

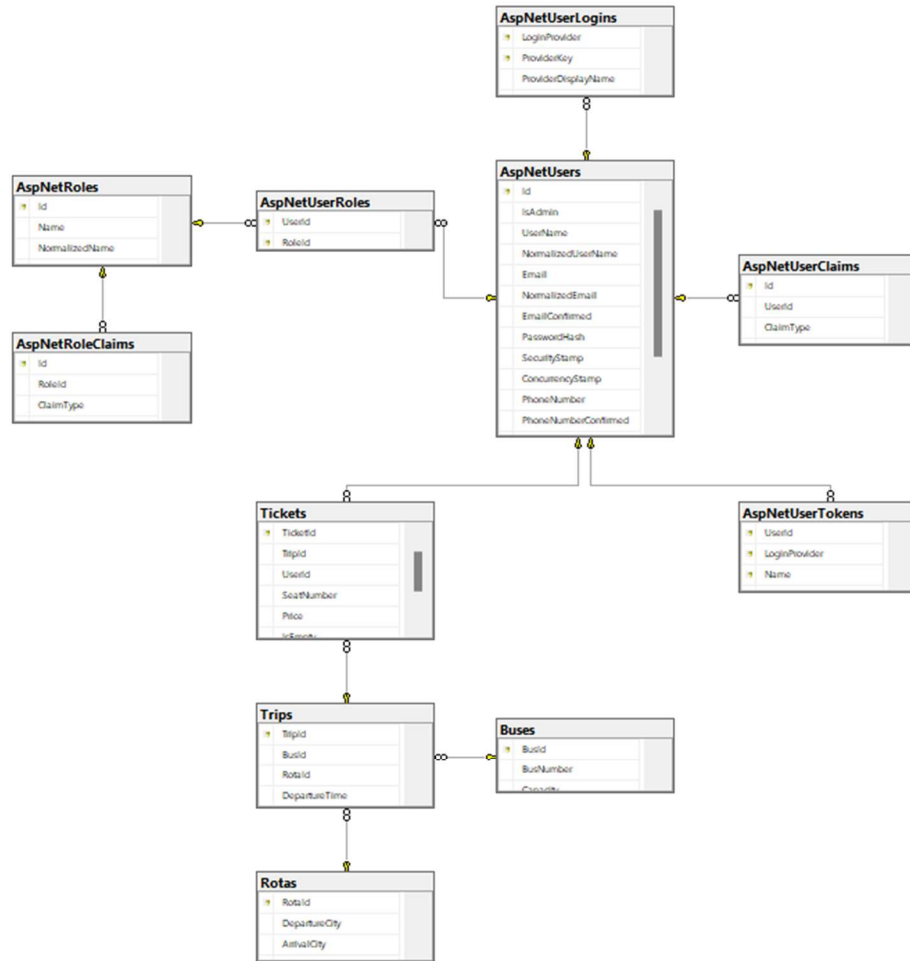
Özet

Çalışmada ASP.NET Core Identity ve Iyzico ödeme sistemi kullanarak MVC mimarisi ile ve aynı zamanda entityframework kullanılarak MSSQL DB ‘den gelen verilerin işlenerek bir otobüs bilet satış uygulaması oluşturmak amaçlanmıştır.

