

ANKARA ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ



BLM 4531 PROJE RAPORU

Hastane Randevu Alma Sitesi

Ömer Faruk CENGİZ

19290224

Enver BAĞCI

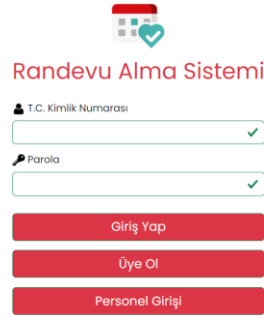
11.01.2023

Projenin Tanıtımı

Projem hastanelerden randevu alma sitesidir. Kaydını yapan bir kullanıcı, ardından giriş yaptıktan sonra hastane doktoru, aşı ve aile hekimi randevuları alabilmektedir. Doktor randevusu için istediği şehir, ilçe, hastane ve klinik bilgisini seçtikten sonra filtrelenen doktorlardan randevu saati alabilmektedir. Ayrıca sitede personel girişi bulunmaktadır. Personel girişi yapan kişi admin yetkisinde olup hastane, klinik ve doktor ekleme,silme ve güncelleme yetkilerine sahiptir.

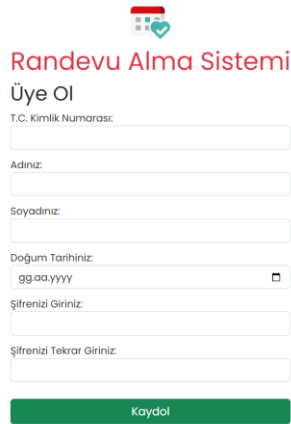
Sitenin Görünümü

Siteye ilk giriş yapıldığında giriş(login) ekranı gelmektedir.



The image shows the login screen of the 'Randevu Alma Sistemi'. At the top, there is a logo consisting of a red calendar icon and a green heart icon. Below the logo, the title 'Randevu Alma Sistemi' is displayed in red. The login form includes two input fields: 'T.C. Kimlik Numarası' and 'Parola', both with green checkmarks indicating successful input. Below these fields are three red buttons: 'Giriş Yap', 'Üye Ol', and 'Personel Girişi'.

Kullanıcının eğer bir hesabı bulunuyorsa doğrudan giriş yapabilir. Fakat hesabı yoksa önce üye olmalıdır.



The image shows the 'Üye Ol' (Sign Up) screen of the 'Randevu Alma Sistemi'. At the top, there is a logo consisting of a red calendar icon and a green heart icon. Below the logo, the title 'Randevu Alma Sistemi' is displayed in red, followed by 'Üye Ol' in black. The registration form includes several input fields: 'T.C. Kimlik Numarası', 'Adınız', 'Soyadınız', 'Doğum Tarihiniz' (with a date picker showing 'gg.aa.yyyy'), 'Şifrenizi Giriniz', and 'Şifrenizi Tekrar Giriniz'. A green 'Kaydol' (Register) button is at the bottom.

Buradan gerekli bilgileri girip üye olduktan sonra artık giriş yapabilir.

Kullanıcı giriş yaptıktan sonra karşısına şöyle bir sayfa çıkmaktadır:

Randevu Alma Sistemi

Hosgeldin, !!!!!!!

Aşı Randevusu Al

Hastane Randevusu Al

Aile Hekimi Randevusu Al

Hastanelerimiz

İstanbul	Fatih	Cerrahpaşa Tıp Fakültesi Hastanesi
İstanbul	Bahçelievler	Bahçelievler Devlet Hastanesi
İstanbul	Küçükçekmece	İstanbul Eğitim ve Araştırma Hastanesi
Ankara	Keçiören	Ankara Etik Şehir Hastanesi
Ankara	Çankaya	Ankara Şehir Hastanesi
Ankara	Altındağ	Ankara Eğitim ve Araştırma Hastanesi
İzmir	Bornova	Ege Üniversitesi Hastanesi
İzmir	Konak	Tepecik Eğitim ve Araştırma Hastanesi
İzmir	Gaziemir	Gaziemir Devlet Hastanesi

Bu sayfada üst menü kısmında kullanıcının tc bilgisi, ona tıklandığında ise randevularım ve çıkış yap kısmı çıkmaktadır. Sayfanın üst kısmında aşı, hastane ve aile hekimi randevusu alma butonları bulunmaktadır. Altta ise randevu alınabilecek hastanelerin bilgisi bulunmaktadır. Aşı veya aile hekimi randevusu alma butonlarına basıldığında benzer sayfalar gelmektedir. Bunların farkı ise randevu alındığında birinin aşı randevusu diğerinin aile hekimi randevusu olarak gözükmesidir.

Randevu Alma Sistemi

Hosgeldin, !!!!!!!

< Aşı Randevusu Al

13 Ocak Cuma

14 Ocak Cumartesi

15 Ocak Pazar

16 Ocak Pazartesi

17 Ocak Salı

18 Ocak Çarşamba

19 Ocak Perşembe

9:00						
10:00						
11:00						
12:00						
13:00						
14:00						
15:00						
16:00						

Randevu Al

Yukarıda bulunan sayfa aşı randevusu alma butonuna tıkladıktan sonra gelen sayfadır. Burada seçtiğim gün gri olmuştur ve seçtiğim saat butonu ise yeşil olmuştur. Saatlerin yanındaki aşağı inme butonlarına basarak o saatlere ait randevu saatleri görülebilir. Gün ve saati seçtikten sonra randevu al butonuna tıkladığında artık randevu alınmıştır ve kullanıcı bunu randevularım sayfasında görüntüleyebilir.



Üst kısımda da görüldüğü üzere kullanıcı tc numarasının üstüne basarak randevu bilgisi sayfası ve çıkış yap butonunu görebilir. Bir önceki görselde aşı randevusu alındıktan sonra ve randevu bilgisine tıklandıktan sonra gelen ekran şöyledir:



Randevu bilgisi sayfasında aldığım randevu gözükmemektedir. Burada aşı randevusu aldığım için randevunun türü aşıdır. Gün ve saatte doğru bir şekilde gözükmemektedir. Bu ekrandaki hastane, klinik ve doktor adı bilgisi ise kullanıcılara kayıt yaparken otomatik olarak atanan doktor bilgisinden gelmektedir.

Ana sayfaya döndüğümüzde aile hekimi randevusu alma butonuna tıklandığında önceden dediğim gibi aşı randevu sayfasına benzer bir sayfa çıkmaktadır. Fakat aile hekimi randevusu alındığında randevularım kısmında randevu türü aile hekimi randevusu olarak gözükmemektedir.

