E Ticaret Sitesi Dokümantasyon

Proje Açıklaması

E-Ticaret sitesi olarak yazılan bu proje de kullanıcı sitede istediği kategorideki ürünler arasında dolaşabilir ve ürünlerde filtreleme yapabilir. Kullanıcı istediği ürünü sepetine atabilir ve istediğinde sepetinde hangi ürünlerin bulunduğunu ve toplam olarak ne kadar ödemesi gerektiğini görebilir. Satın alma işlemini yapmak istediğinde kart bilgilerini ve adresini girerek satın alma işlemini tamamlar.

Projenin Teknik Bilgileri

* Ntier ve MVC Mimari
* Code First EAV modeli Veri Tabanı
* Mail Gönderme, Fotoğraf Yükleme ve Şifreleme Sınıfları
* Web Api Kullanımı:

1. Kargo Api
2. Loglama Api
3. Fake Data Api

* Önyüz tasarımında Kullanılanlar:

1. Javascript
2. Ajax

* Admin Özel Sayfa (Urun ekleme vb. için)
* Member için sayfalar

Master’ın Görevleri

* Proje Başlangıcında Projenin Mimarisini oluşturmak, gerekli sınıfları projeye entegre etmek ve referansları vermek
* Projenin yapılış zamanı sürecinde proje çalışanlarının yaptıkları işlemlerde onları gözlemlemek ve çıkan sorunlarda eğer takım arkadaşı o sorunu çözemiyorsa sorunun çözümünü devir alarak sorunu çözmek
* Takım arkadaşlarının psikolojik durumlarını göz önünde bulundurmak ve takım arkadaşlarına destekçi olmak
* Proje çalışanı kendi üzerine düşen görevi tamamladıktan sonra commit ve push işlemlerinden önce güncel projede çalıştığının teyidini almak ve projenin güncel versiyonunda çalıştığının onayını altıktan sonra commit etmesi sonucunda Master olarak projeyi o kişiden çekerek Master’i güncel tutmak.

Önemli Bilgi: Proje deki göreve başlarken master’ın söylediği zamanda başlamaya dikkat edilmelidir. Master’in sizin proje üzerinde çalıştığınızdan haberi olmalıdır. Önemli işlemleri (commit vb.) işlemler yapılırken master’a bilgi verilmeli ve onay beklenmelidir. Projeye başlarken projenin güncel halini master’dan çektiğinize emin olmalısınız.

Görev Dağılımı

* Furkan BÖREK (Master)

1. Ntier ve MVC mimarisinin kurulması
2. Gerekli Framework’lerin indirilmesi
3. Katmanlar arasında referansların verilmesi
4. Web. Config’e bağlantı yazısının verilmesi
5. Mail Gönderme, Fotoğraf Yükleme, Şifreleme Sınıflarının Yapılması
6. Web sitesi görsellerinin MVCUI katmanında mevcut hale getirilmesi
7. Katmanlardaki gerekli klasörlerin açılması (Design Patterns vb.)

* Esra SANCAK

1. Veri tabanın Code First yazılması
2. Veri tabanına EAV modelinin entegrasyonu
3. Ürün ile Kategori arasında ilişkinin çoka çok olarak yapılması
4. Veritabanı sınıflarının MAP katmanında düzenlenmesi(AppuserMap vb.)
5. DAL katmanında MyContext sınıfının oluşturulup Veritabanı işlemlerini sonlandırmak. Connection name =”MyConnection”
6. Strategy patterni
7. BLL katmanında Repository’lerin oluşturulması

* Tarık

1. İlk olarak Kargo Firmasının Web Api’si yazılacak
2. Backend kodları yazılacak(Enes ile ortak çalışarak)

* ENES

1. İlk olarak Loglama Api’si yazılacak
2. Ek veritabanı
3. Backend kodları yazılacak(Tarık ile ortak çalışarak)

