KİTLE KAYNAK TEMELLİ COĞRAFİ BİLGİ SİSTEMLERİ PLATFORMUNUN GELİŞTİRİLMESİ

Yüksek Lisans Tezi

Barış İNALTAY

Eskişehir 2019

KİTLE KAYNAK TEMELLİ COĞRAFİ BİLGİ SİSTEMLERİ PLATFORMUNUN GELİŞTİRİLMESİ

Barış İNALTAY

YÜKSEK LİSANS TEZİ

Uzaktan Algılama ve Coğrafi Bilgi Sistemleri Anabilim Dalı
Danışman: Prof. Dr. Alper ÇABUK
İkinci Danışman: Dr. Öğr. Üyesi Hakan UYGUÇGİL

Eskişehir
Eskişehir Teknik Üniversitesi
Lisansüstü Eğitim Enstitüsü
Mayıs 2019

JÜRİ VE ENSTİTÜ ONAYI

Barış İNALTAY'ın "Coğrafi Bilgi Sistemleri Kullanımının Yaygınlaştırılması Maksadıyla Kitle Kaynak Cbs Sözlük Platformu Oluşturulması" başlıklı tezi 30/05/2019 tarihinde aşağıdaki jüri tarafından değerlendirilerek "Eskişehir Teknik Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliği"nin ilgili maddeleri uyarınca, Uzaktan Algılama ve Coğrafi Bilgi Sistemleri Anabilim dalında Yüksek Lisans tezi olarak kabul edilmiştir.

Jüri Üyeleri	**Unvanı Adı Soyadı**	**İmza**
Üye (Tez Danışmanı)	: Prof. Dr. Alper ÇABUK	.....
Üye (İkinci Danışman)	: Dr. Öğr. Üyesi Hakan UYGUÇGİL	.....
Üye	: Doç. Dr. H. Rafet YÜNCÜ	.....
Üye	: Dr. Öğr. Üyesi H. Cem SAYIN	.....
Üye	: Dr. Öğr. Üyesi Muammer TÜN	.....

Lisansüstü Eğitim Enstitüsü Müdürü

ÖZET

KİTLE KAYNAK TEMELLİ COĞRAFİ BİLGİ SİSTEMLERİ PLATFORMUNUN GELİŞTİRİLMESİ

Barış İNALTAY

Uzaktan Algılama ve Coğrafi Bilgi Sistemleri Anabilim Dalı
Eskişehir Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü, Mayıs, 2019

Danışman: Prof. Dr. Alper ÇABUK
İkinci Danışman: Dr. Öğr. Üyesi Hakan UYGUÇGİL

Bu çalışmada Coğrafi Bilgi Sistemlerinin Türkiye'deki tanınırlığını artıracak ve Coğrafi Bilgi Sistemleri ile ilgilenenlerin ortak bir dil kullanımını sağlayacak kitle kaynaklı sözlük uygulaması yapılması amaçlanmıştır. Çalışmada dünya çapında yapılmış kitle kaynaklı uygulama ve platformlar değerlendirilerek izledikleri yollar araştırılmıştır. Kitle kaynak projelerin; yönetimi, pazarlama metotları ve kitleyi bağlama yöntemleri değerlendirilmiştir. Yapılan yazılım için gereken dil, .Net Core yapısında C#, veritabanı olarak da Postgresql seçilmiştir. Yapılan araştırmalar sonucunda uygulamanın ana hatları ve yazılım yöntemi belirlenmiştir. Belirlenen yöntemler sonucu yazılımın adımları açık bir biçimde ortaya konmuştur.

Anahtar Sözcükler: Coğrafi Bilgi Sistemleri, Kitle kaynak, Kitle fonlaması, Yazılım, Veritabanı.

ABSTRACT

DEVELOPMENT OF CROWDSOURCING BASED GEOGRAPHICAL INFORMATION SYSTEMS PLATFORM

Barış İNALTAY

Department of Remote Sensing and Geographical Information Systems
Eskişehir Technical University, Graduate School, January, 2019

Supervisor: Prof. Dr. Alper ÇABUK
Second Supervisor: Assoc. Prof. Hakan UYGUÇGİL

This study will raise awareness of Geographic Information Systems in Turkey and a common language that will increase the use of Geographic Information Systems is aimed at determining the origin of crowdsourcing Dictionary application. In the study, crowdsourcing based applications and platforms worldwide have been evaluated and methods they followed were investigated. Crowdsourcing projects; management, marketing methods and methods of linking the audience were evaluated. The language required for the software is C # in the .NetCore structure and Postgresql in the database. As a result of the researches, the outline of the application and software method were determined. As a result of the determined methods, the sections of the software are clearly revealed.

Keywords: Geographical Information Systems, Crowdsourcing, Crowd Funding, Software, Database.

TEŞEKKÜR

Bu çalışmanın gerçekleşmesi için bana her türlü kolaylığı sağlayan ve desteğini esirgemeyen tez danışmanım Prof. Dr. Alper ÇABUK'a, çalışma süresince tüm bilgi ve vaktini paylaşan Dr. Öğr. Üyesi Hakan UYGUÇGİL'e ve süreç boyunca bütün sorularımı cevaplayarak beni yalnız bırakmayan Serhan TUNCER ve Arş. Gör. M. Emre BEKTÖRE'ye tüm bu süreç boyunca verdikleri katkı ve destekleri nedeniyle teşekkürü borç bilirim.

Bu çalışma süresince beni yüreklendiren ve maddi ve manevi yönden sürekli yanımda olan kuzenlerim Serdar Utku KARTAL ve Dr. Öğr. Üyesi Mine SÖNMEZ KARTAL'a teşekkür ederim.

Maddi ve manevi desteklerini benden esirgemeyen ve eğitim hayatım boyunca hep yanımda olarak beni yönlendiren ailem; annem Sevim ve babam Bülent İNALTAY'a en içten dileklerimle teşekkür ederim.

Barış İNALTAY
Mayıs 2019

30/05/2019

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Bu tezin bana ait, özgün bir çalışma olduğunu; çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı; bu çalışma kapsamında elde edilen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi; bu çalışmanın Eskişehir Teknik Üniversitesi tarafından kullanılan "bilimsel intihal tespit programı"yla tarandığını ve hiçbir şekilde "intihal içermediğini" beyan ederim. Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçları kabul ettiğimi bildiririm.

Barış İNALTAY

İÇİNDEKİLER

Sayfa

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

CBS : Coğrafi Bilgi Sistemleri

SYMAP : Synagraphic Mapping System

ESA : European Space Agency

BBC : British Broadcasting Corporation

ESRI : Environmental System Research Institute

SETI : Search for Extraterrestrial Intelligence

BOINC : Berkeley Open Infrastructure for Network Computing

RNA : Ribo Nükleik Asit

MVC : Model – View - Controller

SQL : Structured Query Language

1. GİRİŞ

1.1. Çalışmanın Amacı ve Önemi

Coğrafi Bilgi Sistemleri yapılan pek çok yatırım ve proje için öngörü oluşturabilmek adına kullanılan bir bilim dalıdır. Bu bilimin gerekli olduğu tüm dünyada kabul görmekte ve uygulanmaktadır. Fakat Türkiye için geçerli bir olgu değildir. Yapılacak projelerin çoğu günümüzde halen kulaktan dolma kurallar çerçevesinde sonucun gideceği yer bilinmeden yapılmaktadır. CBS bir projenin adımlarını, yerini, maliyetini ve sonuçlarını açıkça ortaya koyabilmektedir. CBS'yi en çok kullanması gereken yerel yönetimler dahi yaptıkları projelerde CBS kaynaklarından yararlanmamaktadır. Günümüzde bu yönde birçok adım atılmakta ve CBS Türkiye içerisinde daha çok bilinir ve kullanılır bir hale gelmektedir.

Bu çalışmada CBS'yi Türkiye'de bilinir bir hale getirebilmek için gereken alt yapı araştırılacak ve uygulanacaktır. Günümüzde teknolojinin gelişmesiyle birlikte kitle-kaynaklı uygulamalar insanlar tarafından çok kullanılır bir hale gelmiştir. Bu nedenle CBS kullanımını artırmak amacıyla kitle-kaynaklı bir sözlük platformu oluşturulması, var olan kullanıcıların da bu amaca katkıda bulunmaları sağlanmaya çalışılacaktır.

1.2. Literatür Özeti

Çalışmanın genel başlığı olarak kitle-kaynak ele alınmıştır. Kitle-kaynak başlığında değerlendirilen çalışmalar aşağıda sıralanmıştır:

Diana Lynn McLean ve Jeffrey Heer (2013) "Identifying medical terms in patient-authored text: a crowdsourcing-based approach" başlıklı makalesinde, iki ayrı çalışma ortaya koymuştur. İlk çalışma olarak sağlık sektöründen olmayan bireylerin tıbbi terimleri tanıması, ikinci olarak hâlihazırda kullanılan tıbbi terim tanımlayıcılarının kitle-kaynaklı veri setleriyle karşılaştırılarak performans değerlendirmesi yapmak amaçlanmıştır. Makalenin ilk çalışma konusu üzerindeki çalışmalar doğrultusunda bir kitle-kaynak uygulaması olan Amazon Mechanical Turk (https://www.mturk.com) kullanılarak, kullanıcılara yeni bir görev oluşturulmuştur. Kullanıcılardan belirlenen kelimeler içerisinden tıp ile alakalı olanı seçmeleri istenmiştir. Amazon Mechanical Turk için kullanılan örnekler ile aynı uygulama hemşireler üzerinde de uygulanmıştır. İlk hipotezin sonucu olarak Amazon Mechanical Turk kullanıcıları ile hemşireler arasında MedHelp (https://www.medhelp.org) altın standardına göre güvenilirlik puanları binde iki gibi bir fark oluşturmuştur ki, Amazon Mechanical Turk kullanıcıları bu yaklaşıma

göre kabul edilebilir sonuçlar elde etmiştir. Makalenin bir diğer çalışma konusu için yapılan araştırmada Kitle-kaynak ve uzmanlardan alınan veri setleri kullanılarak çapraz doğrulama yapılmıştır. Bu doğrulama sonucu ile daha önceden oluşturulmuş modeller olan MetaMap, OBA ve TerMINE modelleri CureTogether (https://curetogether.com) altın standardına göre karşılaştırılmıştır. Sonuç olarak diğer modellerden daha yüksek performans göstermiştir. Makalede değerlendirilen iki hipotez sonucunda kitle-kaynak ile elde edilen verinin daha ucuz ve güvenilir olduğu sonucuna ulaşılmıştır.

Aynur Abdulnasyrov ve arkadaşlarının (2015) "Crowd-sourced automated vocabulary learning system." başlıklı patentinde, kullanıcıların yazdıkları sözcük veya sözcük öbeklerini istedikleri dile anında çevirecek bir yazılım tasarlamışlardır. Bu yazılımı diğer çeviri yazılımlarından ayıran fark kullanıcı tabanlı bir sözlük oluşturarak hizmet vermesinin amaçlanmıştır. Yazılımın, kullandığı istatistiksel yöntemler ile kullanıcı girdilerini değerlendirerek aynı kullanıcının sonraki çevirilerinde ilgilendiği konular ve kullandığı üsluba göre öneriler sunması sağlanmıştır.

Tim Causer and Melissa Terras (2014) "Crowdsourcing Bentham: beyond the traditional boundaries of academic history" başlıklı makalesinde kitle kaynak olarak başlatılan ve Jeremy Bentham'ın makalelerinin dijital kopyalarını çıkartmayı amaçlayan Bentham Makalelerini Kopyalana Girişimi (The Bentham Papers Transcription Initiative) projesinin beşeri bilimlere yaptığı katkıyı ve bu makaleler üzerinden yeni keşifler yapılabilme potansiyelini incelemişlerdir. Bentham Makalelerini Kopyalana Girişimi, oluşturulan internet ortamında üyelik sistemi ile dünyanın her yerinden ulaşılabilen bir uygulama ile gönüllü üyelerin el yazmalarını dijital ortama aktardığı bir proje olarak sunulmuştur. Proje sonrasında yapılan çalışmalar, makalenin geniş bir felsefi ve tarihsel konuya katkıda bulunacağı ortaya konmuştur. Makale sonucunda kitle-kaynak yapının geliştirilmesi, hatta gönüllülere burs verilerek cesaretlendirilmesi gerektiği öne sürülmektedir.

Mark Cartwright and Bryan Pardo (2013) "SocialEQ: Crowdsourcing an Equalization Descriptor Map" başlıklı bildirilerinde, yaptıkları çalışma için kitle-kaynak uygulaması geliştirmiş ve bu uygulama sonuçlarını veri seti olarak kullanmışlardır. Araştırmanın temel konusu, müzik üretimi ve müzik dinleme ara yüzlerindeki karmaşayı ortadan kaldırıp işleri basite indirgemeye çalışmaktır. Bir müzik programındaki veya müzik aletindeki dengeleyici (equalizer) karmaşık bir yapıya sahip iken bu yapıyı herkesin anlayabileceği bir düzeye getirmek amaçlanmıştır. Araştırma sonucunda

insanlardan kitle kaynak yöntemi ile elde edilen veriler ışığında sesler ile alakalı bazı kalıp kelimeler ve eş anlamlıları belirlenmiştir. Araştırmanın temel amacındaki akıllı ses üretim sisteminin temelindeki adımlar atılmıştır.

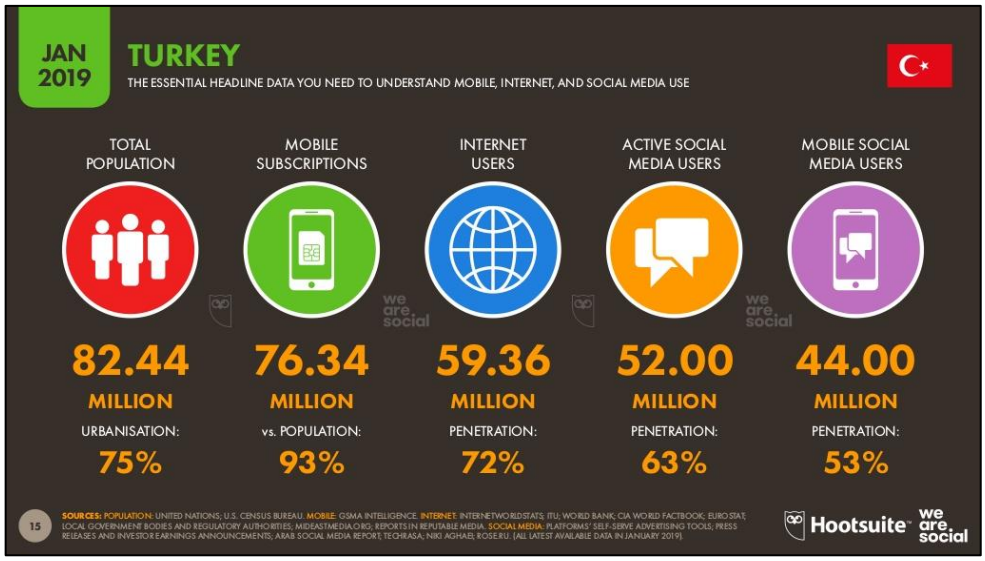
Tün vd. (2018) "Depremlerde gözlenen etkilerin gönüllü katılımıyla hızlı bir şekilde toplanması" başlıklı makalelerinde bir doğal afet yönetim sürecini, kitle-kaynaklı bir uygulama üzerindeki yönetim süreçlerini belirlemek üzerine bir alt yapı hazırlamışlardır. Doğal bir afet meydana geldiğinde afetzedelerin büyük bir çoğunluğunun yaşayacağı iletişim sorunlarını ve acil müdahalenin gecikmesi durumunu durumlarını minimize edebilmek amacıyla alt yapı ve mobil uygulama geliştirilmiştir. Uygulamanın acil müdahale ekiplerince izlenebilen bir de yönetim paneli mevcuttur. Bu panel üzerinde öncelikli müdahale yerinin koordinatları belirlenerek bu bölgelere yapılacak müdahalelerle can ve mal kaybının azaltılacağı öne sürülmüştür. Sadece afet yönetimi üzerine değil afet sonrası hasar kaydını da bu uygulama üzerinden kitle-kaynak olarak kullanıcılar tarafından belirlenebileceği ön görülmüştür.