Ana sayfada hastane randevusu al butonuna tıklandığında ise şöyle bir sayfa gelmektedir:

Randevu Alma Sistemi

Hosgeldin, !!!!!!!

Şehirler

---Şehir Seçiniz---

İlçeler

---İlçe Seçiniz---

Hastaneler

---Hastane Seçiniz---

Klinikler

---Klinik Seçiniz---

Bu ekran doktor filtrelemek için kullanılır. Burada istediğiniz şehri, ilçeyi, hastaneyi ve kliniği seçerek ona bağlı olan doktorların listesini görüntülenmesini sağlarsınız. İlk başta şehir dışındaki diğer seçeneklerin kilitli olmasının sebebi birbirlerine bağlı olmaları ve bir üstteki seçenek seçildikçe açılmalıdır. İstenilen filtrelemeler yapıldıktan sonra ekran şöyle olur:

Randevu Alma Sistemi

Hosgeldin, !!!!!!!

Şehirler

İstanbul

İlçeler

Fatih

Hastaneler

Cerrahpaşa Tıp Fakültesi Hastanesi

Klinikler

Cildiye(Dermatoloji)

Doktor Ara

Doktor ara butonuna basıldığında bu kriterlere uygun doktor listesi görülmektedir.

Randevu Alma Sistemi

Hosgeldin, !!!!!!!

< Hastane Listesi

Ömer Faruk Cengiz	Cerrahpaşa Tıp Fakültesi Hastanesi	Cildiye(Dermatoloji)	Randevu Al
Sedat Toprak	Cerrahpaşa Tıp Fakültesi Hastanesi	Cildiye(Dermatoloji)	Randevu Al
Berat Yılmaz	Cerrahpaşa Tıp Fakültesi Hastanesi	Cildiye(Dermatoloji)	Randevu Al

Listelenen doktorlardan biri seçilip randevu al butonuna basıldıktan sonra bizi artık o doktora ait randevu sayfasına yönlendirmektedir. Burada en üstteki doktor seçildikten sonra karşımıza şöyle bir ekran gelmektedir:

Randevu Alma Sistemi

Hosgeldin,!!!!!!!

< Doktor Randevusu Al

13 Ocak Cuma

14 Ocak Cumartesi

15 Ocak Pazar

16 Ocak Pazartesi

17 Ocak Salı

18 Ocak Çarşamba

19 Ocak Perşembe

9:00

10:00

11:00

12:00

13:00

14:00

15:00

16:00

15:00

15:20

15:40

Randevu Al

Burada gün ve saat tercihleri yapıldıktan ve randevu al butonuna basıldıktan sonra randevu alınmış olur. Randevu kontrolü için randevu bilgi sayfasına bakalım:

Randevu Alma Sistemi

Hosgeldin,!!!!!!!

Randevularım

Randevu Türü	Hastane Adı	Klinik Adı	Doktor Adı	Randevu Tarihi
Aşı Randevusu	Sağlık Ocağı	Aile Hekimi Polikliniği	Ekrem Kuru	17 Ocak Salı 11:20
Doktor Randevusu	Cerrahpaşa Tıp Fakültesi Hastanesi	Cildiye(Dermatoloji)	Ömer Faruk Cengiz	18 Ocak Çarşamba 15:00

Görüldüğü üzere seçtiğim hastane, klinik, doktor, gün ve saat bilgilerine göre randevum gözükmemektedir. Doktor randevusu olduğu için randevu türüm doktor randevusudur. Diğer randevum ise bir önceki yaptığım işlemde kalan randevudur. Son olarak seçtiğim randevu tarihleri başka kullanıcılar tarafından seçilmesin diye o gün ve saate ait buton disabled(seçilemez) olur.

Kullanıcı sayfası dışında admin paneli de bulunur. Admin işlemleri yapmak için giriş ekranında personel girişin seçilmesi gerekir ve sistemde tanımlanan kullanıcı adı ve şifre bilgisiyle giriş yapılır. Admin panelinde hastane, klinik, doktor ekleme, silme ve güncelleme işlemleri yapılabilir.

Admin Paneli Hastaneler Klinikler Doktorlar						Çıkış Yap
+ Yeni Hastane Ekle						
Hastane ID	İl Adı	İlçe Adı	Hastane Adı	Sil	Güncelle	
1	İstanbul	Fatih	Cerrahpaşa Tıp Fakültesi Hastanesi	Sil	Güncelle	
2	İstanbul	Bahçelievler	Bahçelievler Devlet Hastanesi	Sil	Güncelle	
3	İstanbul	Küçükçekmece	İstanbul Eğitim ve Araştırma Hastanesi	Sil	Güncelle	
4	Ankara	Keçiören	Ankara Etilik Şehir Hastanesi	Sil	Güncelle	
5	Ankara	Çankaya	Ankara Şehir Hastanesi	Sil	Güncelle	
6	Ankara	Altındağ	Ankara Eğitim ve Araştırma Hastanesi	Sil	Güncelle	
7	İzmir	Bornova	Ege Üniversitesi Hastanesi	Sil	Güncelle	
8	İzmir	Konak	Tepecik Eğitim ve Araştırma Hastanesi	Sil	Güncelle	
9	İzmir	Gaziemir	Gaziemir Devlet Hastanesi	Sil	Güncelle	
1012	Ankara	Çankaya	Sağlık Ocağı	Sil	Güncelle	

Hastane listesi ve düzenleme ekranı

Admin Paneli Hastaneler Klinikler Doktorlar						Çıkış Yap
+ Yeni Klinik Ekle						
Klinik ID	Hastane Adı	Klinik Adı	Sil	Güncelle		
1	Cerrahpaşa Tıp Fakültesi Hastanesi	Cildiye(Dermatoloji)	Sil	Güncelle		
2	Cerrahpaşa Tıp Fakültesi Hastanesi	Genel Cerrahi	Sil	Güncelle		
3	Cerrahpaşa Tıp Fakültesi Hastanesi	Çocuk Sağlığı ve Hastalıkları	Sil	Güncelle		
4	Cerrahpaşa Tıp Fakültesi Hastanesi	İç Hastalıkları(Dahiliye)	Sil	Güncelle		
5	Cerrahpaşa Tıp Fakültesi Hastanesi	Kulak-Burun-Bogaz(KBB)	Sil	Güncelle		
6	Cerrahpaşa Tıp Fakültesi Hastanesi	Gastroenteroloji	Sil	Güncelle		
7	Cerrahpaşa Tıp Fakültesi Hastanesi	Kardiyoloji	Sil	Güncelle		
8	Cerrahpaşa Tıp Fakültesi Hastanesi	Nöroloji	Sil	Güncelle		
1010	Bahçelievler Devlet Hastanesi	Cildiye(Dermatoloji)	Sil	Güncelle		
1011	Bahçelievler Devlet Hastanesi	Genel Cerrahi	Sil	Güncelle		

