**T.C.**

**SAKARYA ÜNİVERSİTESİ**

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

**BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ**

**WEB PROGRAMLAMA PROJE RAPORU**

**HAZIRLAYAN**

Esra AYVAZ

B211200045

**DANIŞMAN**

Dr. Öğr. Üyesi Fatma AKALIN

**ARALIK 2024**

**İÇİNDEKİLER**

[**GİRİŞ** 1](file:///C:\Users\DELL\AppData\Local\Temp\Temp1_B211200029_NurefşanErbilen_Ödev1.zip\B211200029_NurefşanErbilen.docx#_Toc121847145)

[**TEMEL TASARIM - LAYOUT** 2](file:///C:\Users\DELL\AppData\Local\Temp\Temp1_B211200029_NurefşanErbilen_Ödev1.zip\B211200029_NurefşanErbilen.docx#_Toc121847146)

[**TANITIM SAYFALARI** 3](file:///C:\Users\DELL\AppData\Local\Temp\Temp1_B211200029_NurefşanErbilen_Ödev1.zip\B211200029_NurefşanErbilen.docx#_Toc121847159)

**RANDEVU ALMA VE GÖRÜNTÜLEME** [5](file:///C:\Users\DELL\AppData\Local\Temp\Temp1_B211200029_NurefşanErbilen_Ödev1.zip\B211200029_NurefşanErbilen.docx#_Toc121847159)

**GİRİŞ EKRANI** 6

**EKLEME-SİLME-DÜZENLEME (CRUD)** 7

**TAKVİM YAPISI** 9

**ACİL DURUM BELİRTME** 12

**CODE-FIRST YAKLAŞIMI** 13

**PROJENİN SON HALİ** 14

**GİRİŞ**

Bu proje, bir tıp fakültesinin pediatri bölümüne yönelik olarak, dijital bir yönetim ve iletişim platformunun geliştirilmesi amacıyla hayata geçirilmiştir. Projenin temel amacı, pediatri asistanlarının nöbet süreçlerini düzenlemek, bölüm ve öğretim üyeleri ile olan iletişimlerini kolaylaştırmak ve hastane içi iletişim ağını daha etkin bir şekilde yönetmektir. Bu doğrultuda, modern sağlık hizmetleri ve akademik gerekliliklere uygun, kapsamlı bir web tabanlı uygulama geliştirilmiştir.

Proje kapsamında, hem asistanların hem de öğretim üyelerinin ihtiyaçlarını karşılamak amacıyla çeşitli özellikler tasarlanmıştır. Asistanların nöbet programlarının oluşturulması ve yönetilmesi, öğretim üyeleriyle olan iletişimlerinin planlanması ve gerekli duyuruların etkin bir şekilde paylaşılabilmesi için admin paneli ve kullanıcı paneli olmak üzere iki ayrı arayüz geliştirilmiştir. Bu paneller, kullanıcı dostu bir tasarım anlayışıyla hazırlanmış ve erişilebilir bir deneyim sunmayı hedeflemiştir.

Admin paneli, asistan, öğretim üyesi, bölüm ve acil durum bilgilerini düzenlemek için kapsamlı bir araç sunmaktadır. Nöbet takvimlerinin oluşturulması, randevu zamanlarının belirlenmesi ve acil durum duyurularının eklenmesi gibi kritik işlevler, bu panel aracılığıyla yönetilmektedir. Ayrıca, kullanıcıların ihtiyaçlarına göre kolayca güncellenebilir bir yapı sağlanmıştır.

Kullanıcı paneli, asistanlar ve öğretim üyeleri için planlanan bilgilerin görüntülenmesini ve kullanıcının ihtiyaç duyduğu işlemleri gerçekleştirmesini sağlamaktadır. Bu panel aracılığıyla asistanlar, nöbet programlarını görebilir, öğretim üyelerinin müsaitlik durumlarına göre randevu alabilir ve acil durum duyurularına erişebilirler. Bölüm tanıtım sayfaları ise asistanların nöbet tutacakları bölümlere ilişkin detaylı bilgilere ulaşmasını sağlayarak, ön hazırlık yapmalarına olanak tanımaktadır.

