|  |  |  |
| --- | --- | --- |
| **egelog** | **EGE ÜNİVERSİTESİ** |  |
|  | **LİSANS TEZİ**  **MAKALE YAYINLARINI KATEGORİ BAZLI SUNAN SOSYAL WEB UYGULAMASI**  **Tolga SALTAN**  **Tez Danışmanı : Dr. Öğretim Üyesi Şebnem BORA**  Bilgisayar Mühendisliği Anabilim Dalı  **Sunuş Tarihi : 28.06.2019**  **Bornova-İZMİR**  **2019** |  |

EÜ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

EGE ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

**(LİSANS TEZİ)**

**MAKALE YAYINLARINI KATEGORİ BAZLI**

**SUNAN SOSYAL WEB UYGULAMASI**

**Tolga SALTAN**

**Tez Danışmanı : Dr. Öğretim Üyesi Şebnem BORA**

Bilgisayar Mühendisliği Anabilim Dalı

**Sunuş Tarihi : 28.06.2019**

**Bornova-İZMİR**

**2019**

**ÖZET**

**MAKALE YAYINLARINI KATEGORİ BAZLI**

**SUNAN SOSYAL WEB UYGULAMASI**

SALTAN, Tolga

Lisans Tezi, Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı: Dr. Öğretim Üyesi Şebnem BORA

Haziran 2019, 48 sayfa

Günümüzde internet teknolojisinin gelişmesiyle beraber her yazılı ve sözlü kaynaklar internet ortamına aktarıldı. Bununla beraber bilginin toplanması, dağıtılması, ulaşılabilmesi kolaylaştı. Herhangi bir bilgiye ulaşmak istediğimizde internetten arama yaparak birçok makaleye erişim sağlayabiliyoruz. Bu makaleler genellikle “popüler bilim yayıncılığı” diye adlandırılır. Popüler bilim, akademik düzey bilimsel çalışmaların toplum tarafından anlaşılabilir hale getirilmesidir.

Bizim bu projedeki amaçlarımızdan biri yapılan bilimsel çalışmaları toplumun yararına elektronik ortamda yayılmasını sağlamak. E-Science bu amaçla toplumun dünyadaki bilimsel gelişmeleri yakından takip etmesini sağlar.

Bir diğer amacımız ise herhangi bir üniversite veya başka bir kurumun kendi makalelerini veya başka içerikleri yönetici paneli aracılığı ile yayınlayabildikleri bir web uygulaması oluşturmak.

Bu amaçlar doğrultusunda elektronik bir ortamda makale paylaşımı yapılabilecek, kullanıcılar kendi kurumlarının ilgili makale, duyuru veya not dokümanlarına web aracılığı ile hızlı bir şekilde ulaşabilecekler. Projemizde işlemlerin güvenilir ve hızlı bir şekilde gerçekleşmesini sağlamak amacıyla çok katmanlı mimari kullanılmıştır.

**Anahtar sözcükler:** Web Teknolojileri, Katmanlı Mimari, Makale Yönetimi, Web İçerik Yönetimi, Popüler Bilim

**ABSTRACT**

**SOCIAL WEB APPLICATION OFFERING**

**PUBLICATIONS BY CATEGORY**

SALTAN, Tolga

BSc in Computer Eng.

Supervisor: Assoc. Prof. Dr. Şebnem BORA

June 2019, 48 pages

Nowadays, with the development of internet technology, all written and verbal resources were transferred to the internet. Howbeit, it became easier to collect, distribute and access information. When we want to access any information we can access many articles by searching from the internet. The meaning of these articles is called “popular science publishing”. Popular science is the making of academic level scientific studies understandable by the society.

One of our aims in this project is to ensure that the scientific studies are spread to the benefit of the society in electronic environment. To this end, E-Science enables the society to closely follow scientific developments in the world.

Another objective is to create a web application where any university or other institution can publish their articles or other content through the admin panel.

For these purposes, articles can be shared in an electronic environment and users will be able to quickly access the relevant articles, announcements or note documents of their institutions via the web. N-layered architecture was used in our project to ensure reliable and fast operations.

**Keywords:** Web Technologies, Layered Architecture, Article Management, Web Content Management, Popular Science

**TEŞEKKÜR**

Lisans tezimin hazırlanmasında, benimle ilgilenen, bana her aşamada yol gösteren; desteğini, kaynak ve yardımlarını hiçbir zaman esirgemeyen danışman hocam Sayın Dr. Öğretim Üyesi Şebnem BORA’ya; maddi, manevi destekleriyle değerli aileme çok teşekkür ederim.

İÇİNDEKİLER

Sayfa

ÖZET …iv

ABSTRACT ..vi

TEŞEKKÜR viii

ŞEKİLLER DİZİNİ xiv

tablolar DİZİNİ xvii

SİMGELER VE KISALTMALAR dİZİNİ xviii

1.GİRİŞ 1

1.1 Popüler Bilim Makaleleri 1

1.2 Tezin Amacı ve Önemi 2

2.BİLİMSEL MAKALELERİN NİTELİKLERİ 4

2.1 Akademik Makale Yayınlayan Web Uygulamaları 5

2.1.1 Science direct 5

2.1.2 Springer link 6

2.1.3 Web of science 6

2.1.4 Scopus 6

2.1.5 Research gate 6

2.1.6 Google scholar 6

**İÇİNDEKİLER (devam)**

Sayfa

3.KULLANILACAK YÖNTEM VE TEKNOLOJİLER 7

3.1 Katmanlı Mimari 7

3.1.1 Desen açıklaması 8

3.1.2 Anahtar kavramlar 10

3.2 C# 12

3.3 Asp.Net MVC 14

3.3.1 Model 15

3.3.2 View 15

3.3.3 Controller 16

3.4 Microsoft SQL Server 17

3.5 Visual Studio 15 18

4**.** E-scıence analizi ve tasarımı 20

4.1 Analiz 20

4.1.1 Fonksiyonel gereksinimler 20

4.1.2 Fonksiyonel olmayan gereksinimler 21

4.1.3 Domain gereksinimleri 22

4.1.4 Kullanım senaryoları 22

**İÇİNDEKİLER (devam)**

Sayfa

4.2 Tasarım 27

4.2.1 Veritabanı tasarımı 28

4.2.2 Kullanım senaryoları ssd 28

5.E-scıence kullanıcı kılavuzu 31

5.1 Sitenin Ana Sayfası 31

5.2 Kategori Bazlı Makale Sayfası 31

5.3 Makale Görüntüleme Sayfası 32

5.4 Makale Yazarı Sisteme Giriş Sayfası 33

5.5 Profil İşlemleri 34

5.6 Makale İşlemleri 35

5.7 Makale Düzenleme Sayfası 36

5.8 Makale Filtreleme 36

5.9 Kategori İşlemleri 37

5.10 Çıkış İşlemleri 37

6. Gerçekleştirim 38

6.1 Core Katmanı Entities 38

6.2 Data Katmanı 39

**İÇİNDEKİLER (devam)**

Sayfa

6.2.1 Context yapısı 39

6.2.2 Repository pattern 40

6.2.3 Unit of work pattern 41

6.3 DTO Katmanı 43

6.3.1 Inversion of control katmanı (IoC) 44

6.4 Service Katmanı 45

6.5 Presentation Katmanı 46

7. SONUÇ VE ÖNERİLER 47

KAYNAKLAR DİZİNİ 48

**ŞEKİLLER DİZİNİ**

Şekil Sayfa

3.1. Katmanlı mimari deseni 8

3.2 Katmanlar arası veri akışı 9

3.3 Servis katmanı etkileşimi 10

3.4 Kapalı katmanlar ve erişim isteği 11

3.5 C# çalışma mantığı 14

3.6 MVC yapısı 15

3.7 Controller yapısı 16

3.8 Web uygulamasının aygıtlarla etkileşimi 18

4.1 E-Science veritabanı ilişkisi 28

4.2 Makale ekleme SSD 28

4.3 Makale görüntüleme SSD 29

4.4 Şifre değiştirme SSD 29

4.5 Profil bilgilerini değiştirme SSD 30

5.1 Ana sayfa 31

5.2 Kategori sayfası 32

5.3 Makale görüntüleme sayfası 32

5.4 Giriş sayfası 33

**ŞEKİLLER DİZİNİ (devam)**

5.5 Profil sayfası 33

5.6 Bilgileri değiştirme sayfası 34

5.7 Şifre değiştirme sayfası 34

5.8 Profil resmi değiştirme 35

5.9 Makale işlemleri 35

5.10 Makale düzenleme sayfası 36

5.11 Makale filtreleme ekranı 36

5.12 Kategori işlemleri 37

5.13 Çıkış işlemi 37

6.1a Base class 38

6.1b User class 38

6.1c Post class 39

6.1d Category class 39

6.2 Context yapısı kod bloğu 40

6.3 Repository pattern kod bloğu 41

6.4 Unit of work yapısı 42

6.5 Unit of work pattern kod bloğu 43

6.6 ELoginDTO sınıfı 44

**ŞEKİLLER DİZİNİ (devam)**

6.7 IoC pattern kod bloğu 44

6.8 Servis katmanı kod bloğu 45

**TABLOLAR DİZİNİ**

Tablo Sayfa

3.1 Kullanılacak yöntem ve teknolojiler 7

**SİMGELER VE KISALTMALAR DİZİNİ**

Kısaltmalar Açıklama

WEB Dünya çapında ağ

TÜBİTAK Türkiye bilimsel ve teknolojik araştırma kurumu

ABD Amerika birleşik devletleri

TÜBA Türkiye bilimler akademisi

ISSN Uluslararası standart dizi numarası

SQL Yapılandırılmış sorgu dili

CRUD Oluşturma okuma güncelleme silme

HTML Hiper metin işaretleme dili

ECMA Avrupa bilgisayar imalatçıları birliği

ISO Uluslararası standartlar teşkilatı

PDA Kişisel veri yardımcısı

CLR Ortak dil çalışması

MSIL Microsoft orta seviye dil

XML Genişletilebilir işaretleme dili

JSON Javascript object notation

CSS3 Basamaklı biçim sayfaları

PHP Kişisel anasayfa

**SİMGELER VE KISALTMALAR DİZİNİ (devam)**

IDE Tümleşik geliştirme ortamı

API Uygulama programlama arayüzü

GUI Grafiksel kullanıcı arayüzü

XSLT Genişletilebilir biçimleme dili

SSD Sistem akış diagramı

1. **GİRİŞ**

Günümüz topluluklarında sosyal ağ ve yazılımların, insanların iletişim kurma biçimini ve bilgi paylaşımını da değiştirdi (Murray C., 2008). Teknolojinin gelişmesiyle beraber cep telefonları, tabletler, bilgisayarlar aracılığı ile her an istediğimiz akademik veya başka tür makalelere erişebiliyoruz.

Teknoloji çok hızlı ilerlerken beraberinde makale yazıları da artmaktadır. Artan makale yazıları ile internette aranılan makaleye ulaşmak biraz zaman alabiliyor. Bunun için makale yazılarını kategorilere ayırarak ücretli veya ücretsiz yayınlayarak kullanıcının hangi tür makale aradığını bulmasına yardımcı olan siteler mevcut. Web içerik yönetimi sistemleri web sayfaları ve portallar ile kurum içinde genel bilgi akışını ve kurumdaki birimlerin ihtiyaç duyabilecekleri bilgileri web üzerinden sağlamaya yönelik sistemlerdir (Manas, O., 2001).

Teknoloji hayatın her alanına yayılmış ve günlük okuma alışkanlıkları yavaş yavaş kendini gazete, dergi ortamından elektronik ortama bırakmıştır. Yayınlanan dergiler artık sadece kağıda basılmakla kalmayıp geniş kitlelere ulaşabilmek, kolaylıkla uzaktaki kişilere ulaştırabilmek amacıyla e-dergi olarak internet ortamında sunulmaktadır.

* 1. **Popüler Bilim Makaleleri**

Bilim adamları, yapmış oldukları bilimsel araştırmalarının sonuçlarını kongre, sempozyum ve bilimsel dergilerde sözlü veya yazılı olarak bilim dünyasına sunmak zorundadırlar. Aksi halde harcanmış olan bütün emeklerin ve elde edilen sonuçların boşa gitmiş veya hiç yapılmamış gibi olmaktan, kişisel tatminin dışında bir farkı kalmaz. Evrensel bilgi üretimi, kişilerin birer birer yapmış oldukları araştırmaların sonuçlarının bilim kamuoyu tarafından paylaşılması ile katlanarak çoğalır, büyür, gelişir ve belirli bir birikimden sonra ekonomik, teknolojik ve sosyal yararlar sağlayan ikincil ürünler vermeye başlar.

Toplumlar, bilimle ve bilimin ürünü olan teknolojiyle günden güne artan bir ivmeyle hayatlarını paylaşmaktalar. Bu paylaşım ister istemez insanları az ya da çok bilimle ilgili hale getirmektedir. Fakat bilim, insanlar tarafından her zaman çokta anlaşılabilir bir kavram olmamıştır. Bilimi anlamanın temel sebeplerini Yıldırım (Yıldırım, C., 1991); “1- Bilimin uygulama sonuçları yaşamımızı giderek artan ölçülerde her cephesinde etkilemektedir ve 2- Bilimsel düşünceyi tanımak çağımız aydını için entelektüel bir zorunluluktur.” olarak sıralamaktadır. Bilimin kendi içinde, farklı dallar arasında yaşanan kopukluk nedeniyle, bir alanda uzmanlaşmış bir bilim insanı, başka bir alandaki gelişmeleri izlemekte zorlanmaktadır. İşte bu nedenle, bilimle toplum arasında bir köprü kurmak gerekiyor. Popüler bilim yayıncılığı ve amatör bilim etkinlikleri, bunu sağlayan en önemli araçlardır (Akoğlu, A., 2005). Popüler bilim, bilim insanlarının, bilimle uğraşmayan halkın anlayamayacağı dünyalarını, halkın anlayabileceği ve yararlanabileceği bir hale getirip halka sunar. Popüler bilim, çok ağır ve karmaşık olan, teknik terimlerin sıkça kullanıldığı bilim dilini, halkın diline yakın bir duruma indirger. Böylece halk, bilim ve teknoloji ile ilgili güncel olayları takip edebilir hale gelir.