Klinik listesi ve düzenleme ekranı

Admin Paneli Hastaneler Klinikler Doktorlar						Çıkış Yap
+ Yeni Doktor Ekle						
Aranacak doktor adını giriniz						Arar
Doktor ID	Hastane Adı	Klinik Adı	Doktor Adı	Sil	Güncelle	
1	Cerrahpaşa Tıp Fakültesi Hastanesi	Cildiye(Dermatoloji)	Ömer Faruk Cengiz	Sil	Güncelle	
2	Cerrahpaşa Tıp Fakültesi Hastanesi	Genel Cerrahi	Berkay Akparlar	Sil	Güncelle	
3	Cerrahpaşa Tıp Fakültesi Hastanesi	Çocuk Sağlığı ve Hastalıkları	Eren Kıranatlı	Sil	Güncelle	
4	Cerrahpaşa Tıp Fakültesi Hastanesi	İç Hastalıkları(Dahiliye)	Ahmet Çakal	Sil	Güncelle	
5	Cerrahpaşa Tıp Fakültesi Hastanesi	Kulak-Burun-Bogaz(KBB)	Zeynep Gürler	Sil	Güncelle	
6	Cerrahpaşa Tıp Fakültesi Hastanesi	Gastroenteroloji	Ebru Gün	Sil	Güncelle	
7	Cerrahpaşa Tıp Fakültesi Hastanesi	Kardiyoloji	Hakan İlkay	Sil	Güncelle	
8	Cerrahpaşa Tıp Fakültesi Hastanesi	Nöroloji	Buket Dağdeviren	Sil	Güncelle	
9	Bahçelievler Devlet Hastanesi	Cildiye(Dermatoloji)	Soner Kabak	Sil	Güncelle	
10	Bahçelievler Devlet Hastanesi	Genel Cerrahi	Duygu Özü	Sil	Güncelle	

Doktor listesi ve düzenleme ekranı(Doktor arama ve sayfalama seçeneği mevcuttur)

Doktor sayfasından örnek vererek tüm sayfalar için yapılabilecek işlemleri anlatacağım. Yeni doktor ekle tuşuna basıldığında şöyle bir ekran gelmektedir:

Admin Paneli

Hastaneler

Klinikler

Doktorlar

Çıkış Yap

Hastaneler

---Hastane Seçiniz---

Klinikler

---Klinik Seçiniz---

Doktor Adı

Doktor Ekle

Burada seçeneklere tıklandığında gelen dropdownlistten istenilen bilgiler seçilerek o bilgilere ait yani o hastane adı ve kliniğine ait doktor adı da girildikten sonra doktor eklemesi yapılabilir.

Güncelleme butonuna basıldığında aşağıdaki gibi bir ekran gelmektedir. Bu ekranda değiştirilmek istenen bilgiler değiştirilip onaylanırsa o doktora ait bilgiler güncellenir.

Admin Paneli

Hastaneler

Klinikler

Doktorlar

Çıkış Yap

Hastane Adı

Cerrahpaşa Tıp Fakültesi Hastanesi

Klinik Adı

Cildiye(Dermatoloji)

Doktor Adı

Ömer Faruk Cengiz

Doktor Güncelle

Sil tuşuna basıldığında ise pop-up olarak silme onayı sormaktadır, işlem onaylanınca silme gerçekleşir.

localhost:44308 web sitesinin mesajı

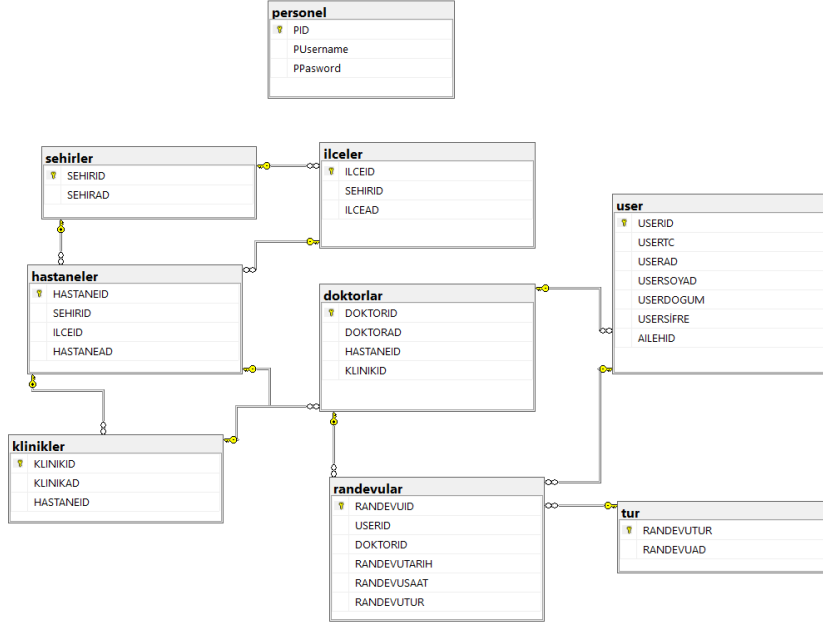
Gerçekten Silmek İstiyor musunuz?

Tamam

İptal

Database Kısmı

Veri tabanımı diyagramım üzerinden açıklayacağım.



Diyagramımda tüm tablolarım ve aralarındaki ilişkiler bulunmaktadır.

user tablosu: Kayıt yapan kullanıcının bilgileri bulunmaktadır.

personel tablosu: Admin giriş bilgisini tutmaktadır.

sehirler tablosu: Hastanelerimin bulunduğu şehirleri içermektedir.

ilceler tablosu: Bu şehirlere bağlı olan ilçeleri içermektedir.

hastaneler tablosu: Şehire ve ilçeye bağlı olan hastaneler bulunmaktadır.