* CEM

1. Frontend Kodları yazılacak ve Gökhan ile birlikte ortaklaşa çalışacaklar.

* GÖKHAN

1. Frontend Kodları yazılacak ve Cem ile birlikte ortaklaşa çalışacaklar.

* OZAN

1. Api Entegrasyonu E Ticaret

* Alp

1. Proje master’a gelmeden önce detaylı test işleminden geçecek.

Yapılanlar

* Master

1. Bogus Dal katmanına yüklendi
2. EntityFramework 6.2.0 BLL, COMMON, DAL, MAP, MVCUI katmanlarına yüklendi
3. Microsoft.AspNet.WebApi.Client 5.2.7 MVCUI katmanına yüklendi
4. BLL katmanı COMMON, DAL, MODEL katmanında referans aldı
5. DAL katmanı MAP ve MODEL katmanında referans aldı
6. MAP katmanı MODEL katmanından referans aldı
7. MVCUI katmanı BLL, COMMON ve MODEL katmanında referans aldı.
8. MVCUI katmanının Web. Config’e Connection String yazıldı name "MyConnection" olarak verildi.
9. COMMON katmanına Mail gönderme, Fotoğraf yükleme, Şifreleme Sınıfları eklendi
10. MVC FlatAdmin Eklendi

* Esra
* **Proje iskelet halinde teslim alındı. Projenin içerisinde hazır bulunanlar:**

1. Proje mimarisi hazır geldi.
2. Gerekli FrameWorkler hazır geldi.
3. Katmanlar arasındaki referanslar hazır geldi.
4. Web.config bağlantısındaki düzenlemeler hazır geldi.
5. Mail gönderme, fotoğraf yükleme, şifreleme sınıfları hazır geldi.
6. Proje, Web sitesi görsellerinin MVCUI katmanına entegre edilmiş halde geldi.
7. Katmanlardaki gerekli klasörler (design pattern, strategy pattern vs. gibi) hazır geldi.

* **Projede yaptıkları:**

1. Veri tabanı code-first ile yazıldı. (Veritabanı EAV modelinde hazırlandı):
   * + - 1. Projedeki enumslar oluşturuldu.
         2. Entities’ler hazırlandı. Hazırlanan sınıflar:

Abstract **BaseEntity** sınıfı

**AppUser**

**AppUserDetail** (AppUser ile Bire bir ilişkili)

**Category**

**Product** (Category sınıfı ile Product sınıfı çoka çok ilişkili)

**Feature** (özellik sınıfı=> Product sınıfı ile çoka çok ilişkili (junction table’ı => **ProductFeature** tablosu. **Value** değerleri bu tablo içerisinde)

**Order** sınıfı (Product sınıfı ile çoka çok ilişkili. Junction table’i **OrderDetail**)

* + - * 1. Gerekli özellikler sınıflara verildi.
        2. Relational propertyler hazırlandı.

1. Veritabanı sınıfları MAP Katmanında düzenlendi. Birebir ve çok a çok tablolar için gerekli ayarlamalar hazırlandı. ColoumnName’ler Türkçeleştirildi. Gerekli coloumnTyplerin ayarlanmaları yapıldı.(datetime => datetime2 , decimal=> Money gibi)
2. DAL katmanında MyContext sınıfı oluşturuldu. Hazırlanan Map katmanları çağırıldı. Veritabanında tablolar oluşturuldu. Connection name ayarlamaları yapıldı.
3. BLL katmanında design patternler hazırlandı. İlk olarak; Singletton pattern projeye entegre edildi.
4. Unit of Works pattern olarak da bilinen, Repository pattern oluşturuldu. IntRep klasörü içerisine; IRepository interface ‘i hazırlandı/ eklendi.
5. BaseRep klasörünün içerisinde, abstract BaseRepository sınıfı oluştuldu. IRepository interface’inden miras verildi. İmplement edildi.
6. BaseReporsitory de gerekli Repository ayarlamaları yapıldı.:
   * + - 1. **Listelemeler:**

**GetAll =>** Hepsini Listeler

**GetActives=>** Aktifleri Listeler

**GetUpdates=>** Güncellemeleri Listeler

**GetPassives=>** Pasife çekilmişleri listeler

* + - * 1. **Modifikasyonlar:**

**Add=>** Ekleme metodu

**Update=>** Güncelleme metodu

**Delete=>** Pasife çekme metodu

**SpecialDelete=>**  Tamamen silme metodu

* + - * 1. **Queries:**

**T GetByID(int id)=>** id’Den yakala

(Diğer kısımların isimlerinde herhangi bir değişiklik yapılmadı.)