Günümüzde bilimle ilgilenen kesimin başvuracağı temel kaynakların başında popüler bilim yayınları gelmektedir. Bilimsel olayları halka ulaştırmayı amaç edinen bu tür yayıncılık, “popüler bilim yayıncılığı” olarak isimlendirilir. Halkın bilimi daha iyi takip edebilmesi amacıyla popüler bilimin yaygınlaştırılmasında birçok argüman kullanılır. Bunların başlıcaları; gazete, dergi, televizyon programı ve belgeseldir. Günümüzde birçok televizyon kanalı ve gazeteler günlük olarak popüler bilim yayınları yaparak halka ulaşmaya çalışırken, belgesellerde, bilim ve teknolojideki gelişmelerin halkın anlayabileceği bir şekilde sunumu yapılmaktadır. Popüler bilim dergileri ise, yayınlanma amacı gereği güncel olayları takip edip, bilimsel gündemi yakalayan ve genellikle de aylık yayın yapan dergiler olmaktadır. Uluslararası ve ulusal düzeyde yayın yapan basılı popüler bilim dergilerinden birkaç örnek verilebilir. Ulusal düzeyde ülkemizde popüler bilim alanında 1967 yılından beri TÜBİTAK tarafından düzenli olarak yayımlanan ‘Bilim ve Teknik’ dergisi dikkat çekici bir örnek olarak öne çıkmaktadır. Uluslararası düzeyde ise ABD menşeili ve 1888 yılından beri yayımlanan, ülkemizde ise 2001 yılından beri Türkçe olarak okuyucuyla buluşan ‘National Geografic’ dergisi örnek olarak verilebilir.

* 1. **Tezin Amacı ve Önemi**

Bizim tez projemizdeki amaçlarımızdan biri yapılan bilimsel çalışmaları toplumun yararına elektronik ortamda yayılmasını sağlamak. E-Science bu amaçla toplumun dünyadaki bilimsel gelişmeleri yakından takip etmesini sağlar. Çalışmalar internet ortamında saklanıp paylaşılarak sonraki çalışmalar için bilgi edinilmesini kolaylaştırır. Toplum dünyadaki gelişmeleri yakından takip edebilme şansı elde eder.

Bir diğer amacımız ise, bu tür bir web uygulaması geliştirip kurumların kendi makalelerini yayınlayabileceği bir ortam sunmak. Örnek vermek gerekirse bir üniversite kendi bünyesinde bulundurduğu fakülteleri kategorilendirip veya bir fakülte kendi öğretim üyelerinin makalelerini, not dokümanlarını, duyurularını yayınlayabileceği ortam sağlamak. Böylelikle kurum içi herhangi bir duyuru veya bilgi paylaşımı yapılabilir. Aynı zamanda projemiz genişletilerek ücretli, ücretsiz makale yayınlayan uluslararası siteler gibi düzenlenebilir.

Projemizde kullanılabilirliği sağlamak için makaleleri kategori bazlı olarak ayırdık. Kullanılabilirlik, kullanıcının bir ürün veya sistemle olan etkileşimini etkileyen faktörlerin birleşimidir (Kılıç, E. ve Güngör, Z., 2006). Daha kullanılabilir sistemlerin geliştirilebilmesi tasarım aşamasında bu faktörlerin dikkate alınmasıyla mümkündür. Kullanıcı odaklı ve kullanılabilirlik testleri yapılarak tasarlanmış arayüzlere sahip web sayfaları benzer içeriğe sahip diğer web sitelerine göre kullanıcılar tarafından daha çok tercih edilmektedir. Bu nedenle web sitelerinin tasarlanmasında kullanıcı özelliklerinin dikkate alınması gerekmektedir. Kullanıcı özelliklerine göre tasarlanan web siteleri, kullanıcılara daha etkin bir içerik sunmakta, site içerisinde kaybolmadan ve gereksiz işlemler yapmadan bilgiye erişme olanağı sağlamaktadır (Uçak, N. Ö. ve Çakmak, T., 2009). Web sitelerinde yer alan içeriğin kullanıcı beklenti ve özelliklerine göre tasarlanarak sunulması bu içeriğin etkin kullanımı açısından önemlidir.

Projemizde farklı türden yazarlar ile çeşitlilik ve fazla kitleye hitap edilmesi amaçlanmıştır. Sisteme kayıtlı olan yazarların kimlik girişleri doğrulandıktan sonra makale ekleyip, silip, düzenleyebileceği bir yönetici paneli ve profil bilgilerinin görüntülenebildiği arayüz bulunmakta. Eklenen makalenin uygun olup olmadığını kontrol amacıyla bir hakem (yönetici) kontrolünden geçirilmesi düşünülmüştür. Eklenen makale direk yayınlanmadan önce hakem tarafından kontrol edilip, uygun görülmesi sonucu ilgili internet sitesinde kullanıcıya gösterilir. Kategoriye göre eklenen makale, makaleye ulaşmak isteyen kullanıcılar için arama kolaylığı sağlamaktadır. Kullanıcının okumuş olduğu makaleye göre benzer makaleler önerilmektedir. Bir üniversite için yapılmış olan makale yönetiminde, öğretim üyeleri kendi dersleriyle ilgili not dokümanı veya duyuru yayınlayabilirler. Böylelikle öğrenciler uygulama üzerinden ders takibini gerçekleştirebilir, güncel teknolojiyi takip etme imkanı bulabilirler. Üniversitenin böyle bir makale yönetim sitesinin mevcut olması, fakülte öğrencilerini daha çok araştırma yapmaya ve makale okumaya, öğretim üyelerinin ise makale yazımına teşvik edecektir.

1. **BİLİMSEL MAKALELERİN NİTELİKLERİ**

Uluslararası bilimsel dergilerde aranan nitelikler pek çok kitap, dergi ve broşürlerde dolaylı yoldan genel olarak belirtilmiştir. Bunlara örnek olmak üzere aşağıda TÜBİTAK' ın ve TÜBA'nın aradığı nitelikler verilmiştir.

TÜBİTAK 1995 yılında Sağlık Bilimleri Araştırma Grubu bünyesinde oluşturduğu bir komisyon marifetiyle "Türk Tıp Dizini" kitaplarını çıkartmaktadır. Bilimsel dergilerin "Türk Tıp Dizini”ne kabul edilebilmesi için aşağıdaki maddelerde uygunluk esas alınmaktadır (TÜBİTAK, 1996).

1. Dergiler periyodik olarak ve düzenli şekilde yayınlanmalıdır. Dergilerin her sayısı, yayınlanması gereken tarihten itibaren en geç 6 ay içerisinde TÜBİTAK Sağlık Bilimleri Araştırma Grubu'na ulaştırılmalıdır.

2. Dergilerde kesinlikle hakem değerlendirme sistemi bulunmalı ve bu değerlendirmeler en az beş yıl süreyle saklı tutulmalıdır. Komisyon, ilgili yıl içinde dergilerden rastgele yöntemle seçtiği bir veya birkaç makalenin hakem raporlarını denetleyebilmelidir.

3. Dergiler araştırma yazıları içermelidir. Derleme yazılarından oluşan veya derleme ve olgu sunularının toplamı, makale sayısının yarısından fazla olan dergiler "Türk Tıp Dizini"ne kabul edilmeyecektir.

4. Makaleler İngilizce özet ve başlık içermelidir.

5. Makalelerin "Index Medicus" standartlarına uygun İngilizce anahtar sözcükleri olmalıdır.

6. Dergiler her yayın yılı sonunda kendi yıllık dizinlerini çıkartmalıdır.

7. Dergi için sürekliliği belgeleyen ISSN numarası Kültür Bakanlığından alınmalıdır.

TÜBA Akademi Konseyi 15.10.1996 günlü toplantısında Türkiye'de yayınlanmakta olan sosyal bilimler alanındaki dergileri desteklemek amacıyla bir "Bilimsel Dergileri Destekleme Programı" başlatmıştır. TÜBA (TÜBA, 1997)'ye göre nakdi veya bilimsel onay niteliğinde yardım desteği yapılacak dergilerde şu nitelikler aranmaktadır:

a) Derginin bir yayın kurulu bulunmalı ve dergilerde yer alan yazıların seçiminde hakemli değerlendirme yöntemi kullanılmış olmalıdır.

b) Dergide yayınlanan yazılar özgün araştırma ürünü ya da bilimsel tarama- değerlendirme (review) niteliğinde olmalıdır.

c) Derginin yılda kaç sayı yayınlanacağı ve hangi tarihlerde yayınlanacağı belirtilmiş olmalı ve buna titizlikle uyulmalıdır.

d) Dergi Türkiye çapında etkili bir bilimsel vakıf, dernek ya da üniversite tarafından yayınlanmalıdır.

e) Dergide Türkçe'nin yanı sıra, yazı konusu bakımından geçerli bir yabancı dilde makaleler de yer almalıdır. Türkçe makalelerin yabancı dilde, yabancı dildeki yazıların Türkçe özetleri mutlaka bulunmalıdır.

f) Uluslararası atıf endekslerince taranan dergilere öncelik verilir.

* 1. **Akademik Makale Yayınlayan Web Uygulamaları**

Aşağıda akademik makale yayınlayan benzer web uygulamalarının özellikleri kısaca belirtilmektedir.

* + 1. **Science direct**

Elsevier’in önde gelen akademik literatür programı olan Science direct ile araştırmacılar tam metinli dergi makaleleri, kitapları içeren milyonlarca yayına ulaşabilirler. Burada yayınlanan makalelerin hepsi ücretsiz değildir. Ücretli olan birçok makale bulunmaktadır.

Science direct, moleküler biyoloji, genetik, biyokimya, kimya, mühendislik, tıp, veterinerlik gibi konularda birçok koleksiyona sahiptir.

Science direct ile belli bir başlık, ya da yazar adı, yayın, cilt sayısı taraması ile belirli bir makaleye ulaşma imkanına mevcut.

* + 1. **Springer link**

Dünyanın lider araştırma platformudur. Çevrimiçi dergiler, e-kitaplar, ve referans kaynaklarına sahiptir. Springer link en kaliteli akademik araştırmalara hızlı ve kolay erişim sağlar.

Sisteme kayıt yaptırarak daha iyi araştırma sonuçları almamıza imkan sağlar. Ya da springerlink sayfasından ‘search’ başlığı ile ‘easy search’ alanına gelerek tarama kutucuğuna yazılan anahtar kelime ile basit tarama yapılabilir ‘expert search’ ile de gelişmiş tarama yapılabilir.

* + 1. **Web of science**

Thomas Reuters’in bir ürünü olan veri tabanı fen bilimleri, sosyal bilimler, insani bilimleri konu alan en kaliteli araştırmalara ulabilmeyi sağlar.

Başlık, kaynak adı, yayın tarihi, yazar adı ile tarama yapabilmektedir.

* + 1. **Scopus**

Hızlı, kolay ulaşımı ile araştırma konusunda oldukça faydalı olan scopus, fizik, kimya, matematik, mühendislik, sağlık bilimleri, sosyal bilimler, biyoloji, çevre bilimleri gibi konuları kapsar.

* + 1. **Research gate**

Akademisyenlerin sosyal ağlarından biri olan bu sitede oldukça fazla makale bulunmaktadır. Beğenilen yazarları takip etme seçeneği mevcut. Yazarlarla irtibata geçerek beğenilen makale ücretsiz bir şekilde istenilebilir.

* + 1. **Google Scholar**

Google’ın bütün ürününde olduğu gibi bu arama motorunda da kullanım kolaylığı ile ön plana çıkmaktadır. İstenilen konunun başlığını, yazarın adını arama kutusuna yazmak yeterli. Diğer makale tarama sitelerinden daha kolay ve hemen hemen aynı sonuçları veriyor.

1. **KULLANILACAK YÖNTEM VE TEKNOLOJİLER**

Proje kapsamında, kullanılan yöntem ve teknolojiler Tablo 3.1’de görülmektedir.

Tablo 3.1 Kullanılacak yöntem ve teknolojiler

|  |  |
| --- | --- |
| Donanım | PC |
| Yöntem | Katmanlı Mimari |
| Programlama Dili | C# |
| Kütüphaneler | ASP.NET MVC |
| Veri Tabanı | Microsoft SQL Server |
| Ortamlar | Visual Studio 15 |
| İşletim Sistemi | Windows 10 |

* 1. **Katmanlı Mimari**

Modern dünya da istemciler tarafından o kadar çok istek ve problem var ki yazılım teknolojilerin bunları yerine getirmesi, çözmesi pek de kolay olmuyor. İstenilen her bir fonksiyonellik yazılım projelerine uygulanması anlamına gelir. İstenilen fonksiyonelliklerin çok fazla olduğunu düşünürsek yazılım projeleri de kompleks bir hale geliyor. İşte bu kompleks yapıyı yönetmek ve değişimi kolaylaştırmak adına projeler belirli bir standarta ve düzene ihtiyaç duyuyor. Çok katmanlı mimari yaklaşım da bu karmaşık yapıyı sorumluluklara, fonksiyonelliklere, işlere ve görevlere göre birbirinden ayırarak daha basit ve yönetilebilir uygulamalar yapılmasını sağlar. Aslında çok katmanlı mimariyle yapılmak istenen, kendi içerisinde aynı amaca hizmet eden aynı mantık için oluşturulmuş kısımları ayrı bir katmanda bir araya getirmektir.

Örneğin, veri tabanı bağlantısı için yazılan kodlar bir katmana, işin fonksiyonelliğini içeren kodlar başka bir katmana gibi.

* + 1. **Desen açıklaması**

Katmanlı mimari desen içindeki bileşenler, her bir uygulama içinde belirli bir rol oynayan (örneğin sunum mantığı veya iş mantığı) yatay katmanlar halinde düzenlenir (Mark Richards, 2015). Katmanlı mimari desen, desende bulunması gereken katman sayısını ve türlerini belirtmese de, katmanlı katmanların çoğu dört standart katmandan oluşur: sunum, iş, kalıcılık ve veri tabanı (Şekil 3.1). Bazı durumlarda, iş katmanı ve kalıcılık katmanı, özellikle kalıcılık mantığı (örneğin, SQL veya HSQL) iş katmanı bileşenlerinin içine gömülü olduğunda, tek bir iş katmanı olarak birleştirilir. Bu nedenle, daha küçük uygulamalar sadece üç katmana sahip olabilir, oysa ki daha büyük ve daha karmaşık iş uygulamaları beş veya daha fazla katman içerebilir.