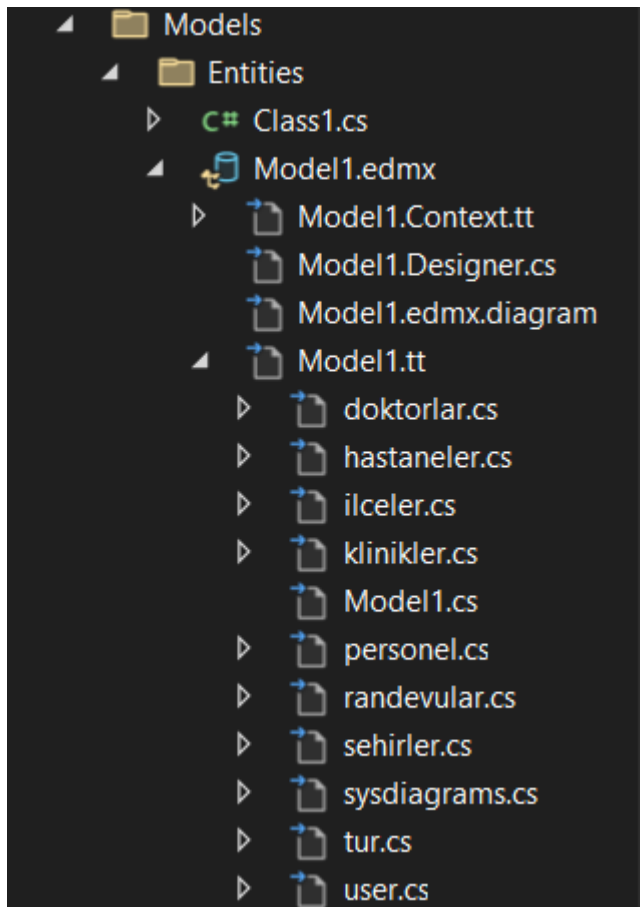
kliniker tablosu: Hastaneye bağlı olan polikliniklerin bilgisi bulunmaktadır.

doktorlar tablosu: Hastane ve ona bağlı olan kliniklerdeki doktorların bilgisi bulunmaktadır.

randevular tablosu: Alınan randevuların tür, gün, saat ve doktor bilgisini içermektedir.

tur tablosu: Randevunun tür bilgisini bulundurmaktadır.

Model Kısmı



Birkaç örnek:

```
namespace hastanerandevu.Models.Entities
{
    using System;
    using System.Collections.Generic;

    15 references
    public partial class doktorlar
    {
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]
        1 reference
        public doktorlar()
        {
            this.randevular = new HashSet<randevular>();
            this.user = new HashSet<user>();
        }

        3 references
        public int DOKTORID { get; set; }
        5 references
        public string DOKTORAD { get; set; }
        3 references
        public int HASTANEID { get; set; }
        6 references
        public int KLINIKID { get; set; }

        1 reference
        public virtual hastaneler hastaneler { get; set; }
        1 reference
        public virtual klinikler klinikler { get; set; }
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
        1 reference
        public virtual ICollection<randevular> randevular { get; set; }
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
        1 reference
        public virtual ICollection<user> user { get; set; }
    }
}
```

Yukarıda doktorlar tabloma ait doktorlar.cs dosyası ve içindeki değişken, ilişkiler gibi bilgiler bulunmaktadır.

```
namespace hastanerandevu.Models.Entities
{
    using System;
    using System.Collections.Generic;
    using System.ComponentModel;
    using System.ComponentModel.DataAnnotations;
    7 references
    public partial class user
    {
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]
        0 references
        public user()
        {
            this.randevular = new HashSet<randevular>();
        }
        4 references
        public int USERID { get; set; }
        [Required(ErrorMessage = "Lütfen TC Kimlik No Giriniz", AllowEmptyStrings = false)]
        10 references
        public string USERTC { get; set; }
        [Required(ErrorMessage = "Lütfen Adınızı Giriniz", AllowEmptyStrings = false)]
        1 reference
        public string USERAD { get; set; }
        [Required(ErrorMessage = "Lütfen Soyadınızı Giriniz", AllowEmptyStrings = false)]
        0 references
        public string USERSOYAD { get; set; }

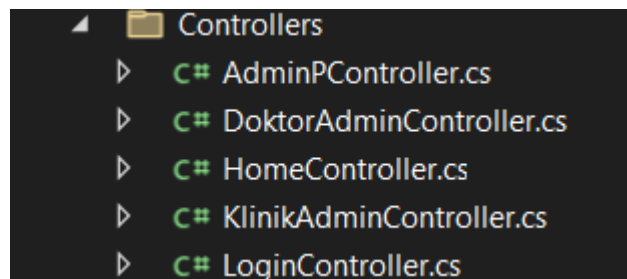
        [DataType(DataType.Date)]
        [DisplayFormat(ApplyFormatInEditMode = true, DataFormatString = "{0:MM/dd/yyyy}")]
        0 references
        public System.DateTime USERDOGUM { get; set; }
        [Required(ErrorMessage = "Lütfen Şifrenizi Giriniz", AllowEmptyStrings = false)]
        [DataType(System.ComponentModel.DataAnnotations.DataType.Password)]
        [StringLength(50, MinimumLength = 6, ErrorMessage = "Şifreniz en az 6 hane olmalıdır")]
        2 references
        public string USERSİFRE { get; set; }
        [Compare("USERSİFRE", ErrorMessage = "Şifre eşleşmiyor")]
        [DataType(System.ComponentModel.DataAnnotations.DataType.Password)]
        0 references
        public string USERRESİFRE { get; set; }
        0 references
        public int AILEHID { get; set; }

        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
        1 reference
        public virtual ICollection<randevular> randevular { get; set; }
        0 references
        public virtual doktorlar doktorlar { get; set; }
    }
}
```

Yukarıda user tabloma ait user.cs dosyası bulunmaktadır. Değişkenler dışında giriş yapıldığında veya kayıt olunduğunda gerekli mesajlar, data type tanımlamaları ve display bilgileri içermektedir.