1. Diğer sınıflar için Respository sınıfları düzenlendi. Hazırlanan BaseRepository, bu sınıflara miras verildi.
2. Strategy patterni oluşturuldu. Projeyle hazır hale gelen bogus’tan fake datalar çekildi. (AppUser, AppUserDetail, Product ve Category için fake datalar çekildi)
3. Proje test amacı ile ayağa kaldırıldı. (package Manage Console’dan)

* **Proje GitHub’ a yüklenmeden önce:**
  + - Migration klasörü kaldırıldı.
    - Son kontroller yapıldı.
* **Proje üzerinde çalışılmadan önce yapılması gerekenler:**

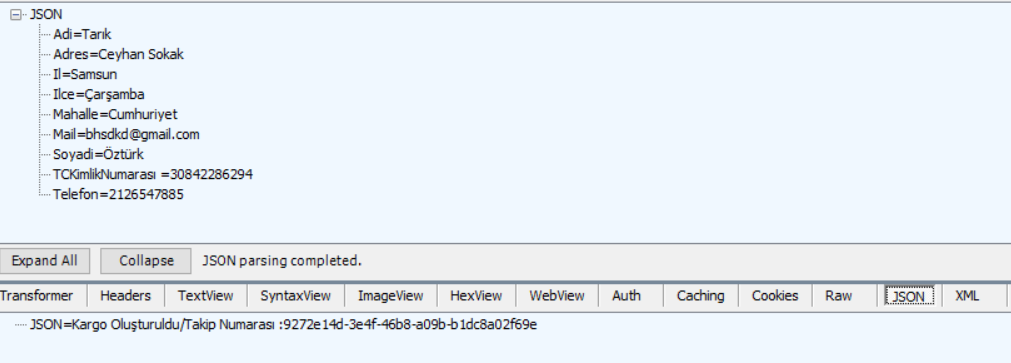
1. Projeyi Build etmeyi unutmayın. Genelde git hub’dan bağlanırken entityframeworklerde sorun çıkıyor. Build ederek bu sorun çözülüyor.
2. Bogus bazen patlıyor. Böyle bir durumda birkaç daha deneyin.
3. MVCUI katmanında oluşturulan HomeContorller, Test amaçlı oluşturulmuştur. Keza view-Home klasöründeki “index” view’u da test amaçlı oluşturulmuştur. Çalışmalarınızı testlerini yaparken belki kullanmak istenir belki diye özellikle silmedim. Sizler istediğiniz modifikasyonları orada uygulayabilirsiniz.