Şekil 3.1 Katmanlı mimari deseni

Katmanlı mimari modelin her katmanının uygulama içinde belirli bir rolü ve sorumluluğu vardır. Örneğin, bir sunum katmanı tüm kullanıcı arayüzünü ve tarayıcı iletişim mantığını idare etmekten sorumlu olurken, bir iş katmanı, taleple ilgili belirli iş kurallarını uygulamaktan sorumlu olacaktır. Mimarideki her katman, belirli bir iş talebini karşılamak için yapılması gereken işin etrafında bir soyutlama oluşturur. Örneğin, sunum katmanının müşteri verilerinin nasıl alınacağını bilmesi veya endişelenmesi gerekmez; bu bilgiyi sadece belirli bir formatta ekranda göstermesi gerekir. Benzer şekilde, iş katmanının bir ekranda görüntülenmek üzere müşteri verilerinin nasıl biçimlendirileceği veya müşteri verilerinin geldiği yerde bile endişelenmesi gerekmez; verileri kalıcılık katmanından alması, verilere karşı iş mantığı gerçekleştirmesi (örneğin, değerleri veya toplam verileri hesaplaması) ve bu bilgileri sunum katmanına aktarması gerekir.

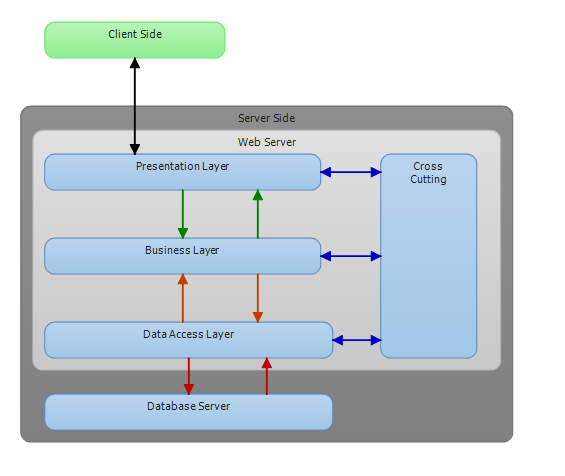
Veri Katmanı (Data Access Layer): Sadece veri tabanına erişimi sağlamakla sorumlu katmandır. Her veri tabanı için ayrı bir veriye erişim sınıfını içerebilir. Kurumsal uygulamalarda kullanıcı tabloları, raporlara ait tablolar, temel uygulama tabloları farklı farklı veri tabanlarında bulunabilir. Bu katmanda yazılan veri erişim sınıfı biraz daha soyut ve daha genelleyici bir yapıda olmasına dikkat edilmelidir.

İş Katmanı (Business Layer): Uygulamanın iş mantığı, veri tabanına ait operasyonel (CRUD) işlemler, kullanıcı rolleri, yönetimi, yetkilendirmeler bu katmanda yapılan projelerin sorumluluğundadır.

Sunum Katmanı (Presentation Layer): Yapılan uygulamanın kullanıcı arayüzüne ait projeler bu katmanda oluşturulur. Bu katmanda yapılan projeler; web uygulamaları, mobil siteler, mobil uygulamalar, Silverlight, Winform veya Html 5 gibi uygulamalarının kullanıcı arayüzüne ait dosyalar yer alır diyebiliriz.

Cross-Cutting Layer: Diğer 3 katmana ait projelerin ihtiyaç duyduğu ve projelerdeki teknik fonksiyonelliği içeren projeleri barındırır. Caching, ExceptionHandler, Loging gibi mekanizmalar bu katmanda yer alır ve ihtiyaca göre diğer katmanda bulunan projelere referans edilir. Burada kullanılan projeler kendi geliştirdiğiniz projelerde veya third party projelerde kullanılır. Örneğin Microsoft best-practiceslerinde olan Enterprise Library içerisindeki bloklar kullanılabilir.

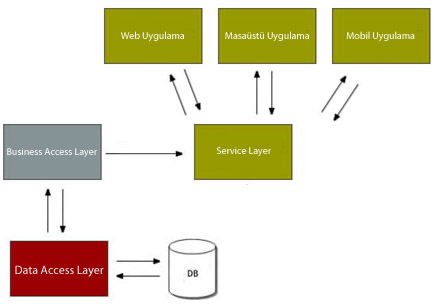
Aşağıdaki şekilde (Şekil 3.2) katmanlar arası veri akışı gösterilmektedir.



Şekil 3.2 Katmanlar arası veri akışı

Katmanlı mimarideki katmanların yapısı ve oluşumu projenin ihtiyaçlarına göre belirlenir. İhtiyaçlar doğrultusunda proje katman sayısı artırılıp azaltılabilir.

Servis Katmanı (Servis Katmanı): Servis katmanı, uygulamalarımızı servis ettiğimiz katmandır. Bu katman üzerinden, iş katmanı üzerinde yazdığımız fonksiyonellikleri web uygulaması, masaüstü uygulama, mobil uygulama gibi uygulamalara servis edebiliriz (Şekil 3.3).



Şekil 3.3 Servis katmanı etkileşimi

* + 1. **Anahtar kavramlar**

Şekil 3.1'de, mimarideki katmanların her birinin kapalı olarak işaretlenmiş olduğuna dikkat edin. Bu katmanlı mimari modelinde çok önemli bir kavramdır. Kapalı bir katman, bir istek katmandan katmana geçerken, altındaki katmandan bir sonraki katmana geçmek için hemen altındaki katmandan geçmesi gerektiği anlamına gelir. Örneğin, sunum katmanından kaynaklanan bir istek, önce iş katmanından ve daha sonra da veri tabanı katmanına çarpmadan önce kalıcılık katmanından geçmelidir.

Aşağıda kapalı katmanlı mimari (Şekil 3.4) içeren bir yapı gösterilmektedir.



Şekil 3.4 Kapalı katmanlar ve erişim isteği

İzolasyon kavramı katmanları, mimarinin bir katmanında yapılan değişikliklerin genellikle diğer katmanlardaki bileşenleri etkilememesi veya etkilememesi anlamına gelir: değişiklik, bu katman içindeki bileşenlere ve muhtemelen başka bir ilişkili katmanı (örneğin kalıcı bir katman gibi SQL). Sunum katmanının kalıcılık katmanına doğrudan erişmesine izin verirseniz, kalıcılık katmanında SQL'de yapılan değişiklikler hem iş katmanını hem de sunum katmanını etkiler, böylece bileşenler arasında çok fazla birbirine bağlı olan çok sıkı birleştirilmiş uygulama üretilir. Bu tür bir mimari daha sonra değiştirmek çok zor ve pahalı hale gelir.

İzolasyon kavramı katmanları, mimarinin bir katmanında yapılan değişikliklerin genellikle diğer katmanlardaki bileşenleri etkilememesi veya etkilememesi anlamına gelir: değişiklik, bu katman içindeki bileşenlere ve muhtemelen başka bir ilişkili katmanı (örneğin kalıcı bir katman gibi SQL). Sunum katmanının kalıcılık katmanına doğrudan erişmesine izin verirseniz, kalıcılık katmanında SQL'de yapılan değişiklikler hem iş katmanını hem de sunum katmanını etkiler, böylece bileşenler arasında çok fazla birbirine bağlı olan çok sıkı birleştirilmiş uygulama üretilir. Bu tür bir mimari daha sonra değiştirmek çok zor ve pahalı hale gelir.

Katmanlı mimarinin avantajları;

* Kompleks ve gitgide karmaşıklaşan projelere daha kolay hakimiyet
* Hata ayıklamanın kolaylaşması
* Projede ekip olarak çalışmayı kolaylaştırır
* Değişim sürecini hızlandırır
* Esnek bir yapı sağlar

Çok katmanlı mimarinin dezavantajları ise;

* Deployment süresinin uzunluğu
  1. **C#**

C # Anders Heljsberg ve ekibi tarafından, Microsoft'un .NET platformu için geliştirilen nesne yönelimli bir programlama dilidir. ilk versiyonu 2002 yılında .NET Framework 1.0 ile yayınlanmıştır. C# dili güçlü nesne yaklaşımlı (object-oriented) yapısı ile Microsoft .NET platformunun temelini oluşturur. Söz dizimi bakımından C ve C++ benzemesi programcılar tarafından öğrenilmesini kolaylaştırmaktadır.

C# yapısı itibari ile daha kolay, esnek ve taşınabilir olması bakımından da ön plana çıkmaktadır. Bu nedenle 4 tane + işaretinin birleşimini andıran bemol işareti (#) ile ifade edilir. C# dili her ne kadar Microsoft'a ait olsa da, ECMA (ECMA-334) ve ISO (ISO/IEC 23270:2006) tarafından uluslararası standart olarak tanımlanmıştır.

C# dilinin genel özellikleri:

* Basit ve genel amaçlı bir dil
* Platformdan bağımsız bir dil
* Kullanımı kolay modern bir dil
* % 100 nesneye yönelik programlama dili
* Gelecek tabanlı uygulamalar için uygun ve sağlam bir dil
* Çöp toplama ve kaynak yönetimi ile programların belleğini çok daha verimli kullanan bir dil

ve bunlar gibi birçok özelliğe sahiptir.

C# ile şunlar yapılabilir:

* Konsol uygulaması geliştirme
* Windows uygulaması geliştirme
* NET uygulaması geliştirme
* Web servisleri yazma
* Mobil uygulama geliştirme (PDA, cep telefonları vb. için)
* DLL yazma

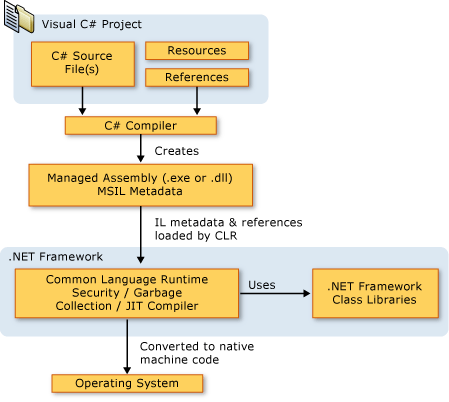
C#’ın çalışma mantığı:

C# Java dili gibi çalışma anında derleme prensibi ile çalışan temelli bir dildir. Visual Studio'da yazdığımız C# kodu derlendiğinde ; .NET Framework İşletim sistemi ve program arasında arabirim görevindeki CLR tarafından, .NET platformunun ortak ara dili olan MSIL'a dönüştürülerek derlenir.

Derlenen program çalıştırıldığında; MSIL Assembly formatındaki kodumuz .NET Framework tarafından işleme alınır ve CLR tarafından bellek temizleme (Garbage Collection) gibi işlemlerden geçirilerek, çalışma anında derleme ile bilgisayar üzerinde çalışan bir makine diline dönüşür.

Bu çalışma prensibi, C gibi doğrudan derlenerek çalıştırılan programlama dillerden daha yavaş çalışmasına neden olur. Fakat bu çalışma yapısı C# ile yazılan kodun platformdan bağımsız, farklı işletim sistemleri arasında taşınabilir olmasını sağlamaktadır. .Net platformunun gelişmiş CLR sistemi, C# rakibi olan Java Programlama dilinden daha hızlı çalışmasını sağlamaktadır.

Aşağıdaki şekilde (Şekil 3.5) C#’ın çalışma mantığı gösterilmektedir.



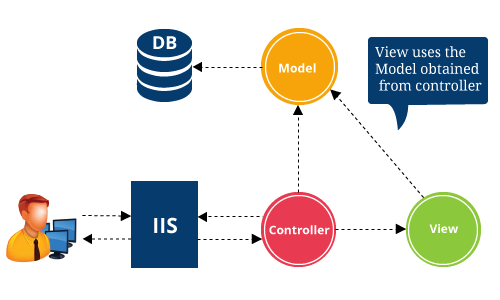
Şekil 3.5 C# çalışma mantığı

* 1. **Asp.NET Mvc**

ASP.NET MVC, MVC pattern’ini ASP.NET’e eklemek için Microsoft’un geliştirdiği framework’tür. MVC, uygulama geliştirmede (özellikle web uygulaması geliştirmede) önemli yere sahip mimari desenlerden biridir. MVC (Model-View-Controller), yazdığımız uygulamanın iş mantığı ile (business logic) kullanıcı arayüzünü birbirinden ayrıştıran, uygulamanın farklı amaçlara hizmet eden kısımlarının birbirine girmesini engelleyen yazılım mimarisidir.

Kodun farklı amaçlara hizmet eden yapılarını birbirinden ayırarak, kodu daha rahat geliştirilebilir ve test edilebilir (yani daha az hata çıkartma potansiyeline sahip) hale getirmiş oluruz.

Aşağıdaki şekilde (Şekil 3.6) MVC yapısının birbirleri ile arasındaki etkileşimi ve sunucu, istemci, veri tabanı arasındaki veri akışları gösterilmektedir.



Şekil 3.6 MVC yapısı

* + 1. **Model**

MVC dünyasında model uygulama verisinin veya durumunun saklandığı yerdir, genellikle veritabanı veya xml/json dosyası formatındadır.

Model, veri katmanını (database, xml, json dosyası, vb.) uygulamadan izole eder, böylece diğer katmanlarda veri katmanının neresi olduğunun bilinmesine gerek kalmaz.

Model katmanı sıklıkla Entity Framework, Nhibernate, LLBLGen, vb. gibi araçlar kullanılarak oluşturulur.

* + 1. **View**

View, istemcinin gördüğü arayüzü içeren katmandır, genellikle model katmanındaki verinin kullanılması ile oluşturulur. View katmanının model ve controller katmanlarından ayrılması ile arayüz değişikliklerinin uygulamanın diğer katmanlarını değiştirmeye gerek kalmadan yapılabilmesi sağlanmıştır.