Backend Kısmı

Backend kısmında controllerlarım bulunmaktadır. Bu controllerlardaki kodlar veri tabanında gelen bilgilerle işlemler yaparak view kısmında görüntülenecek bilgilerin arka plan kodlarını bulundurur. Controller'larım;



Login Controller:

```

1  using hastanerandevu.Models.Entities;
2  using System;
3  using System.Collections.Generic;
4  using System.EnterpriseServices.CompensatingResourceManager;
5  using System.Linq;
6  using System.Threading.Tasks;
7  using System.Web;
8  using System.Web.Mvc;
9  using System.Web.Security;
10
11 namespace hastanerandevu.Controllers
12 {
13     public class LoginController : Controller
14     {
15         // GET: Login
16         hastaneEntities db = new hastaneEntities();
17         public ActionResult Register()
18         {
19             return View();
20         }
21         [HttpPost]
22         [ValidateAntiForgeryToken]
23         public ActionResult Register(user U)
24         {
25             if (ModelState.IsValid)
26             {
27                 using (hastaneEntities dc = new hastaneEntities())
28                 {
29                     if (dc.user.Any(x => x.USERTC == U.USERTC))
30                     {
31                         ViewBag.DuplicateMessage = "Kullanıcı zaten mevcut";
32                         return View(U);
33                     }
34                     dc.user.Add(U);
35                     dc.SaveChanges();
36                     ModelState.Clear();
37                     U = null;
38                     ViewBag.Message = "Kaydınız başarıyla tamamlanmıştır";
39                 }
40             }
41             return View(U);
42         }
43         [HttpGet]
44         public ActionResult Login()
45         {
46             return View();
47         }
48         [HttpPost]
49         [ValidateAntiForgeryToken]

```

Yukardaki controller kısmında yeni kayıt yapacak bir kullanıcının girdiği bilgilerin veritabanına eklenmesini sağlayan fonksiyonum bulunmaktadır. Ayrıca işlemlerin doğru veya yanlış yapılması halinde gönderilen mesaj bilgilerini içermektedir.

```

    }
    [HttpPost]
    [ValidateAntiForgeryToken]
    0 references
    public ActionResult Login(user US)
    {
        var checkLogin = db.user.Where(x => x.USERTC.Equals(US.USERTC) && x.USERSİFRE.Equals(US.USERSİFRE)).FirstOrDefault();
        if (checkLogin != null)
        {
            Session["USERCSS"] = US.USERTC.ToString();
            TempData["UserName"] = checkLogin.USERAD;
            FormsAuthentication.SetAuthCookie(US.USERTC, false);
            return RedirectToAction("Index", "Home");
        }
        else
        {
            ViewBag.Notification = "Yanlış TC veya şifre";
        }
        return View();
    }
    [HttpGet]
    0 references
    public ActionResult PersonalG()
    {
        return View();
    }
    [HttpPost]
    [ValidateAntiForgeryToken]
    0 references
    public ActionResult PersonalG(personel PS)
    {
        var checkLogin = db.personel.Where(x => x.PUsername.Equals(PS.PUsername) && x.PPassword.Equals(PS.PPassword)).FirstOrDefault();
        if (checkLogin != null)
        {
            Session["PUsernameSS"] = PS.PUsername.ToString();
            return RedirectToAction("Index", "AdminP");
        }
        else
        {
            ViewBag.Notification = "Yanlış kullanıcı adı veya şifre";
        }
        return View();
    }
    0 references
    public ActionResult Logout()
    {
        FormsAuthentication.SignOut();
        return RedirectToAction("Login", "Login");
    }
}

```

Yukardaki controller kısmında ise sisteme giriş yapacak bir kullanıcının girdiği bilgilerin veri tabanındaki bilgilerle karşılaştırılması ve bu bilgilerin doğru veya yanlış olmasına göre yapılacak işlemlerin fonksiyonu bulunmaktadır. Ayrıca giriş yapan bir üyenin çıkmasını sağlayan logout işlemi bulunmaktadır.

Home Controller:

Bu controllerda kullanıcının giriş yapmasının ardından seçebileceği işlemlerin arka plan kodları bulunmaktadır.

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Net.Sockets;
5  using System.Security.Policy;
6  using System.Web;
7  using System.Web.Mvc;
8  using System.Web.Security;
9  using Newtonsoft.Json;
10 using hastanera randevu.Models.Entities;
11 using System.Security.Cryptography.Xml;
12
13 namespace hastanera randevu.Controllers
14 {
15     // GET: Home
16     public class HomeController : Controller
17     {
18         // GET: Home
19         hastaneEntities db=new hastaneEntities();
20         public ActionResult Index()
21         {
22             var degerler=db.hastaneler.ToList();
23             return View(degerler);
24         }
25         Class1 cs = new Class1();
26
27         public ActionResult Cascading()
28         {
29             ViewBag.SehirList = new SelectList(ilgetir(), "SEHIRID", "SEHIRAD");
30             return View();
31         }
32         public List<sehirler> ilgetir()
33         {
34             List<sehirler> sehirler=db.sehirler.ToList();
35             return sehirler;
36         }
37         public ActionResult ilcegetir(int SEMIRID)
38         {
39             List<ilceler> selectlist=db.ilceler.Where(x=>x.SEMIRID==SEMIRID).ToList();
40             ViewBag.ilceList= new SelectList(selectlist,"ILCEID", "ILCEAD");
41             return PartialView("ilcegoster");
42         }
43         public ActionResult hastanegetir(int ILCEID)
44         {
45             List<hastaneler> selectlist = db.hastaneler.Where(x => x.ILCEID == ILCEID).ToList();
46             ViewBag.Hastanelist = new SelectList(selectlist, "HASTANEID", "HASTANEAD");
47             return PartialView("hastanegoster");
48         }
49         public ActionResult klinigetir(int HASTANEID)
50         {
51             List<kliniker> selectlist = db.klinikler.Where(x => x.HASTANEID == HASTANEID).ToList();
52             ViewBag.Kliniklist = new SelectList(selectlist, "KLINIKID", "KLINIKAD");
53             return PartialView("klinikgoster");
54         }
55         public ActionResult doktorgetir(int KLINIKID)
56         {
57             List<doktorlar> selectlist = db.doktorlar.Where(x => x.KLINIKID == KLINIKID).ToList();
58             ViewBag.Doktorlist = new SelectList(selectlist, "DOKTORID", "DOKTORAD");
59             return PartialView("doktorgoster");
60         }
61     }
62 }

```

Yukardaki kısımda giriş yaptıktan sonra gelen ana sayfanın fonksiyonu bulunmaktadır(Index) ve listeleme için değer döndürülmüştür. Geri kalan kısımda ise hastaneden randevu alma kısmına tıklandığında doktor aramak için ekrana çıkan filtreleme sayfasının fonksiyonlarıdır. Bu sayfa bir üstteki seçeneğe bağlı olacak şekilde cascading dropdownlist kullandığı için arama kriterlerine ait fonksiyonlar yazılmıştır ve sistemin çalışması için bunlar üzerinden partialviewlar oluşturulmuştur. Birinin üzerinden örnek verecek olursam ilcegetir fonksiyonu bir üstte seçilen şehire ait ilçelerin bilgisini getirmek için yazılan arkaplan kodunu içermektedir.