Projenin bir diğer önemli bileşeni olan randevu sistemi, asistanların öğretim üyeleriyle görüşmelerini planlı ve düzenli bir şekilde yapmalarını sağlamaktadır. Öğretim üyelerinin belirlediği müsaitlik saatleri üzerinden randevu talebi oluşturma, düzenleme ve iptal etme gibi işlemler kolayca gerçekleştirilebilmektedir. Son olarak, acil durum sayfası, hastane içi iletişim ve koordinasyonun güçlendirilmesi için tasarlanmıştır. Bu sayfa, admin paneli üzerinden girilen acil durum haberlerinin, kullanıcı paneli aracılığıyla son kullanıcılara hızlı bir şekilde ulaştırılmasını sağlamaktadır.

Bu proje, pediatri bölümünün asistanlar, öğretim üyeleri ve yöneticiler arasındaki işleyişini dijitalleştirerek, daha etkin ve organize bir sistem oluşturmayı hedeflemektedir. Geliştirilen platform, kullanıcı dostu arayüzü ve geniş kapsamlı özellikleri ile modern hastane yönetimi için örnek teşkil etmektedir.

**TEMEL TASARIM – LAYOUT**

Projenin temel tasarımında, tutarlı ve kullanıcı dostu bir deneyim sağlamak amacıyla layout yapısı kullanılmıştır. Bu yapı sayesinde, arka plan rengi, navbar (navigasyon çubuğu) ve footer gibi tasarım öğeleri merkezi bir noktada tanımlanmış ve tüm sayfalarda tekrar kullanılabilir hale getirilmiştir.

Arka plan tasarımı, proje temasına uygun bir renk seçimiyle sade ve modern bir görünüm sunacak şekilde belirlenmiştir. Navbar, kullanıcıların sayfalar arasında kolaylıkla gezinmesini sağlamak amacıyla işlevsel ve minimalist bir yapıda tasarlanmıştır. Footer kısmında ise proje ile ilgili genel bilgiler ve bağlantılar düzenli bir şekilde sunulmuştur.

Layout yapısının kullanılması, hem tasarımın tüm sayfalarda tutarlı görünmesini sağlamış hem de kod tekrarını azaltarak geliştirme sürecini daha verimli hale getirmiştir. Bu sayede, tüm sayfalar ortak bir çerçeveye oturtulmuş ve kullanıcılara estetik ve düzenli bir deneyim sunulmuştur.

**NAVBAR**;

<nav class="navbar navbar-expand-lg navbar-dark" style="background: #4682B4; font-family: sans-serif; font-weight:bold;">

<div class="container">

<a class="navbar-brand" href="/" style=" font-size: 40px; padding: 25px;">ATSAHLAS</a>

<button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarNav">

<ul class="navbar-nav ms-auto">

<li class="nav-item mr-10">

<a class="nav-link active" href="/">ANASAYFA</a>

</li>

@if (Context.Session.GetString("Role") != "Admin")

{

<li class="nav-item mr-10">

<a class="nav-link active" href="/Asistantanitim">ASİSTANLAR</a>

</li>

<li class="nav-item mr-10">

<a class="nav-link active" href="/Ogretimuyesitanitim">ÖĞRETİM ÜYELERİ</a>

</li>

<li class="nav-item mr-10">

<a class="nav-link active" href="/Bolumlertanitim">BÖLÜMLER</a>

</li>

<li class="nav-item mr-10">

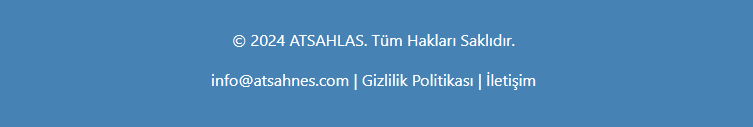
<a class="nav-link" href="/Appointments/Create">Randevu Al</a>

</li>

<li class="nav-item mr-10">

<a class="nav-link" href="/Account/Login">Giriş Yap</a>

</li>

**FOOTER;**

**TANITIM SAYFALARI**

**TANITIM SAYFALARI**

Asistanlar, öğretim üyeleri ve bölümler için tanıtım sayfaları oluşturulmuştur. Asistan ve öğretim üyeleri, estetik bir görünüm sunan kart formatında tasarlanmış, iletişim bilgileri kolayca erişilebilir hale getirilmiştir. Bölümler için ise her fotoğrafın üzerine gelindiğinde tanıtıcı bilgilerin görüntülenmesini sağlayan dinamik bir CSS efekti uygulanmıştır. Bu sayfalar, kullanıcıların ihtiyaç duyduğu bilgilere hızlı ve şık bir şekilde ulaşmasını sağlamaktadır.