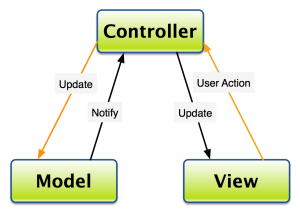
View katmanında HTML5 ve CSS3 gibi son versiyon teknolojiler kullanmak mümkündür. HTML5 ve CSS3 ile masaüstü ve mobil tarayıcılarda çalışabilen uygulamalar geliştirmek çok kolaylaşmıştır. Hatta Windows Store uygulamaları geliştirmek için HTML5 ve CSS3 teknolojilerinden yararlanılabilir.

* + 1. **Controller**

Model ve view arasında getir götür işlemlerini gerçekleştirir.

Kullanıcıların view üzerinden gerçekleştirdiği işlemlerle alınan veriyi modele taşır, modelden aldığı veriyi view üzerinden kullanıcıya gösterir.

MVC yapısında ana mantık model ve view yapısının ayrılmasıdır. Bu iki yapı arasındaki haberleşmeyi sağlayan köprüye controller (Şekil 3.7) diyoruz.



Şekil 3.7 Controller yapısı

MVC’nin diğer bir önemli yapıtaşı routing mekanizmasıdır. Routing, istemcinin uygulamaya yaptığı isteği uygun controller ve action’a yönlendiren yapıdır. İstemci, isteği uygulamanın belli bir adresine gönderir, routing mekanizması sayesinde ilgili adres için en uygun controller ve içerisindeki action tespit edilir ve çalıştırılır.

ASP.Net MVC’nin tercih edilme sebepleri:

MVC ile istemcinin isteğine karşılık üretilen çıktı üzerinde çok büyük kontrol imkanı vardır. Bu sayede her alanına müdahale edebildiğimiz ve isteğe en uygun çıktının üretilebilmesi sağlanmıştır.

MVC ile tekrar kullanılabilir (reusable) kod üretmek mümkündür. MVC’nin katmanları birbirinden ayrıldığı için her bir katmanın başka projelerde kullanılabilmesi sağlanmıştır.

MVC ile istemcinin istek göndereceği adresler üzerinde çok büyük kontrol imkanı vardır. Bu sayede adresin içerik ile tam bir ilişki içerisinde olması sağlanmıştır. Arama motorları için adres-içerik ilişkisi önemli olduğu için uygulamanın bulunabilirliğine katkısı yüksektir.

MVC ile test edilebilir uygulamalar geliştirme çok kolaylaşmıştır. Katmanların birbirinden ayrı olması ve Test Driven Development yapmayı kolaylaştırıcı mimarisi sayesinde test edilebilir uygulama geliştirilebilmesi sağlanmıştır.

Sonuç olarak; MVC uzun yıllardır bir çok frameworkte ve programlama dilinde kullanılmış (Java, PHP, vb.) ve olgunlaşmış bir desendir. ASP.NET MVC sayesinde .Net framework dilleri ile MVC pattern kullanılarak hızlı çalışan, test edilebilir, tekrar kullanılabilir parçaları olabilen web uygulamaları geliştirilebilmesi sağlanmıştır.

* 1. **Microsoft SQL Server**

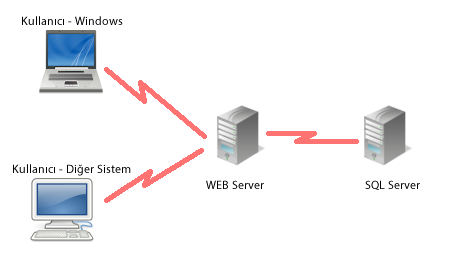
SQL (Structured Query Language), ilişkisel veritabanı yönetim sistemlerinden veri almak, veritabanında bulunan veriyi düzenlemek veya sisteme veri girişi yapmak için kullanılan en popüler sorgulama dilidir. SQL temelde, nesne-ilişkili (object-relational) veritabanı yönetim sistemlerini desteklemek için tasarlanmıştır.

MSSQL (Microsoft SQL Server), herhangi bir web sitesi veya yazılımın içerisinde kullanılan verilerin içerisinde sakladığında bir veritabanı sistemidir. Örneğin bir blog içerisindeki yazılar, yorumlar, kullanıcı bilgileri ve daha birçok veri MSSQL yardımıyla depolanabilmektedir. Windows tabanlı sunucular ve programlama dillerinde MSSQL en çok kullanılan veri tabanı tipidir. MSSQL ücretsiz bir veritabanı sistemi olmamasına karşın kullanıcılara gelişmiş ve öne çıkan özellikler sunması nedeniyle daha çok tercih edilmektedir.

MSSQL ne için kullanılır:

MSSQL, Windows platformlar üzerinde .NET veya ASP programlama dili aracılığıyla oluşturulan web siteleri ve web yazılımlarda veritabanı görevi görmesi amacıyla kullanılır. Bu yazılım dilleri içerisinde MS SQL veri tabanına bağlanmak ve bu veri tabanı üzerinde işlemler gerçekleştirmek birçok programlama diline göre çok daha kolaydır.

Aşağıdaki şekilde (Şekil 3.8), kullanıcı sistemi hangisi olursa olsun bir WEB uygulamasına bağlandığında öncelikle WEB uygulamasının server’ı çalıştırılacaktır ve bu server üzerinden kullanıcıya yansıyan bilgilerin hepsi arkada planda yer alan bir veritabanı yönetim sisteminden gelecektir.



Şekil 3.8 Web uygulamasının aygıtlarla etkileşimleri

* 1. **Visual Studio 15**

Visual Studio, birçok programlama dilini kullanarak program, uygulama ya da web sitesi yapabileceğiniz bir IDE yani entegre geliştirme ortamıdır. Microsoft Windows için bilgisayar programları, web siteleri, web uygulamaları, web hizmetleri ve mobil uygulamalar geliştirmek için kullanılır.

Visual Studio, Windows API, Windows Forms, Windows Presentation Foundation, Windows Store ve Microsoft Silverlight gibi Microsoft yazılım geliştirme platformlarını kullanır. Hem yerel kod hem de yönetilen kod üretebilir.

Visual Studio, IntelliSense'i (kod tamamlama bileşeni) ve kod yeniden düzenleme işlemini destekleyen bir kod düzenleyici içerir. Entegre hata ayıklayıcı, hem kaynak düzeyinde hata ayıklayıcı hem de makine düzeyinde hata ayıklayıcı olarak çalışır. Diğer yerleşik araçlar arasında bir kod profili oluşturucu, GUI uygulamaları oluşturmak için form tasarımcısı, web tasarımcısı, sınıf tasarımcısı ve veritabanı şeması tasarımcısı bulunur. Neredeyse her düzeyde işlevselliği artıran eklentileri kabul eder.

Visual Studio, farklı programlama dillerini destekler ve dile özgü bir hizmet olması koşuluyla, kod düzenleyicisinin ve hata ayıklayıcının hemen hemen tüm programlama dillerini desteklemesine olanak tanır. Yerleşik diller arasında C, C ++ ve C ++ / CLI (Visual C ++ ile), VB.NET (Visual Basic .NET ile), C # (Visual C # ile), F # (Visual Studio 2010’dan itibaren) ve TypeScript (Visual Studio 2013 Update 2’den itibaren) bulunur.

Diğer dillerin yanı sıra Python, Ruby, Node.js ve M gibi diğer dilleri destekleme, ayrı olarak yüklenen dil servisleri aracılığıyla sağlanır. Aynı zamanda XML / XSLT, HTML / XHTML, JavaScript ve CSS'yi de destekler. Java geçmişte destekleniyordu.

1. **E-SCIENCE ANALİZİ VE TASARIMI**

E-Science’ın kapsamı doğrultusunda yapılan analiz ve tasarım aşamaları aşağıda detaylı olarak anlatılmıştır.

* 1. **Analiz**

Proje analizi kapsamında fonksiyonel ve fonksiyonel olmayan gereksinimleri ve de sistemin ana senaryoları aşağıda incelenmiştir.

* + 1. **Fonksiyonel gereksinimler**
* Yönetici, sisteme makale yazarı için profil oluşturabilmeli.
* Yönetici, sistemdeki makale yazarının profilini düzenleyebilmeli.
* Yönetici, sistemdeki makale yazarının profilini silebilmeli.
* Makale yazarı, sistemde kendisine ait şifre ile uygulamaya giriş yapabilmeli.
* Makale yazarı, sistemden çıkış yapabilmeli.
* Makale yazarı, kendisine ait profil bilgilerini değiştirebilmeli.
* Makale yazarı, kendisine ait şifreyi değiştirebilmeli.
* Makale yazarı, kendisine ait profil resmini değiştirebilmeli.
* Makale yazarı, sistemin kendisine verdiği kısıtlara göre makale ekleyebilmeli.
* Makale yazarı, kendisine ait makaleyi düzenleyebilmeli.
* Makale yazarı, kendisine ait makaleyi yayından kaldırabilmeli.
* Makale yazarı, kendisine ait makaleleri filtreleyip görüntüleyebilmeli.
* Sistem; makale yazarının makaleyi düzenleme, silme, yayından kaldırma eylemlerini güvenlik amacıyla teyit edecek şekilde soru sormalı.
* Sistem; makale yazarının makaleyi değiştirme, silme, yayından kaldırma eylemlerini gerçekleştirmeli.
* Sistem, makale yazarının eklediği makaleyi uygulamada ilgili kategoride ekleyip görüntülenebilir olmasını sağlamalı.
* Sistem, eklenen makaleyi kullanıcıya eskiden güncele olacak şekilde görüntülenebilir olmasını sağlamalı.
* Kullanıcı, uygulama içerisinde okumuş olduğu makaleye benzer içerikleri görüntüleyebilmeli.
* Kullanıcı, ilgili makalenin hangi yazara ait olduğunu görüntüleyebilmeli.
* Kullanıcı, ilgili makalenin detayını, yayınlandığı tarihi görüntüleyebilmeli.
  + 1. **Fonksiyonel olmayan gereksinimler**
* Sistem, 5000’den fazla isteğe yanıt verebilmeli.
* Sisteme atılan sorgulara karşı, en geç 5 saniye içerisinde cevap verebilmeli.
* Sitenin isteklere hızlı cevap verebilmesi için hızlı bir sunucu üzerine kurulması gerekmektedir.
* Sistem herhangi bir sunucu hatası vb. çökme hatalarına karşı önlem amacı ve verileri yedeklemek için yeterli bir kapasitede yedek sunucu üzerinde günlük olarak yedeklemelidir.
* Sistem, yazar tarafından yayınlanan makaleyi düzenlemek isterken süre aşımına uğrayan form isteğini tekrar göndermesini istemelidir.
* Kullanıcının uygulamayı kullanabilmesi için internet bağlantısı olmalıdır.
* Web siteye erişim için bilgisayar, tablet, cep telefonunda internet tarayıcısı yüklü olmalıdır.
* Programlar, .Net Framework 4.7.2, MSSQL kullanılarak implemente edilmeli.
  + 1. **Domain gereksinimleri**
* Kullanıcı, makale yazarı verileri şirketlerle paylaşılmamalıdır. (Veri Gizliliği)
* Kullanıcı cihazları uygulama tarafından izlenmemelidir. (Kişisel Güvenlik)
* Yöneticiler, geliştiriciler kullanıcıların verilerini işlememelidir.
  + 1. **Kullanım senaryoları**

Projenin kullanım senaryoları şu şekildedir:

* Makale Ekleme
* Makale Görüntüleme
* Şifre Değiştirme
* Profil Bilgilerini Değiştirme
  + - 1. **Use case uc1: makale ekleme**

Level: User Goal

Primary Actor: Makale Yazarı

Stakeholders and Interest:

* Makale Yazarı, doğru ve hızlı bir şekilde makalesini siteye eklemek ister.
* Sistem, makale yazarının eklemek istediği makaleyi ilgili yazarın kategorisine hızlı bir şekilde yayınlamak ister.

Pre-Conditions:

* Makale yazarının uygulama içerisinde kayıtlı bir hesabı olmalı.
* Makale yazarı uygulamaya giriş yapmış olmalı.
* Web sunucusu aktif olmalı.

Post-Conditions:

* Eklenmek istenen makale ilgili kategoride yayınlanır.

Main Success Scenario:

1. Makale yazarı makale oluşturma isteğinde bulunur.
2. Sistem, yazardan eklenecek makalenin bilgilerinin girilmesini ister.
3. Makale yazarı sırasıyla makalenin; başlığını, kısa açıklamasını, resmini, kategori ve tarihini, içeriğini girerek kaydetme isteğinde bulunur.
4. Sistem makaleyi veri tabanına ekler ve sunucu üzerinde yayınlar.

Alternative Flow (Extension):

* Sistem herhangi bir anda çökerse;
* Yedek sunucu üzerinden tarafından ilgili veriler getirilir ve sistem yeniden başlatılır.

3.1. Makale bilgileri yanlış veya eksik girilirse eğer;

1. Yanlış veya eksik girilen bilginin girilmesi istenir.

3.2. Seçilecek makale kategorisi yoksa eğer;

1. Makale yazarı, ilgili konu için yeni kategori eklemeli, ardından ilk adıma dönülerek aynı işlemler tekrarlanmalıdır.
   * + 1. **Use case uc2: makale görüntüleme**

Level: User Goal

Primary Actor: Kullanıcı

Stakeholders and Interest:

* Kullanıcı, okumak istediği makaleyi hızlı bir şekilde görüntülemek ister.
* Sistem, kullanıcının görüntülemek istediği makaleyi en hızlı bir şekilde kullanıcıya iletmek ister.

Pre-Conditions:

* Kullanıcı uygulamayı çalıştırmış olmalı.
* Site içerisinde yayınlanan en az bir makale olmalı.
* Web sunucusu aktif olmalı.

Post-Conditions:

* Kullanıcı okumak istediği makaleyi görüntüler.

Main Success Scenario:

1. Kullanıcı uygulamayı çalıştırarak makale görüntüleme isteğinde bulunur.
2. Sistem kullanıcıya yayınlanan makaleleri gösterir.
3. Kullanıcı görüntülemek istediği makaleyi seçer.
4. Sistem seçilen makale detayını kullanıcıya gösterir.

Alternative Flow (Extension):

* Sistem herhangi bir anda çökerse;
* Yedek sunucu üzerinden tarafından ilgili veriler getirilir ve sistem yeniden başlatılır.