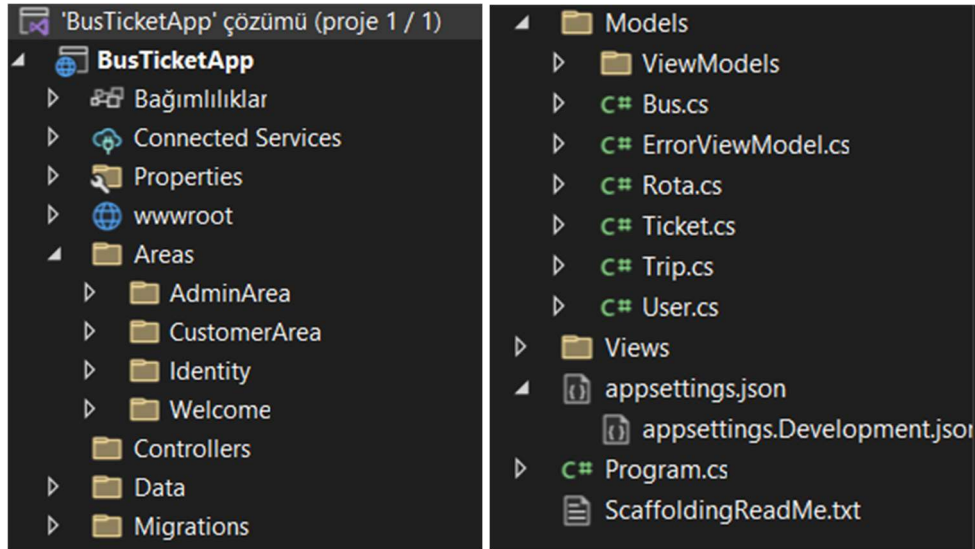
Database Modeli

ApplicationDbContext adlı bir DbContext sınıfı kullanılarak çeşitli modeller arasında ilişkiler kurulmuş ve veritabanındaki tabloların temsili yapılmıştır. Bu sınıf, uygulamanın veritabanına erişim sağlar.

Database Diyagramı



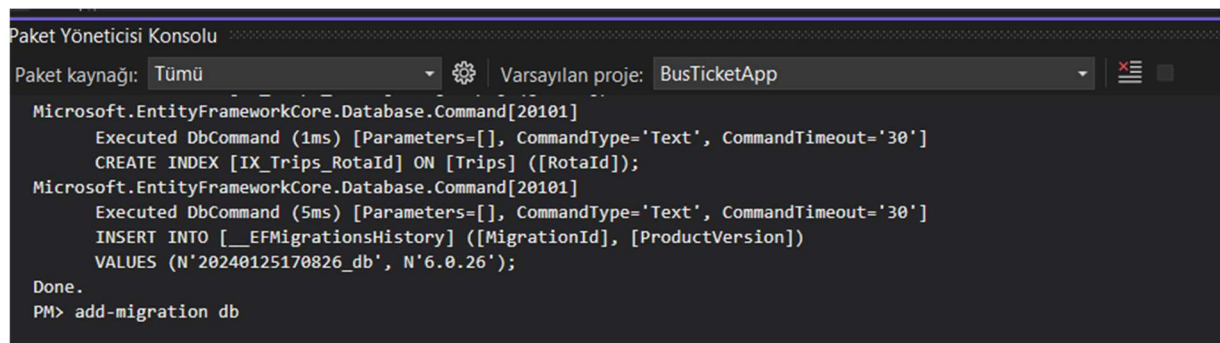
Çözüm Gezini



Proje Arealar ile oluşturulmuştur. Çözüm gezgininde görüldüğü üzere 3 adet Area kullanılmıştır. Her bir area kendi içerisinde MVC yapısına sahiptir. Böylece admin ve customer alanları ayrılmıştır. Ek olarak uygulamaya giriş yapan kişinin karşılanacağı bir sayfa Welcome Area içerisinde oluşturulmuştur.

EntityFramework Bağlantısı

Oluşturulan Database modelleri entity yardımıyla mssql tarafında tablolara karşılık gelmektedir. Ayrıca Identity kütüphanesi kullanıldığı için User tablosu ASPNETUSERS tablosuna karşılık gelmektedir. Bu noktada kullanıcı işlemleri b tablodan kontrol edilmektedir.



Paket Yönetici konsolu sayesinde gerekli migrationlar düzenlenmiş ve db update edilmiştir.