**ÖĞRETİM ÜYESİ;**

<div class="container">

<h2>ÖĞRETİM ÜYELERİ</h2>

<div class="row">

<!-- Öğretim Üyesi 1 -->

<div class="col-md-6 col-lg-3">

<div class="card">

<img src="/css/unnamed.jpg" class="card-img-top" alt="Öğretim Üyesi Fotoğrafı">

<div class="card-body">

<h5 class="card-title">Dr. Selim Kara</h5>

<p class="card-text">Telefon: +90 555 112 34 56</p>

<p class="card-text">E-posta: selim.kara@university.com</p>

</div>

</div>

</div>

**ASİSTANLAR;**

<div class="container">

<h2 style="color: #4682B4 !important;">PEDİATRİ ASİSTANLARI</h2>

<div class="row row-cols-1 row-cols-md-3 row-cols-lg-5 g-4">

<!-- Asistan 1 -->

<div class="col-md-2">

<div class="card">

<img src="/css/unnamed.jpg" class="card-img-top" alt="Asistan Fotoğrafı">

<div class="card-body">

<h5 class="card-title">Emine Dönmez</h5>

<p class="card-text">Telefon: +90 532 123 45 67</p>

<p class="card-text">E-posta: emine.donmez@hospital.com</p>

</div>

</div>

 </div>

**metin, ekran görüntüsü, yazı tipi, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**BÖLÜMLER(CSS);**

.section-card {

position: relative;

overflow: hidden;

border-radius: 15px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

transition: transform 0.3s;

margin-bottom: 30px;

}

.section-card:hover {

transform: scale(1.05);

}

.section-img {

width: 100%;

height: 400px; /\* Daha büyük fotoğraf \*/

object-fit: cover;

transition: opacity 0.3s;

}

.section-card:hover .section-img {

opacity: 0.3;

}

.section-info {

Asitan, öğretim üyesi ve bölümler kısmında kullandığım fotoğrafları internetten bilgisayarıma indirip projemdeki wwrooot dosyasının içerisindeki css dosyasına ekledikten sonra css kısmında fotoğrafları kullanabilir hale geldim.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

position: absolute;

bottom: 0;

left: 0;

width: 100%;

background-color: rgba(0, 0, 0, 0.7);

color: white;

padding: 20px;

opacity: 0;

transition: opacity 0.3s;

text-align: center;

}

.section-card:hover .section-info {

opacity: 1;

}

.section-title {

font-size: 24px;

font-weight: bold;

}

.section-description {

font-size: 16px;

margin-top: 10px;

**metin, ekran görüntüsü, insan yüzü, adam, insan içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**RANDEVU ALMA VE GÖRÜNTÜLEME**

Projede randevu sistemi için bir Controller, Model, Migration ve View kullandım. Modelde randevu bilgilerini (ad, tarih, bölüm, iletişim) tanımladım ve bu yapıya uygun veritabanı tablosunu migration ile oluşturdum. Controller, randevu oluşturma, listeleme, düzenleme ve silme işlemlerini yönetti. View kısmında kullanıcıların kolayca randevu talebi oluşturabileceği bir form tasarladım. Bu yapı sayesinde randevu işlemleri düzenli ve kullanıcı dostu bir şekilde yönetildi.

**CONTROLLER;**

public class AppointmentsController : Controller

{

private readonly AppDbContext \_context;

public AppointmentsController(AppDbContext context)

{

\_context = context;

}

**MODEL;**

public class Appointment

{

[Key, DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

[Required]

public int DoctorId { get; set; } // Hangi doktorla randevu

public int AssistantId { get; set; } // Hangi asistanla randevu

public DateTime Date { get; set; } // Tarih (Gün)

public TimeSpan Time { get; set; } // Saat

public required string PatientName { get; set; } // Hasta adı

}

**VİEW;**

<div class="container">

<div class="row justify-content-center">

<div class="col-md-8 col-lg-6">

<div class="card shadow-lg border-0 rounded-lg">

<div class="card-header text-center" style="background-color: #4682B4; color: white; border-radius: 10px 10px 0 0;">

<h4 class="font-weight-bold">Randevu Formu</h4>

</div>

**metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

Eklenilen randevular, admin paneline girdikten sonra navbardaki randevular kısmından görülebilir.

metin, ekran görüntüsü, yazı tipi, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

@foreach (var appointment in Model)

