musteriler.findIndex((musteri) => musteri.isim === isim);
78      if (index !== -1) {
79        this.musteriler.splice(index, 1);
80      }
81    }
82  }

```

Müşteri Modülü:

Musteri sınıfı, müşteri nesnelerini oluşturmak için kullanılır. Müşteri nesneleri isim, soyisim, e-posta, şifre, telefon ve TC kimlik numarası gibi bilgileri içerir. Bu modül, müşteri nesnelerini oluşturur.

```
const customer = new Musteri();
const cars = new Araba();
const myObject = new System();
```

Burada müşteriler, arabalar ve objeler için değişkenler atandı.

```
const musteri0 = new Musteri(
    "Admin",
    "Admin",
    "admin@gmail.com",
    "admin",
    "admin123",
    "000000000000"
);
const musteri1 = new Musteri(
    "Berke",
    "Tangör",
    "berke@gmail.com",
    "sifre",
    "11111111111",
    "11111111111"
);
```

```
customer.addMusteri(musteri0);
customer.addMusteri(musteri1);
customer.addMusteri(musteri2);
customer.addMusteri(musteri3);
```

```
const araba1 = new Araba(
    "26mrc45",
    "Mercedes",
    "Benz CLS",
    "2021",
    "2500",
    "Kullanılabilir",
    "./assets/images/mercedes1.jpg"
);
const araba2 = new Araba(
    "34mrc87",
    "Mercedes",
    "AMG GT",
    "2011",
    "1500",
    "Serviste",
    "./assets/images/mercedes2.jpg"
);
const araba3 = new Araba(
    "41audi",
    "Audi",
    "A5",
    "2016",
    "2500",
    "Kullanılabilir",
    "./assets/images/audi1.jpg"
);
const araba4 = new Araba(
    "61audi1",
    "Audi",
    "A6",
    "2021",
    "2500",
    "Kullanılabilir",
    "./assets/images/audi2.jpg"
);
```

```
const araba5 = new Araba(
    "44volvo63",
    "Volvo",
    "V40",
    "2017",
    "1800",
    "Serviste",
    "./assets/images/volvo1.jpg"
);
const araba6 = new Araba(
    "45fiat45",
    "Fiat",
    "500",
    "2017",
    "500",
    "Rezerve",
    "./assets/images/fiat1.jpg"
);
const araba7 = new Araba(
    "35volvo45",
    "Volvo",
    "240",
    "2015",
    "2500",
    "Kullanılabilir",
    "./assets/images/volvo2.jpg"
);
const araba8 = new Araba(
    "37fiat85",
    "Fiat",
    "500",
    "1975",
    "700",
    "Rezerve",
    "./assets/images/fiat2.jpg"
);
```

Burada da tek tek atamasını yapmış olduğumuz 8 araç değişkenimizin marka, model, yıl, kiralama bedeli ve kiralama ücreti eklendi.

```
cars.addAraba(araba1);
cars.addAraba(araba2);
cars.addAraba(araba3);
cars.addAraba(araba4);
cars.addAraba(araba5);
cars.addAraba(araba6);
cars.addAraba(araba7);
cars.addAraba(araba8);
```

Burada da tek tek atamasını yapmış olduğumuz 8 araç değişkenimizin marka, model, yıl, kiralama bedeli ve kiralama ücreti eklendi.

```
const main = document.getElementById("row");
main.className = "row gap";

function arabaFonksiyon(sonuc) {
    main.innerHTML = "";
    cars.getMyAraba().forEach((araba) => [
        //başlık Yapısı
        const card = document.createElement("div");
        card.className = "col-4 col-sm-6 mt0 card";
        const cardItem = document.createElement("div");
        cardItem.className = "row";
        let baslik = document.createElement("h2");
        baslik.className = "col-6-11";
        baslik.textContent = araba.marka + " " + araba.model;
        const imageDiv = document.createElement("div");
        imageDiv.className = "image-height-center col-6-11";
        let image = document.createElement("img");
        image.src = araba.resim;
        cardItem.appendChild(baslik);
        cardItem.appendChild(imageDiv);
        card.appendChild(cardItem);
        sonuc.appendChild(card);
    ]);
}
```

Kartın Ana Yapısının Oluşturulması:

- card adında bir div ögesi oluşturulur ve bu kartın ana bileşenini temsil eder.
- card ögesine col-4 col-sm-6 mt0 card sınıfları atanır. Bu sınıflar kartın stilini ve görünümünü belirler.

