**Piri Reis Üniversitesi**

**İnternet Porgramcılığı Dersi**

**HAZIRLAYANLAR:**

Beyza Muratoğlu 2021 010 80 11

Salih Can Erkek 2021 010 80 15

Mert Can Doğan 2022 010 80 58

Kerem Ketenci 2022 010 80 18

**DANIŞMAN:**

Erkan Aydın

**KONU:**

E-Ticaret Web Sitesi

**Proje Amacı:**

**E-Ticaret Platformu Oluşturma:**

Amacı, kullanıcıların çeşitli ürünleri inceleyebileceği, sepete ekleyebileceği ve satın alabileceği bir e-ticaret platformu oluşturmaktır.

**Kullanıcı Deneyimini Artırma:**

Projenin temel amacı, kullanıcıların platformu kolayca kullanabilmesi ve keyifli bir alışveriş deneyimi yaşayabilmesidir. Bu nedenle, kullanıcı dostu bir arayüz ve etkili bir kullanıcı deneyimi tasarlanmıştır.

**Admin Paneli Yönetimi:**

Proje, yöneticilerin platformu kontrol edebilmeleri için bir admin paneli içermektedir. Admin paneli, ürünleri yönetme, kategorileri düzenleme, kullanıcıları takip etme gibi yetenekleri içerir.

**Gelişmiş Veri Tabanı Yönetimi:**

Veri tabanı, en az 10 kayıt içerir ve ilişkilendirilmiş bir yapıya sahiptir. Bu, ürünlerin kategorilere atanabilmesi, kullanıcıların geçmiş alışverişleri ile ilgili bilgilerin saklanabilmesi gibi özellikleri destekler.

**Güvenlik ve Yetkilendirme:**

Kullanıcıların giriş yapma ve kayıt olma işlemleri için güvenli bir kimlik doğrulama mekanizması sağlanmıştır. Admin paneli girişi, sadece yetkilendirilmiş kişiler tarafından yapılabilir.

**Ürün ve Kategori Yönetimi:**

Admin paneli üzerinden, ürünlerin kategorilere atanması, stok durumlarının takibi, yeni ürün eklemesi gibi işlemler gerçekleştirilebilir.

**Gelişmiş Sepet Yönetimi:**

Kullanıcılar, sepete ürün ekleyebilir, çıkartabilir ve alışveriş işlemlerini tamamlayabilir. Sepetteki ürün sayısı ve toplam fiyat gibi bilgiler sürekli güncellenir.

**ASP.NET Core MVC :**

Projede ASP.NET Core MVC kullanılarak bir web uygulaması geliştirilmiştir. MVC mimarisi, uygulamanın modüler ve sürdürülebilir olmasına katkı sağlar.

**Projenin Kim İçin ve Neden Yapıldığı:**

**Kullanıcılar İçin:**

Kullanıcılar için bu proje, çeşitli ürünleri inceleyebilecekleri, seçtikleri ürünleri sepete ekleyip satın alabilecekleri bir e-ticaret platformu sunar. Hızlı ve güvenli alışveriş deneyimi sağlamak amacıyla kullanıcı dostu bir arayüz tasarlanmıştır.

**Yöneticiler İçin:**

Admin paneli, yöneticilere platformu etkili bir şekilde yönetme imkanı sunar. Ürün ve kategori yönetimi, kullanıcı takibi, stok kontrolü gibi işlemleri kolayca gerçekleştirebilmeleri için tasarlanmıştır.

**Geliştiriciler İçin:**

Geliştiriciler için bu proje, ASP.NET MVC framework kullanarak web uygulaması geliştirme becerilerini artırmak ve modern bir e-ticaret platformu oluşturarak proje yönetimi ve kodlama konusundaki deneyimlerini geliştirmek amacıyla yapılmıştır.

Bu proje, kullanıcıların, yöneticilerin ve geliştiricilerin ihtiyaçlarına cevap verecek şekilde tasarlanmış bir e-ticaret platformu oluşturmak amacıyla gerçekleştirilmiştir.

**E-Ticaret Uygulaması Kod Dökümantasyonu**

**AdminService.cs:**

Bu sınıf, kullanıcı yönetimi ile ilgili çeşitli işlemleri gerçekleştirmek için tasarlanmıştır.