{

<tr>

<td>@appointment.PatientName</td>

<td>

@doctors?.FirstOrDefault(d => d.Id == appointment.DoctorId)?.Name

</td>

<td>

@assistants?.FirstOrDefault(a => a.Id == appointment.AssistantId)?.Name

</td>

<td>@appointment.Date.ToShortDateString()</td>

<td>@appointment.Time</td>

</tr>

}

**GİRİŞ EKRANI**

Projede kullanıcı giriş sistemi için bir **Controller, Model** ve **View** kullandım. Modelde kullanıcı bilgilerini (kullanıcı adı, şifre) tanımladım. Controller, kullanıcı giriş işlemlerini doğrulama ve yönlendirme işlevlerini üstlendi. View kısmında kullanıcıların giriş yapabileceği sade ve işlevsel bir ekran tasarladım. Bu yapı sayesinde güvenli ve kullanıcı dostu bir login sistemi oluşturuldu.

**CONTROLLER;**

[HttpGet]

public IActionResult Login()

{

return View();

}

// POST: Account/Login

[HttpPost]

[ValidateAntiForgeryToken]

public IActionResult Login(string username, string password)

{

// Kullanıcıyı doğrulama

var user = \_context.Users.FirstOrDefault(u => u.Username == username && u.Password == password);

**MODEL;**

public class LoginViewModel

{

[Required]

[Display(Name = "Username")]

public string Username { get; set; }

[Required]

[DataType(DataType.Password)]

[Display(Name = "Password")]

public string Password { get; set; }

}

**VİEW;**

<div class="form-group mb-4">

<**label** **asp-for**="Username" class="control-label" style="font-weight: 600;"></**label**>

<**input** **asp-for**="Username" class="form-control form-control-lg shadow-sm neon-border" placeholder="Kullanıcı Adı" />

<**span** **asp-validation-for**="Username" class="text-danger"></**span**>

</div>

<!-- Password Field -->

<div class="form-group mb-4">

<**label** **asp-for**="Password" class="control-label" style="font-weight: 600;"></**label**>

<**input** **asp-for**="Password" **type**="password" class="form-control form-control-lg shadow-sm neon-border" placeholder="Şifre" />

<**span** **asp-validation-for**="Password" class="text-danger"></**span**</div>

**metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**EKLEME-SİLME-DÜZENLEME (CRUD)**

Admin panelinde, asistan, öğretim üyesi ve bölümler için **CRUD** işlemleri (ekleme, düzenleme, silme) yapılabilmektedir. Bu işlemler için önceden tanımladığımız **controller** ve **modelleri** kullandım. Her bir işlem için ayrı **view** dosyaları oluşturdum ve **CSS** tasarımını düzenlemek için ayrıca bir view ekledim. Bu sayede admin paneli üzerinden tüm veriler kolayca yönetilebilir hale getirildi.

**ASİSTAN EKLEME;**

<div class="card shadow-lg border-0 rounded-lg">

<div class="card-header text-center" style="background-color: #4682B4; color: white; border-radius: 10px 10px 0 0;">

<h4 class="font-weight-bold">Yeni Asistan Ekle</h4>

</div>

<div class="card-body p-5">

<**form** **asp-action**="Create" method="post">

**ÖĞRETİM ÜYESİ DÜZENLEME;**

<div class="card shadow-lg border-0 rounded-lg">

<div class="card-header text-center" style="background-color: #4682B4; color: white; border-radius: 10px 10px 0 0;">

<h4 class="font-weight-bold">Öğretim Üyesini Düzenle</h4>

</div>

<div class="card-body p-5">

<h3 class="text-center">Bu öğretim üyesini düzenlemek istediğinizden emin misiniz?</h3>