Berk Anbaroğlu (2017) "Gönüllü coğrafi bilgi: mekânsal bilişim çalışmalarına web 2.0 devrinde yeni bir yaklaşım" başlıklı makalesinde bir kitle-kaynak kavramı olarak ortaya çıkan gönüllü coğrafi bilgi olarak Türkçeye çevrilen volunteered geographic information'ı değerlendirmektedir. Gönüllü coğrafi bilgi kavramından yola çıkılarak küçük büyük birçok proje oluşturulmaktadır. Gönüllü coğrafi bilgi internet ortamındaki konumsal veri toplama ve bu verilerin analizi gerekliliğinden ortaya çıkmıştır. Gönüllü coğrafi bilginin kullanıldığı ve dünyaca bilinen en büyük proje Open Street Map'tir. Birçok gönüllü tarafından Dünya'nın haritalandırılması üzerine bir kitle-kaynak çalışmasıdır. Bu çalışmada gönüllü coğrafi bilgi ile oluşturulan projelerin güvenilirliği ve katılımcıların motivasyonu araştırılmıştır. Çalışma sonucu projelerdeki veri güvenilirliği ve doğruluğunu yüzde yüz sağlayacak bir yöntemin halen geliştirilememiş olduğu ortaya çıkmaktadır. Yapılan projelerde oluşan veri setlerini, referans veri setleri ile karşılaştırarak doğrulama yapılabilmektedir. Katılımcı motivasyonları üzerindeki araştırmada ise katılımcıların gönüllü coğrafi bilgiye yaptıkları katkı ile söz sahibi ve tanınır olmak istekleri gibi düşüncelerin ön plana çıktığı belirlenmiştir.

2. KURAMSAL TEMELLER

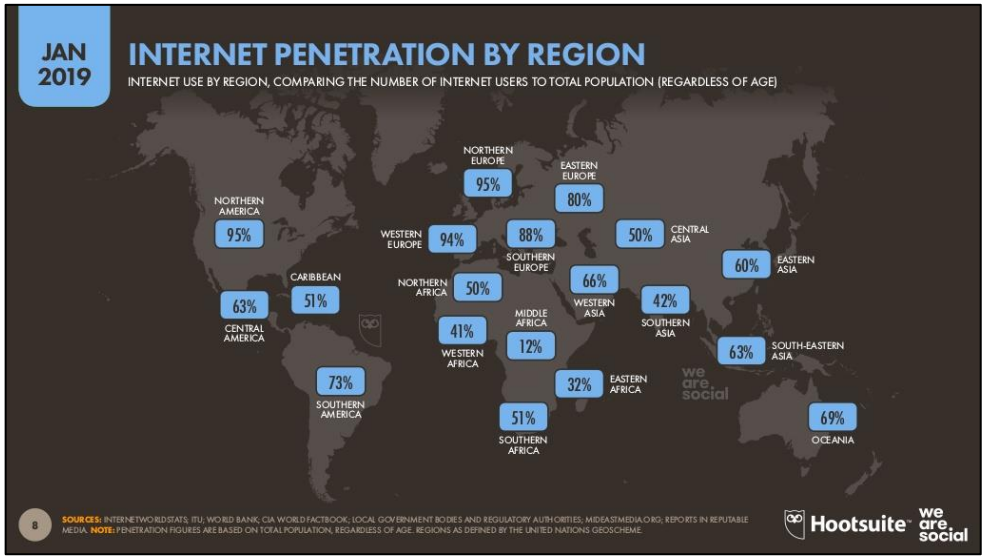
Günümüzde teknolojinin gelişmesi ile birlikte artık ceplerimize kadar gelen internet teknolojisi sayesinde internet, yaşayan bir ortama dönüşmüştür. İnternetin yaşayan bir organizma gibi sürekli beslenmesi, internet kullanıcıları tarafından sağlanmaktadır. Kullanıcılar gönüllü veya ücretli olarak içerik üretip paylaşmakta ve internete katkı sağlamaktadır. Kullanıcıların sağladığı bu içerikler daha büyük teknolojilerin ve bilimin gelişmesinde dahi katkıda bulunabilmektedirler. Dünya üzerindeki internet ve sosyal medya kullanım oranları We Are Social'ın 2019 yılında yayınladığı raporu 4.388 milyar internet kullanıcısı varken 3.484 milyar sosyal medya kullanıcısı olduğu yönündedir. Aynı raporun ülkemiz için verdiği istatistikler ise 59.36 milyon internet kullanıcısı ve 52 milyon sosyal medya kullanıcısı olduğu yönündedir (Şekil 2.1, Şekil 2.2). İnternet ve sosyal medya kullanımının bu kadar yaygın bir halde olması, internet üzerindeki projelerin büyük bir çoğunluğunun kullanıcılar tarafından yürütülebilir olmasını sağlamaktadır. Günümüzde artık internet sitelerinin yapıları, içeriklerinin tamamı kullanıcılar tarafından sağlanacak şekilde yapılar oluşturulmaya başlanmıştır. Bu yapılara ise kitle-kaynak yapılar adı verilmektedir.

Şekil 2.1 *Dünya'da internet ve sosyal medya kullanımı (http-1)*



Şekil 2.2 *Türkiye'de internet ve sosyal medya kullanımı (http-1)*

Dünya üzerinde gerçekleşen birçok olay kayıt altına alınırken konum verisi ile birlikte kaydedilmektedir. Artık günümüzde internet üzerinden yapılan bütün işlemler içerisinde konum verisi yer almaktadır. Konum verileri ve bu verilerin analizi ile elde edilen haritalar yine analiz veya bilgilendirme amacıyla kullanılmaktadır. Haritalama ve analiz içeren bu sisteme Coğrafi Bilgi Sistemleri (CBS) adı verilmektedir. Şekil 2.1 ile gösterilen Dünyada internet kullanımı, CBS kullanılarak dünya haritasına yayıldığında Şekil 2.3 gibi gösterilebilir.



Şekil 2.3 *Dünyada kıtalara göre internet kullanımı (http-1)*

Bu çalışma CBS ve kitle-kaynak alanlarını birleştirmeye çalışmaktadır. Bu çalışmada kullanılan iki ana başlık olarak kitle-kaynak ve CBS bir arada kullanılarak en etkin modeller belirlenmeye çalışılacaktır.

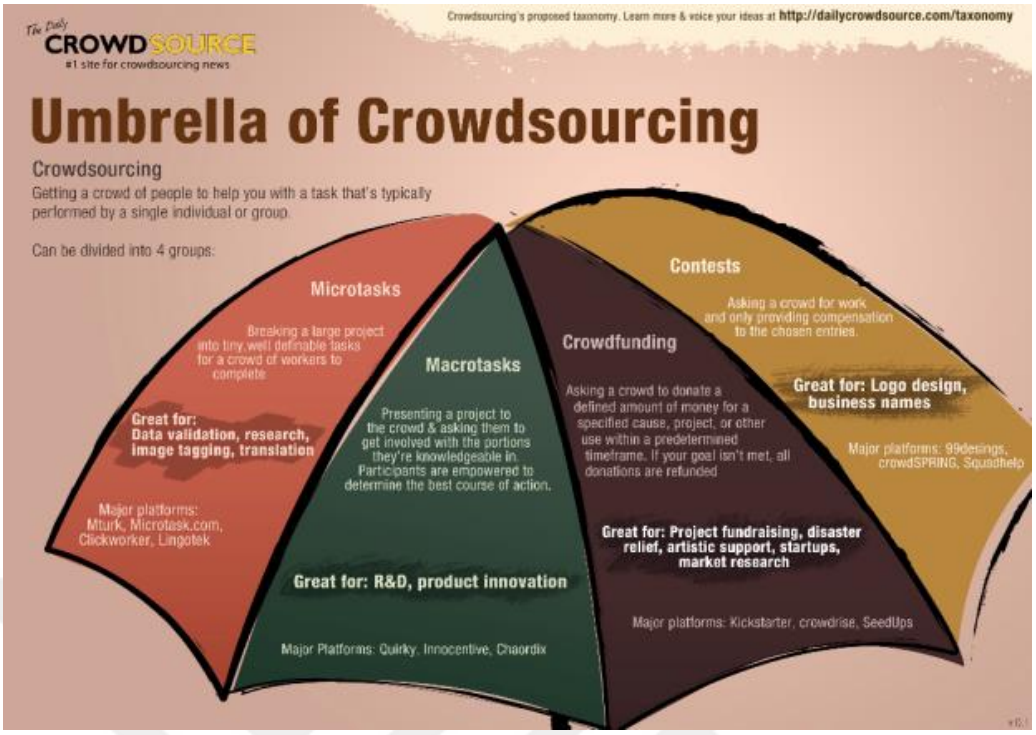
2.1. Kitle-Kaynak

Kitle-kaynak, terim olarak Jeff Howe tarafından, dış kaynak (outsource) ve kalabalık (crowd) kelimelerinin birleşmesiyle türetilmiştir (Howe, 2006). Yapılmak istenen bir işin, dış kaynak kullanarak daha fazla kişi tarafından yapılmasını sağlamak anlamında kullanılmaktadır. Kitle-kaynak birçok kişi tarafından farklı şekillerde tanımlanmıştır.

- Bir projedeki işlerin dış kaynak olarak bir grup insandan yararlanılarak yaptırılmasındaki dış kaynaktır (Howe, 2006).
- Projedeki işlerin, dış kaynak grupların bilgi ve yeteneklerine göre eşleştirildiği bir mekanizmadır (Howe, 2008).
- Bazı kâr amaçlı firmaların çevrimiçi olarak kullanılan uygulamalarının problem çözme ve üretim modelidir. (Brabham, 2008)
- Özel bir sorunun ortak aklıdır (Buecheler vd., 2010).
- İnterneti, dağınık işçi çalıştırmak için kullanma yoludur (Grier, 2011).

Kitle-kaynak tanımı gün geçtikçe çeşitlenmiş ve farklılaşmıştır. Temelindeki amaç, görev ve kalabalık grup kavramlarının değişmediği görülmüştür.

Kitle-kaynak kavramı 2006 yılında ilk kez adlandırılmış olsa da kullanımı daha eskiye dayanmaktadır. İnsanlar bazı projeleri geliştirebilmek için yarışmalar düzenlemişlerdir. Projeler ve yöntemler geliştikçe internet kavramının da yardımıyla birlikte kitle-kaynak kavramı da gelişmiştir. Kitle-kaynak, uygulanacak projeye göre dört farklı başlık altında toplanabilir (Şekil 2.4). Bu başlıklar; küçük görevler (microtasks), büyük görevler (macrotasks), kitle fonlama (crowdfunding) ve yarışma (contest) olarak belirlenmiştir (Rose 2011).



Şekil 2.4 *Kitle kaynak şemsiyesi (Rose, 2011)*

Küçük görevler, büyük bir projeyi küçük, iyi tanımlanmış parçalara ayırarak, kitle-kaynaklı işçi gruplarına sunarak yapılmasını sağlamaktır. Projenin küçük parçaları tamamlandıkça bütünü ortaya çıkacak ve proje kolaylıkla tamamlanmış olacaktır. Her proje, bu tarz kitle-kaynak projesi değildir. Küçük görevler tarzındaki projeler genellikle veri doğrulama, araştırma, resim etiketleme ve çeviri gibi projelerde daha uygun olarak kullanılmaktadır.

Büyük görevler, büyük bir projede kitle-kaynağı olan çalışanlardan, projenin bilgi sahibi oldukları kısımlarını tamamlamalarını istemektir. Proje içerisinde katılımcıların en iyi ve başarılı oldukları işlerde yetkilendirilmeleri, projenin doğru bir şekilde ilerlemesini de sağlamaktadır. Şirketler için büyük projelerini kitle kaynak bir platforma taşımak bünyelerinde bu kadar çalışan barındırma gerekliliklerini de ortadan kaldırmaktadır. Böylece hem proje yöneticilerini hem de çalışanlarını kalabalık gruplardan oluşturabilmektedirler. Bu tarz projeler genellikle Ar-Ge ve ürün yenileme tarzında projelerde daha yaygın olarak kullanılmaktadır.

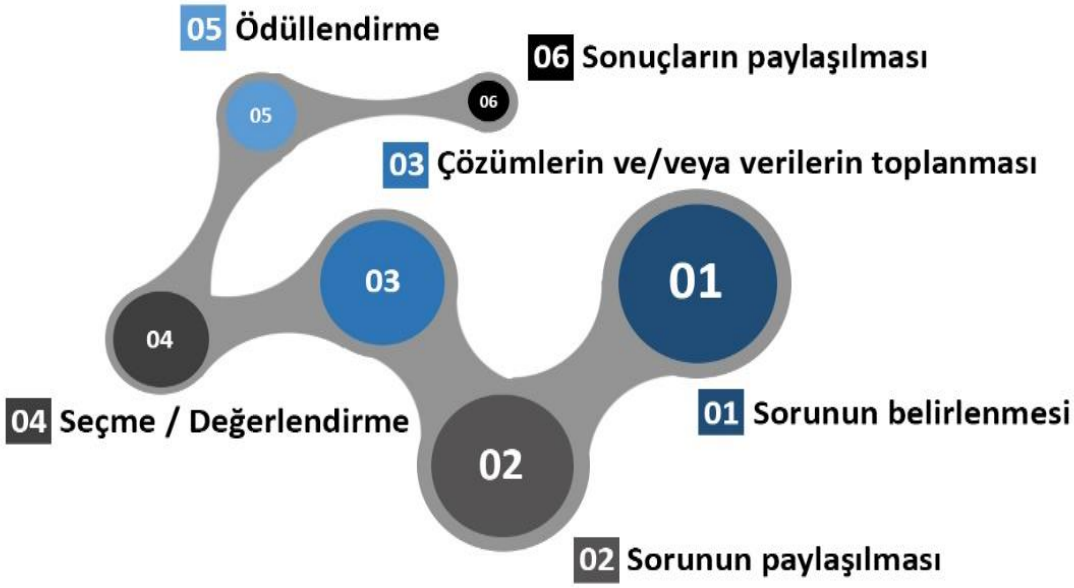
Kitle fonlaması yapılmak istenen bir proje, başlangıç olarak bir sermayeye ihtiyaç duymakta ve bu sermayeyi bir yatırımcıdan sağlayamaması durumunda gerekli sermayenin kitle kaynak olarak bir kalabalıktan karşılanmasıdır. Eğer kitle kaynaklı

toplanan sermaye, projenin başlaması için yeterli değilse yapılan yatırımlar geri dağıtılarak kitle kaynaklı yatırımcılar korunur. Sermaye yeterli ise projeye başlanır ve proje sonunda sermaye sağlayan kitle-kaynak yatırımcıya bir ödül verilir. Proje bir yardım etkinliği de olabilir. Kitle fonlamasıyla ihtiyaç sahiplerine bir yardım şeklinde de düzenlenebilir. İhtiyaç sahiplerine yapılacak yardım bir kişi veya kurumdan sağlanmaktansa kitle-kaynak olarak yardım edebilecek kalabalıktan sağlanabilir. Kitle fonlamasıyla yapılabilecek projeler, kaynak yaratma, afet yardımı, sanatsal destek, girişim projeleri ve pazar araştırması olarak sıralanabilir.