Welcome Controller

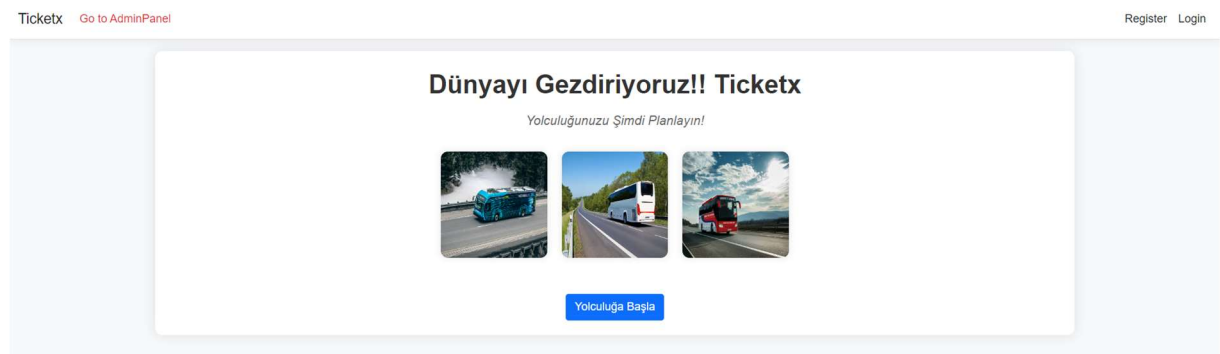
```
namespace BusTicketApp.Areas.Welcome.Controllers
{
    [Area("Welcome")]
    1 başvuru
    public class WelcomeController : Controller
    {
        private readonly UserManager<User> _userManager;
        ApplicationDbContext db;

        0 başvuru
        public WelcomeController(ApplicationDbContext db, UserManager<User> userManager)
        {
            this.db = db;
            _userManager = userManager;
        }
        0 başvuru
        public IActionResult Index()
        {
            var userId = _userManager.GetUserId(User);

            return View();
        }
    }
}
```

Figürde uygulamaya giriş yapan kullanıcının yönlendirildiği controller görünmektedir. En üst satırda [Area("Welcome")] ile bu controller'ın Welcome Area'ya ait olduğu gösterilmiştir. Aynı zamanda Identity kütüphanesinin sayesinde giriş yapan kullanıcının bilgilerini almak için UserManager sınıfı kontroller metotuna imza olarak verilmiştir. Bu kontroller sadece görüntü amaçlı olduğundan dolayı herhangi bir işlem yapmaksızın View razor sayfasına yönlendirme yapar.

Welcome Page Ekran Görüntüsü



Identity Service

Identity kütüphanesi kendi içerisinde Interface'ler servisler barındırır. Bu projede Login ve Register Sayfaları implamente edilmiştir. Kullanılan login ve register servisleri sayesinde

kullanıcı giriş ve kayıt işlemleri yapabilmektedir. Db'deki ASPNETUSERS tablosu içerisinde bulunan kolonlara erişim sağlayan kütüphane verileri güncellemektedir.

Login ve Register Page Ekran Görüntüleri

Log in

☐ Remember me?

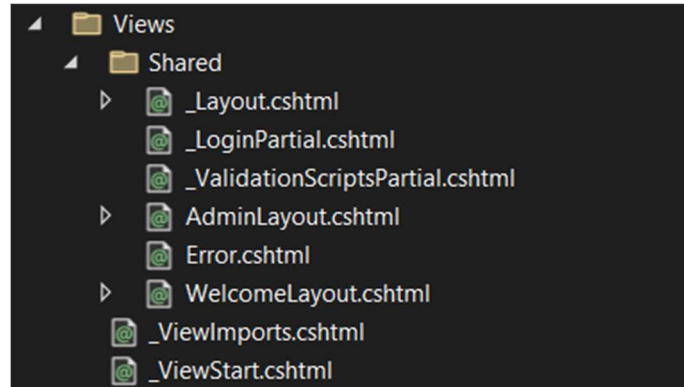
Log in

[Register as a new user](#)

Register

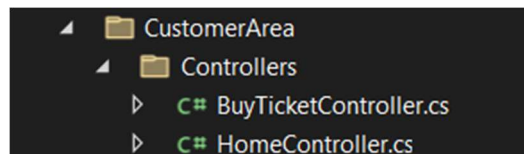
Register

Layout Sayfaları



Görüntüde görüldüğü üzere 3 farklı layout yapısı bulunmaktadır. Her bir Area için farklı hazırlanan bu layoutlar sayesinde kullanıcı dostu bir arayüz oluşturmak istenmiştir.