**BÖLÜM SİLME;**

<div class="card shadow-lg border-0 rounded-lg">

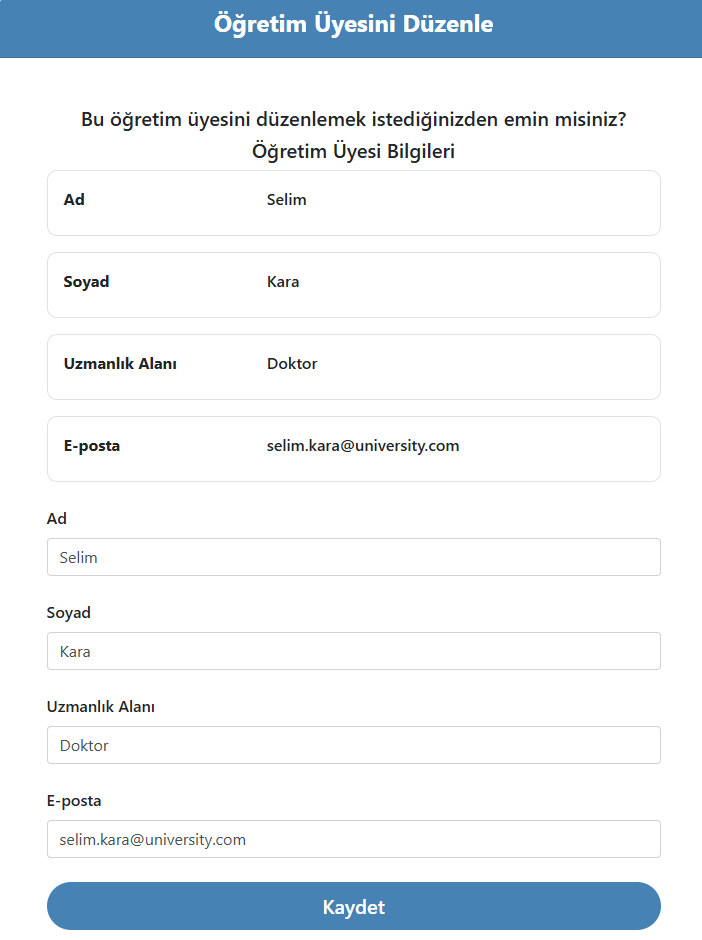
<div class="card-header text-center" style="background-color: #4682B4; color: white; border-radius: 10px 10px 0 0;">

<h4 class="font-weight-bold">Bölümü Sil</h4>

</div>

<div class="card-body p-5">

<h3 class="text-center">Bu bölümü silmek istediğinizden emin misiniz?</h3>

****

**metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

****

**Adminin gördüğü tasarım kısmında hepsi için index.cshtml sayfası oluşturduğumu belirtmiştim;**

<a href="@Url.Action("Create")" class="btn btn-success">Yeni ……… Oluştur</a>

.

.

.

<td>

<a href="@Url.Action("Edit", new { id = department.Id })" class="btn btn-warning btn-sm">Düzenle</a>

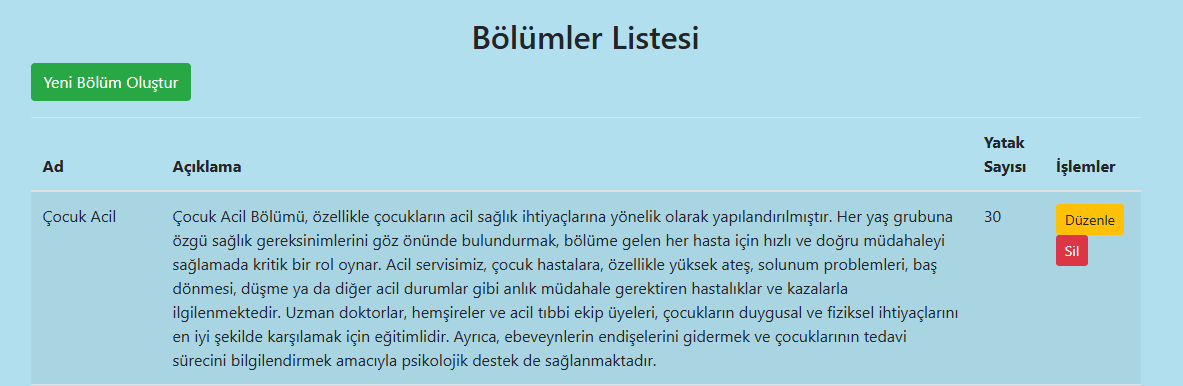
<a href="@Url.Action("Delete", new { id = department.Id })" class="btn btn-danger btn-sm">Sil</a>

</td>

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

****

****

**TAKVİM YAPISI**

Admin Paneline girdikten sonra anasayfada bir mesajla karşılaşırsınız. Bu mesajda yeni takvim oluşturabildiğinizden bahseder. Yeni takvim oluştur butonuna bastıktan sonra bir nöbet oluşturabilirsiniz. Bu takvimi silebilir veya düzenleyebilirsiniz. Bu oluşturulan takvimler sadece admin panelinde silinip düzenlenebilir veya oluşturulabilir ancak kullanıcı panelinde bu işlemler yapılamaz sadece oluşturulmuş hazırda bulunan takvimler görüntülenir.