Yarışma olarak ele alınabilecek projeler, işi bir firmaya vermek yerine bir kitle kaynak projesi olarak duyurulan projelerdir. Kitle-kaynak tarafından yapılan bu projelerde, projeye katılım gösteren adayların yaptıkları işler arasından bir seçim yapılır ve kazanan projenin sahibine bir ödül verilir. Böylece projeden beklenen başarı bir firma veya bir kişiye bağlı değil proje sahibinin beklentilerine bağlı olarak belirlenmiş olur. Genellikle yarışmacı olarak ele alınan kitle-kaynak grubunun ürünleri, logo tasarımı, kurumsal kimlik tasarımı ve mimari tasarımlar gibi projelerdir.

Kitle kaynak projeleri diğer çok büyük şirket projeleri gibi farklı modeller kullanılarak geliştirilseler de bu projelerin genel adımları belirlidir. Kitle-kaynak projeleri 6 ana adımdan oluşmaktadır (Şekil 2.5). Diğer projelerdeki gibi proje yöneticilerinin geliştirmesi gereken adımlar önceden belirlenmiştir. Bu adımlar projenin başlangıcından bitişine kadar takip edilmesi gereken modeli belirlemektedir.

- Sorunun belirlenmesi
- Sorunun paylaşılması
- Çözümlerin ve/veya verilerin toplanması
- Seçme / Değerlendirme
- Ödüllendirme
- Sonuçların paylaşılması



Şekil 2.5 *Kitle-kaynak proje adımları*

Sunulacak projenin ya da sorunun kitle-kaynak ile bir çözüme ulaşabilir olması gerekmektedir. Öncelikle proje ya da sorun iyi belirlenmeli ve tanımlanmalıdır. Sorun iyi tanımlanmış olmazsa kitleye anlatmakta zorluk yaşanıp projenin başarısız olmasına yol açabilir. Kitle kaynak projelerinde öncelik kitleye sorulacak sorunun iyi tanımlanmış olmasıdır.

Proje ya da sorunun belirlenmesinden sonra kitleye duyurulması gerekmektedir. Proje kitleye duyurulurken duyuru kaynakları hedef kitleye göre belirlenmelidir. Hedef kitle, sorunun çözümüne yönelik bilgiye sahip olmalıdır ki başarıya ulaşılabilsin. Kitleye duyuru eski dönemlerde gazete ilanı ile yapılırken günümüzde sosyal medya reklamları aracılığıyla yapılmaktadır.

Çözüm sürecinde kitle-kaynak çalışanları için bir platform sağlanmalıdır. Günümüzde genellikle bu platformlar internet uygulamalarıdır. Oluşturulan uygulamalar üzerinde üye girişi ile girilebilen bir platform hazırlanmalıdır. Üyeler sorun çözümlerini bu platform üzerinden göndermelidirler. Daha önceden belirlenmiş ise üyeler birbirlerinin çözümlerini görerek daha da geliştirme imkânı sağlayabilirler. Böylece çözümler soruna daha iyi cevap verecektir.

Proje çözüm süreci bittikten sonra kitleden elde edilen çözümler arasından en iyi ve uygun olanı seçilir. Kitlede çözümü üreten kişi ya da kişilerin bilgi birikimi, çözümler için en iyi sonuç olabilir.

Sorun için seçilen en iyi çözümü bulan kişi ya da kişiler daha önceden belirlenen ödüllerin sahibi olarak yaptıkları çözümlemenin mükâfatını elde ederler. Kitleye bir şekilde verilen bu ödüller daha sonraki proje ya da sorunlarda kitlelerin daha motive bir şekilde çalışmalarını sağlamaktadır.

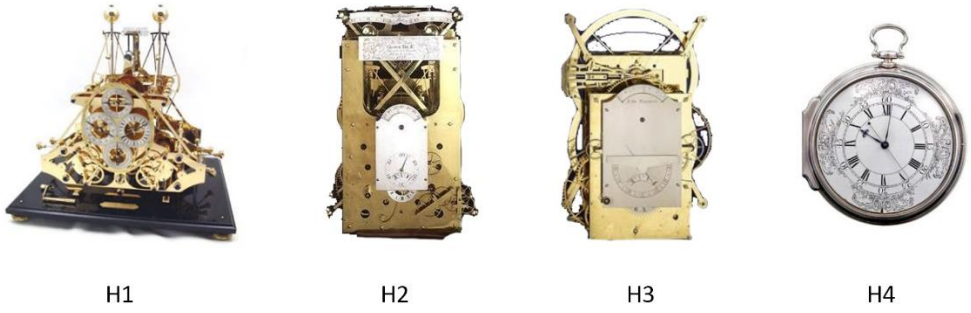
Proje tamamlandığında, kitleden elde edilen çözümün projeyi hangi konuma getirdiği değerlendirilmelidir. Elde edilen sonuç projenin geleceğini belirlemektedir (http-2).

2.1.1. Kitle-kaynağın tarihsel gelişimi

Kitle kaynak 1700'lü yıllardan beri kullanılan bir yapı olmuştur. Günümüzde yoğun bir şekilde kullanılıyor olsa da geçmişten gelen uygulama örnekleri mevcuttur. Proje geliştirme sürecinde kitlelerden yararlanmanın o dönemlerde henüz bir adı olmasa da yöntem olarak kullanılmıştır.

İngiliz denizciler, denizde yönlerini bulmak için gökyüzünden faydalanmaktadır. Bilinen hesaplama yöntemi kesin bir sonuç vermemektedir. Devamında da İngiliz Hükümeti, 1714 yılında Boylam Ödülü (Longitude Prize) adını verdikleri bir yarışma düzenlemiştir. Yarışma sonunda 20.000 poundluk bir ödül verilecektir. Düzenlenen yarışmada John Harrison (Şekil 2.6) isimli bir saatçi H1 adını verdiği bir saat üretir. Saatin yeterli olmadığını savunan kurul Harrison'ın maddi destek alarak saati geliştirmesine sağlar. Harrison saati sürekli geliştirerek H2, H3 ve H4 sürümlerine ulaştırır (Şekil 2.7) (http-3, http-4).

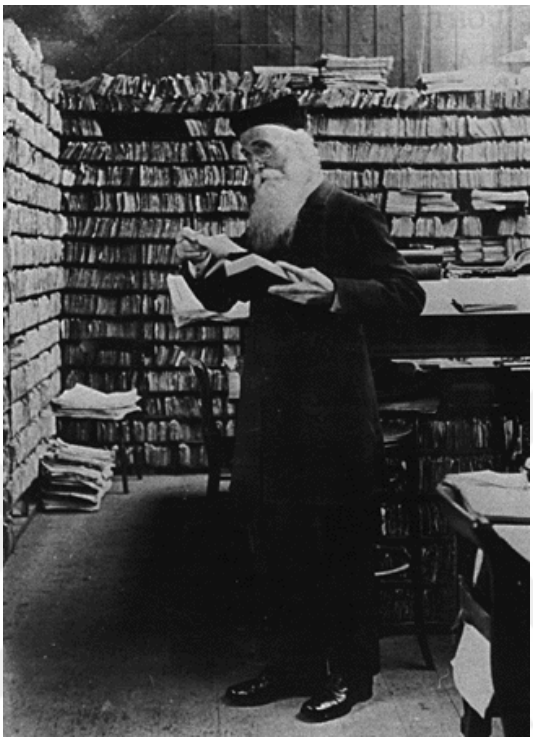
Şekil 2.6 *John Harrison(http-3)*



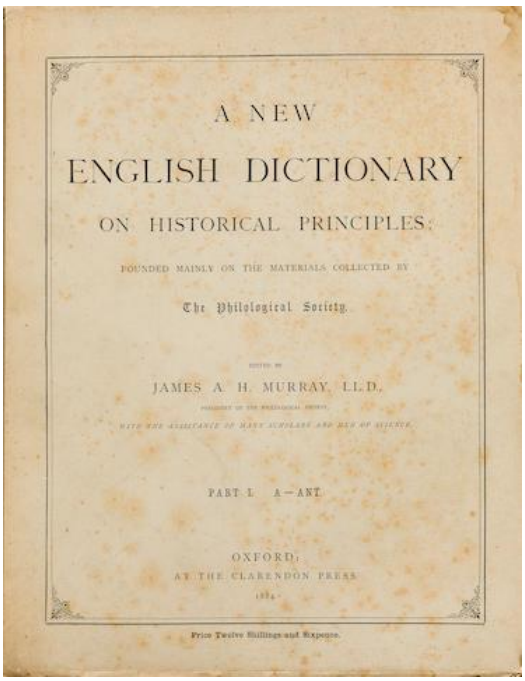
Şekil 2.7 *John Harrison tarafından geliştirilen saatler (http-3)*

İngilizler sözlüklerini düzenlemek ve geliştirmek için birçok girişimde bulunmuştur. Fakat bu girişimler 1879 yılına kadar hep başarısızlıkla sonuçlanmıştır. Sözlük projesinin Oxford University'ye devrinden sonra üniversite sözlük editörü olarak James Murray adlı bir filolog görevlendirmiştir (Şekil 2.8). James Murray yaptığı çok yenilikçi bir atılımla sözlükte yer alması gereken sözcükleri, sözcük kökenlerini ve sözcüklerin örneklendirilmesi ile ilgili kalıpları toplamak için kamuoyuna bir duyuru yayınlatmıştır. Yapılan bu duyuru için 2000 kişiden fazla insan 10.000'den fazla kayıt

göndermiştir. Bu şekilde oluşturulan ilk baskı 1884 yılında yayınlanmıştır (Şekil 2.9) (Liu, 2016, http-5).



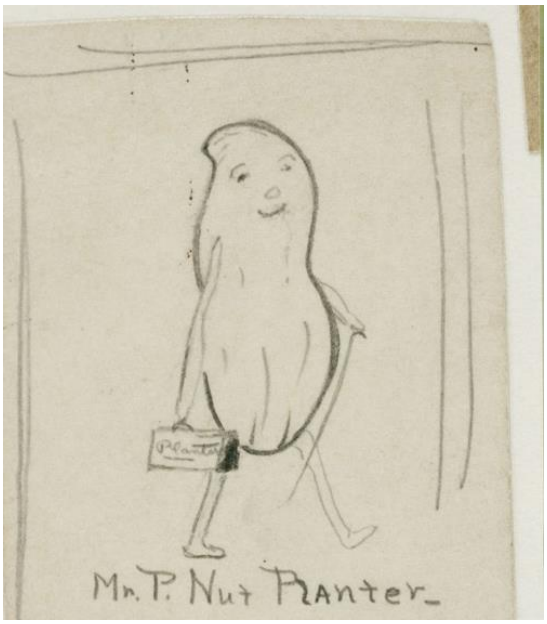
Şekil 2.8 *James Murray (http-5)*



Şekil 2.9 *James Murray'ın editörlüğünde yayınlanan sözlük*

Planters Nut and Chocolate Company, üretip pazarladığı yer fıstığı için bir logo arayışına girmiştir. Logo arayışı için kitle kaynaklı bir tasarım yarışması düzenleyeceğini duyurmuştur. 1916 yılında bir lise öğrencisi olan Antonio Gentile yaptığı eskizleri yarışma için göndermiştir (Şekil 2.10). Antonio Gentile'nin çizimi olan Mr. Peanut çizimi yarışmayı ve 5 dolarlık ödülü kazanmıştır (Şekil 2.11) (Kietzmann, 2017, http-6).

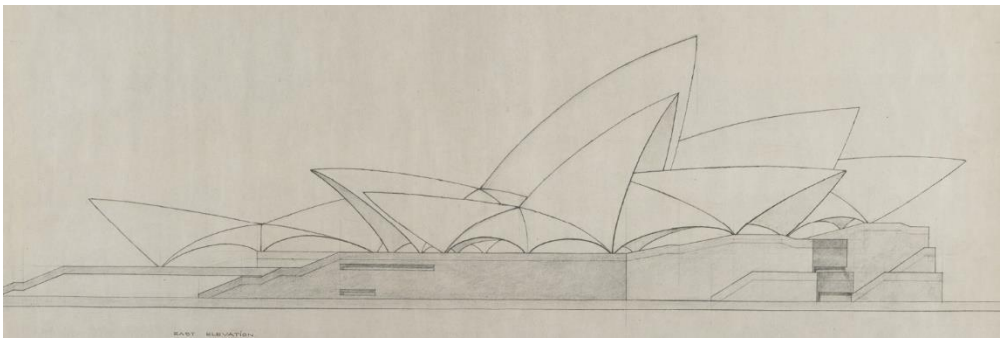
Şekil 2.10 *Antonio Gentile (http-6)*



Şekil 2.11 *Antonio Gentile'nin Mr. Peanut eskizi (http-6)*

İkinci dünya savaşı sonrası Avustralya bir toparlanma süreci geçirmiştir. Bu süreçte belirli bir aşama kaydedildikten sonra hükümet, tesislerin yetersiz olduğuna karar vermiştir. Verilen bu karar 1952'de tekrar gündeme gelene kadar uygulanmamıştır. 1956 yılına kadar konferanslar ve paneller ile tesislerin yetersiz olduğu düşüncesi desteklenmeye çalışılmıştır. Nihayet 1956 yılında Ulusal Opera Binası için uluslararası bir mimari yarışma yapılacağı duyurulmuştur. Yarışmaya 12 çizimiyle katılan Jørn Utzon (Şekil 2.12) 28 ülkeden katılan 233 yarışmacı arasından 218 numaralı tasarımı Şekil 2.13 ile birinci olmuştur (http-7).

Şekil 2.12 *Jørn Utzon (http-7)*



Şekil 2.13 *Sydney Opera Binası (http-7)*

Kitle-kaynak henüz bir kavram olarak isimlendirilmemişken birçok projede uygulanmış ve başarılı sonuçlara imza atılmıştır. Geçmişten günümüze gelene kadar projelerin çapları da genişlemiş, 1956 yılında yapılan Sydney Opera Binası yarışması dünya çapında bir proje olmuştur. Bu yıllardan sonra internetin ve sosyal medyanın hızla gelişmesiyle birlikte yapılan neredeyse bütün projeler dünya çapında projeler olarak yerlerini almaktadır. Günümüz teknolojisiyle ulaşılan noktada birçok sektör işlerinde kitle kaynaklı projeler üretmekte ve uygulamaktadır.

2.1.2. Kitle-kaynağın kullanım alanları

Sosyal medya kullanımı kitlelere ulaşmayı kolaylaştırmıştır. Bu sebeple de bir projenin duyurulması çok daha kolaylaşmıştır. Projenin bir tek alandan duyurulması dünyanın her noktasına ulaşımı sağlamakta ve daha geniş kitlelerin projelere katılımını sağlamaktadır. Kitlelerin genişlemesi projelerin çoğalmasıyla birlikte kitle-kaynaklı projelerin farklı sektörlerde kullanılabilir olmasını sağlamıştır. Kitle-kaynağın kullanıldığı sektörler 8 başlık altında örneklendirilebilir:

- Reklamcılık
- Yapay zekâ
- Film
- Bilim
- Televizyon programcılığı
- Oyun
- Tasarım
- Kitle fonlaması

Kitle-kaynak kullanıldığı sektörler ve farklı projeler açısından farklı avantaj ve dezavantajlara sahiptir. Kitle-kaynağın kullanıldığı her projenin başarılı olma ihtimali bulunmamaktadır.