Customer Area Controller Yapıları



Bilet satın alma işlemi yapmak isteyen kullanıcı welcome ekranından sonra customer area tarafından HomeController'a yönlendirilir ve burada karşılanır. HomeController yapısı içerisinde 2 IActionResult metodu bulundurulur.


```
0 başvuru  
public IActionResult Index()  
{  
    return View();  
}  
[HttpPost]  
0 başvuru  
public IActionResult Index(string from, string to, DateTime date) ...  
  
0 başvuru  
public IActionResult FoundedTrips() ...  
  
[HttpPost]  
  
0 başvuru  
public IActionResult FoundedTrips(Trip fromviewmodel) ...
```

Index metodu sayesinde kullanıcıya seyahat bilgilerinin gösterildiği bir form UI gösterilir ve buradan alınan veriler ile DB'de karşılaştırma yapılır. Eğer db'de uygun sefer bulunursa FoundedTrips metodu viewinde listelenir.

Ticketx

Nereden:

Nereye:

Tarih:
 

[Bilet Ara](#)

Aranan Tarihte Bulunan Biletler

Trip Id	Departure Time	Bus Id	Bilet Seç
11	1.01.2024 10:00:00	1	Bileti Al
27	1.01.2024 10:00:00	10	Bileti Al

[Back to Home](#)

Burada tercih edilen otobüs seferi seçilir ve kullanıcı otobüs koltuk seçim ekranına yönlendirilir. Otobüs seçim ekranı ise BuyTicketController tarafından yönetilir.

```
0 başvuru  
public IActionResult Index()  
[HttpPost]  
0 başvuru  
public IActionResult Index(string koltuknumber,string secilensefernumber,string totalamount)  
0 başvuru  
public IActionResult Order()  
  
0 başvuru  
public IActionResult Success() {
```

BuyTicketController bünyesinde 3 adet metod bulundurulur. Index metodu otobüs koltuğu seçmesi için kullanıcıya arayüz sağlar ve post metodu sayesinde seçilen koltuk numarasını, seçilen sefer numarasını ve toplam ödenmesi gereken miktarı alır.

En Ticketx Koltuğu Seçiniz

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40

Seçilen Koltuklar:

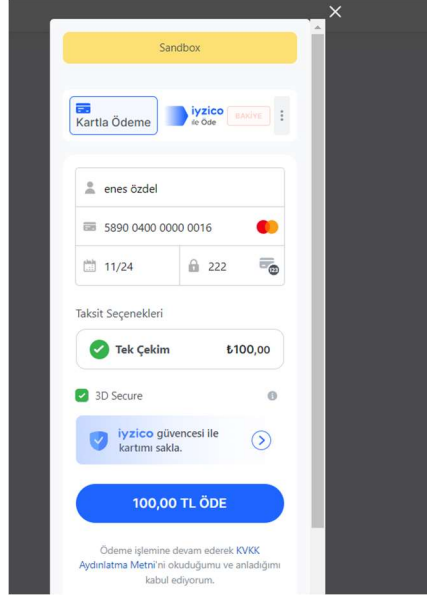
Seçilen Koltuk Sayısı: 2

Toplam Ödeme: 200 TL

Confirm Reservation

Bu görselde koyu gri ile gösterilen numaralı koltuklar daha önce satın alınmış koltuklardır. Hangi koltukların satın alınabileceği bilgisi db kontrolü sonrasında belirlenir.