**CONTROLLER;**

public class ScheduleController : Controller

{

private readonly AppDbContext \_context;

public ScheduleController(AppDbContext context)

{

\_context = context;

}

**MODEL;**

public class Schedule

{

public int Id { get; set; }

public string Title { get; set; }

public DateTime StartTime { get; set; } // Nöbet başlangıç zamanı

public DateTime EndTime { get; set; } // Nöbet bitiş zamanı

public string AssignedTo { get; set; } // Asistan veya doktor adı

public string Department { get; set; } // Bölüm bilgisi

}

**VİEW;**

<!-- Takvim Görünümü -->

<div id="calendar" class="mb-4"></div>

<h3>TAKVİM YAPISI</h3>

<!-- Kart Görünümü -->

<div class="row row-cols-1 row-cols-md-2 g-4">

@foreach (var schedule in Model)

{

<div class="col">

<div class="card shadow-sm">

<div class="card-body">

<h5 class="card-title">@schedule.Title</h5>

<p class="card-text">

<strong>Başlangıç:</strong> @schedule.StartTime<br />

<strong>Bitiş:</strong> @schedule.EndTime<br />

<strong>Atanan Kişi:</strong> @schedule.AssignedTo<br />

<strong>Bölüm:</strong> @schedule.Department

**metin, ekran görüntüsü, yazı tipi, daire içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**TAKVİMİ OLUŞTURMA;**

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

<div id="calendar" class="mb-4"></div>

<script>

document.addEventListener('DOMContentLoaded', function () {

var calendarEl = document.getElementById('calendar');

var calendar = new FullCalendar.Calendar(calendarEl, {

initialView: 'dayGridMonth', // Month view

headerToolbar: {

left: 'prev,next today',

center: 'title',

right: 'dayGridMonth,timeGridWeek,timeGridDay'

},

events: [

@foreach (var schedule in Model)

{

@:

@:{

@:title: "@schedule.Title",

@:start: "@schedule.StartTime.ToString("yyyy-MM-ddTHH:mm:ss")",

@:end: "@schedule.EndTime.ToString("yyyy-MM-ddTHH:mm:ss")",

@: },

}

]

});

calendar.render();

});

**TAKVİMİ DÜZENLEME;**

@model webOdev.Models.Schedule

<div class="container">

<div class="row justify-content-center">

<div class="col-md-10 col-lg-8">

<div class="card shadow-lg border-0 rounded-lg">

<div class="card-header text-center" style="background-color: #4682B4; color: white; border-radius: 10px 10px 0 0;">