3.1. Kullanıcı kategori bazlı arama yaparak makale görüntülemek isterse eğer;

1. Aranacak yayınlar için kategori seçilir.
2. Seçilen kategoriye ait yayınlar kullanıcıya gösterilir.

3.2. Kullanıcı ana sayfada makale görüntülemek isterse eğer;

1. Yayınlar tüm kategorileri içerecek şekilde güncelden eskiye sıralı şekilde kullanıcıya gösterilir.
   * + 1. **Use case uc3: şifre değiştirme**

Level: User Goal

Primary Actor: Makale Yazarı

Stakeholders and Interest:

* Makale yazarı, güvenli bir şekilde şifresini değiştirmek ister.
* Sistem, şifre değişikliğini doğru ve güvenli bir şekilde güncellemek ister.

Pre-Conditions:

* Makale yazarının uygulama içerisinde kayıtlı bir hesabı olmalı.
* Makale yazarı, uygulamaya giriş yapmış olmalı.
* Web sunucusu aktif olmalı.

Post-Conditions:

* Makale yazarının, şifresi güncellenir.

Main Success Scenario:

1. Makale yazarı sisteme şifre değiştirme talebinde bulunur.
2. Sistem, yazara şifre bilgilerini gösteren arayüz gösterir.
3. Yazar, kullanıcı adı ve yeni şifresini yazar.
4. Sistem yazarın şifre bilgilerini günceller.

Alternative Flow (Extension):

* Sistem herhangi bir anda çökerse;
* Yedek sunucu üzerinden tarafından ilgili veriler getirilir ve sistem yeniden başlatılır.

3.1. Doğrulama işlemi sırasında iki şifre birbiriyle aynı değilse;

1. Sistem girilen iki şifrenin aynı olmadığını belirten uyarı mesajı verir.

2. Makale yazarı, şifreleri tekrar kontrol ederek baştan girer.

* + - 1. **Use case uc4: profil bilgilerini değiştirme**

Level: User Goal

Primary Actor: Makale Yazarı

Stakeholders and Interest:

* Makale yazarı, güvenli bir şekilde profil bilgilerini değiştirmek ister.
* Sistem, profil bilgi değişikliğini doğru ve güvenli bir şekilde güncellemek ister.

Pre-Conditions:

* Makale yazarının uygulama içerisinde kayıtlı bir hesabı olmalı.
* Makale yazarı, uygulamaya giriş yapmış olmalı.
* Web sunucusu aktif olmalı.

Post-Conditions:

* Makale yazarının, profil bilgileri güncellenir.

Main Success Scenario:

1. Makale yazarı sisteme profil bilgilerini değiştirme talebinde bulunur.
2. Sistem, yazara profil bilgilerini içeren pencereyi gösterir.
3. Yazar, değiştirmek istediği bilgileri (ismi ve mesleği) girer.
4. Sistem yazarın bilgilerini günceller.

Alternative Flow (Extension):

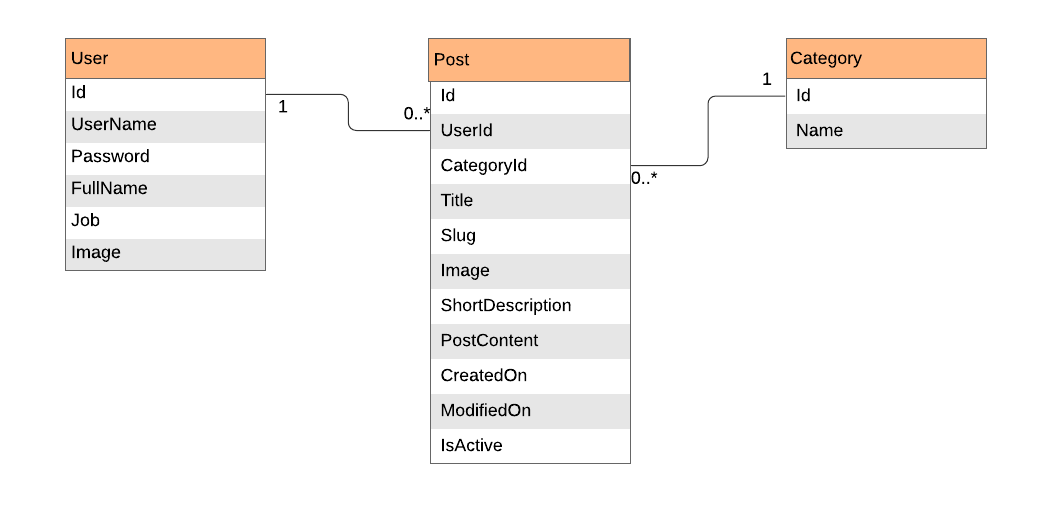
* Sistem herhangi bir anda çökerse;
* Yedek sunucu üzerinden tarafından ilgili veriler getirilir ve sistem yeniden başlatılır.
  1. **Tasarım**

Proje tasarımı için öncelikle veritabanı, sonrasında kullanım senaryolarının SSD’leri oluşturuldu.

Tasarım, analiz sonucu elde edilen bulgular ışığında gerçekleştirilmiştir.

* + 1. **Veritabanı tasarımı**

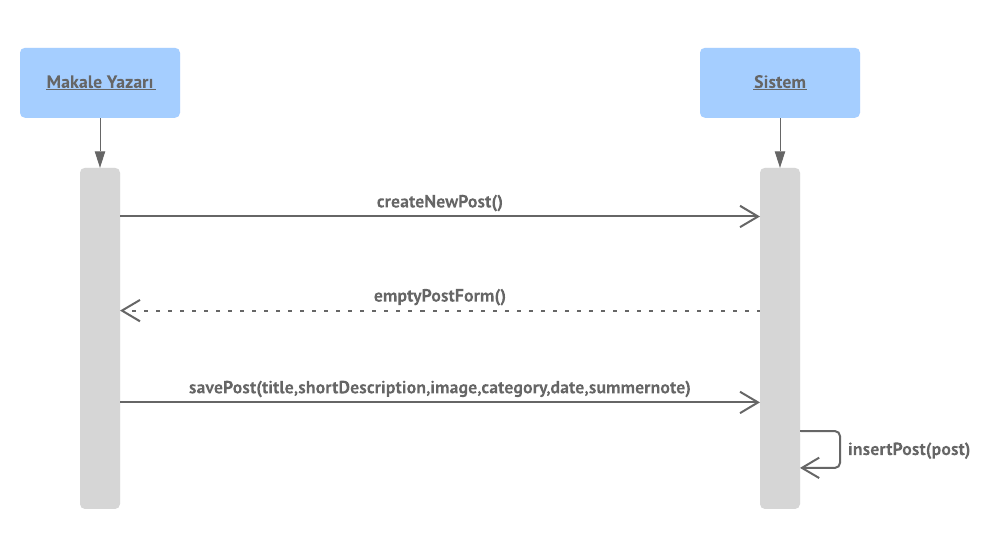
Aşağıdaki şekilde veritabanı tablosu ve bağlantıları gösterilmiştir (Şekil 4.1).



Şekil 4.1 E-Science veritabanı ilişkisi

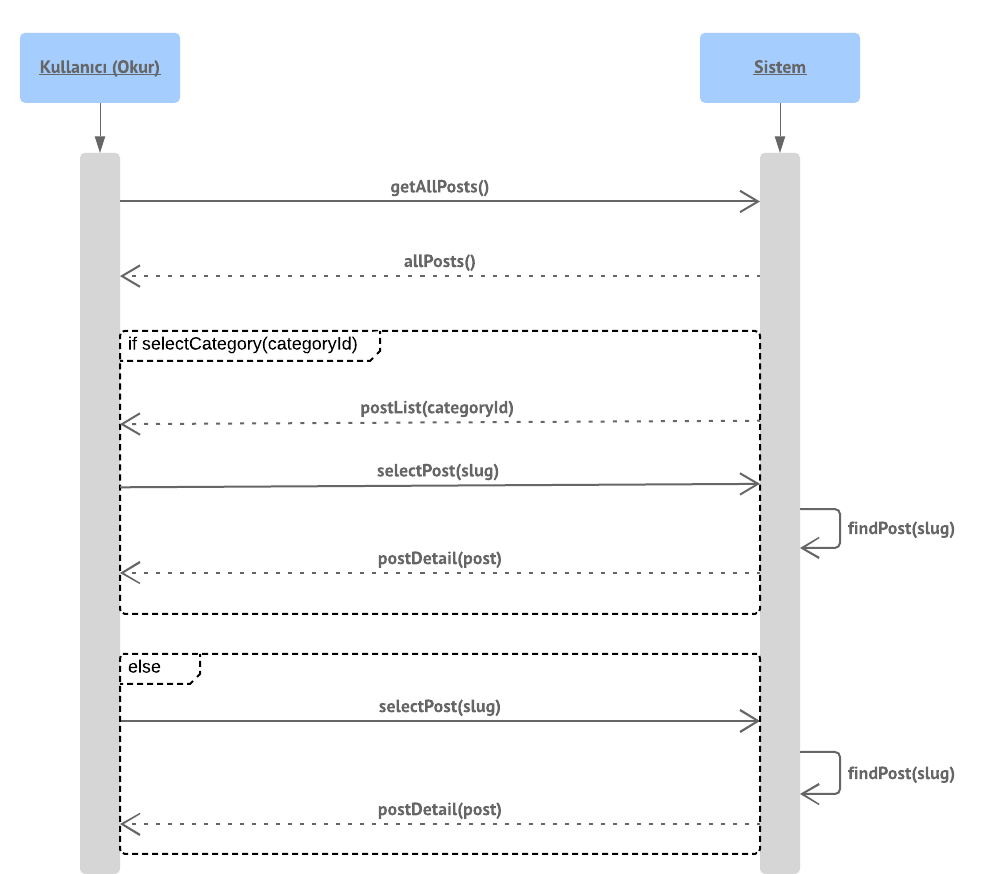
* + 1. **Kullanım senaryoları ssd**

Makale ekleme SSD’si aşağıdaki şekildeki gibidir (Şekil 4.2).



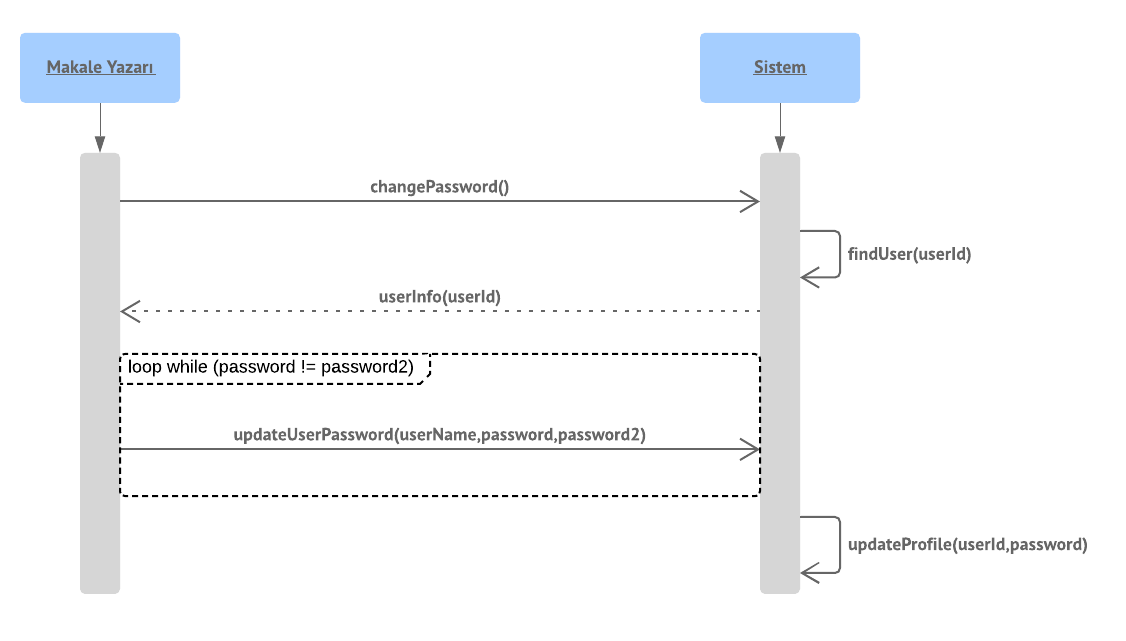
Şekil 4.2 Makale ekleme SSD

Makale görüntüleme SSD’si aşağıdaki şekildeki gibidir (Şekil 4.3).



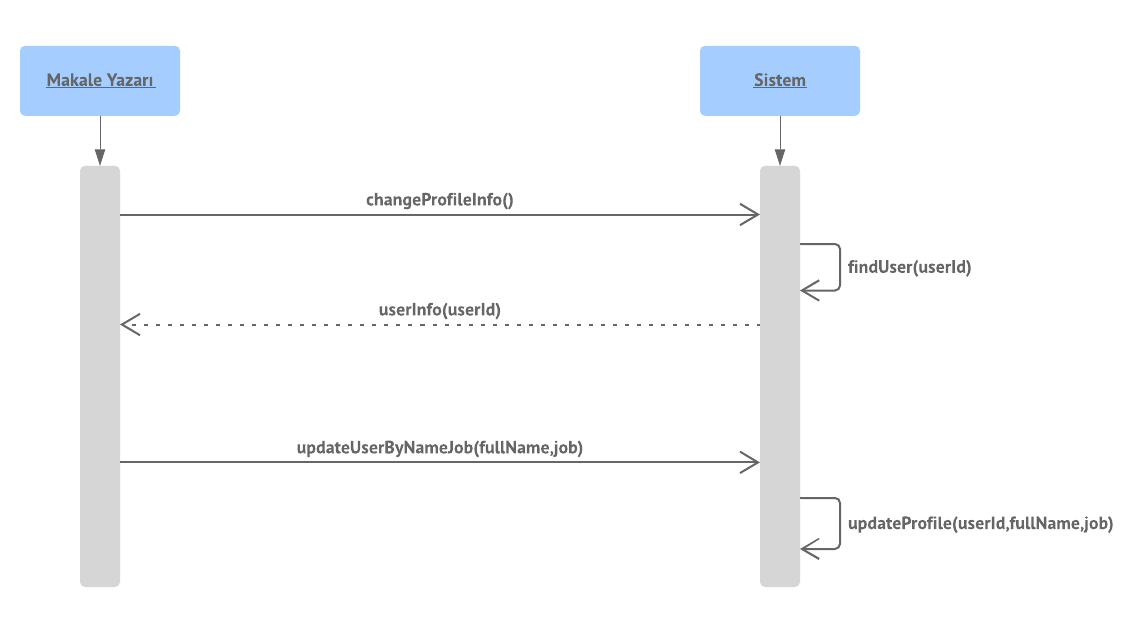
Şekil 4.3 Makale görüntüleme SSD

Şifre değiştirme SSD’si aşağıdaki şekildeki gibidir (Şekil 4.4).