```
//Table Yapısı
const tableSection = document.createElement("div");
tableSection.className = "navbar-center";
const tableDiv = document.createElement("div");
tableDiv.className = "table-responsive";
const table = document.createElement("table");
const tbody = document.createElement("tbody");
const tr1 = document.createElement("tr");
const tr2 = document.createElement("tr");
const tr3 = document.createElement("tr");
const tr4 = document.createElement("tr");
const tr5 = document.createElement("tr");
const tr6 = document.createElement("tr");
let markaTd = document.createElement("td");
markaTd.textContent = "Marka:";
let markaTextTd = document.createElement("td");
markaTextTd.textContent = araba.marka;
let modelTd = document.createElement("td");
modelTd.textContent = "Model:";
let modelTextTd = document.createElement("td");
modelTextTd.textContent = araba.model;
let yilTd = document.createElement("td");
yilTd.textContent = "yıl:";
let yilTextTd = document.createElement("td");
yilTextTd.textContent = araba.yil;
let plakaTd = document.createElement("td");
plakaTd.textContent = "Plaka:";
let plakaTextTd = document.createElement("td");
plakaTextTd.textContent = araba.plaka;
let kiraBedeliTd = document.createElement("td");
```

Başlık öğesinin metni, arabaya ait marka ve model bilgilerini içerir, imageDiv adında bir div öğesi oluşturulur. image adında bir img öğesi oluşturulur ve bu resmin görüntüsünü temsil eder. Kartın sağ tarafında bir tablo yapısı oluşturulur. tableSection, tableDiv, table, ve tbody adında sırasıyla bir div, div, table, ve tbody öğeleri oluşturulur. Bu öğeler, tablonun düzenini ve içeriğini tanımlar. tr1, tr2, ..., tr6 adında altı farklı tr (tablo satırı) öğesi oluşturulur. Bu satırlar, arabaya ait bilgileri içerir. Her satırın içine td öğeleri (tablo hücresi) eklenir ve bu hücrelere sırasıyla "Marka:", "Model:", "Yıl:", "Plaka:", "Kira Ücreti:" metinleri eklenir. İlgili bilgileri içeren td öğeleri de oluşturulur ve bu bilgilerin içeriği bu td öğelerine eklenir.

```

if (sonuc != NaN) {
    kiraBedeliTextTd.innerHTML =
    | Number(araba.kiraBedeli) * sonuc + " " + "TL";
} else {
    kiraBedeliTextTd.innerHTML = araba.kiraBedeli + "TL";
}
let durumTd = document.createElement("td");
durumTd.textContent = "Durumu:";
let durumTextTd = document.createElement("td");
durumTextTd.textContent = araba.durum;
let kiralabuton = document.createElement("button");
kiralabuton.type = "button";
kiralabuton.className = "btn-out-white mb2";
kiralabuton.textContent = "Kirala";
kiralabuton.addEventListener("click", () => {
    kiralamaYapilanAraba(araba);
    console.log(araba.durum);
    if (dateGiris.value && dateBitis && hesaplaButonuCheck == true) {
        if (durumTextTd.textContent == "Kullanılabilir") {
            cars.updateAraba(araba.marka, "kiralandı");
            durumTextTd.textContent = araba.durum;
        } else {
            alert("Araç kiralananamaz " + durumTextTd.textContent);
        }
    } else {
        alert("Tarih Seçiniz ve Hesapla Butonuna Basınız");
    }
});

```

Bu kod bölümü, araba bilgilerini içeren kartları oluşturur. Her kart, arabanın marka, model, yıl, plaka ve kiralama ücreti gibi bilgilerini içerir. Kira ücreti, günlük kira bedeli ve kiralama süresi baz alınarak hesaplanır. Eğer kiralama işlemi uygun koşullarda ise "Kirala" düğmesi aracılığıyla gerçekleştirilir. Bu işlem, aracın durumunu güncelleyerek "Kullanılabilir"den "Kiralandı"ya çevirir. Ayrıca, tarih seçimi ve hesaplama butonu kontrol edilir. Arabanın durumu "Kullanılabilir" değilse, kullanıcıya uygun bir uyarı gösterilir. Bu kod parçası, araç listesini görüntülemek ve kiralama işlemi yapmak için önemlidir.