<h4 class="font-weight-bold">Takvimi Düzenle</h4>

</div>

**TAKVİMİ SİLME;**

@model webOdev.Models.Schedule

<div class="container">

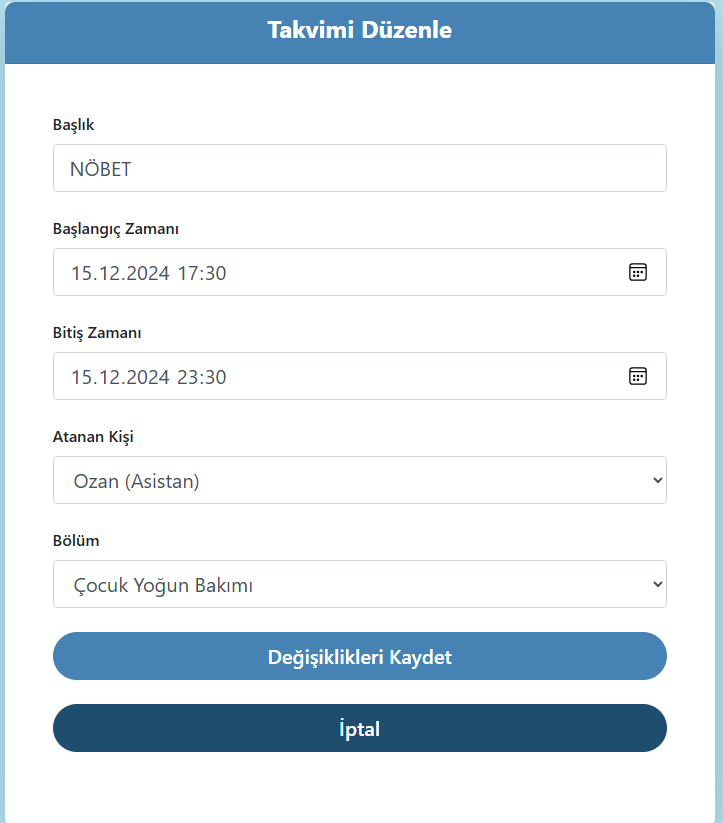
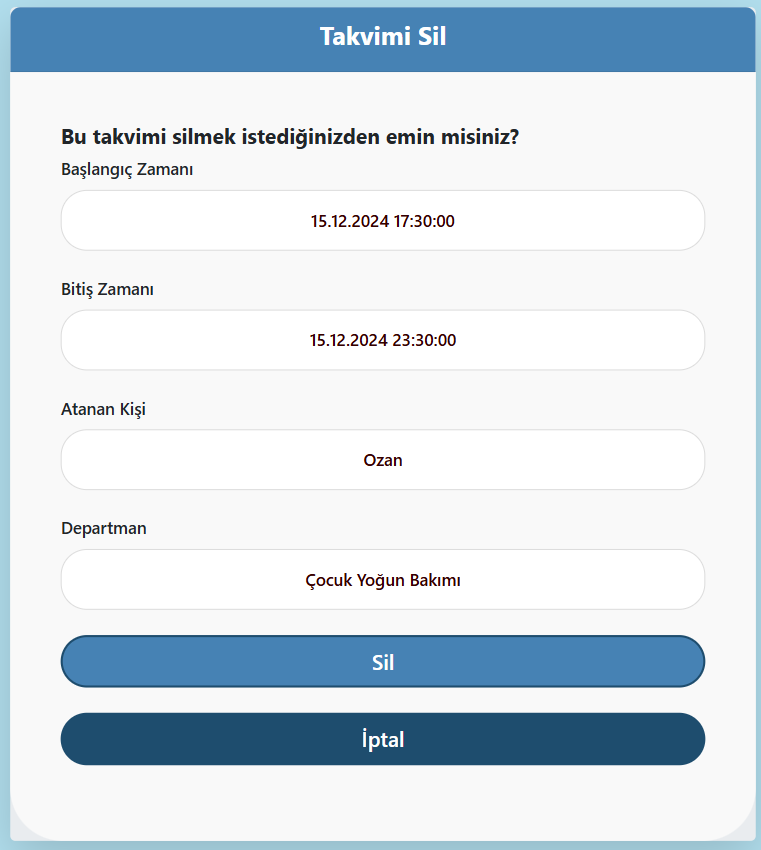
<div class="row justify-content-center">

<div class="col-md-10 col-lg-8">

<div class="card shadow-lg border-0 rounded-lg">

<div class="card-header text-center" style="background-color: #4682B4; color: white; border-radius: 10px 10px 0 0;">

<h4 class="font-weight-bold">Takvimi Sil</h4>

**** </div>

**ACİL DURUM BELİRTME**

Admin panelinin anasayfasında takvim oluştur butonun yanında acil durum oluştur butonu yer almaktadır. Bu butona basınca acil durumun başlığı, açıklaması yer almaktadır. Bunları doldurduktan sonra gönder tuşuna basınca acil durum gönderilmiştir başarı mesajı anasayfa ekranında belirir.

**MODEL;**

public class Emergency

{

public int Id { get; set; }

public string Title { get; set; }

public string Description { get; set; }

public DateTime CreatedAt { get; set; }

}

**VİEW;**

<**form** **asp-action**="CreateEmergency" method="post">

<!-- Acil Durum Başlığı Alanı -->

<div class="form-group mb-4">

<**label** **asp-for**="Title" class="control-label" style="font-weight: 600;">Acil Durum Başlığı</**label**>

<**input** **asp-for**="Title" class="form-control form-control-lg shadow-sm rounded-pill p-3" placeholder="Acil durum başlığını girin" />

<**span** **asp-validation-for**="Title" class="text-danger"></**span**>

</div>

<!-- Acil Durum Mesajı Alanı -->

<div class="form-group mb-4">

<**label** **asp-for**="Message" class="control-label" style="font-weight: 600;">Acil Durum Mesajı</**label**>

<**textarea** **asp-for**="Message" class="form-control form-control-lg shadow-sm p-3" placeholder="Acil durumu açıklayın" rows="6"></**textarea**>