```

0 references
public ActionResult AsiRandevu()
{
    return View();
}

[HttpPost]
0 references
public ActionResult AsiRandevuAl(string data)
{
    randevular rande = new randevular();
    var dataObject = JsonConvert.DeserializeObject<randevular>(data);
    var kullaniciadi = User.Identity.Name;
    var kullanici = db.user.FirstOrDefault(x => x.USERTC == kullaniciadi);
    rande.USERID = kullanici.USERID;
    rande.DOKTORID = dataObject.DOKTORID;
    rande.RANDEVUTARIH = dataObject.RANDEVUTARIH;
    rande.RANDEVUSAAT = dataObject.RANDEVUSAAT;
    rande.RANDEVUTUR = dataObject.RANDEVUTUR;

    db.randevular.Add(rande);
    db.SaveChanges();
    return Json(true);
}

0 references
public ActionResult AileHekimi()
{
    return View();
}

[HttpPost]
0 references
public ActionResult AilehekimRandevuAl(string data)
{
    randevular rande = new randevular();
    var dataObject = JsonConvert.DeserializeObject<randevular>(data);
    var kullaniciadi = User.Identity.Name;
    var kullanici = db.user.FirstOrDefault(x => x.USERTC == kullaniciadi);
    rande.USERID = kullanici.USERID;
    rande.DOKTORID = dataObject.DOKTORID;
    rande.RANDEVUTARIH = dataObject.RANDEVUTARIH;
    rande.RANDEVUSAAT = dataObject.RANDEVUSAAT;
    rande.RANDEVUTUR = dataObject.RANDEVUTUR;
    db.randevular.Add(rande);
    db.SaveChanges();
    return Json(true);
}

```

Controllerların bu kısmında ise aşı ve aile hekimi randevusu alma işlemlerinin fonksiyonları bulunmaktadır. Bu fonksiyonlar viewdan gelen bilgilerin veri tabanına eklenmesini sağlar. Viewdan gelen bilgiler, kullanıcının ekranda seçtiği gün ve saat butonları gibi bilgilerdir. Bu bilgiler de veri tabanına eklenerek kullanıcıya ait randevuyu oluşturur.

```

105
106
107 0 references
108 public ActionResult Arama()
109 {
110     return View();
111 }
112
113 [HttpPost]
114 0 references
115 public ActionResult Arama(Class1 id)
116 {
117     var model=db.doktorlar.Where(x=>x.KLINIKID==id.KLINIKID).ToList();
118     var firma = db.doktorlar.Where(x => x.KLINIKID == id.KLINIKID).Select(x => x.DOKTORID).FirstOrDefault();
119     ViewBag.viewfirma = firma;
120     return View("Arama",model);
121 }
122 0 references
123 public ActionResult Doktor()
124 {
125     return View();
126 }
127 [HttpPost]
128 0 references
129 public ActionResult DoktorRandevuAl(string data,doktorlar p1)
130 {
131     var u = db.doktorlar.Find(p1.DOKTORID);
132     randevular rande = new randevular();
133     var dataObject = JsonConvert.DeserializeObject<randevular>(data);
134     var kullaniciadi = User.Identity.Name;
135     var kullanici = db.user.FirstOrDefault(x => x.USERTC == kullaniciadi);
136     rande.USERID = kullanici.USERID;
137     rande.DOKTORID = dataObject.DOKTORID;
138     rande.RANDEVUTARIH = dataObject.RANDEVUTARIH;
139     rande.RANDEVUSAAT = dataObject.RANDEVUSAAT;
140     rande.RANDEVUTUR = dataObject.RANDEVUTUR;
141     db.randevular.Add(rande);
142     db.SaveChanges();
143     return Json(true);
144 }
145 0 references
146 public ActionResult Goster()
147 {
148     var kullaniciadi = User.Identity.Name;
149     var kullanici = db.user.FirstOrDefault(x => x.USERTC == kullaniciadi);
150     var model = db.randevular.Where(x => x.USERID == kullanici.USERID).ToList();
151     return View(model);
152 }
153

```

Bu kısımda ise arama fonksiyonu,iki önceki kısımda bahsettiğim filtreleme işine yarayan fonksiyon sonucunda belli kriterlere göre şehir, ilçe, hastane ve klinik araması yapıldıktan sonra o kriterleri sağlayan doktorların bilgisini döndürülmesini sağlar. Doktorrandevual bölümü ise randevu alma ekranında kullanıcıların gününü ve saatini seçtikten sonra aldığı randevunun veri tabanına eklenmesini sağlar. Son olarak goster fonksiyonu ise kullanıcıların randevusunu görüntülemek için veritabanında karşılaştırma yaparak o kullanıcıya ait randevu bilgisini döndürür.

AdminP Controller:

Bu controller ve DoktorAdmin, KlinikAdmin controllerları aynı işlevi görmektedir. Farkları AdminP controllerı admin panelindeki hastane sayfası için, DoktorAdmin controllerı admin panelindeki doktor sayfası için, KlinikAdmin controllerı ise admin panelindeki klinik sayfası için yapılmıştır. Ama içerdiği fonksiyonlar bakımından hepsi farklı tablolar için CRUD işlemlerini gerçekleştirir. O yüzden bu 3 controllerı sadece Adminp controllerı üzerinden analatacağım.


```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Web;
5  using System.Web.Helpers;
6  using System.Web.Mvc;
7  using hastanerandevu.Models.Entities;
8
9  namespace hastanerandevu.Controllers
10 {
11     // GET: AdminP
12     public class AdminPController : Controller
13     {
14         // GET: AdminP
15         hastaneEntities db = new hastaneEntities();
16         public ActionResult Index()
17         {
18             var degerler = db.hastaneler.ToList();
19             return View(degerler);
20         }
21         [HttpGet]
22         public ActionResult Ekleme()
23         {
24             List<SelectListItem> firmas = (from x in db.sehirler.ToList()
25                                         select new SelectListItem
26                                         {
27                                             Text = x.SEHIRAD,
28                                             Value = x.SEHIRID.ToString(),
29                                         }).ToList();
30             ViewBag.firms = firmas;
31             List<SelectListItem> degerler = (from i in db.ilceler.ToList()
32                                         select new SelectListItem
33                                         {
34                                             Text = i.ILCEAD,
35                                             Value = i.ILCEID.ToString(),
36                                         }).ToList();
37             ViewBag.dgr = degerler;
38             ViewBag.SehirList = new SelectList(ilgetir(), "SEHIRID", "SEHIRAD");
39             return View();
40         }
41         [HttpPost]
42         public ActionResult Ekleme(hastaneler p1)
43         {
44             hastaneler ab = new hastaneler();
45             ab.HASTANEAD = p1.HASTANEAD;
46             ab.SEHIRID = p1.SEHIRID;
47             ab.ILCEID = p1.ILCEID;
48             db.hastaneler.Add(ab);
49             db.SaveChanges();
50             return RedirectToAction("Index");
51         }
52     }
53 }

```

Yukardaki kısımda admin panelinde hastanelerin listelenmesi için değer döndüren bir fonksiyon bulunuyor. Ardından ise adından da anlaşılacağı üzere ekleme işlemini sağlayan ve burdaki dropdownlistlerin id olarak değil string olarak dönmesini sağlayan fonksiyonlar bulunmaktadır.