Kitle kaynak kullanımının avantajları şunlardır:

- Proje bütçesinin uygun bir maliyette kalmasını sağlar.
- Kitlenin bilgi ve deneyiminin kullanılmasını sağlar.
- Projenin test aşamasında farklı kullanımların test edilmesini sağlar.
- Hata bulma olasılığını yükseltir.
- Proje testinin maliyetini düşürür.

Kitle kaynak kullanımının dezavantajları şunlardır:

- Projenin gizliliğine bir tehdittir.
- Test edenler arasında bir bağlantı kurulması zordur.
- Test kapsamını kontrol altında tutmak zorlaşır.
- Bulunan hatalar daha çok para kazanabilmek için manipüle edilebilir.

2.2. CBS

Günümüz teknolojilerinin gelişmesi ve şehirlerin büyümesi ile birlikte meydana gelen olayların konumdan bağımsız olması mümkün olmamaktadır. İnternet üzerinden yapılan bütün işlemleri de kapsar şekilde her işlemin bir konum verisi bulunmaktadır. Teknoloji ve teknolojiyi kullanan girişimler bu konum verilerini bir fırsata dönüştürerek analiz verisi olarak kullanmaya başlamışlardır. Teknolojiden önce de kullanılan bu analiz yöntemleri daha sınırlı ve maliyetli işler olarak değerlendirilebilir. Teknoloji, harita, konum verileri ve bu verilerden üretilen istatistiksel değerleri kullanabildiğimiz yapıya Coğrafi Bilgi Sistemleri adını verebiliriz. CBS birçok yazar tarafından farklı tanımlamalarla kullanılmıştır.

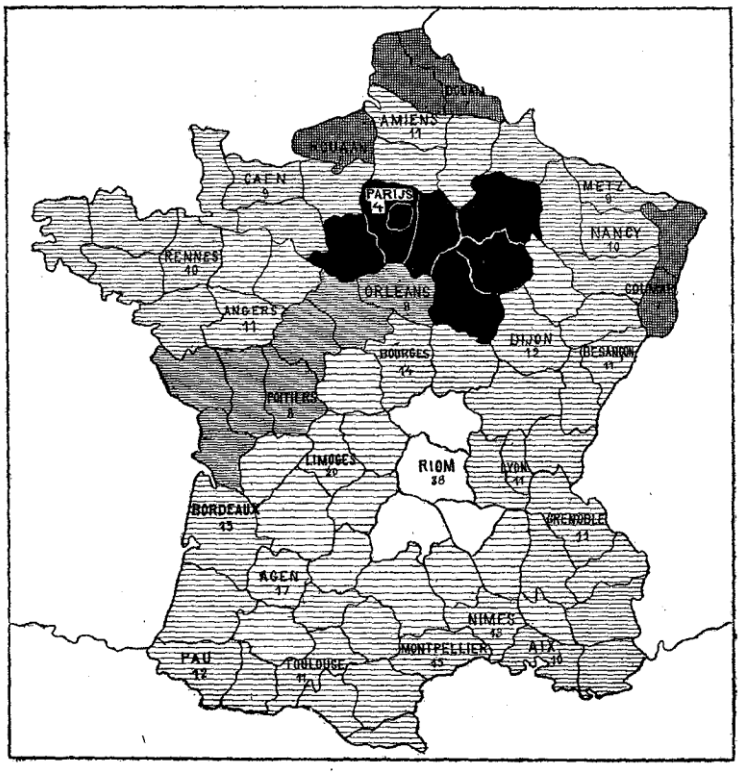
CBS, verileri işleyerek değer kazandırıp onları bilgiye çevirmektedir. Bilgiye dönüşen bu veriler farklı yollardan elde edilerek CBS yapısına uyarlanmaktadır. CBS'nin temel yapısı üzerinde iki çeşit veri yer almaktadır.

- Öznitelik verisi
- Konumsal veri
    - Vektör veri modeli
        - Nokta
        - Çizgi
        - Poligon
    - Raster veri modeli

CBS'yi kullanılabilir yapan veri türlerinin az ve kolay üretilebilir olmasıdır. Veri üretimi kolaylaştıkça yapılan analizler ve CBS'yi kullanan alanlar gün geçtikçe artmaktadır.

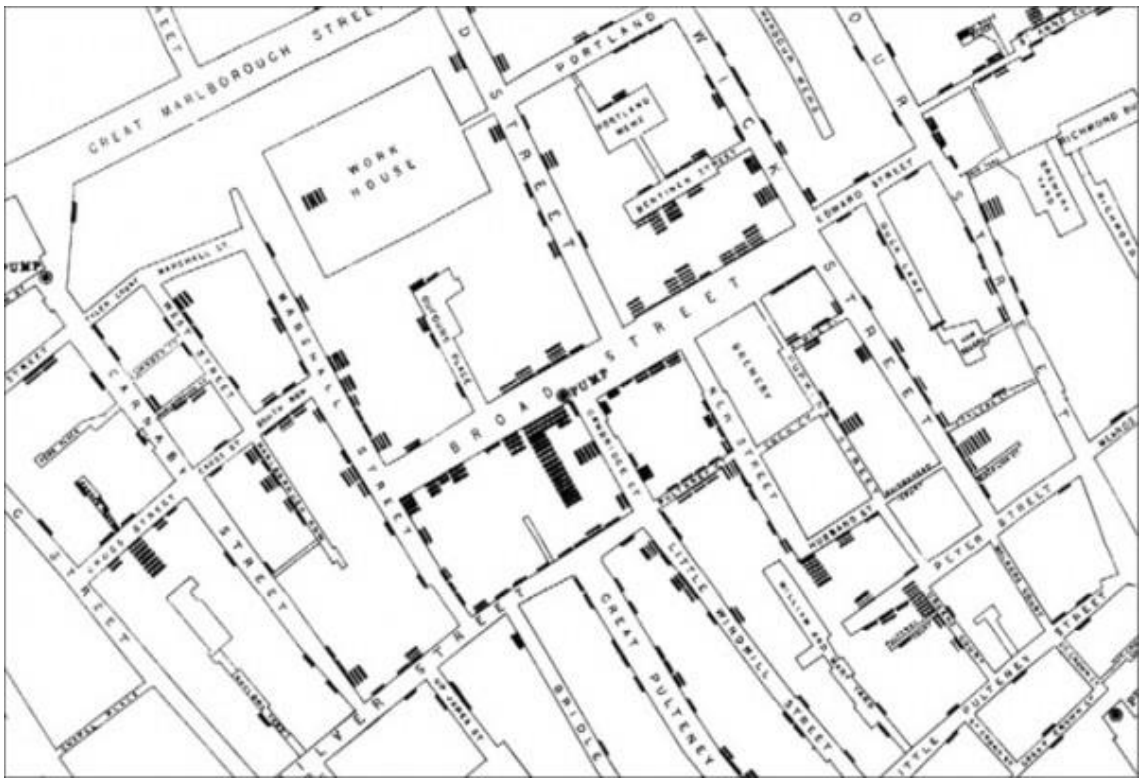
2.2.1. CBS' nin tarihsel gelişimi

CBS'nin asıl amacı analize yardımcı olacak verileri haritalandırarak anlamlandırmaktır. Bu amaçla yola çıkan istatistikçi Baron Pierre Charles Dupin, Fransa'daki okuma yazma oranını haritalandırmıştır (Şekil 2.14). 1819 yılında okuryazar oranlarının haritalandırıldığı çalışma, CBS'de çakıştırma analizleri için kullanılabilecek tematik haritaların ilki olarak kabul edilebilir (http-8).



Şekil 2.14 *Baron Pierre Charles Dupin'ın okuryazar haritası (http-8)*

1854 yılında Londra'nın Soho Mahallesinde meydana gelen kolera salgını üzerine Dr. John Snow bir harita üzerinde ölüm olaylarını işaretlemiş ve koleranın kaynağını belirlemeye çalışmıştır (Şekil 2.15). Kolera o yıllarda kötü havadan bulaşan bir hastalık olarak görülmektedir. Fakat Dr. John Snow'un haritasına göre ölümler bir sokağın köşesindeki tulumbanın etrafında daha yoğun olarak görülmektedir. Bu buluştan sonra tulumba kapatılır ve hastalık kısa bir süre sonra durur. Dr. John Snow kolera salgınını çözebilmek için CBS'den yararlanmıştır (http-9).

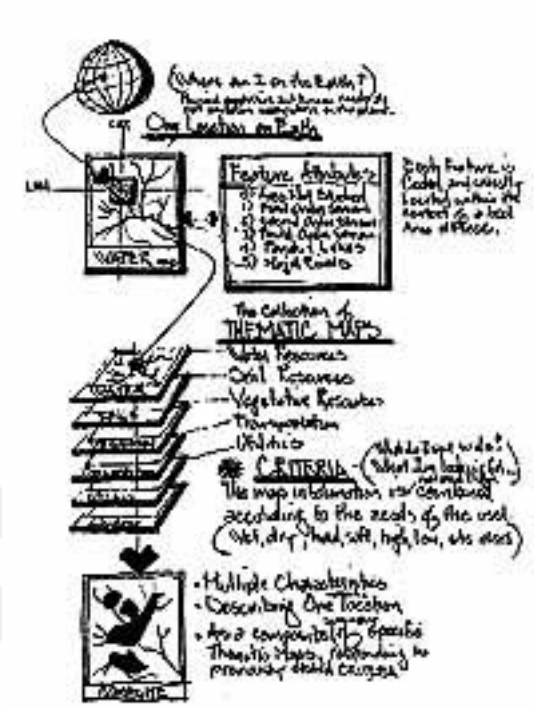


Şekil 2.15 *Dr. John Snow'un kolera haritası (http-9)*

Teknolojinin gelişmesiyle birlikte 1950'lerde Washington University'de bilgisayar destekli haritalama kullanılmaya başlamıştır. Bunu takip eden yıllarda artık CBS'nin akademik çalışmaları başlamış ve Kanada Coğrafi Bilgi Sistemi geliştirilmiştir. Bu proje Roger Tomlison tarafından 1963 yılında kurulan ilk bilgisayarlı CBS projesidir. İlk kez geliştirilen sistemde arazi, doğal kaynak gibi yapıların dökümleri çıkartılmaya başlanmıştır (http-10).

CBS, Ian L. McHarg tarafından geliştirilen teknikler sayesinde, döküm, toplama ve haritalandırma dışında kullanılmaya ve bugünkü halini almaya başlamıştır . Design with Nature kitabında Ian L. McHarg kullandığı teknikler hala CBS'nin en çok başvurulan

teknikleridir. CBS'de altyapı, en uygun yer bulma gibi konularda kullanılan çakıştırma analizini literatüre kazandırmıştır (Şekil 2.16).



Şekil 2.16 *Ian L. McHarg'ın çakıştırma analizi (http-11)*

Howard Fisher, 1964 yılında Harvard Konumsal Analiz ve Bilgisayar Grafikleri Laboratuvarları'nı kurarak CBS için önemli adımlar atmıştır. CBS'yi bilgisayar ortamında kullanmak üzerine yoğunlaştığı çalışmalar sonucunda Synagraphic Mapping System (SYMAP) yazılımı geliştirilmiştir. SYMAP sayesinde CBS'nin temel bileşenleri olan nokta, çizgi ve poligon mantığı ortaya çıkmıştır. Sonraki yıllarda bilgisayar sistemlerinin gelişmesi ile birlikte CBS üzerine birçok uygulama geliştirilmiş ve bu uygulamalar sayesinde CBS gelişmeye devam etmiştir.

1969 yılında Jack & Laura Dangermond tarafından Çevresel Sistemler Araştırma Enstitüsü (Environmental Systems Research Institute - ESRI) şirketi kurulmuştur. Şirket kurulduktan sonra sadece yazılım geliştirme ile ilgilenmeyip veritabanı uygulamaları, otomasyonlar ve daha çok eğitim ile ilgilenmiştir. 1981 yılında son kullanıcıya yönelik bir uygulama olan ArcInfo yazılımı piyasaya sürülmüştür. Bu yazılım sayesinde pek çok analizin yapılabilmesi kolaylaşmıştır. ESRI 1996 yılında Virtual Campus ile internet

üzerinden CBS eğitimi vermeye başlamıştır (Şekil 2.17). Verilen CBS eğitimi dünya çapında bir eğitimdir (http-12).

Şekil 2.17 *ESRI - Virtual Campus 1998 ekran görüntüsü (http-13)*

1990 yılından sonra CBS veri setleri gün geçtikçe artış göstermiştir. Veri setleri arttıkça üzerinde yapılan analizler çeşitlenmeye başlamış ve bu analizler farklı bilimlerin devreye girmesini sağlamıştır. İstatistiksel yöntemler ile analizler, rota ve en kısa yol hesaplamaları gibi farklı analiz uygulamaları ortaya çıkarak CBS'yi bir bilim haline getirmiştir.

2.2.2. CBS 'nin kullanım alanları

CBS konuma bağlı bir yapıda oluştuğundan, konuma bağlı yapılabilecek her türlü analiz ve yorumlamayı içeren alanlarda kullanılmaktadır. Bu alanlar, teknoloji geliştikçe ve verileri sayıları arttıkça artmaktadır.

- Mühendislik Uygulamaları
  - Elektrik, Su, Doğalgaz Şebekeleri
  - Telekomünikasyon Ağı
  - Araç Takip Sistemleri
- Tarım Uygulamaları
  - Bitki Örtüsü
  - Ekilebilir Arazi
  - Sulama Sistemleri

- Tahıl Rekolte Tahmini
- Toprak Haritaları

- Çevre Uygulamaları
  - Erozyon Risk Analizleri
  - Su Kaynakları ve Kirlilik Analizleri
  - İklim Bilim Çalışmaları
  - Sel Bölgeleri Risk Analizleri
  - Doğal Hayatı Koruma
- Yerbilimleri Uygulamaları
  - Maden, Petrol Alanlarının Tespit ve Envanteri
  - Jeolojik Haritalar
  - Deprem Risk Analizleri
- Ormancılık Uygulamaları
  - Orman ve Ağaç Envanteri
  - Orman Bölgelerinin Korunması ve Sürdürülebilirlik Analizleri
  - Acil Orman Yangın Tespiti ve Müdahale Sistemleri
- Arkeoloji
  - Arkeolojik Kazıların Haritalanması ve Kazı Envanteri
  - Tarihsel Sit Alanlarının Envanteri
  - Uydu ve Hava Fotoğraflarının İşlenerek Arkeolojik Yer Tespit Çalışmaları
  - Arkeolojik Ölçümler
- Kamu Uygulamaları
  - Yerel Yönetimlerde Kent Bilgi Sistemleri
  - Şehir Bölge Planlama Çalışmaları
  - Arazi Kullanım Haritaları
  - Altlık Harita ve İmar Planları
  - Kent Taşınmaz Envanteri
  - Tapu ve Kadastro Müdürlüklerine Mülkiyet Envanteri
  - CBS Tabanlı E-Devlet Projeleri
  - Kent İçi Ulaşım Planları
  - Toplum Sağlığı Analizleri ve Haritaları
  - Suç Analizleri ve Haritaları

CBS'nin kullanıldığı bazı uygulamalar yukarıdaki listede verildiği gibidir (Uyguçgil, 2011).