```
    main.appendChild(card);
    card.appendChild(cardItem);
    cardItem.appendChild(baslik);
    cardItem.appendChild(imageDiv);
    imageDiv.appendChild(image);
    card.appendChild(tableSection);
    tableSection.appendChild(tableDiv);
    tableDiv.appendChild(table);
    table.appendChild(tbody);
    tbody.appendChild(tr1);
    tbody.appendChild(tr2);
    tbody.appendChild(tr3);
    tbody.appendChild(tr4);
    tbody.appendChild(tr5);
    tbody.appendChild(tr6);
    tr1.appendChild(markaTd);
    tr1.appendChild(markaTextTd);
    tr2.appendChild(modelTd);
    tr2.appendChild(modelTextTd);
    tr3.appendChild(yilTd);
    tr3.appendChild(yilTextTd);
    tr4.appendChild(plakaTd);
    tr4.appendChild(plakaTextTd);
    tr5.appendChild(kiraBedeliTd);
    tr5.appendChild(kiraBedeliTextTd);
    tr6.appendChild(durumTd);
    tr6.appendChild(durumTextTd);
    tableSection.appendChild(kiralabuton);
  });
}
```

Bu kod bölümü, bir HTML sayfasında araba bilgilerini içeren kartlar oluşturur. Her kart, arabanın marka, model, yıl, plaka ve kiralama ücreti gibi bilgilerini içerir. Kartlar, sayfanın ana bölümüğe eklenir ve görsel olarak düzenlenir. Her kartın içeriği, bir başlık, resim, ve bir tablo içerir. Tablo, araba bilgilerini düzenli bir şekilde gösterir. Bu kod aynı zamanda her kartın altında bir "Kiral" düğmesi ekler. Tıklanabilir düğme, araba kiralama işlemini başlatır. Tüm bu işlemler, araçların listesini görsel olarak oluşturarak kullanıcıların kolayca araba kiralamasına olanak tanır.

```

const musterilisteleButton = document.getElementById("musterilistesi");
musterilisteleButton.addEventListener("click", function () {
    let kiralanacakOlanAraba = myObject.getMyAraba()[0].marka;
    let kiralayanMüsteri = myObject.getMyMusteri()[0].isim;

    customer.updateMusteri(
        kiralayanMüsteri,
        kiralanacakOlanAraba,
        dateGiris.value + dateBitis.value
    );
    console.log(customer.musteriler);
    console.log(kiralayanMüsteri);
    musterilistele(kiralanacakOlanAraba, kiralayanMüsteri);
});

```

Bu kod, bir araba kiralama işlemi gerçekleştirildiğinde ve bir müşteri arabayı kiraladığında çalışır. Bu işlem, kiralayan müşteriyi ve kiralanacak araba bilgisini günceller ve bu bilgilerle birlikte müşteri listesini yeniden oluşturur.

```

function musterilistele(kiralananAraba, kiralayanMüsteri) {
    const musteriliste = document.getElementById("musteriliste");
    musteriliste.innerHTML = "";
    customer.getMyMusteri().forEach((musteri) => {
        console.log(musteri);
        const tr = document.createElement("tr");
        let isimTd = document.createElement("td");
        isimTd.textContent = musteri.isim;
        let soyisimTd = document.createElement("td");
        soyisimTd.textContent = musteri.soyisim;
        let emailTd = document.createElement("td");
        emailTd.textContent = musteri.email;
        let sifreTd = document.createElement("td");
        sifreTd.textContent = musteri.sifre;
        let telefonTd = document.createElement("td");
        telefonTd.textContent = musteri.telefon;
        let tcTd = document.createElement("td");
        tcTd.textContent = musteri.tc;
        let kiraladigiArabaTd = document.createElement("td");
        kiraladigiArabaTd.textContent =
            kiralayanMüsteri == musteri.isim
                ? kiralananAraba
                : musteri.kiraladigiAraba;
        let kiraladigiZamanTd = document.createElement("td");
        kiraladigiZamanTd.textContent =
            kiralayanMüsteri == musteri.isim
                ? dateGiris.value + "/" + dateBitis.value
                : musteri.kiraladigiTarih;
        const settingsBtnTd = document.createElement("td");
        const deleteBtn = document.createElement("button");
        deleteBtn.textContent = "Delete";
        deleteBtn.className = "btn-red me5";
        deleteBtn.addEventListener("click", () => {
            deleteMusteri(musteri);
        });
    });
}

```

Burada yer alan JavaScript kod parçasığında kiralanan araba ve kiralayan müşteriye göre müşteri bilgilerinin listelenmesi amaçlanmaktadır.

Müşteri ismi, soy ismi, e-mail, şifre, müşterinin kiralamış olduğu araba ve aracın kiralandığı tarih bilgileri tutulmaktadır.