Şekil 4.4 Şifre değiştirme SSD

Profil bilgilerini değiştirme SSD’si aşağıdaki şekildeki gibidir (Şekil 4.5).



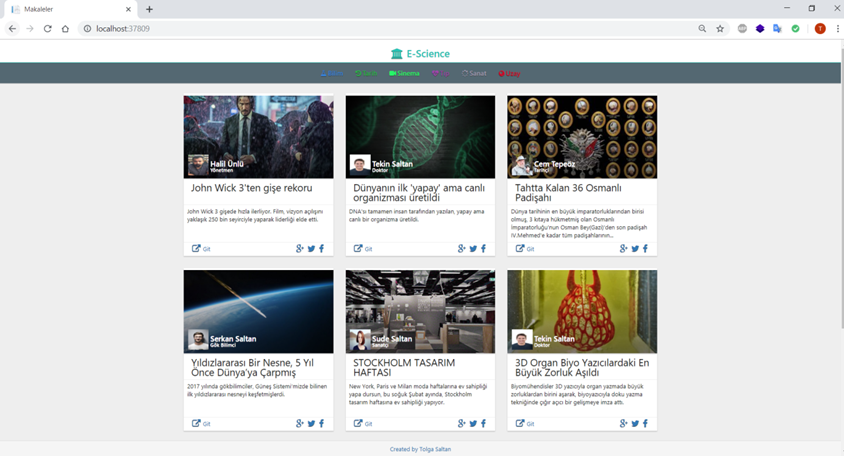
Şekil 4.5 Profil bilgilerini değiştirme SSD

1. **E-SCIENCE KULLANICI KILAVUZU**

Bu bölümde E-Science’in sunucu üzerinde site kullanıcılarının ve ziyaretçilerine aktarılan arayüz ve makale yazarlarının sistemi nasıl kullanması gerektiğini belirten açıklamalar içermektedir. Sistemin kullanımı ve menüler aşağıdaki ekran görüntüleriyle birlikte tanıtılmıştır.

* 1. **Sitenin Ana Sayfası**

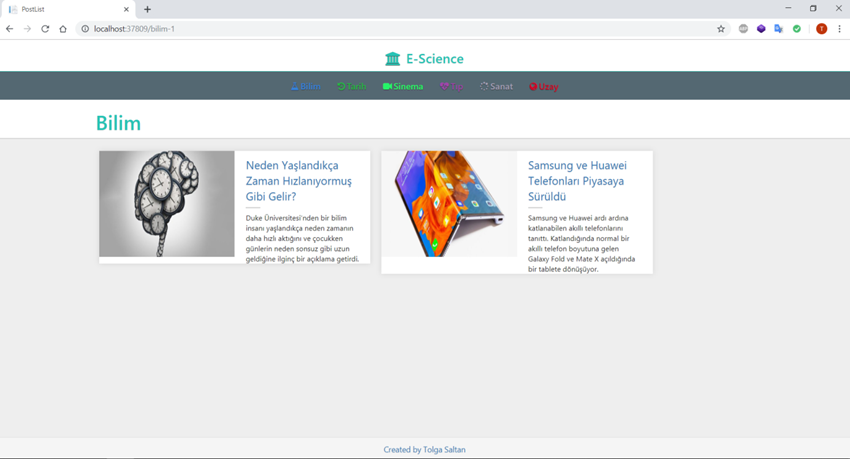
Site kullanıcıları veya ziyaretçiler domain adresini yazıp siteye girmek istediklerinde ilk olarak aşağıdaki şekildeki (Şekil 5.1) gibi ana sayfa ile karşılaşmaktadır. Bu sayfada site üzerinde yayınlanan makaleler güncelden eskiye bir şekilde kategori olarak ayrım yapılmadan rastgele bir şekilde makalenin başlığı, kısa içeriği ve yazarı ile birlikte gösterilmektedir.



Şekil 5.1 Ana sayfa

* 1. **Kategori Bazlı Makale Sayfası**

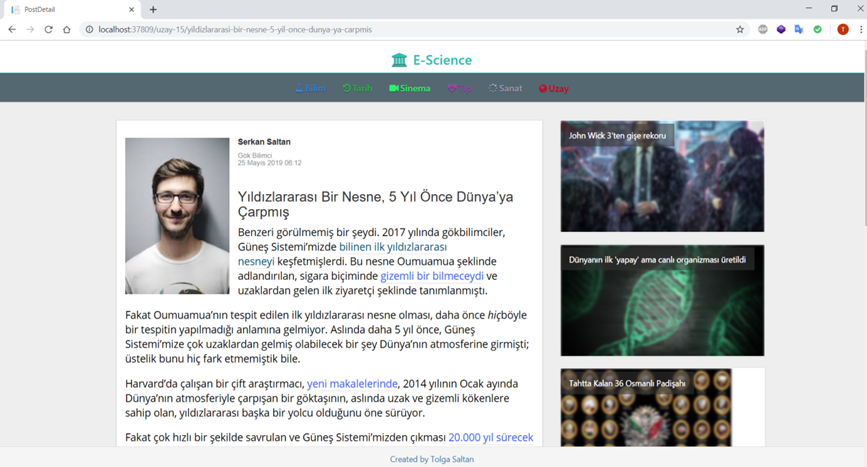
Kategori seçerek, filtreleme işlemi sonrası sadece ilgili kategoriye ait yayınlanan makaleler görüntülenebilir. Kategori seçilerek makale arama işlemi şekildeki gibidir (Şekil 5.2).



Şekil 5.2 Kategori sayfası

* 1. **Makale Görüntüleme Sayfası**

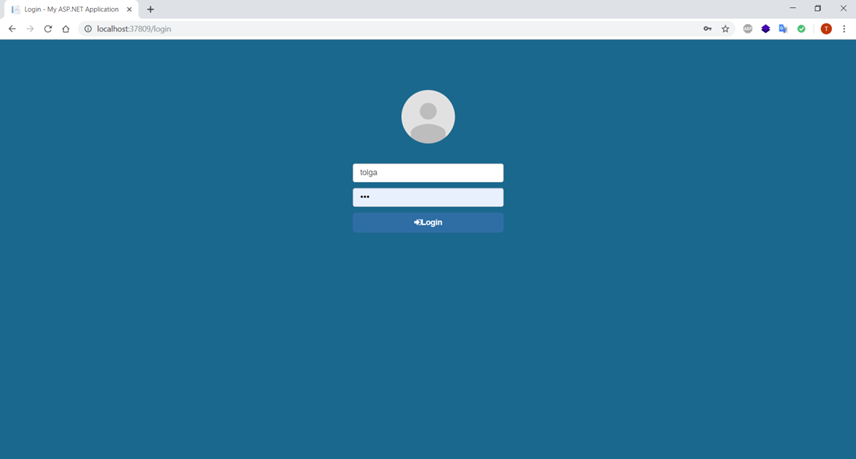
Yayınlanan makalenin detaylı bir şekilde içeriğine ulaşılmak istenirse ilgili makalenin “git” veya “makale ismine” tıklanarak aşağıdaki şekildeki (Şekil 5.3) gibi makale detaylı bir şekilde görüntülenebilir. Görüntülenen makalenin detayında aynı zamanda makalenin yayınlanma tarihi, yazarı ve makale yazarının mesleği de bulunmaktadır. Sağ panelde yayınlanan benzer makaleler ile hızlı bir şekilde görüntülenmek istenen makaleye geçiş yapılabilir.



Şekil 5.3 Makale görüntüleme sayfası

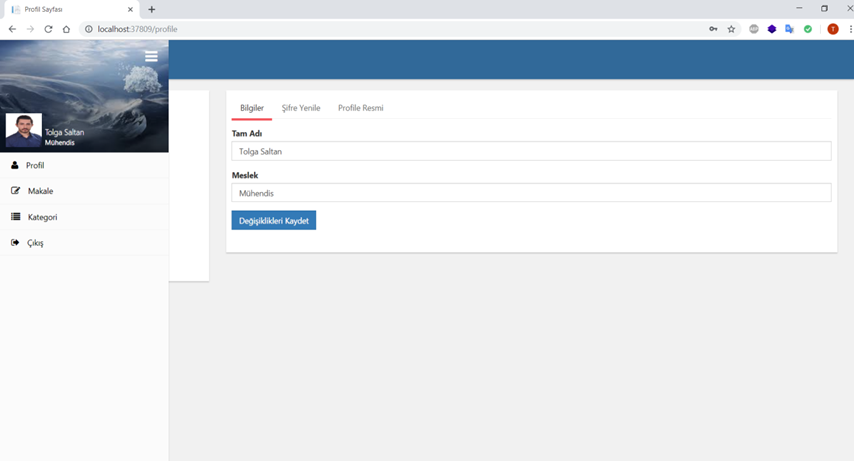
* 1. **Makale Yazarı Sisteme Giriş Sayfası**

Makale yazarları, sisteme giriş yaparken kendisine ait kullanıcı adı ve şifre bilgilerini doğrulayarak aşağıdaki şekilde (Şekil 5.4) verilen ekrandan giriş yapar.



Şekil 5.4 Giriş sayfası

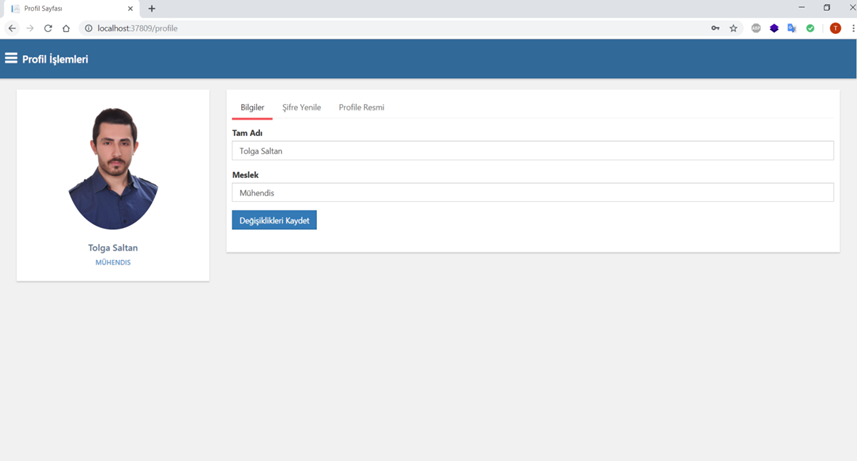
Sisteme giriş yapan yazarların profil bilgilerini içeren sayfa aşağıdaki gibidir (Şekil 5.5). Sol panelde bulunan dashboard “bars” denilen yapı ile kapatıp açılabiliyor. Bu dashboarddan yazar gerçekleştirmek istediği aksiyonu seçebilir.



Şekil 5.5 Profil sayfası

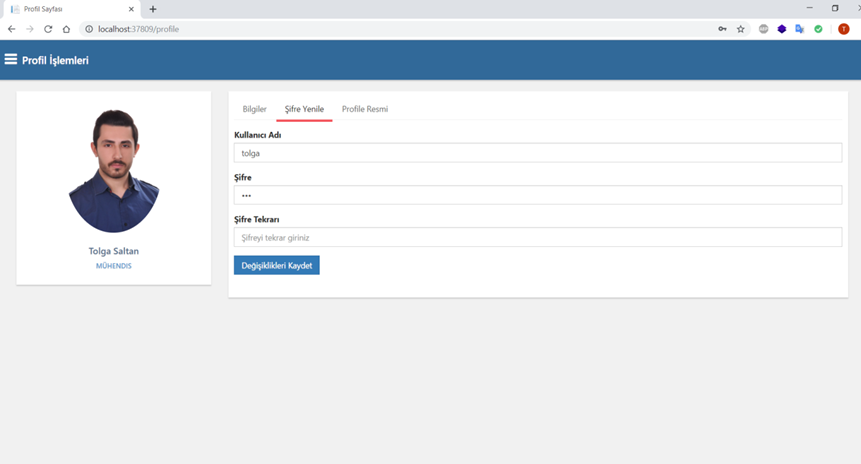
* 1. **Profil İşlemleri**

Bilgiler seçeneği ile yazar, adını ve mesleğini aşağıdaki şekilde (Şekil 5.6) bulunan ekrandan değiştirebilir.



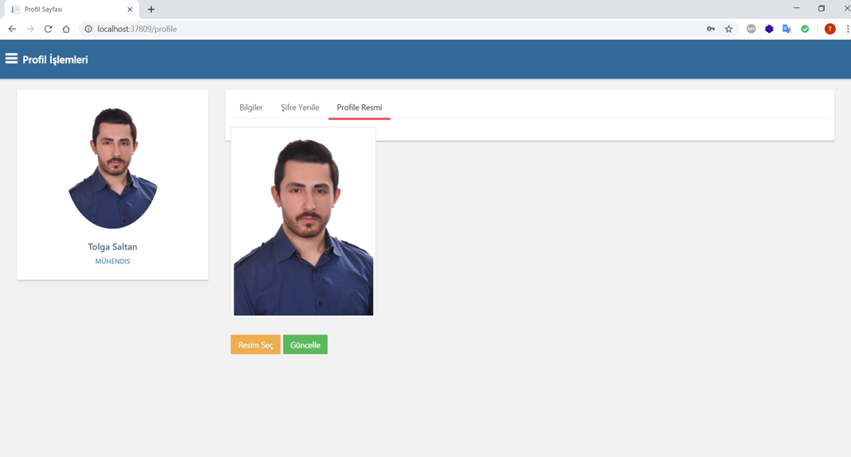
Şekil 5.6 Bilgileri değiştirme sayfası

Yazar, şifre yenileme seçeneği ile kendisine verilen şifreyi doğrulama işlemi ile beraber aşağıdaki şekilde (Şekil 5.7) bulunan ekrandan değiştirebilir. Sisteme kayıt edilen şifre encrypted bir şekilde veritabanına kayıt edilirken, sistemden şifre kontrolü de aynı şekilde decrypted bir şekilde gerçekleşir.