```

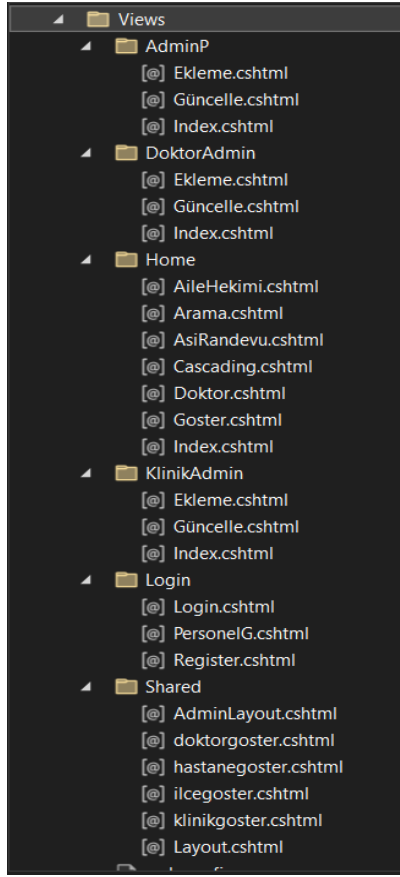
54 public ActionResult Sil(int id)
55 {
56     var hstnlr = db.hastaneler.Find(id);
57     db.hastaneler.Remove(hstnlr);
58     db.SaveChanges();
59     return RedirectToAction("Index");
60 }
61
62 0 references
63 public ActionResult Güncelle(int id)
64 {
65     var ur = db.hastaneler.Find(id);
66     List<SelectListItem> ils = (from x in db.sehirler.ToList()
67                                select new SelectListItem
68                                {
69                                    Text = x.SEHIRAD,
70                                    Value = x.SEHIRID.ToString(),
71                                }).ToList();
72     ViewBag.shr = ils;
73     List<SelectListItem> ilces = (from i in db.ilceler.ToList()
74                                select new SelectListItem
75                                {
76                                    Text = i.ILCEAD,
77                                    Value = i.ILCEID.ToString(),
78                                }).ToList();
79     ViewBag.ilc = ilces;
80     return View("Güncelle", ur);
81 }
82
83 0 references
84 public ActionResult Guncel(hastaneler p1)
85 {
86     var u = db.hastaneler.Find(p1.HASTANEID);
87     u.HASTANEAD = p1.HASTANEAD;
88     var fr = db.sehirler.Where(m => m.SEHIRID == p1.sehirler.SEHIRID).FirstOrDefault();
89     u.SEHIRID = fr.SEHIRID;
90     var mrk = db.ilceler.Where(m => m.ILCEID == p1.ilceler.ILCEID).FirstOrDefault();
91     u.ILCEID = mrk.ILCEID;
92     db.SaveChanges();
93     return RedirectToAction("Index");
94 }
95
96 1 reference
97 public List<sehirler> ilgetir()
98 {
99     List<sehirler> sehirler = db.sehirler.ToList();
100     return sehirler;
101 }
102
103 0 references
104 public ActionResult ilcegetir(int SEHIRID)
105 {
106     List<ilceler> selectlist = db.ilceler.Where(x => x.SEHIRID == SEHIRID).ToList();
107     ViewBag.Ilcelist = new SelectList(selectlist, "ILCEID", "ILCEAD");
108     return PartialView("ilcegoster");
109 }
110

```

Bu kısımda ise silme işlemini sağlayan bir fonksiyon ve hastanelerin bilgisini güncellemek için arka plan işlemlerini sağlayan kodlar bulunmaktadır. En alttaki kısım ise ekleme ve güncelleme ekranındaki dropdownlistlerin cascading olarak görüntülenmesi için yazılan fonksiyonlardır.

Frontend Kısım

Frontend kısmında yazdığım kodlar view dosyasında bulunmaktadır.



Frontend kısmında çok fazla dosyam olduğu için bunlardan en önemli olanların kodlarını paylaşıp anlatacağım. Öncelikle normal kullanıcı sayfasına ait dosyalarla başlayacağım. Login kısmında giriş sayfamaya(Login.cshtml) ait kodlar:

```
1  <@page "~/Views/Login/Login.cshtml" Layout="Layout.cshtml">
2
3  @{
4      Layout = null;
5  }
6
7  <doctype html>
8  <html>
9  <head>
10     <title>
11         Randevu Giriş
12     </title>
13     <meta charset="utf-8">
14     <meta name="viewport" content="width=device-width, initial-scale=1">
15     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet">
16     <style>
17         {
18             margin: 0;
19             padding: 0;
20             box-sizing: border-box;
21         }
22         body {
23             height: 90vh;
24             display: flex;
25             align-items: center;
26             justify-content: center;
27             font-family: 'Poppins';
28         }
29         h1 {
30             font-size: 36px;
31             margin-bottom: 25px;
32         }
33         .giris {
34             height: min-content;
35             padding: 20px;
36             border-radius: 12px;
37             background: #ffffff;
38         }
39         input[type="button"] {
40             font-size: 20px;
41             margin-top: 15px;
42         }
43         input[type="submit"] {
44             font-size: 20px;
45             margin-top: 15px;
46         }
47         .abc {
48             margin-bottom: 12px;
49         }
50         #image {
51             display: block;
52             margin-left: auto;
53             margin-right: auto;
54             width: 100px;
55         }
56         #icon {
57             width: 20px;
58         }
59         input::-webkit-outer-spin-button,
60         input::-webkit-inner-spin-button {
61             -webkit-appearance: none;
62             margin: 0;
63         }
64     </style>
65 </head>
66 <body>
67     <div>
68         <h1>
69             Randevu Giriş
70         </h1>
71         <div>
72             <div>
73                 <input type="button" value="Giriş" />
74             </div>
75             <div>
76                 <input type="button" value="Kayıt Ol" />
77             </div>
78         </div>
79     </div>
80 </body>
81 </html>
```