```
//Düzenleme işlemi
const duzenleBtn = document.createElement("button");
duzenleBtn.textContent = "Düzenle";
duzenleBtn.className = "btn-green";
duzenleBtn.addEventListener("click", () => {
    duzenleMusteri(musteri);
});
settingsBtnTd.appendChild(deleteBtn);
settingsBtnTd.appendChild(duzenleBtn);
musteriListe.appendChild(tr);
tr.appendChild(isimTd);
tr.appendChild(soyisimTd);
tr.appendChild(emailTd);
tr.appendChild(sifreTd);
tr.appendChild(telefonTd);
tr.appendChild(tcTd);
tr.appendChild(kiraladigiArabaTd);
tr.appendChild(kiraladigiZamanTd);
tr.appendChild(settingsBtnTd);
});
myObject.bosalt();
}
```

Bu JavaScript kod parçası, bir müşteri listesini HTML tablosunda görüntülemek için kullanılan “musteriListele” adlı bir fonksiyonun devamını içermektedir. Bu kod, her müşteri için bir “Düzenle” düğmesi eklemeyi amaçlamaktadır.

```
/Müşteri Ekle/;
function musteriEkle() {
    const isim = document.getElementById("isim").value;
    const soyisim = document.getElementById("soyisim").value;
    const email = document.getElementById("email").value;
    const sifre = document.getElementById("sifre").value;
    const telefon = document.getElementById("telefon").value;
    const tc = document.getElementById("tc").value;
    if (isim && soyisim && email && sifre && telefon && tc) {
        let musteri = new Musteri(isim, soyisim, email, sifre, telefon, tc);
        customer.addMusteri(musteri);
        musterilistele();
    } else {
        alert("Eksik bilgi girdiniz.");
    }
}
```

Burada yer alan JavaScript kod parçası ile müşteri listeleme müşteri ekleme fonksiyonları yer almaktadır. Musteriekle fonksiyonu alınan bilgilerin dolu olup olmadığını kontrol etmektedir. Eğer tüm bilgiler doluysa, yeni bir Musteri nesnesi oluşturur ve bu bilgilerle doldurur. Bu nesne, customer adlı bir nesnenin addMusteri metoduna eklenir. Bu işlem, yeni bir müşteri eklemeyi temsil eder. Eğer bir veya daha fazla bilgi eksikse, bir uyarı mesajı görüntülenir ve müşteri eklenmez.

```

// ...
// Düzenleme formunu gizle
const duzenleForm = document.getElementById("duzenleForm");
// Düzenleme işlemi başlat
function duzenleMusteri(musteri) {
    // Düzenleme formunu göster
    duzenleForm.style.display = "flex"; // Düzenlenen müşteri bilgilerini formda göster
    duzenleForm.style.flexDirection = "column"; // Düzenlenen müşteri bilgilerini formda göster
    duzenleForm.style.alignItems = "center"; // Düzenlenen müşteri bilgilerini formda göster
    document.getElementById("duzenlenecekMusteriIndex").value = customer
        .getMyMusteri()
        .indexOf(musteri);
    document.getElementById("duzenleIsim").value = musteri.isim;
    document.getElementById("duzenleSoyisim").value = musteri.soyisim;
    document.getElementById("duzenleEmail").value = musteri.email;
    document.getElementById("duzenleSifre").value = musteri.sifre;
    document.getElementById("duzenleTelefon").value = musteri.telefon;
    document.getElementById("duzenleTc").value = musteri.tc;
}

```

Burada yer alan JavaScript kod parçası ile bir düzenleme formunu görüntülemek ve bu formu doldurmak için kullanılan duzenleMusteri adlı bir fonksiyonu içermektedir. Bu kod parçası müşterilere ait bilgilerle dolu olan formun düzenlenmesini sağlayan ara yüz oluşturmaktadır.