Şekil 5.7 Şifre değiştirme sayfası

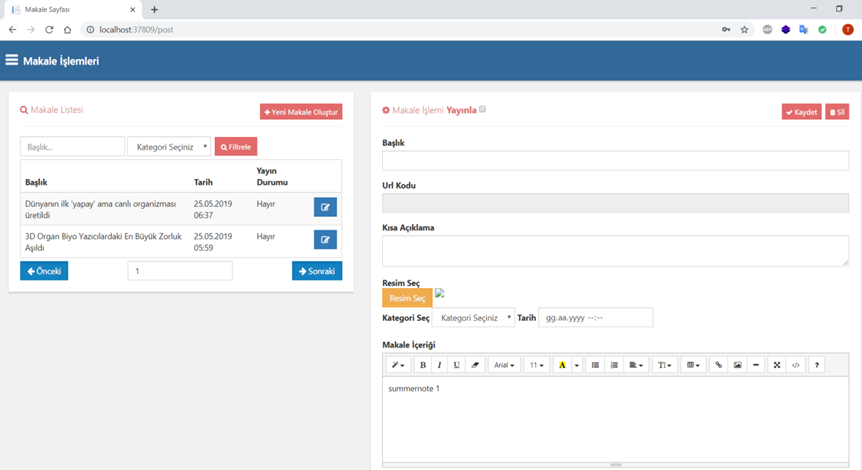
Yazar, profil resmini değiştirmek isterse aşağıdaki şekildeki gibi gerçekleştirebilir (Şekil 5.8).



Şekil 5.8 Profil resmi değiştirme

* 1. **Makale İşlemleri**

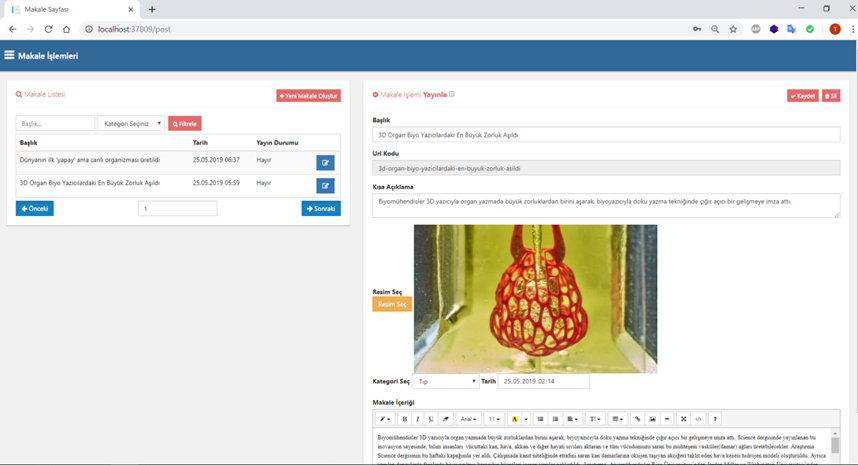
Yazarın, sistemdeki makalelerini görüntüleyip düzenleyebileceği, silebileceği, yeni bir makale oluşturup sisteme ekleyebileceği sayfa aşağıdaki gibidir (Şekil 5.9).



Şekil 5.9 Makale işlemleri

* 1. **Makale Düzenleme Sayfası**

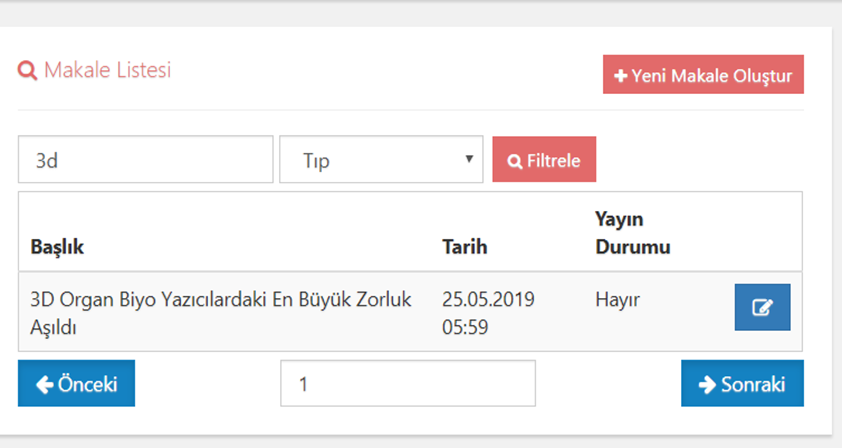
Sol panelde bulunan makalelerden biri seçilerek makale içeriği tekrardan düzenleme sayfası aşağıdaki gibidir.



Şekil 5.10 Makale düzenleme sayfası

* 1. **Makale Filtreleme**

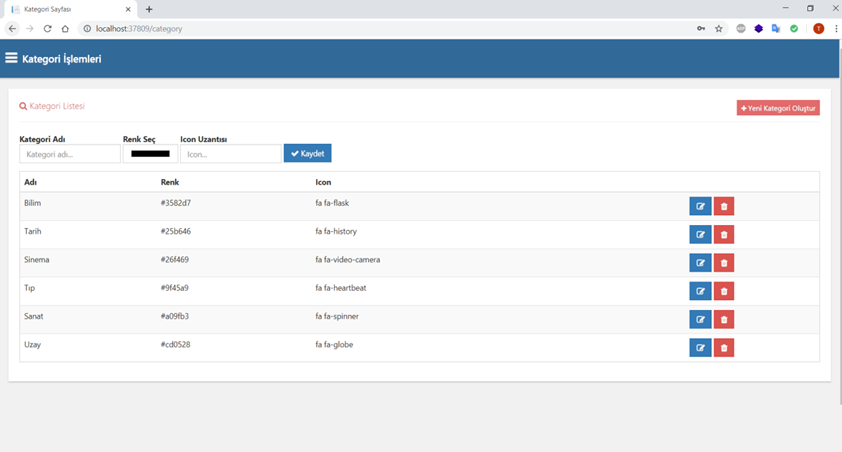
Yazarın başlık içeriğinde bulunan herhangi bir kelime veya ilgili kategori seçimi yapılarak makale filtreleme işlemi gerçekleştirilmesi aşağıdaki şekildeki gibidir (Şekil 5.11).



Şekil 5.11 Makale filtreleme ekranı

* 1. **Kategori İşlemleri**

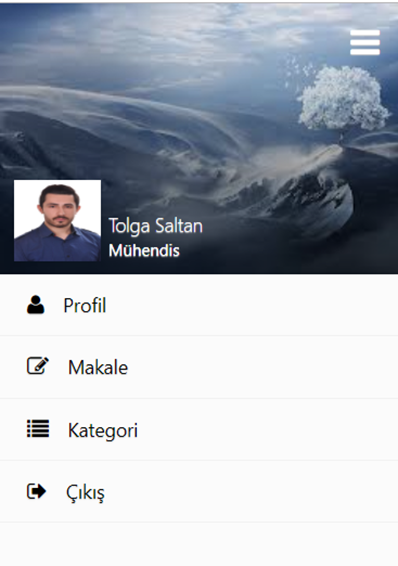
Kategori ekleme, kategorinin rengi veya simgesini değiştiren sayfa aşağıdaki şekildeki gibidir (Şekil 5.12).

****

Şekil 5.12 Kategori işlemleri

* 1. **Çıkış İşlemi**

Yazar, sistemden çıkış yapmak isterse aşağıdaki şekilde (Şekil 5.13) bulunan gösterge panelindeki çıkış seçeneği ile bu işlemi gerçekleştirebilir.



Şekil 5.13 Çıkış ekranı

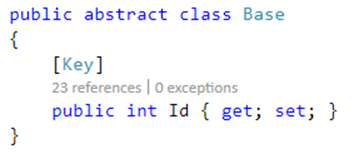
1. **GERÇEKLEŞTİRİM**

Proje gerçekleştirim aşamasında kullanmış olduğumu yöntem olan katmanlı mimari, Code First mimarisinin nasıl bir yapıda kullanıldığı kısa açıklamalar ve kod blokları bu bölümde incelenecektir.

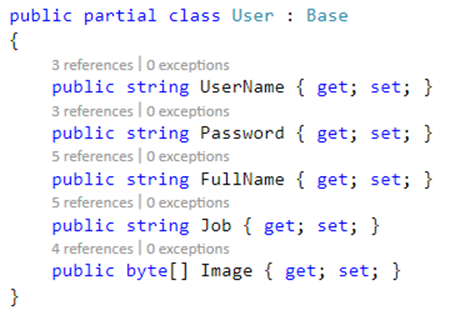
* 1. **Core Katmanı Entities**

Model katmanı entity sınıflarımızı içeren bir katmandır. Her entity ayrı bir class dosyasında yazılır. Enum’lar da aynı şekilde ayrı dosyalarda yer alır. Entity’lerin neredeyse tamamının miras aldığı BaseEntity adında bir sınıf bulunur. Sadece many-to-many ilişkileri kurmaya yarayan ara entityler BaseEntity’i miras almazlar. Entitylerde Data Annotation attribute’ları kullanılmaz. Bu attribute’lar yerine data katmanında Builder class’ları ile fluent-api kodları yazılacaktır.

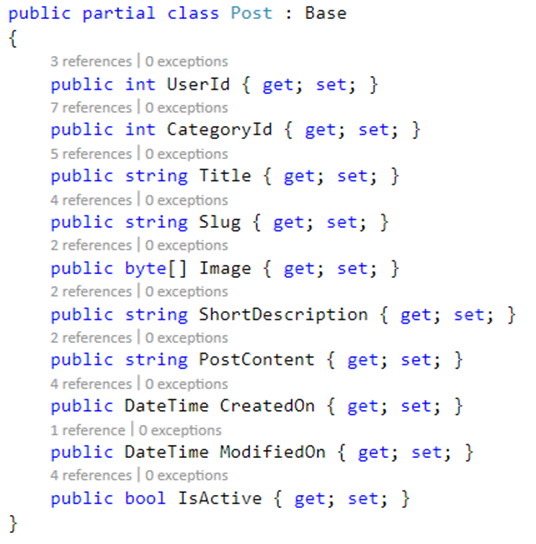
Aşağıdaki şekilde core katmanında kullanılan sınıflar sırasıyla gösterilmiştir (Şekil 6.1).



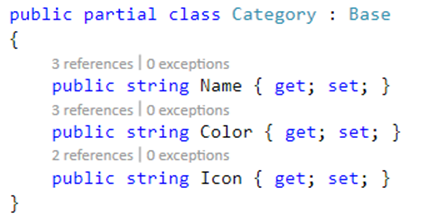
Şekil 6.1 a) Base class



Şekil 6.1 b) User class



Şekil 6.1 c) Post class



Şekil 6.1 d) Category class

**Şekil 6.1** Core Katmanı a) Base class, b) User class, c) Post class, d) Category class

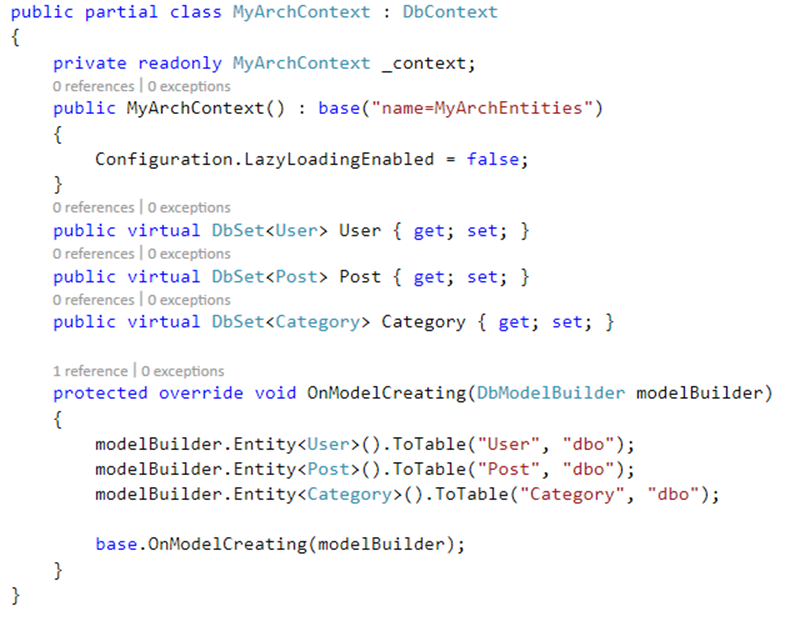
* 1. **Data Katmanı**

Data katmanı; contex yapısı, repository pattern ve unit of work pattern olmak üzere 3 bölüme ayrılır.

* + 1. **Context yapısı**

Veritabanı ve tabloların veri transferi için kod ile veritabanı arasındaki bağlantının sağlandığı yapıdır. Veritabanına erişim için entity framework kurulumu sırasında veritabanı bağlantı ismi aracılığı ile bağlantı sağlanır. Bağlantının isimlendirildiği yapı olan MyArchEntities, Web.config içerisinde connectionStrings bloğu içerisinde belirtilerek veritabanı bağlantısı sağlanır.

Projede kullanılan context yapısı aşağıdaki şekildeki gibidir (Şekil 6.2).