```

76 <body>
77 <@using (Html.BeginForm())
78 {
79 <@Html.AntiForgeryToken()
80 <@Html.ValidationSummary(true)
81
82 <div class="girisi">
83
84 
85 <h1 class="text-danger">Randevu Alma Sistemi</h1>
86
87 <form class="needs-validation">
88
89 <div class="abc was-validated">
90 
91 <label class="form-label" for="USERTC">T.C. Kimlik Numarası</label>
92 <br>
93 <@Html.EditorFor(model => model.USERTC, new
94 {
95 htmlAttributes = new { @class = "form-control" }
96 })
97 <div class="text-danger">
98 <@Html.ValidationMessageFor(model => model.USERTC, "", new { @class = "text-danger" })
99 </div>
100 </div>
101
102 <div class="abc was-validated">
103 
104 <label class="form-label" for="USERSIFRE">Parola</label>
105 <br>
106 <@Html.EditorFor(model => model.USERSIFRE, new { htmlAttributes = new { @class = "form-control" } })
107 <div class="text-danger">
108 <@Html.ValidationMessageFor(model => model.USERSIFRE, "", new { @class = "text-danger" })
109 </div>
110 </div>
111
112 <input class="btn btn-success w-100 text-bg-danger" type="submit" value="Giriş Yap">
113 <br>
114 <a href="/Login/Register">
115 <input class="btn btn-success w-100 text-bg-danger" type="button" value="Üye Ol">
116 </a>
117 <a href="/Login/PersonelG">
118 <input class="btn btn-success w-100 text-bg-danger" type="button" value="Personel Girişi">
119 </a>
120 <@if (ViewBag.Notification != null)
121 {
122 <div class="bg-light text-danger">
123 <@ViewBag.Notification
124 </div>
125 }
126 </form>
127 </div>
128
129 </body>
130 </html>
131

```

İlk bölümde bootstrap ve özel font kullandığım için onlara ait linkler bulunmaktadır. Ardından sayfamın görünüş olarak tasarımı için style tagı altında classlarımın css bilgileri bulunmaktadır. Body kısmında ise giriş sayfasının html kısmı bulunuyor. Bu sayfadaki çoğu divde formların daha güzel gözükmesi için bootstrap classları kullanılmıştır.

Cascading.cshtml: Bu sayfa hastane randevu al butonuna tıklandıktan sonra yönlendirilen sayfadır. Burada filtreleme yapılır.

Arama.cshtml: Bu sayfada yukardaki yapılan filtreleme işleminin ardından o filtreye uygun doktorların listelenmesi sağlanır.

Doktor.cshtml: Bu sayfa randevu alma sayfasıdır. Aynı zamanda AsiRandevu.cshtml ve AileHekimiRandevu.cshtml dosyalarıyla benzerdir.

Bu kısımda bu sayfaya ait cdnler ve css kısmı bulunmaktadır.

Bu kısımdaki ilk fonksiyonum tab panel içindir. Yani randevu alma sayfamda günlerin üstüne tıklanınca alt tarafına tüm saat butonlarının bulunduğu kısmın gelmesini sağlar. Alttaki fonksiyonlar ise saatin yanında bulunan slide up/down butonunun işlevli hale gelmesini sağlıyor. Yani aşağı kayarak o saate ait 3 tane randevu dakikasının gösterilmesini sağlıyor.

```
290 }
291 let selected = "";
292 document.getElementById('gunler').addEventListener('click', function (event) {
293
294     const collection = document.getElementsByClassName('tablinks');
295
296     for (let i = 0; i < collection.length; i++) {
297
298         if (event.target.classList[i] == collection[i].classList[i]) {
299             selected = event.target.classList[i] + " " + event.target.classList[2] + " " + event.target.classList[3];
300         }
301         else {
302
303         }
304     }
305 }
306 console.log(selected);
307 });
308 let selectedhour = "";
309 document.getElementById('Allbuttons').addEventListener('click', function (event) {
310
311     const collection = document.getElementsByClassName('butons');
312     for (let i = 0; i < collection.length; i++) {
313         if (event.target.classList[0] != "fa-sharp") {
314             collection[i].style.backgroundColor = "blue";
315         }
316     }
317     console.log(event.target.classList[0]);
318     for (let i = 0; i < collection.length; i++) {
319
320         if (event.target.classList[5] == collection[i].classList[5]) {
321
322             if (collection[i].style.backgroundColor != "rgb(0, 171, 182)") {
323                 if (collection[i].style.backgroundColor == "blue") {
324                     collection[i].style.backgroundColor = "green";
325                     selectedhour = collection[i].classList[5];
326                     console.log(selectedhour);
327                 }
328                 else {
329
330                     collection[i].style.backgroundColor = "blue";
331                 }
332             }
333
334         }
335     }
336 }
337 }
338 }
339 }
340 }
341 }
```

Bu iki fonksiyon ise gün ve saat butonlarına basıldığında onları dinlememi(okumamı) sağlıyor. Dinlediğim bilgileri ise bir değişkene atarak orda tutuyorum. Ayrıca alttaki fonksiyon saat butonlarına tıklandığında renginin değişmesini sağlamaktadır.


```
181 </script>
182
183 $( "#buyButton" ).click(function (event) {
184     console.log("aaa");
185     var url = window.location.pathname;
186     var sid = url.substring(url.lastIndexOf('/') + 1);
187     var ticket = {
188         "DOKTORID": sid,
189         "RANDEVUTUR": 3,
190         "RANDEVUTARIH": selected,
191         "RANDEVUSAAT": selectedhour,
192     }
193
194     event.preventDefault();
195
196     $.ajax({
197         type: "POST",
198         dataType: "json",
199         url: "/Home/DoktorRandevuAl",
200         data: {
201             data: JSON.stringify(
202                 ticket
203             ),
204             success: function (response) {
205                 window.location.href = "/Home/Index";
206             }
207         });
208
209 });
210
```

Burda ajax kullanarak kullanıcının randevu alma ekranında tıkladığı gün ve saat bilgilerinin randevu alma controllerına atılması sağlanıyor.

DoktorAdmin-Index.cshtml: Bu sayfada admin panelinde bulunan doktor bilgisi ekranının görüntülenmesi sağlanır. Adminp-Index ve KlinikAdmin-Index sayfaları da bu sayfayla benzerdir.

[illegible]

Bu sayfada doktorların adıyla arama yapabilmek için arama kutusu ve doktor sayısı fazla olduğu için sayfalama(pagedlist) kullanılmıştır.