```

// Düzenleme formunu iptal et
document.getElementById("duzenleIptal").addEventListener("click", function () {
    duzenleForm.style.display = "none";
});
// Düzenleme işlemini kaydet
document.getElementById("duzenleKaydet").addEventListener("click", function () {
    // Düzenlenen müşteri bilgilerini alın
    const duzenlenecekMusteriIndex = document.getElementById(
        "duzenlenecekMusteriIndex"
    ).value;
    const yeniIsim = document.getElementById("duzenleIsim").value;
    const yeniSoyisim = document.getElementById("duzenleSoyisim").value;
    const yeniEmail = document.getElementById("duzenleEmail").value;
    const yeniSifre = document.getElementById("duzenleSifre").value;
    const yeniTelefon = document.getElementById("duzenleTelefon").value;
    const yeniTc = document.getElementById("duzenleTc").value;
    // Müşteri nesnesini güncelle
    const duzenlenecekMusteri = customer.getMyMusteri()[duzenlenecekMusteriIndex];
    duzenlenecekMusteri.isim = yeniIsim;
    duzenlenecekMusteri.soyisim = yeniSoyisim;
    duzenlenecekMusteri.email = yeniEmail;
    duzenlenecekMusteri.sifre = yeniSifre;
    duzenlenecekMusteri.telefon = yeniTelefon;
    duzenlenecekMusteri.tc = yeniTc;
    // Düzenleme formunu gizle
    duzenleForm.style.display = "none";
    // Müşteri listesini yeniden oluştur
    musterilistele();
});

```

Burada yer alan JavaScript kod parçası ile event listener metodu ile düzenleme formunun iptal edilmesi, düzenlenen müşteri bilgilerini kaydedilmesi işlemlerini sağlamaktadır.

Kısacası bu kod parçası, düzenleme formunun işlevselliğini ele almakta ve kullanıcının müşteri bilgilerini düzenlemesini ve değişiklikleri kaydetmesini sağlamaktadır.

```
let dateButton = document.getElementById("tarihbtn");
let dateGiris = document.getElementById("startDate");
let dateBitis = document.getElementById("endDate");
let hesaplaButonuCheck = false;
dateButton.addEventListener("click", () => {
    hesapla();
    hesaplaButonuCheck = true;
});
function hesapla() {
    var dateBitis_ = dateBitis.value.split("-");
    var dateGiris_ = dateGiris.value.split("-");
    var gunSonuc = Number(dateBitis_[2]) - Number(dateGiris_[2]);
    var aySonuc = Number(dateBitis_[1]) - Number(dateGiris_[1]);
    var sonuc = gunSonuc + aySonuc * 30;
    console.log(sonuc);
    if (sonuc > 0) {
        arabaFonksiyon(sonuc);
    } else {
        alert("Yanlış Tarih girdiniz");
    }
}
```

Bu Javascript kod parçasında web sayfasında tarih bilgilerini hesaplayan ve bu tarihlerle ilgili işlemler gerçekleştiren fonksiyonu ve event listener'ı içermektedir. Kullanıcı tarafından rezervasyon başlangıç ve bitiş tarihlerinin girilmesini ve bu tarihler arasındaki farkın hesaplanarak işlem yapılmasını amaçlar.

```
var signupLink = document.getElementById("signup-link");
var signupWindow = document.getElementById("signup-window");
var signupButton = document.getElementById("signup-button");
signupLink.addEventListener("click", function (event) {
    event.preventDefault();
    signupWindow.style.display = "block";
    body.style.display = "none";
});
signupButton.addEventListener("click", function () {
    musteriekle();
    signupWindow.style.display = "none";
    body.style.display = "block";
});
var odemeEkranı = document.getElementById("payment-form");
var loginLink = document.getElementById("login-link");
var loginWindow = document.getElementById("login-window");
var loginButton = document.getElementById("login-button");
var body = document.getElementsByTagName("body");
loginLink.addEventListener("click", function (event) {
    event.preventDefault();
    loginWindow.style.display = "block";
    body.style.display = "none";
});
loginButton.addEventListener("click", function () {
    login();
    loginWindow.style.display = "none";
    body.style.display = "block";
});
```

Bu kod parçası, kullanıcı etkileşimlerini takip edilerek giriş, kayıt ve ödeme işlemleri için pop-up pencerelerinin görünmesini veya gizlenmesini sağlamakatadır.