ConfirmReservation butonuna tıklandığında eğer herhangi bir giriş yapan kullanıcı yoksa Login sayfasına yönlendirirken giriş yapan kullanıcı varsa Iyzico Satın alma ekranına(Modalına) yönlendirir

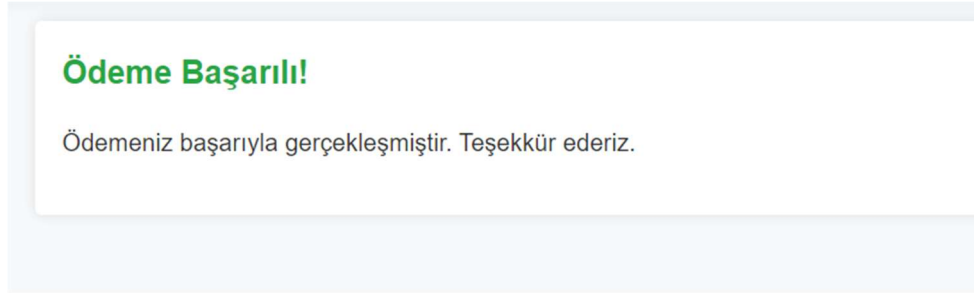


Gerekli bilgiler girildikten sonra sms doğrulama kodu ile kullanıcı otobüs bileti alma işlemini sonlandırır.

Please enter the sms code (283126) that we send to your phone.

Sms Code

Ödeme işlemi tamamlandıktan sonra ödeme işleminin başarılı bir şekilde gerçekleştiğini gösteren Success metodu UI sayfası aşağıdaki gibidir:



Admin Area ve Controller Yapıları

Admin paneli oluşturulurken ASP.NET Core'un sağladığı entityframework kullanılarak controller ve view ekranları oluşturulmuştur. Admin Panelinde db'de bulunan tabloları güncelleyerek veri ekleme, çıkarma ve düzenleme işlemleri yapılabilir. Admin Paneline ulaşım için navbarda bulunan Go To Admin Panel butonuna basılmalıdır.

Ticketx

User

Buses

Tickets

Trips

Hello senesozdel@gmail.com!

Logout

Index

Create New

BusNumber	Capacity	
BUS001	30	<a>Edit <a>Details <a>Delete
BUS002	40	<a>Edit <a>Details <a>Delete
BUS003	35	<a>Edit <a>Details <a>Delete
BUS004	50	<a>Edit <a>Details <a>Delete
BUS005	45	<a>Edit <a>Details <a>Delete
BUS006	55	<a>Edit <a>Details <a>Delete
BUS007	25	<a>Edit <a>Details <a>Delete
BUS008	60	<a>Edit <a>Details <a>Delete
BUS009	30	<a>Edit <a>Details <a>Delete
BUS010	40	<a>Edit <a>Details <a>Delete

Görselde otobüs bilgilerinin listelendiği admin paneli görüntüsü bulunmaktadır. Sağ tarafta bulunan linkler sayesinde istenilen düzenlemeler yapılabilmektedir.

Karşılaşılan Zorluklar

Seçilen seferdeki koltuk numaraları ve toplam ödenecek miktarın belirlenmesinde ve bu verilerin controllera yönlendirilip işlenmesi esnasında zorluklar yaşanmıştır. Aynı zamanda db'ye otobüs bileti verisi girerken fazla bağlantılı veriler olduğu için eklemede zorluklarla karşılaşmıştır. Iyzico entegrasyonu ve ödeme sistemi ile işlem doğruysa db güncelleme projede zaman alan noktalardan birini oluşturmuştur.

Tartışma

Bu proje sonucunda Area yapısı kullanan, Identity kütüphanesi sayesinde Login ve Register işlemleri yapan ve Iyzico ödeme sistemi entegrasyonu ile kullanıcı etkileşimi yüksek bir otobüs bilet seçim uygulaması oluşturmak hedeflenmiştir. Bu noktada istenilen çıktılara ulaşılmıştır. Kullanıcı bilgileri sessionda tutularak oturum sürecinde işlemler sessionda tutulan bilgiler üzerinden tamamlanmıştır. Bu da bir başka veri tutma ve veri erişimi yöntemini öğretmiştir.