<**span** **asp-validation-for**="Message" class="text-danger"></**span**>

</div>

<!-- Acil Durum Gönder Butonu -->

<div class="form-group text-center mb-4">

<button type="submit" class="btn btn-lg btn-block neon-border" style="background-color: #4682B4; color: white; font-weight: 600; border-radius: 30px; padding: 12px 30px;">Acil Durumu Gönder</button>

</div>

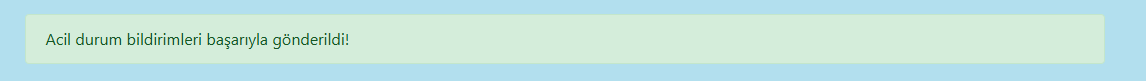
</**form**>

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

TempData["SuccessMessage"] = "Acil durum bildirimleri başarıyla gönderildi!";

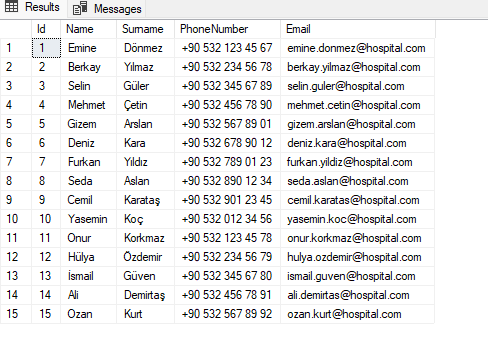
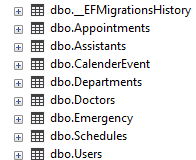
metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldureturn RedirectToAction("Index");

****

**CODE-FIRST YAKLAŞIMI**

Projede Code First yaklaşımını uygulayarak, veritabanı modellerini oluşturmak için önce gerekli model sınıflarını (Appointment.cs, Assistant.cs, CalendarEvent.cs, Department.cs, Doctor.cs, Emergency.cs, Schedule.cs, User.cs) tanımladım. Her bir sınıf, veritabanında karşılık gelen tablonun yapısını belirledi. Model sınıflarını oluşturduktan sonra, Add-Migration komutunu kullanarak bu değişiklikleri içeren migration dosyalarını oluşturdum. Ardından, Update-Database komutunu çalıştırarak bu migration’ları veritabanına uyguladım. Sonuç olarak, her bir model için ilgili veritabanı tabloları (8 tablo) SQL veritabanımda otomatik olarak oluşturuldu. Bu süreç, Code First ilkesine uygun olarak, veritabanı şemasını doğrudan koddan oluşturmayı sağladı.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, diyagram, plan, teknik çizim içeren bir resim

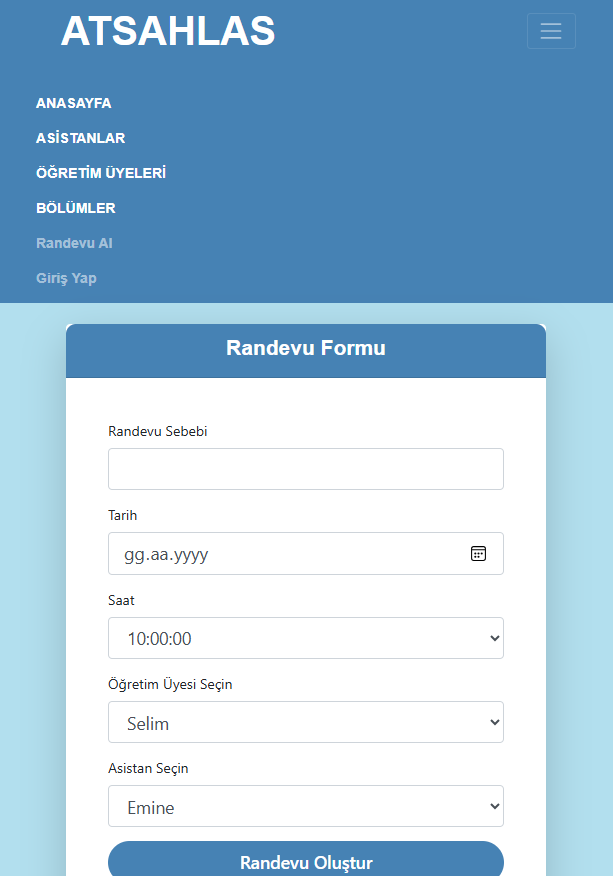
Açıklama otomatik olarak oluşturuldu

**PROJENİN SON HALİ**

**ADMİN PANELİ ANASAYFA;**

**metin, ekran görüntüsü, yazı tipi, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**RESPONSİVE TASARIM;**