CBS kullanıldığı birçok alanda sağladığı faydalar yönünde vazgeçilmez bir bilim olmuştur. Kullanılan alanlara göre farklı avantajlar sağlasa da CBS'nin avantajları aşağıdaki gibi sıralanabilir:

- Hızlı ve kolay kullanım
- Verimli envanter yönetimi
- Bağlantılı ve bağlantısız verilere ulaşma
- Yapılan uygulamalarda yakında ve uzakta veri sorgulama imkânı
- Bilgi analizleri
- Çok farklı sektörde kullanılabilme imkânı
- Acil durum müdahale analizleri
- Veri güncelleştirme ve tanımlama
- Vektör ve raster verileri kullanma
- Ses ve GPS kullanabilme
- Harita katmanlandırabilme
- Veri sınıflandırması, sembol değiştirme, etiketleme
- Veritabanlarıyla uyum
- SQL sorgulaması yapabilme
- Özel analiz ve sorgulama yapabilme
- Adres coğrafyasını bulma
- Geniş veri setinde çalışabilme
- Veri dağılımı mevcuttur

2.3. Örnekler

Bu bölümde Türkiye ve Dünya'da yapılmış kitle-kaynak ve CBS uygulamaları ele alınmıştır.

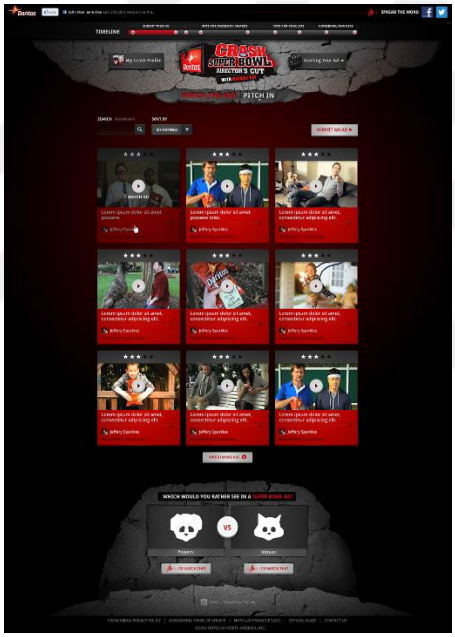
2.3.1. Dünya'dan kitle-kaynak uygulamaları

Dünya'da yapılan kitle-kaynaklı uygulamalar sektörlerine göre ayrılarak araştırılmıştır.

2.3.1.1. *Reklam sektöründe kitle-kaynaklı uygulamalar*

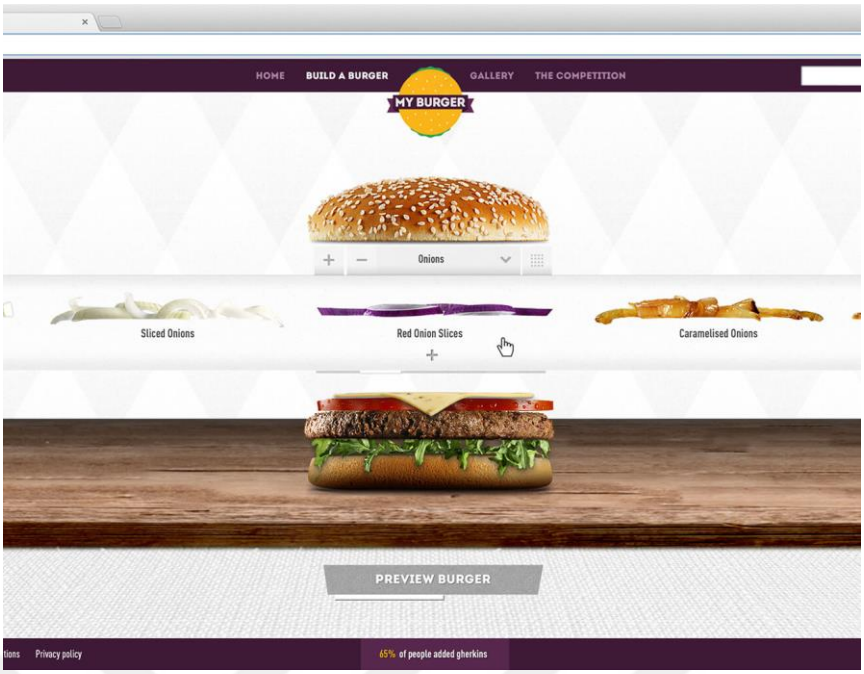
Kitle-kaynak reklam sektöründe uzun süredir kullanılan bir yöntem olmuştur. Reklam sektörü için kitle-kaynaklı bir uygulamadan yararlanmak hem yaratıcılığı artırmakta hem de reklam yapacak firmanın bütçesini hafifletmektedir.

Frito-Lay firması 2006 yılında The Crash The Super Bowl isimli bir yarışma düzenlediğini duyurarak kitleden 30 saniyelik reklam filmlerini göndermelerini istemiştir (Şekil 2.18). Seçilen film televizyonda yayınlanmış ve kazanan yarışmacı 10.000$ ödül kazanmıştır. Frito-Lay daha sonraki yıllarda da bu yarışmayı düzenleyerek çeşitli ödüller dağıtmıştır. En son yarışma 2016 yılında yapılmış ve daha önceki yıllarda kazananların katılması istenmiştir.



Şekil 2.18 *The Crash The Super Bowl yarışma sayfası (http-14)*

2014 yılında McDonald's İngiltere'de bulunan restoranlarında sunmak üzere lezzet üretme yarışması düzenlemiştir. Yarışmacılar oluşturulan kitle-kaynaklı platform üzerinden kendilerine verilen 80 adet malzemeden istediklerini kullanarak kendi hamburgerlerini oluşturmuşlardır (Şekil 2.19). Yarışmaya 98.000 kişi katılmış ve katılan yarışmacıların arasından 5 adet lezzet seçilmiştir. Bir hafta boyunca kitle değerlendirmesi ile denen lezzetler 11'e indirilmiş ve bu 11 lezzet arasından profesyonel jürinin seçimi ile 5 tanesi belirlenmiştir (Şekil 2.20).



Şekil 2.19 *MY BURGER yarışması için hazırlanan kitle kaynak platform (*http-15*)*

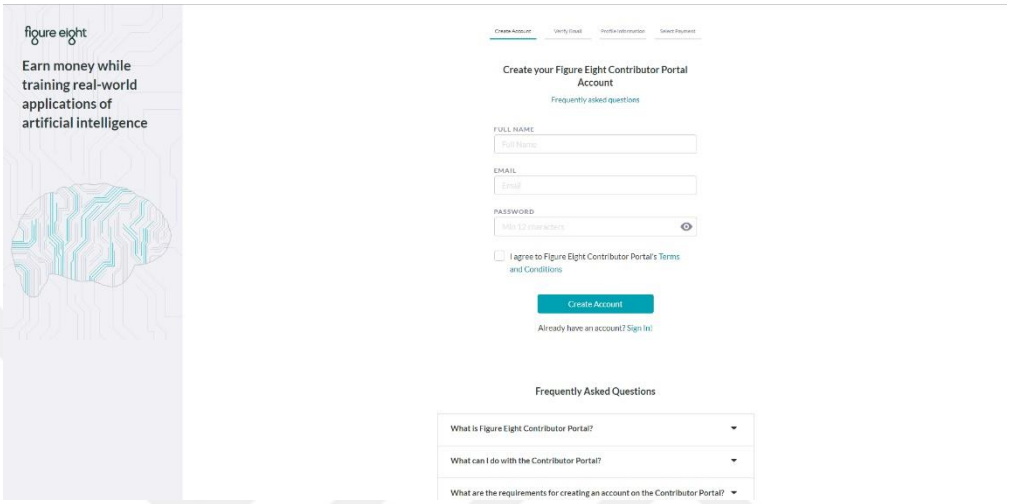
Şekil 2.20 *MY BURGER yarışmasını kazanan lezzetler (http-16)*

2.3.1.2. *Yapay zekânın geliştirilmesinde kitle-kaynaklı uygulamalar*

Yapay zekâ için kullanılan platformlar makine öğrenmesini kolaylaştırmak ve makinanın yapamayacağı işleri kitlelere yaptırmak üzerine tasarlanmışlardır. Bu işlemlerin kitlelere yaptırılmasının en büyük getirisi ise yapılan işlerin doğruluğunu sağlamaktır.

Dolores Lab olarak kurulmuş olan ve şu anki adı Figure-Eigth olan firma tarafından oluşturulan bir yapay zekâ projesi bulunmaktadır. Proje firma ile aynı ismi taşımaktadır. Yapay zekâ projesi olduğu gibi bir kitle fonlama projesi olarak da görülebilir. Oluşturulan internet platformu üzerinden üyelik sistemi ile çalışmaktadır (Şekil 2.21). Üyeler

belirlenen görevlerin zorluk derecesine göre puan almaktadır. Üyelerin aldıkları puanlar birikerek üyelerin seviyelerini oluşturmaktadır. Böylece üyeler yaptıkları görevlere göre belirli miktarda para kazanmaktadır. Projenin asıl amacı makinenin insan davranışlarını doğru ve hızlı olarak öğrenmesini sağlamaktır.

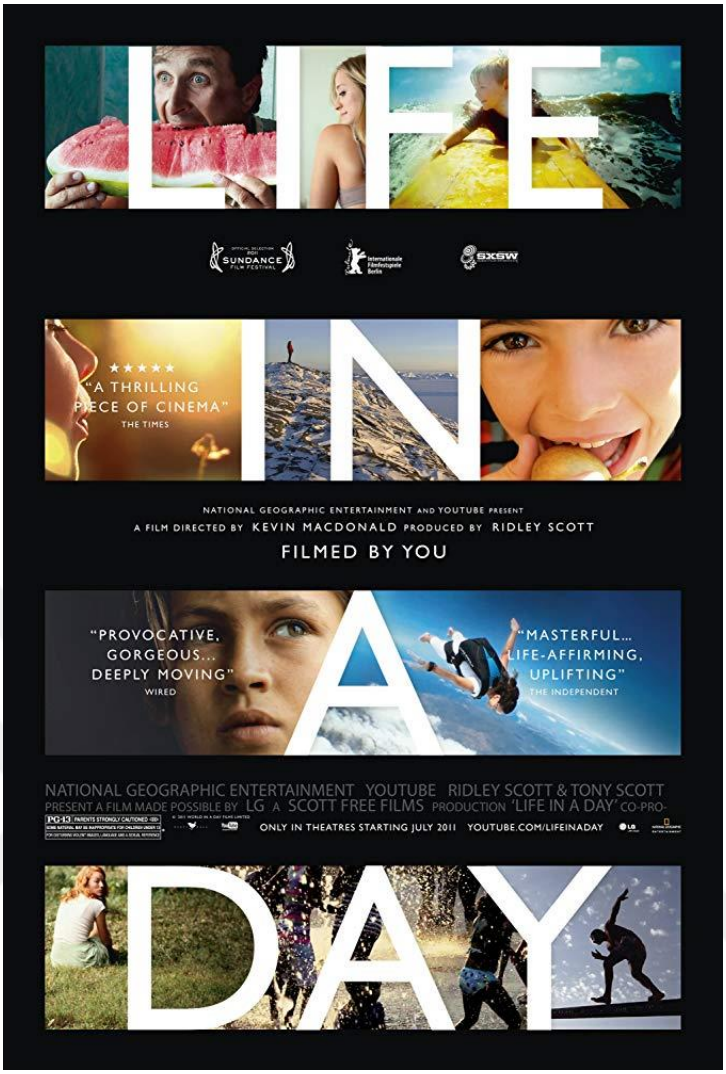


Şekil 2.21 *Figure Eight uygulaması (http-17)*

2.3.1.3. *Film sektöründe kitle-kaynaklı uygulamalar*

Film sektörü adına yapılan kitle-kaynak uygulamaları henüz sadece deneme aşamasındadır. Bir film çekebilmek için gereken tüm kaynakları kitle kaynak uygulama ile sağlayabilecek bir altyapı bulunmamaktadır. Yapılan deneysel yaklaşım sadece görüntü montajı ve hikâye örgüsünü sağlamaktadır. Kitle-kaynak uygulamalarla film için maddi kaynak ve hikâye sağlanması yolunda gelişmeler yaşanmıştır. Bir filmin tamamını kitleye bırakmak daha denenmiş bir yöntem değildir.

Ridley Scott'un yapımcılığını üstlendiği ve yönetmenliğini Oskar ödüllü Kevin Macdonald'ın yaptığı bir proje filmi olan Dünyada Bir Gün (Life in a Day) 2011 yılında gösterime girmiştir (Şekil 2.22). Bu film için 24 Temmuz 2010 tarihinde kullanıcılardan bir günlerini anlatan videoları Youtube'ye yüklemeleri istenmiştir. 192 farklı ülkeden 80.000 kişi videolarını yükleyerek 4500 saatlik bir kayıt oluşturmuşlardır. Bu 4500 saatlik kayıt içerisinden Ridley Scott ve Kevin Macdonald ile birlikte bir ekip tarafından 94 dakika 53 saniyelik bir film oluşturulmuştur. Film 27 Ocak 2011 tarihinde gösterime girmiştir.



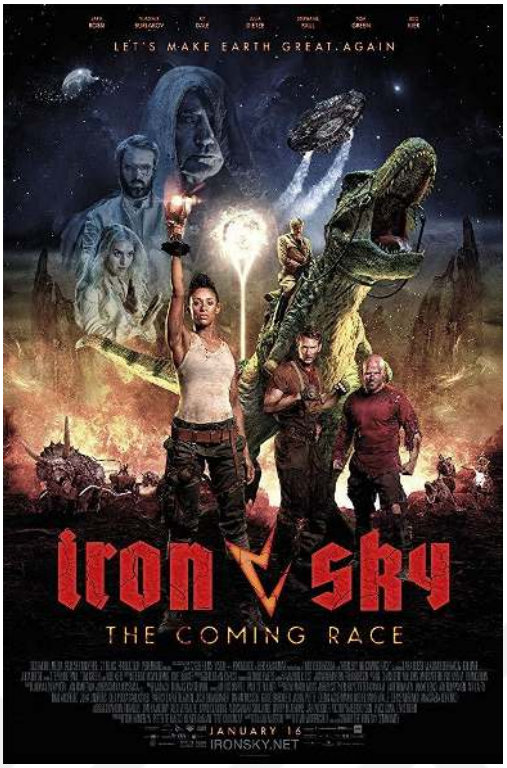
Şekil 2.22 *Life in a Day film afişi (http-18)*

Kitle fonlaması ile yapılan birçok film örneği bulunmaktadır. Filmlerin artan maliyetlerini karşılamak için genellikle filmin hayranlarından oluşan bir kitleden destek alınmaktadır.

2013 yılında Veronica Mars dizisi hayranları bir araya gelerek bir kampanya başlatmışlardır. Veronica Mars filminin çekilebilmesi için başlatılan kampanya dâhilinde 91.585 kişiden 5.702.153 dolar para toplanarak film çekilmiştir (Şekil 2.23). Film kitle fonlaması ile toplanan en yüksek bağış miktarına ulaşmıştır. Filmin gişe hasılatı ise 3,5 milyon dolardır.

Şekil 2.23 *Veronica Mars film afişi (http-19)*

Iron Sky filmi bağımsız ve bilim kurgu bir yapım olduğu için bütçe bulma sıkıntısını yine kitle fonlaması ile çözme yoluna giderek aşmıştır. Film için ilk etapta senaryo ve hikâye için 150 bin dolarlık bir proje duyurusu yapılmıştır. Bu kampanya için 3517 destekçiden 182.557 dolar toplanmıştır. En büyük katkıyı sağlayan destekçiye ise film içerisinde konuşma bulunan bir sahnede oynama şansı verilmiştir. İkinci etapta filmin çekimleri için bir proje daha duyurulmuş ve bu proje için ise 9408 kişinin desteği ile 657.414 dolar toplanmıştır. Film 14 Haziran 2019'da gösterime girmiştir (Şekil 2.24).