* Tarık

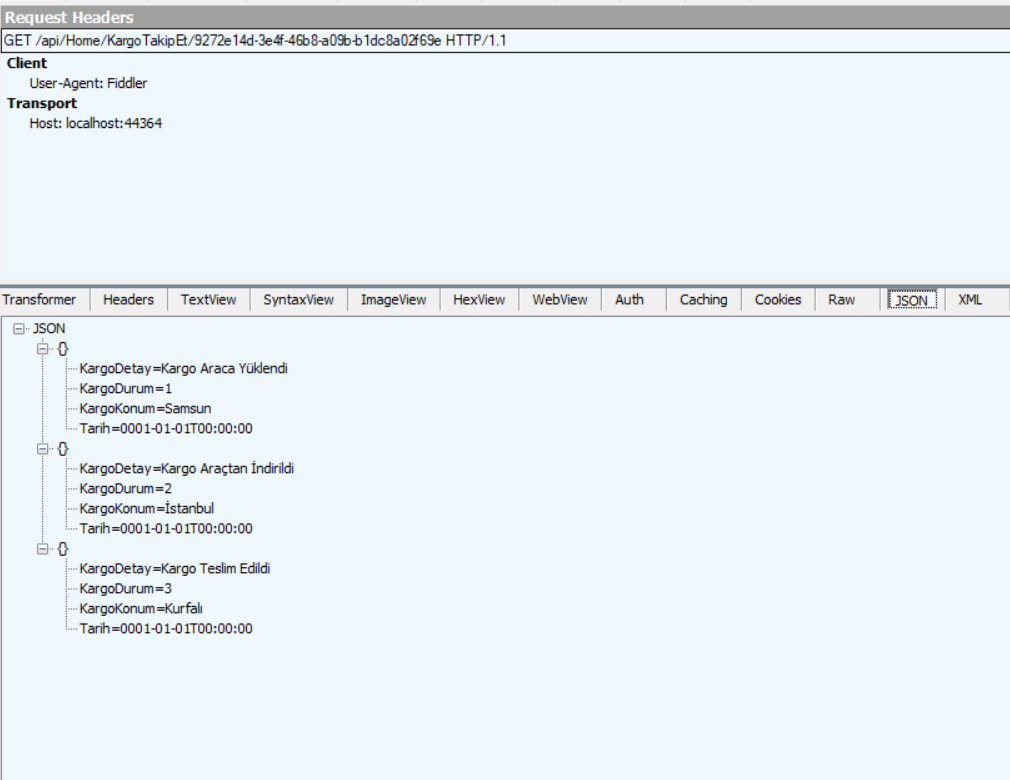
# **Kargo Web Api’si Oluşturma Projesi**

1. NTier mimarisi oluşturuldu(MODEL, MAP, DAL, BLL, WebApi)
2. MODEL katmanında veritabanı sınıfları(Alici, AdresBilgisi, Hareket, Kargo) ve gerekli enum yapıları(DataStatus, Durum) oluşturuldu.
3. MODEL katmanında veritabanı sınıfları arasındaki ilişkiler oluşturuldu.
4. MAP katmanında tablo isimleri ayarlandı, maksimum karakter sayıları belirlendi.
5. DAL katmanında veritabanı sınıfı oluşturuldu ve configürasyon ayarları sağlandı, veritabanı tabloları oluşturuldu.
6. BLL katmanında Singleton Patterni ve Unit Of Work Patternleri uygulandı. Veritabanı işlemlerinde kullanılacak metotlara açıklamaları eklendi.
7. WebApi katmanında connectionString yazıldı.
8. WebApi katmanında VM sınıfları oluşturuldu. Post olarak gelecek olan KargoVM sınıfında validation(DataAnnotations) ayarlamaları yapıldı.
9. WebApi katmanında HomeController oluşturuldu. KargoOlustur ve KargoTakip isimli iki metot oluşturuldu.
10. WebApi katmanında route template "api/{controller}/{action}/{kargoTakip}" olarak ayarlandı. {kargoTakip} ifadesi Postman veya Fiddler araçlarında test edilebilirliği sağlamak için yazıldı. Bu veri form ile geleceği için daha sonra kaldırılacak.
11. Proje Fiddler aracı ile test edildi. Çalıştığı görüldü.

KargoOlustur Metot Çıktısı:



KargoTakipEt Metot Çıktısı



Zaman Yönetimi

1. Esra => 4 gün (veri tabanı) ------ Yapıldı (14.11.2019)
2. Tarık => 4 gün (Api entegrasyonu Kargo)------- Yapıldı (15.11.2019)
3. Enes => 4 gün (Loglama ve ek veri tabanı) ----- (Yapılıyor...)
4. Ozan => (Api Entegrasyonu E Ticaret) -------- daha zamanı var
5. Kasım 18 Tarık (Api),Enes(Loglama) ve Esra veritabanı bitiyor... (sadece Enes kaldı !!!)
6. Aralık 5 Tarık ve Enes Backend bitişi...
7. Aralık 10 Cem ve Gökhan Frontend bitişi...
8. Aralık 13 Ozan Api Entegrasyon Bitişi...
9. 2020 Ocak 3 Emre Alp Test bitişi
10. 2020 Mart 3 Proje Live'a acılısı...