```

signupLink.addEventListener("click", function (event) {
  event.preventDefault();
  signupWindow.style.display = "block";
  body.style.display = "none";
});
signupButton.addEventListener("click", function () {
  signupWindow.style.display = "none";
  body.style.display = "block";
});
let kullaniciEmail = document.getElementById("yeni_email");
let kullaniciPassword = document.getElementById("yeni_password");
let loginSignup = document.getElementById("loginSignup");
let loginItem = document.getElementById("login");
let signupItem = document.getElementById("signup");
let kiralaButtonu = document.getElementById("kiralaButon");
const musteriliste_ = document.getElementById("musteriliste_");

```

Bu kod parçası, kullanıcı kaydı ve giriş için gerekli HTML elementlerine erişim sağlayarak bu işlemlerin gerçekleştirilemesi için gerekli event listener'ları eklemektedir.

```

musterilisteButton.style.display = "none";
musteriliste_.style.display = "none";
function login() {
  var kimlikDogrulama = customer.getMyMusteri().find(function (element)
    return kullaniciEmail.value == element.email &&
    kullaniciPassword.value == element.sifre
    ? true
    : false;
  });

  if (
    kimlikDogrulama.email == "admin@gmail.com" &&
    kimlikDogrulama.sifre == "admin"
  ) {
    musteriliste_.style.display = "table";
    musterilisteButton.style.display = "block";
  } else {
    musteriliste_.style.display = "none";
  }
  if (kimlikDogrulama) {
    myObject.addMusteri(kimlikDogrulama);
  }
}

```

Bu kod parçası, kullanıcının kimlik doğrulamasını yapar ve kullanıcının yönetici olup olmadığını kontrol ederek müşteri listesini görüntülemeye veya gizlemeye karar vermektedir.

```
// console.log(kiralananAraba);
loginSignup.innerHTML = "";
console.log(myObject.getMyMusteri());
alert("Giriş Başarılı");
loginItem.style.display = "none";
signupItem.style.display = "none";
let li = document.createElement("li");
li.className = "nav-item";
let ahref = document.createElement("a");
ahref.textContent = "Hoşgeldin " + kimlikDogrulama.isim;
ahref.className = "nav-link";
loginSignup.appendChild(li);
li.appendChild(ahref);
let li2 = document.createElement("li");
li2.className = "nav-item";
let ahref2 = document.createElement("a");
ahref2.textContent = "Çıkış Yap";
ahref2.className = "nav-link";
ahref2.id = "secondLogin";
loginSignup.appendChild(li2);
li2.appendChild(ahref2);
let secondLogin = document.getElementById("secondLogin");
secondLogin.addEventListener("click", (event) => {
    event.preventDefault();
    loginWindow.style.display = "block";
    body.style.display = "none";
    myObject.bosalt();
});
} else {
    alert("Giriş Başarısız");
}
}
```

Yukarıdaki kod parçasında login işlemi sırasında kullanıcı bilgilerinin doğrulanması durumunda çalışacak fonksiyonlar bulunmaktadır. Başarılı giriş yapıldığında müşteri kendi sistemini görüntüleyerek başarılı giriş uyarısı alır.

```

function kiralamaYapilanAraba(araba) {
    if (araba != undefined) {
        myObject.addAraba(araba);
    }
}
// Ödeme:
let odemeWindow = document.getElementById("odeme-window");
odemeWindow.style.display = "none";
let cardNumber = document.getElementById("cardNumber");
let expirationDate = document.getElementById("expirationDate");
let cvv = document.getElementById("cvv");
let odemeButton = document.getElementById("odeme-button");

odemeButton.addEventListener("click" , ()=>{
    odemeYap();
})

```

Araba kiralama anında kiralama butonuna basıldığında ödeme popupı açılır ve müşteri gerekli kart bilgilerini almaktadır. Böylelikle hem araba kiralama işlemi tamamlanmaktadır hem de kiralama işlemi sırasında kullanılan kart bilgileri sisemde gösterilmektedir.

```

function odemeYap() {
    var cardNumber = document.getElementById("cardNumber").value;
    var expirationDate = document.getElementById("expirationDate").value;
    var cvv = document.getElementById("cvv").value;

    if (cardNumber == "" && expirationDate == "" && cvv == "") {
        alert("Eksik yerleri doldurunuz.");
    } else {
        alert(
            "Ödeme işlemi tamamlandı. Kart Numarası: " +
            cardNumber
        );
    }
}

```

RENT A CAR

RAPOR

Hazırlayanlar

İsmail Arif Güleç
Betül Daşçı
Salih Enes Özdel
Şeyma Çıldır
Berke Tangör
Ahmet Akif Kasım



EKİM 2023

TEŞEKKÜRLER