Şekil 2.24 *Iron Sky film afişi (http-20)*

Veronica Mars ve Iron Sky gibi yapımları takip eden birçok bağımsız ve kısa film bulunmaktadır. Tüm filmler istenilen desteği alamamaktadır. Kitle-kaynağını film sektöründe kullanabilmek için incelenen örnekler film sektörünü, hayranların yönlendirebileceğine işaret etmektedir.

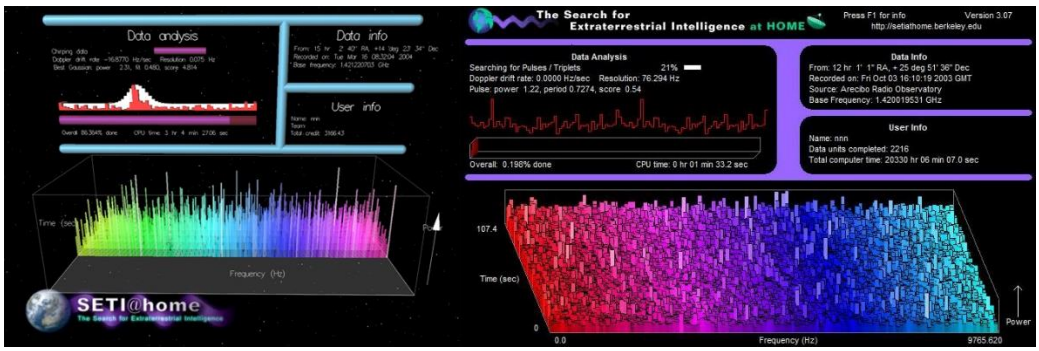
Film sektörü için kitle-kaynağın bir diğer kullanım şekli ise hikâye ve senaryo toplama yolunu seçmektir. Amazon oluşturduğu bir kitle kaynaklı platform üzerinden üyelerin kendi fikirleri olan senaryo ve hikâyeleri toplamaktadır. Amazon'a üye olan herkesin ulaşabileceği bir platform olan Amazon Storybuilder kullanıcıların fikirlerini bir mantar pano gibi saklamaya imkân tanımaktadır (Şekil 2.25).

Şekil 2.25 *Amazon Storybuilder (http-21)*

2.3.1.4. *Bilimde kitle-kaynak uygulamalar*

Bilim için kitle-kaynağından yararlanmak dünya çapında bir projeyi kısa sürede çözme ve daha büyük bir güçten destek alma fikrinden yola çıkılarak oluşturulmuştur.

Kaliforniya Üniversitesi'nde başlanan ve halen devam etmekte olan SETI@home projesi şimdiki zamana kadar en büyük katılım sağlanan kitle kaynak uygulaması olarak Guinnes Rekorlar Kitabı'na girmiştir. Proje 17 Mayıs 1999'da başlamış ve dünyanın en büyük analiz ağını oluşturmuştur. Arecibo Gözlemevindeki bir radyo teleskobundan gelen verileri 5.2 milyon katılımcı ve bilgisayarlarının yardımıyla analiz ederek dünya dışı bir varlık arayışında bulunmaktadır. Projede internete bağlı bir bilgisayar için kullanılan küçük bir programcık sayesinde her bilgisayar sahibinin gönüllü olarak katılması sağlanmıştır. Bu programcık radyo teleskop verilerini indirerek analiz etmektedir. SETI@home projesi 15 Aralık 2005 tarihinde kapatılmış ve BOINC platformuna taşınmıştır (Şekil 2.26).



Şekil 2.26 *SETI@home ve BOINC istemcileri (http-22)*

2014 yılında ESA tarafından gerçekleştirilen bir oyun sayesinde uzay projelerini güçlendirebilecek bir kitle kaynak projesi yaratılmıştır. ESA'nın Astro Drone (Şekil 2.27) adını verdiği artırılmış gerçeklik oyunu Parrot AR Drone (Şekil 2.28) sahibi herkesin oynayabileceği bir oyun geliştirmiştir. Oyun üzerinde drone ile birlikte yapılabilecek çeşitli seviyelerde iniş görevleri mevcuttur. Bu sayede oyuncuların izni dâhilinde oyuncu hareketleri, iniş ve kaçış şekilleri internet üzerinden ESA'ya gönderilmektedir. Verilerin incelenmesi ile ESA uzay görevlerindeki sondaj gibi işlemler için geliştirme sağlamayı amaçlamıştır.

Şekil 2.27 *Astro Drone oyun görüntüsü (http-23)*

Şekil 2.28 *Parrot AR Drone (http-24)*

2.3.1.5. *Televizyon programcılığında kitle-kaynak uygulamaları*

Televizyon sektörü için genellikle formata göre kitlelerin başvurduğu ve yönlendirdiği yarışma programları oluşturulmuştur. Program sahipleri sadece gereken platformu hazırlar ve katılımcıların başvurularını değerlendirir. Geri kalan programın tüm akıbeti izleyici ve katılımcı kitlededir.

Yaratıcısı Franc Roddam 1990 yılında BBC için MasterChef formatını hazırlamıştır (Şekil 2.29). Daha sonra tüm dünya üzerinde bu format uygulanmak üzere pazarlanmıştır.

Format gereği program başlamadan belirli bir süre önce duyurularak yarışmaya katılacak katılımcı seçimleri gerçekleştirilmektedir. Daha sonra yarışma içerisinde profesyonel bir jüri tarafından değerlendirmeler yapılmaktadır. Yarışma sona yaklaştıkça izleyicilerden de bu sürece dâhil olarak kazanacak yarışmacının belirlenmesi üzerine oylamaya katılması istenmektedir.

Şekil 2.29 *2019 MasterChef ekran görüntüsü (http-25)*

CNBC tarafından 2013 yılında kitle kaynaklı bir program olan Crowd Rules programı yayına girmiştir (Şekil 2.30). Program, stüdyoda bulunan yüz kişilik bir kitlenin oylarına başvurarak kazananı belirlemek üzerinedir. Kitle programın her hafta değişen konusu üzerine katılan üç küçük işletme sahibinin hikâyelerini, başarılarını ve gelecek planlarını dinleyerek hangisinin ödülü alacağına karar vermektedir. Kitlenin verdiği karar doğrultusunda işletme sahibi 50.000 doların sahibi olmaktadır.

Şekil 2.30 *Crowd Rules ekran görüntüsü (http-26)*

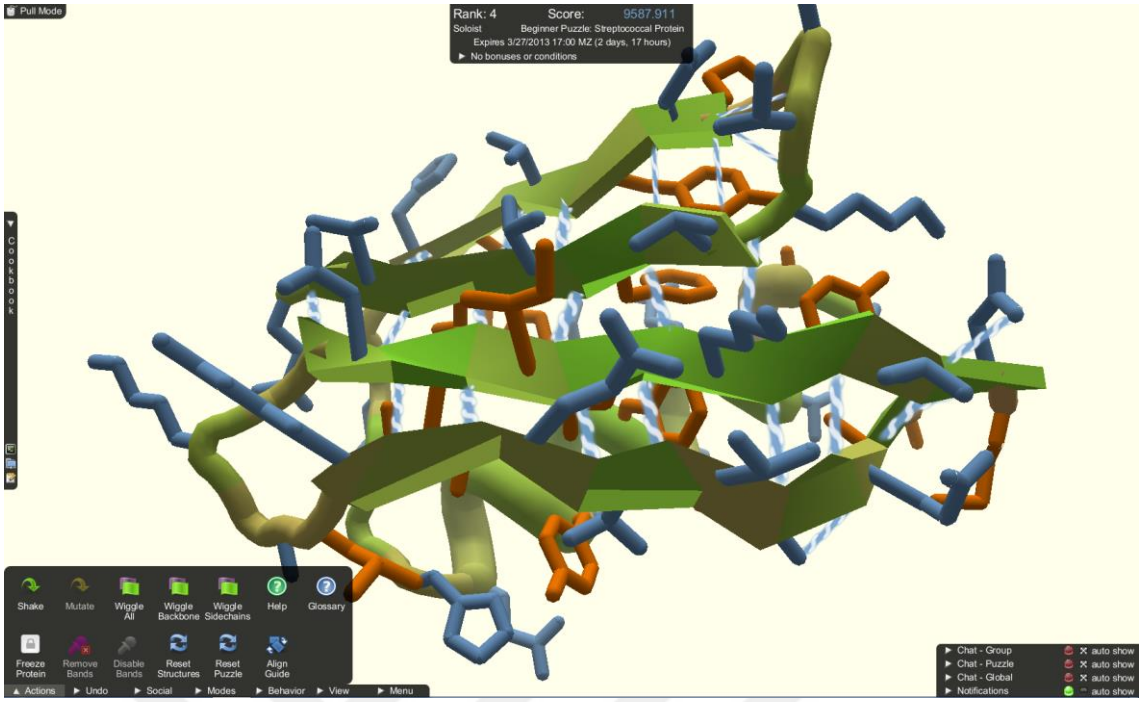
2.3.1.6. *Oyun sektöründe kitle-kaynak uygulamaları*

Oyun sektörü için planlanan uygulamalar ya bilimi geliştirmeye yönelik ya da sanal bir sosyalleşme ortamı yaratmaya yönelik uygulamalardır. Günümüz teknolojisi ve internet sayesinde oyunlar internet üzerinden eş zamanlı olarak oynanabilmektedir. Bu sayede oyuncular kendilerine oyun içerisinde sanal bir hayat kurabilmektedirler. Kurulan bu sanal hayat üzerinde herkesin farklı bir rolü ve özelliği bulunmaktadır. Bu farklı özellikler sayesinde oyun içerisinde bir görev için bir araya gelip işbirliği yapabilmektedirler. Bilim üzerine yapılan oyunlar ise kitleyi kullanarak veri toplamak ve verileri analiz etmek üzerine çalışmaktadır.

Blizzard Entertainment firması tarafından 2004 yılında World of Warcraft adıyla bir oyun piyasaya sürmüştür (Şekil 2.31). World of Warcraft internet üzerinden çok oyunculu olarak oynanan bir oyundur. Blizzard sanal bir dünya ve ırklar yaratarak oyuncuların üyeliklerini bu ırklar üzerinden açmasını amaçlamıştır. World of Warcraft dünya çapında sevilen bir oyun olarak yaklaşık 100 milyondan fazla oyuncuya ev sahipliği yapmaktadır. Üyeler firmaya oyun oynadıkları ay boyunca üyelik ücreti ödemektedirler.

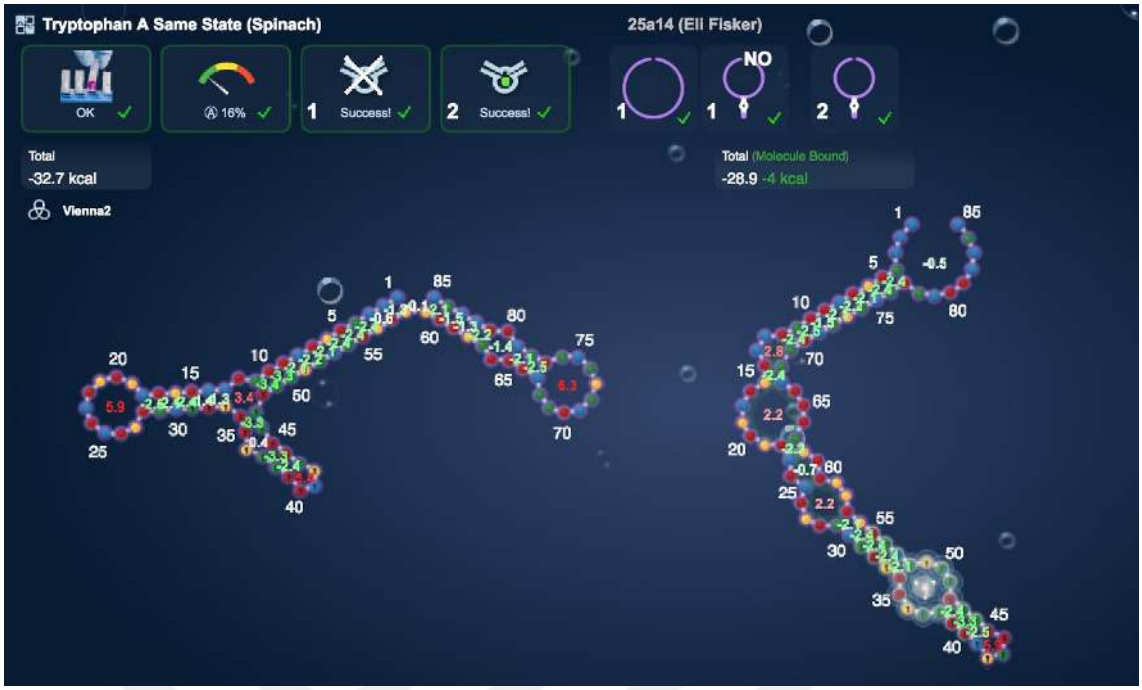
Şekil 2.31 *World of Warcraft ekran görüntüsü (http-27)*

Protein katlanması üzerine araştırmalar yapan Washington Üniversitesi başlangıç olarak SETI@home gibi bir uygulama yapma yoluna gitmiştir. Yapılan bu uygulamaya farklı bir yaklaşım getiren oyun tasarımcısı Seth Cooper protein katlanmasını bir oyun olarak 2008 yılında piyasaya sunmuştur (Şekil 2.32). Piyasaya sunulan Foldit oyunu 240.000 oyuncu sayesinde büyük bir başarı göstermiştir. Oyundaki işbirliği sayesinde protein katlanmasının tahmininden çok, bilinmeyen yapıların bile çözümlenmesine olanak sağlanmıştır.



Şekil 2.32 *Foldit oyunu ekran görüntüsü (http-28)*

Carnegie Mellon Üniversitesi ve Stanford University ortak gerçekleştirdiği bir proje ile Washington Üniversitesi'nin Foldit projesine benzer bir proje geliştirmişlerdir. Proje sayesinde RNA'nın katlanmasını ve sentezini modellemeyi amaçlamışlardır. Projede 2010 yılında Eterna isimli bir oyun geliştirip piyasaya sürülmüştür (Şekil 2.33). Oyunda yaklaşık 250.000 kişinin katılımıyla 90.000 üzerinde RNA modeli oluşturulmuştur.