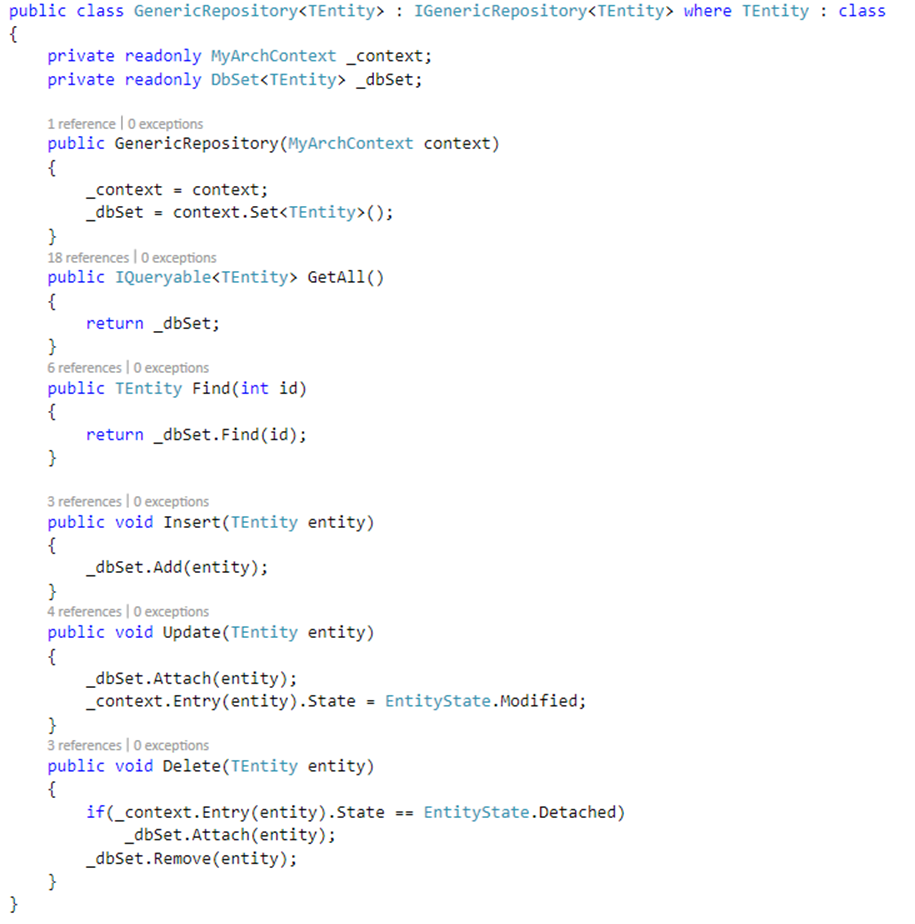
Şekil 6.2 Context yapısı kod bloğu

* + 1. **Repository pattern**

Bu desenin temelinde yatan asıl amaç; veritabanına veri ekleme, güncelleme ve okuma gibi CRUD (Create Read Update Delete) işlemlerimiz için oluşturmuş olduğumuz kodların tekrar kullanılabilirliğini sağlamaktır.

Yazılım geliştirmede, belkide en önemli prensiplerden birisi olan DRY (Don’t repeat yourself) yani kendini tekrarlama prensibine göre kod tekrarlarından sakınmak gerekmektedir. Repository pattern bize bunu sağlamaktadır. Oluşturduğumuz bir sınıf ile tüm tablolar için CRUD işlemlerimizi yapabilmemizi sağlamaktadır ve kod tekrarını önlemektedir.

Projede kullanılan repository pattern yapısı aşağıdaki kod bloğunda gösterilmektedir (Şekil 6.3).



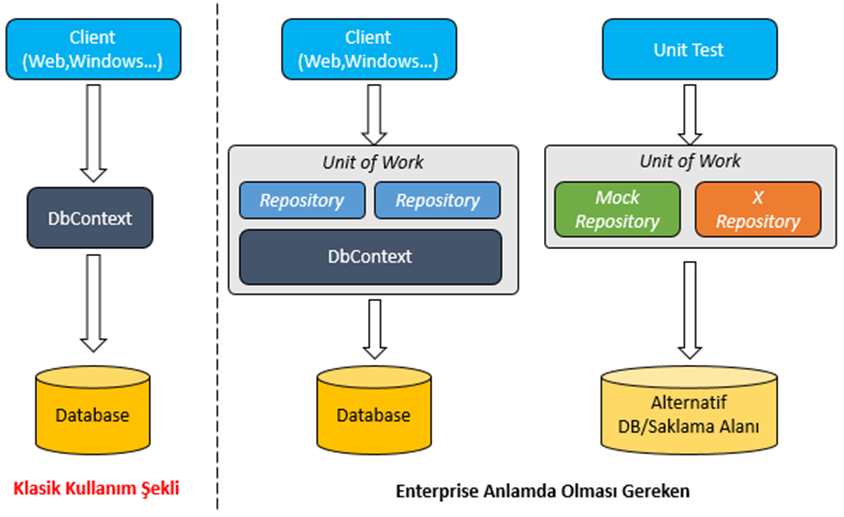
Şekil 6.3 Repository pattern kod bloğu

* + 1. **Unit of work pattern**

Unit of Work ise bize; veritabanı ile yapılacak olan tüm işlemleri, tek bir kanal aracılığı ile gerçekleştirme ve hafızada tutma işlemlerini sunmaktadır. Bu sayede işlemlerin toplu halde gerçekleştirilmesi ve hata durumunda geri alınabilmesi sağlamaktadır.

Aşağıdaki şemayı (Şekil 6.4) incelediğimizde göreceğimiz üzere ilk başta veritabanına direkt erişimi görüyoruz klasik programlamada. İkinci kısımda ise bir tarafta Entity Framework için hazırlanmış olan Unit of work deseni ile ilgili repository’ler aracılığı ile veritabanına erişim var. Bu sayede unit testler için olan mocklanmış sınıflar aracılığı ile test bir veritabanına veya in-memory hazırlanmış veri alanına erişimini görüyoruz.

Mock Object: Unit testler yaparken uygulamamızın tümünü ayakta tutmak/tutabilmek zor olacaktır bir hayli. Bunun yerine uygulamamızın test ortamında çalışabileceği kadar fonksiyonları içeren sınıflar hazırlanır. Bu tür sınıflara Mock Object denir.



Şekil 6.4 Unit of work yapısı

Projede kullanılan unit of work pattern yapısı aşağıdaki kod bloğunda gösterilmektedir (Şekil 6.5).

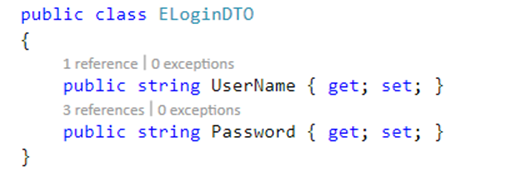


Şekil 6.5 Unit of work pattern kod bloğu

* 1. **DTO Katmanı**

Bu katman ile kullanılması gereken sınıflar için ana sınıfın çağrılması yerine, farklı sınıfların veya tek bir sınıfın gerekli olan değişkenleri alınarak oluşturulmuş olan yardımcı sınıfların bulunduğu katmandır.

Örneğin; aşağıda verilen şekildeki (Şekil 6.6) kod ile yazarın sisteme giriş yaparken sadece kullanıcı adı ve şifresi gereklidir. Yazar için oluşturulmuş olan sınıftaki diğer bilgilere ihtiyaç duyulmamaktadır.



Şekil 6.6 ELoginDTO sınıfı

* + 1. **Inversion of control katmanı (IoC)**

Interface yapısı ile nesne yaratılması işleminin bizden alınıp, bir yapıya dönüşme halidir. Nesne bağımlılıkları azaltılmış olup, nesnenin nereden geldiği umursamadan sadece kullanılır. IoC sayesinde birim test yazmak daha kolay hale gelir.

Projede kullanılan IoC pattern yapısı aşağıdaki kod bloğunda gösterilmektedir (Şekil 6.7).

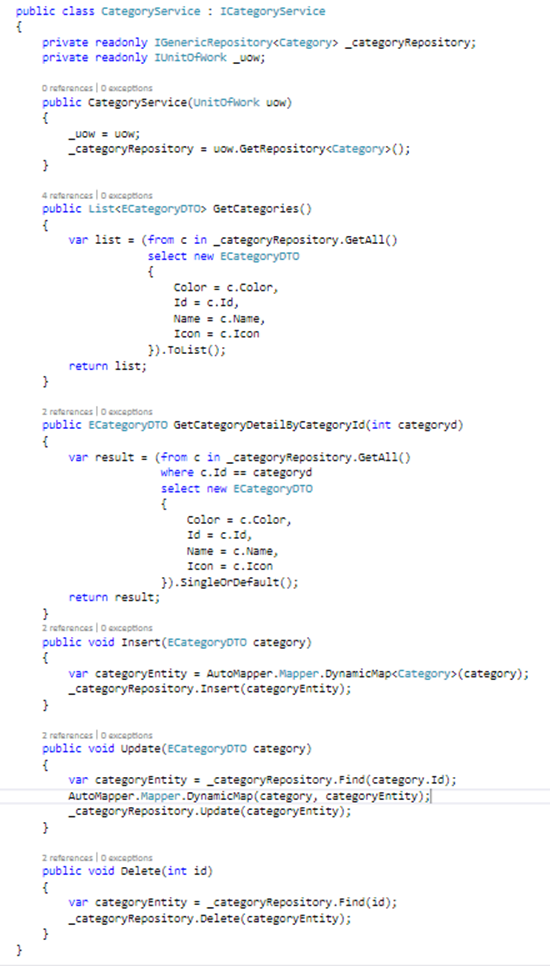


Şekil 6.7 IoC pattern kod bloğu

* 1. **Service Katmanı**

Data katmanındaki repository’lerin bir ya da bir kaçını kullanan bussiness logic katmanımız olan servis katmanında ProductService ya da CategoryService gibi sınıflar tanımlanır. Bu servisler genellikle kayıt ekleme, güncelleme, arama ve silme işlemleri yapmak için repository’leri kullanır ve değişiklikleri veritabanına aynı transaction ve context ile uygulayabilmek için unit of work’tan yararlanır.

Projede kullanılan servis katmanı yapısı aşağıdaki kod bloğunda gösterilmektedir (Şekil 6.8).



Şekil 6.8 Servis katmanı kod bloğu

* 1. **Presentation Katmanı**

MVC projemizin olduğu katman bu katmandır. Yukarda anlattığımız tüm MVC yapıları burada implement edilir. Kullanıcının görüntülediği, yapmış olduğu her işlem bu katmanda başlatılır, diğer katmanlar bu katman için hizmet eder.

1. **SONUÇ VE ÖNERİLER**

E-Science, yapılan bilimsel çalışmaları toplumun yararına elektronik ortamda yayılmasını sağlar. Böylelikle bilimsel çalışmalar, bilgiler internette saklanıp bir sonraki çalışmalar için başvurulacak kaynak olacaktır. Aynı zamanda toplum da dünyadaki bilimsel gelişmeleri yakından takip edebilme imkanı elde eder.

Farklı türden yazarlar ile çok fazla kategoride yayınlar yayınlamış, kategori çeşitliliği sonucu hitap edilen kullanıcı sayısı artırılmıştır. Kullanıcıların benzer özellikte yayınlanan makale önerileri ile ilgisinin çekilmesi amaçlanmaktadır. Makalelerin kategori bazlı eklenmesi sonucu, okuyucuların daha rahat uygulamayı kullanması ve ilgisini çeken kategoriyi belirleyerek hızlı bir şekilde merak ettiği makalelere ulaşabilmesi sağlanmıştır.

Makale yazarlarının, sisteme kayıtları sonrası kendi yayınladığı makaleleri filtreleme özelliği makalelere hızlı ulaşabilmesine imkan sağlamaktadır. Yazarın eklemek istediği makale yayınlanmadan önce bir hakem kontrolü sayesinde uygunluğunun tespit edilmesi ve sonrasında yayınlanması gerekli görülmektedir.

Günümüzde okuryazarlığın yazılı kaynaklardan elektronik ortamlara geçişi sonucu uygulamamızın tercih edileceğini göstermektedir. Aynı zamanda günümüz toplumunda internet kullanımın ciddi oranlarda yüksek olması sonucu gerçekleştirmiş olduğumuz çalışmanın geleceğe yönelik bir çalışma olduğu net bir şekilde görülmektedir. Uygulamamız genişletilerek ücretli veya ücretsiz makale yayınlayan uluslararası siteler gibi düzenlenebilmeye elverişli bir algoritma ile yazılmıştır.

**KAYNAKLAR DİZİNİ**

**Murray, C**. ., 2008, ‘Schools and Social Networking: Fear or Education?’, Synergy Perspectives: Local, Vol. 6 Issue 1, pp. 8-12.

**Manas, O**., 2001, Kurumsal portallar http://bbytezarsivi.hacettepe.edu.tr/jspui/bitstream/2062/50/2/692.pdf (Erişim: 12 Ocak 2019)

**Yıldırım, C.,** 1991, Bilim Felsefesi, Remzi Kitabevi, İstanbul

**Akoğlu, A**., 2005, Popüler Bilim Yayıncılığı ve Gökyüzü Gözlem Etkinlikleri. Journal of Istanbul Kültür University, (2), 1-4.

**Kılıç, E. ve Güngör, Z.,** 2006, Web site tasarımlarında kullanılabilirlik değerlendirme yöntemlerinin önemi. Akademik Bilişim Konferansı.

**Uçak, N. Ö. ve Çakmak, T.,** 2009, Web sayfası kullanılabilirliğinin ölçülmesi: Hacettepe Üniversitesi Bilgi ve Belge Yönetimi Bölümü web sayfası örneği. Türk kütüphaneciliği, 23(2), 278-298.

**TÜBİTAK,** 1996, Türk Tıp Dizini (1993 yılına ait konu ve yazar dizini kitapçıkları), ISSN-1300- 7602, Türkiye Bilimler Akademisi Broşürü, Eylül, 1997.

**TÜBA,** 1997, 1997 Yılı Burs ve Destek Programları, Türkiye Bilimler Akademisi Broşürü, Eylül, 1997

**Science Direct**, https://www.sciencedirect.com/ (Erişim: 12 Haziran 2019)

**Springer Link**, https://link.springer.com/ (Erişim: 12 Haziran 2019)

**Web Of Science**, https://login.webofknowledge.com/ (Erişim: 12 Haziran 2019)

**Scopus**, https://www.scopus.com/ (Erişim: 12 Haziran 2019)

**Research Gate**, https://www.researchgate.net/ (Erişim: 12 Haziran 2019)

**Google Scholar**, https://scholar.google.com.tr/ (Erişim: 12 Haziran 2019)

**Mark Richards**, 2015, Software Architecture Patterns , O’Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472., 978-1-491-92424-2, February 2015.