**Constructor:**

AdminService, AppDbContext, AuthDbContext, UserManager, ve IUserStore örnekleri alır. Bunlar, kullanıcıları yönetmek ve kimlik doğrulama işlemlerini gerçekleştirmek için gereken bağlam ve servisleri temsil eder.

**Metodlar:**

1. **GetUsers():**

Tüm kullanıcıları sorgulayıp IQueryable olarak döndürür.

1. **DetailsUser(string id):**

Verilen id ile kullanıcıyı bulur ve detaylarını getirir. Eğer kullanıcı bulunamazsa bir hata fırlatır.

1. **AddUser(UserPostRequest userPostRequest, string password):**

Yeni bir kullanıcı ekler, şifresini hashler ve rol ataması yapar.

1. **UpdateUser(UserPutRequest userPutRequest, string id):**

Belirtilen id'ye sahip kullanıcının bilgilerini günceller.

1. **DeleteUser(string id):**

Belirtilen id'ye sahip kullanıcıyı siler.

1. **TakeUsers(IQueryable<AppUser> query, int? count):**

Belirtilen sayıda kullanıcıyı döndürmek için sorguyu filtreler.

1. **FilterUsers(IQueryable<AppUser> query, string? firstName, string? lastName):**

İsimlere göre kullanıcıları filtrelemek için sorguyu günceller.

1. **UserStatus(IQueryable<AppUser> query, bool? userStatus):**

Kullanıcıların durumlarına göre filtre uygular.

**AdminController.cs**

Bu sınıf, HTTP isteklerini karşılamak ve AdminService üzerinden kullanıcıları yönetmek için tasarlanmıştır.

**Constructor:**

AdminController, bir IAdminService örneği alır.

**Metodlar:**

1. **Get(string? id, string? firstName, string? lastName, int? count, bool? userStatus = null):**

Belirtilen parametrelere göre kullanıcıları getirir.

1. **Post(UserPostRequest userPostRequest, string password):**

Yeni bir kullanıcı oluşturur.

1. **Put(UserPutRequest userPutRequest, string id):**

Mevcut bir kullanıcının bilgilerini günceller.

1. **Delete(string id):**

Belirtilen id'ye sahip kullanıcıyı siler.

**Bu sınıflar ve metodlar, kullanıcı yönetimi için API tabanlı bir sistem oluşturur ve çeşitli HTTP isteklerini karşılayarak kullanıcı işlemlerini gerçekleştirir.EntityBaseRepository<T>.cs:**

Bu genel repository sınıfı, bir IEntityBaseRepository arayüzü üzerinden belirli işlemleri gerçekleştiren genel CRUD (Create, Read, Update, Delete) işlemlerini sağlar.

**Constructor:**

EntityBaseRepository, bir AppDbContext örneği alır. Bu, veritabanı bağlamını temsil eder ve işlemlerin gerçekleştirileceği yerdir.

**Metodlar:**

1. **GetAllAsync():**

Tüm T tipindeki varlıkları asenkron olarak getirir ve IEnumerable<T> olarak döndürür.

1. **GetAllAsync(params Expression<Func<T, object>>[] includeProperties):**

Belirtilen ilişkili özellikleri içeren tüm T tipindeki varlıkları asenkron olarak getirir ve IEnumerable<T> olarak döndürür.

1. **GetByIdAsync(int id):**

Belirtilen id değerine sahip T tipindeki varlığı asenkron olarak getirir.

1. **GetByIdAsync(int id, params Expression<Func<T, object>>[] includeProperties):**

Belirtilen id değerine sahip T tipindeki varlığı belirtilen ilişkili özellikleri içeren şekilde asenkron olarak getirir.

1. **AddAsync(T entity):**

Yeni bir T tipindeki varlık ekler ve değişiklikleri veritabanına kaydeder.

1. **UpdateAsync(int id, T entity):**

Belirtilen id değerine sahip T tipindeki varlığı günceller ve değişiklikleri veritabanına kaydeder.

1. **DeleteAsync(int id):**

Belirtilen id değerine sahip T tipindeki varlığı siler ve değişiklikleri veritabanına kaydeder.

**Bu sınıf, veritabanı işlemlerini genelleştirerek tekrar kullanılabilir bir yapı sunar ve belirli bir varlık türü üzerinde CRUD işlemlerini gerçekleştirmek için kullanılabilir.**