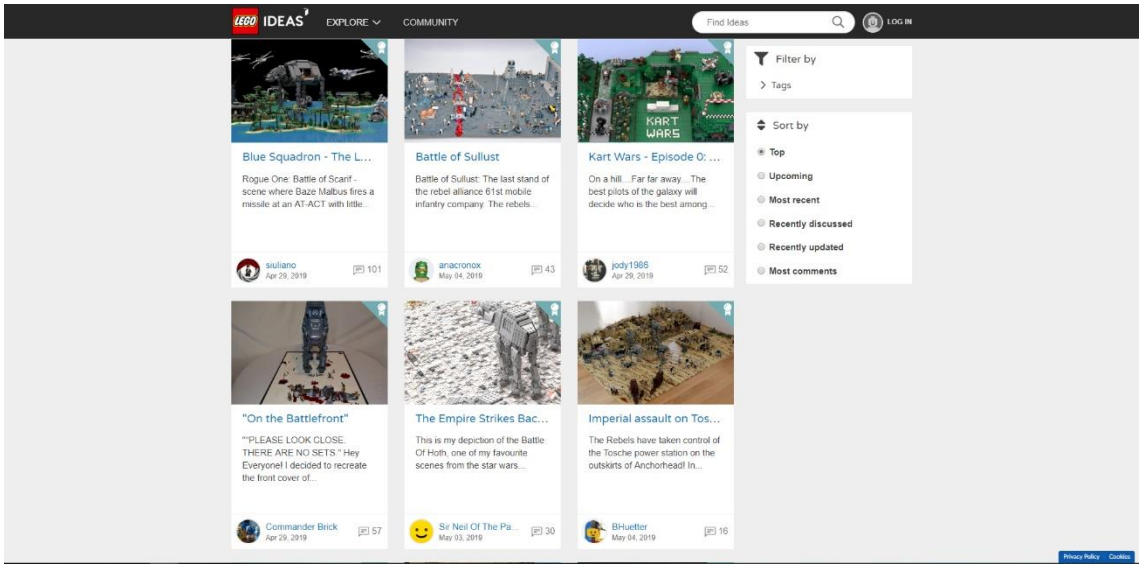


Şekil 2.33 *Eterna oyunu ekran görüntüsü (http-29)*

2.3.1.7. *Tasarım alanında kitle-kaynak uygulamaları*

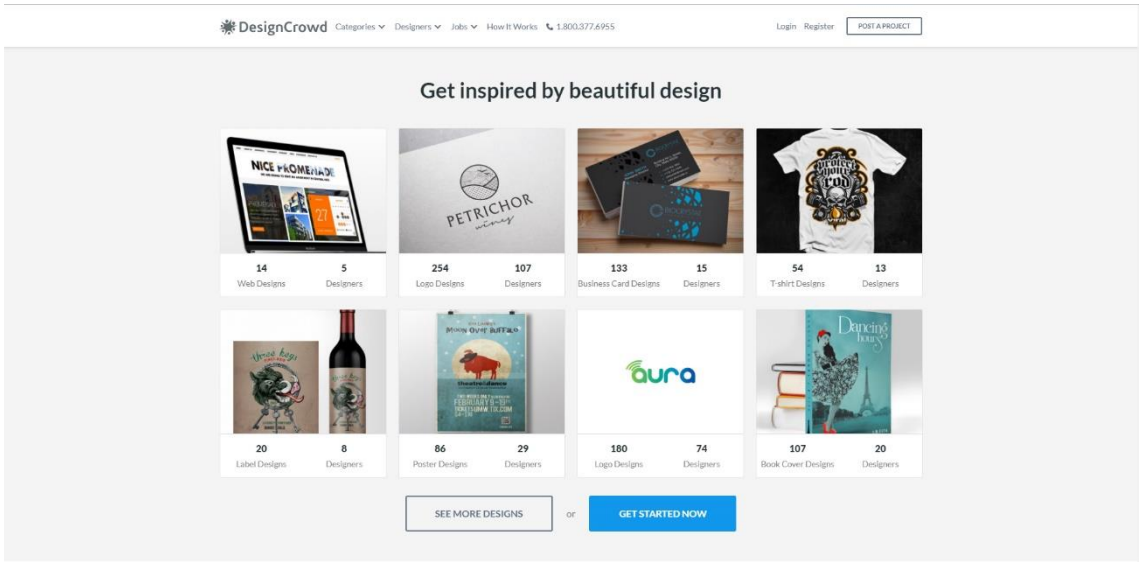
Tasarım alanında kitle-kaynaklı uygulamalar kitlenin yaratıcılığından yararlanmaktadır. Katılımcılar yaptıkları tasarım unsurlarıyla genellikle bir sıralamaya tabi tutulurlar ve kazandıkları takdirde ödül alırlar.

LEGO Group tarafından 2011 yılında oluşturulan bir internet sitesi olan LEGO Ideas katılımcıların kendi tasarımlarını yaratmasına ve tasarımlarının denenmesine olanak sağlamaktadır (Şekil 2.34). Yapılan tasarımlar bu platforma yüklenerek oylamaya açılmakta, oylamada gereken desteği alan tasarım, üretim ve piyasaya sürülme sürecine girmektedir. Süreci tamamlayan tasarımın yaratıcısına o tasarımdan kazanılan telifin %1 'i verilmektedir.



Şekil 2.34 *LEGO Ideas internet platformu (http-30)*

Avustralya'nın Sidney kentinde 2008 yılında tasarımcılar için bir platform oluşturulmuştur (Şekil 2.35). Sistem müşteriler ile tasarımcıları bir araya getirerek iş çözüm süreçlerinin maliyetlerini düşürmek ve tasarımcılara maddi kaynak sağlamak üzerine çalışmaktadır. DesignCrowd isimli platformda 700.000 civarında kayıtlı tasarımcı bulunmaktadır. Logo tasarımı, kurumsal kimlik tasarımı, web tasarımı ve tişört tasarımı gibi konularda tasarım desteği alınabilmektedir.

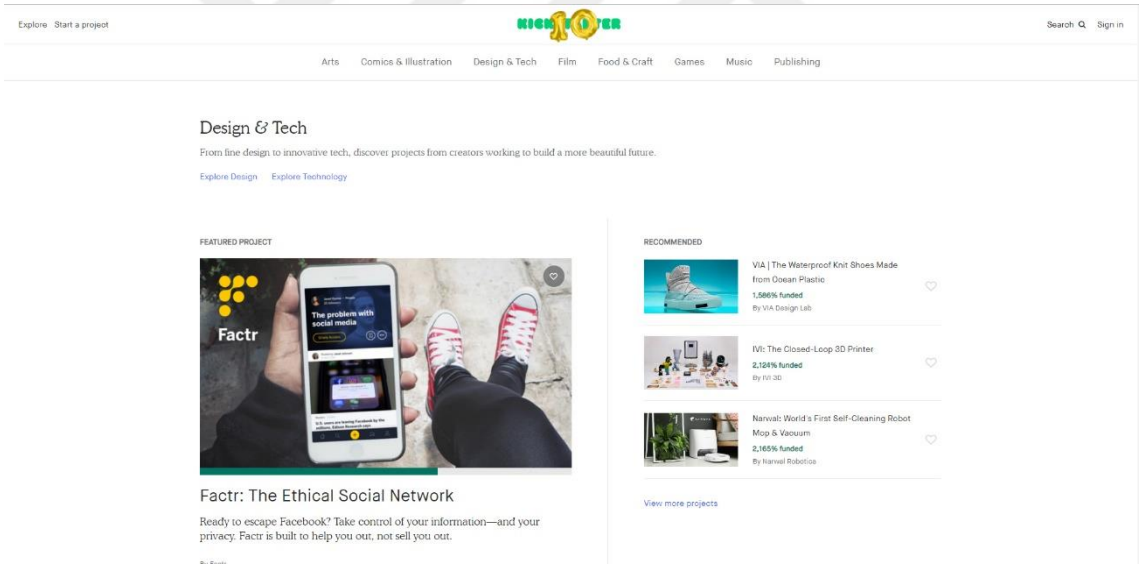


Şekil 2.35 *DesingCrowd platform ekran görüntüsü (http-31)*

2.3.1.8. *Kitle fonlaması uygulamaları*

Kitle fonlaması uygulamalarında üretim sürecine başlamadan bir fikir kitleye duyurulmaktadır. Kitleye duyurulan fikir kitle tarafından beğenilir ve fikre bağış yapılırsa fikrin gerçekleştirilmesi için gereken miktar toplanır ve proje hayata geçirilir.

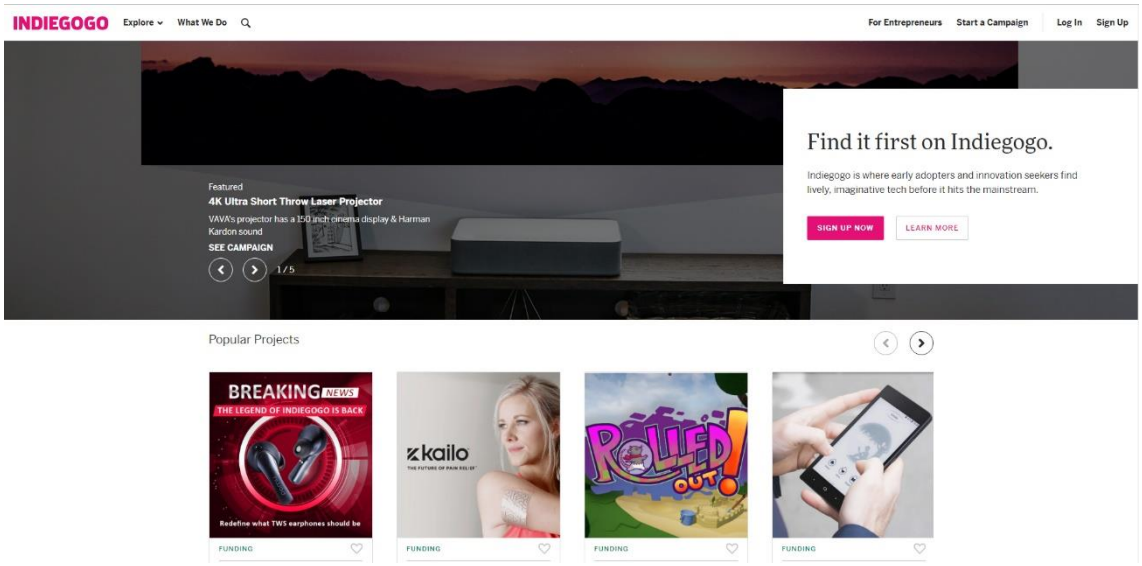
2009 yılında hayata geçen ve projelere bağış toplayarak ünlenen bir uygulama olan Kickstarter günümüze kadar 150.000 üzeri projenin gerçekleştirilmesini sağlamıştır (Şekil 2.36). Proje fikri olanlar Kickstarter üzerinden projelerini tanıtır ve proje için gerekli olan miktarı belirtir. Projeye bağış yapmak isteyenler için farklı bağış miktarları belirlenmekte ve bu miktarlar için bazı ödül ve öncelikler bulunmaktadır. Bağış miktarına göre proje ile ilgili bir anahtarlık, bir tişört veya proje sonucunda üretilecek ürünün ilk gönderimi gibi seçenekler belirlenmektedir. Proje bağışı belirlenen miktara ulaşırsa proje yapımına geçilir, proje bütçesine ulaşılamaz veya proje başarısızlığı uğrarsa bağışçıların paraları iade edilir. Kickstarter ise bu bağışların %5 'ini kâr olarak alır.



Şekil 2.36 *Kickstarter ekran görüntüsü (http-32)*

Indiegogo, Kickstarter gibi bir kitlesel fonlama platformudur. Platform Kickstarter'den önce hayata geçirilmiştir (Şekil 2.37). Indiegogo 2008 yılından bu yana hizmet vermektedir. Mantık olarak her iki platform aynı süreçler ile çalışmaktadır. Indiegogo'nun Kicstarter'den farkı;

- Altyapısal farklılıklar vardır.
- Indiegogo dünyanın daha geniş bir kesmine hizmet vermektedir.
- Kickstarter'de bir ürün projesi oluşturabilmek için örnek bir ürüne ihtiyaç vardır.
- Indiegogo'da proje belirlenen miktarda bağış alamasa bile proje sahibi bağışlanan parayı alır.
- Indiegogo proje süreçlerine destek hizmeti sunmaktadır.



Şekil 2.37 *Indiegogo platform ekran görüntüsü (http-33)*

2.3.2. Türkiye'den kitle-kaynak uygulamaları

Ülkemizde uygulanmış kitle-kaynaklı projeler araştırıldığında, kitle-kaynaklı uygulamaların ülkemizde yaygın olarak kullanılmadığı göze çarpmaktadır. Kullanım örneği bakımından başvurulabilecek kaynak sayısı da yok denecek kadar azdır. Genellikle kullanım alanı olarak televizyon ve reklamcılık sektöründen örnekler ile karşılaşılmıştır.

Yabancı bir firma olan FritoLay tarafından 2009 yılında Doritos ürünü için bir lezzet yarışması düzenlemiştir. Düzenlenen yarışma katılımcıların Doritos adına bir ürün fikri belirlemesi ile gerçekleşmiştir. Kampanyaya www.tytz.com internet adresi üzerinden fikir gönderimi ve yine aynı adres üzerinden oylama yapılmıştır. Kampanyaya 83 bin öneri sunulmuştur. Bu öneriler arasından Hazar İyiduvar 'Hadi Gari Haydari'

isimli lezzet önerisi ile birinci olmuştur (Şekil 2.38). Birinci olan Hazar İyiduvar 50.000 liralık ödül ile ürün cirosunun %1'ine sahip olmuştur.

Şekil 2.38 *Hisseli tatlar kampanyası kazananı Hazar İYİDUVAR (http-34)*

Turkcell firması tarafından yapılan bir kitle-kaynaklı uygulama olan 'Turkcell'in Çekim Gücü' reklam kampanyası 2009 yılında gösterime girmiştir. Turkcell bu kampanya dâhilinde kullanıcılarından gelen videoları reklamlarında kullanmıştır (Şekil 2.39).

Şekil 2.39 *Turkcell reklamı ekran görüntüleri (http-35)*

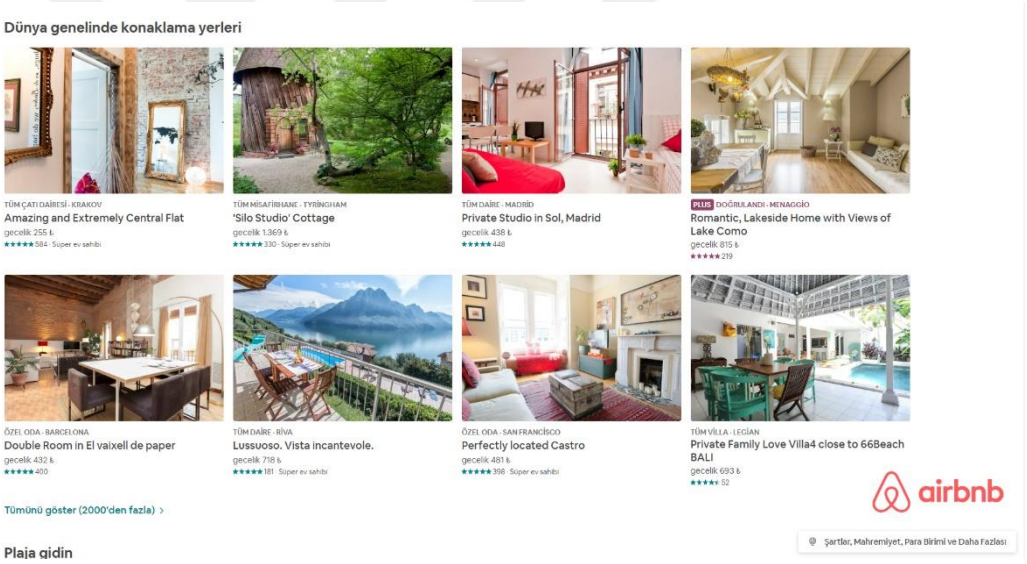
Televizyon yapımcısı Charlie Parsons tarafından üretilen bir içerik olan Survivor ülkemizde bu fikri geliştiren ve pazarlamaya başlayan Acun Ilıcalı tarafından yönetilen kitle-kaynaklı bir uygulama olmuştur. Acun Ilıcalı 1994 yılında ortaya çıkan fikri geliştirerek günümüzdeki haline getirmiştir (Şekil 2.40). Yapım harici tüm katılımcılar kitle kaynaklıdır. Yarışmacılar halktan başvuran kişiler arasından seçilmekte ve yarışmaya katılmaktadır. Yarışmanın seyri ise izleyicilerin attıkları oylarla belirlenmektedir.

Şekil 2.40 *Survivor 2019 (http-36)*

2.3.3. Türkiye ve Dünya'dan CBS içerikli kitle-kaynak örnekleri

Araştırma içeriğine uygun olarak içeriğinde harita ve analiz barındıran uygulama örnekleri evrensel çaptadır. Uygulamalar Türkiye üretimli değildir, fakat Türkiye'de de kullanılabilir durumdadır.

2008 yılında Amerika'da kurulmuş bir firma olan Airbnb kitle kaynaklı bir uygulama olarak evrensel çapta yoğun olarak kullanılan bir uygulamadır (Şekil 2.41). Uygulama özellik olarak kiracıları ve ev sahiplerini bir araya getiren bir platformdur. Evlerini kiraya vermek isteyen kişilerce evin özellikleri, fotoğrafları ve harita üzerinden konumu işaretlenerek fiyat belirlemesi yapılmaktadır. Kiralık ev arayan kişiler harita üzerinden günlük, haftalık ve aylık olarak bütçelerine göre evleri kiralamaktadır. Bu şekilde ev sahipleri evlerini kiraya vererek para kazanmış olmakta ve kiracılar ise kısa süreli olarak konaklayacakları kendilerine uygun evleri kolaylıkla belirleyebilmektedir.



Şekil 2.41 *Airbnb internet platformu ekran görüntüsü (http-37)*

Travis Kalanick ve Garrett Camp tarafından 2009 yılında bir fikir olarak kurulmuş ve daha sonra piyasaya sürülmüş bir uygulama olan Uber, tüm Dünya üzerinde en çok tartışılan uygulamalardan biri olmuştur (Şekil 2.42). Uber kişisel araçları taksi olarak kullanıma açan bir uygulamadır. Uygulama sayesinde araç sahibi para kazanırken yolcular için de ulaşım maliyetini düşürmektedir. Uygulamanın suç işleme konusunda çok açık bir araç olarak kullanılabilir olması nedeniyle birçok ülkede yasaklanmıştır.

Uber yolculuk yapmak isteyen kişi ve sürücüyü harita üzerinde en yakın konuma göre eşleştirerek yolculuğun yapılmasını sağlamaktadır.

Şekil 2.42 *Uber uygulaması kullanan bir sürücü (http-38)*

Bir mağaza takip platformu olarak piyasaya çıkan BeMyEye uygulaması firmalar ile yapılan anlaşmalar üzerine firmaların market stantlarının ve satışlarının kontrolünü kitle kaynaklı olarak kontrol ve analiz eden bir platformdur (Şekil 2.43). Firmaların kontrol edilmesini istediği yerler için görev tanımlaması yapılmaktadır. Bu görev tanımlamaları harita üzerinde belirlenerek kullanıcılara ulaşmaktadır. Kullanıcılar uygulama üzerinden verilen görevi yerine getirerek para kazanmaktadırlar. Görevler genellikle stant düzeninin fotoğrafını çekmek ve fiyatları kontrol etmek gibidir.

Şekil 2.43 *BeMyEye platform ekran görüntüsü (http-39)*

3. YÖNTEM

Türkiye'de oluşturulan CBS yayınları için ortak bir dil kullanımını hedefleyen ve bu ortak dilin oluşturulmasını ülkemizdeki CBS kullanıcılarını dahil ederek sağlayabilecek kitle-kaynaklı bir uygulama geliştirilmiştir.

3.1. Uygulamanın Geliştirilmesi

Bu bölümde uygulamanın geliştirilmesi sırasında kullanılan programlama dili, uygulaması, veritabanı gibi yapılar incelenmiştir.

3.1.1. Uygulamanın dili

İnternet ortamına geliştirilecek uygulamalar için belirli diller bulunmaktadır. Bu diller geliştiricinin tercihi ya da uygulamayı barındıracak sunucunun özelliklerine göre değişiklik göstermektedir. Çevrimiçi ortama uygulama üretebilmek için kullanılan diller;

- PHP
- .NET
- PYTHON

olarak listelenebilir. Listede belirtilen diller sunucu tarafında çalışan (Server-Side) diller olarak adlandırılırlar. Uygulama için gerekenler bu dillerle sınırlı değildir. Kullanıcı arayüzü için kullanılan diller bulunmaktadır. Bu diller ise;

- HTML
- JavaScript
- jQuery
- AngularJS
- Vue.js
- ReactJS

olarak listelenebilir. Listeler içerisinden bu araştırma için .NET, JavaScript ve jQuery seçilmiştir.

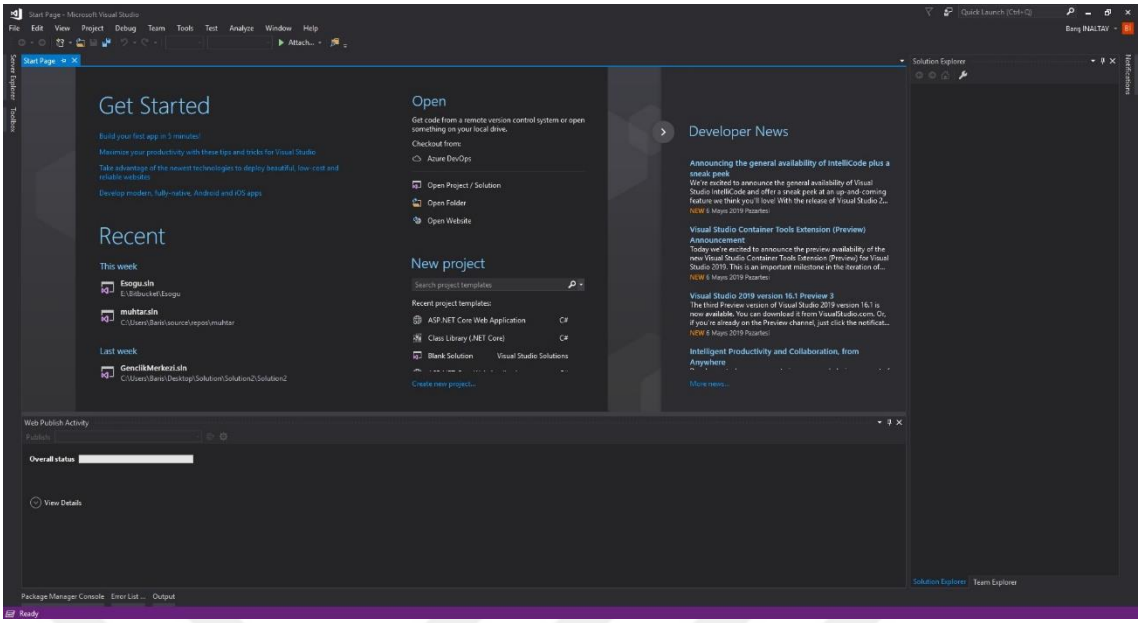
.Net geliştirilmesi için 2015 yılında yayınlanan .Net Core platformu kullanılmıştır. Kullanım olarak platform birçok dil desteği sunmaktadır. C#, F# ve Visual Basic dillerinde geliştirmeye olanak sağlamaktadır. .Net Core sunucu özelliklerinden bağımsız olarak çalışabilmektedir.

3.1.2. Geliştirme ortamı

.Net Core geliştirilmesi için bir geliştirici ortamına ihtiyaç duymaktadır. Geliştirici ortamları yazılımcıların yaptıkları işleri kolaylaştırabilmek için üretilmiş editörlerdir. Editörler yazılan dilin özelliklerine göre otomatik tamamlama, dillere göre renklendirme yaparak kodların okunmasını kolaylaştırmaktadır. Geliştirici editörlerinden bazıları aşağıda listelenmiştir.

- Visual Studio
- Visual Studio Code
- Eclipse
- JetBrains IDE
- Sublime Text
- Vim
- Notepad++

.Net Core ortamı Visual Studio ile birlikte desteklenerek çıkartılmıştır. Visual Studio, .NET Core geliştirme özelliklerini desteklemektedir (Şekil 3.1). Yazılım içerisindeki hatalar, geliştirme süreçlerinde bir betiği adım adım takip edebilme özellikleri sayesinde vazgeçilmez bir editör olmuştur.



Şekil 3.1 *Visual Studio ekran görüntüsü*

3.1.3. Veritabanı

"Veritabanı, farklı çalışmalar sırasında gereksinim duyulan birbirinden farklı, ancak birbiriyle ilişkilendirilmiş veriler topluluğudur" (Uyguçgil, vd., 2011, s. 5). Veritabanları saklanması istenen verilerin düzenli ve ilişkili bir biçimde korunduğu ve yönetildiği bir araçtır. Veritabanları farklı işler için farklı modellere evirilmiştir. Bu modeller aşağıdaki gibi listelenebilir.

- Hiyerarşik veritabanı modeli
- Ağ veritabanı modeli
- İlişkisel veritabanı modeli
- Nesneye dayalı veritabanı modeli
- Varlık-İlişki modeli

Günümüzde en çok ilişkisel veritabanı modeli kullanılmaktadır. İlişkisel veritabanı modeli nesnelerin ilişkilerini tanımlayabildiğimiz bir yapıya sahip olduğundan anlaşılması ve uygulanması kolay bir modeldir.

Veritabanı yapısına uygun kullanılan veritabanı yönetim sistemleri mevcuttur. Bu sistemler veritabanını oluşturma düzenleme gibi işlemlerin yönetilmesini sağlar, ayrıca veritabanına çeşitli eklentilerle farklı özellikler kazandırır. Günümüzde kullanılan bazı veritabanı yönetim sistemleri listelenmiştir.

- Microsoft Access
- MS-SQL
- MySQL
- IBM DB2
- Oracle
- SQLİte
- Postgresql
- NoSQL

Veritabanı yönetim sistemleri, projenin bütçesi, proje çalışanlarının bilgileri veya projenin gereksinimlerine göre belirlenir. Uygulamada kullanılan Postgresql ücretsiz bir dağıtım olduğu ve Postgis eklentisi ile CBS verilerini de barındırabildiği için seçilmiştir.

4. ARAŞTIRMA BULGULARI VE SONUÇLARI

Bu bölümde yapılan araştırmalar sonucu ortaya konulan programın veritabanı yapısı ve programlama mantığı ile ilgili çalışmalar incelenmiştir.

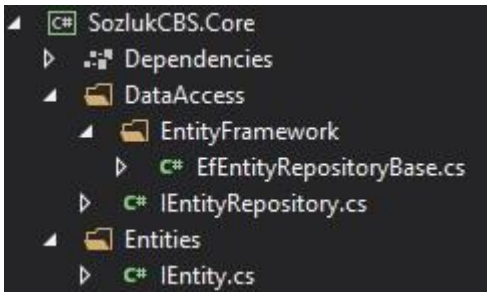
4.1. Programlama

Programın ana dili olarak .Net Core kullanılmıştır. Programlama yapılırken nesne tabanlı bir mimari olan kurumsal yapıda bir mimari kullanılmıştır. Kurumsal mimaride altyapı olarak bir boş çözüm içerisinde bulunan 5 ayrı proje bulunmaktadır. Bulunan bu projeler gerekliliklerine göre birbirlerinden kalıtım almaktadır. Bu proje katmanları

- İş katmanı (Bussiness)
- Ana katman (Core)
- Veri erişim katmanı (DataAccess)
- Varlık katmanı (Entities)
- Web arayüzü katmanı (WebUI)

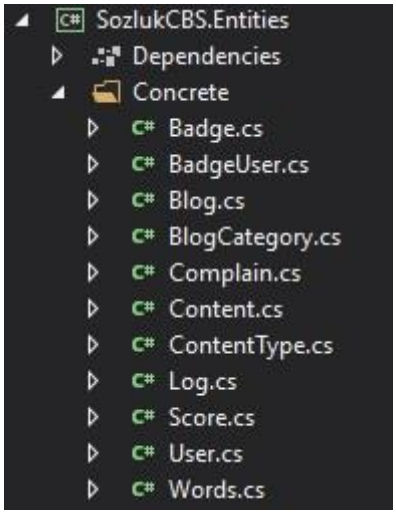
şeklinde oluşturulmuştur.

Programlama yapısında bulunan ve programlanan ilk katman ana katmandır. Ana katman içerisinde nesnelerin türetildiği bir sınıf ve nesneler üzerinde yapılacak işlemlerin tanımlandığı bir sınıf yer almaktadır (Şekil 4.1).



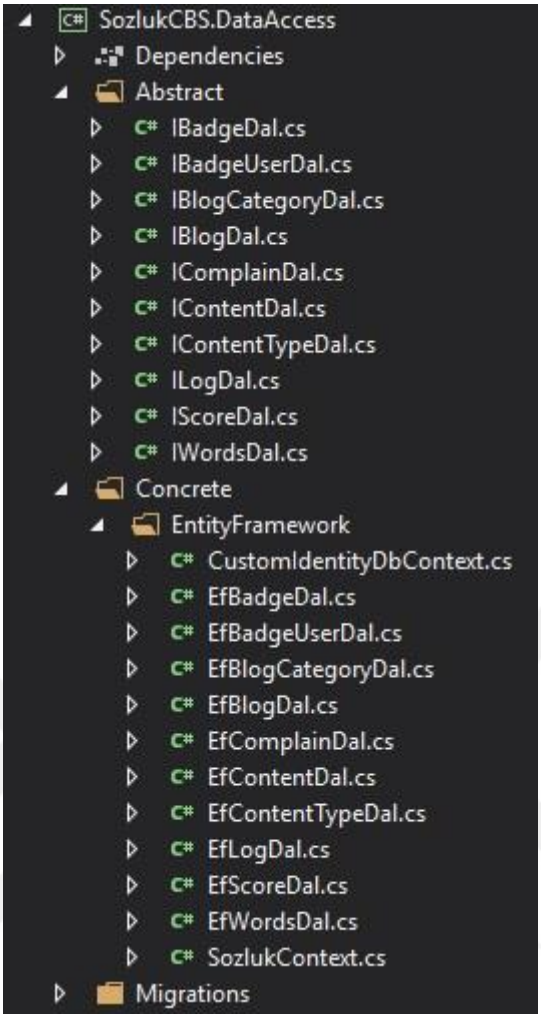
Şekil 4.1 *Ana katman sınıfları*

Nesnelerin tanımlandığı ve aynı zamanda veritabanı yapısının oluşturulduğu katman varlık katmanıdır (Şekil 4.2). Varlık katmanında nesne tabanlı programlamada kullanılan sınıf yapıları ve sınıf bağımlılıkları tanımlanmaktadır.

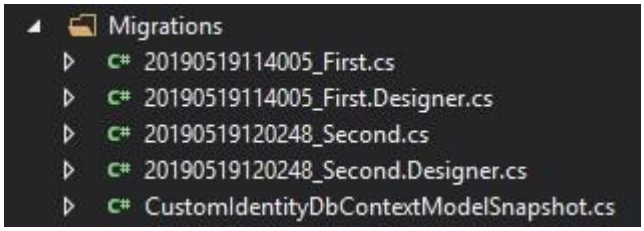


Şekil 4.2 *Varlık katmanı sınıfları*

Veri erişimi ve veritabanı işlemlerinin bulunduğu katman veri erişim katmanıdır. Bu katmana program içerisinden kullanıcıların erişmesi mümkün değildir. Kullanıcılar bir diğer katman olan iş katmanı üzerinde tanımlanan servisler ile bu katmana ulaştırılır. Veri erişim katmanında tanımlanan sınıflar "core" katmanına hangi veritabanı bağlantısının ve hangi nesnenin gönderileceğinin belirlendiği katmandır (Şekil 4.3). Ayrıca veritabanı oluşturulması varlık katmanına göre belirlenen varlıklar üzerinden bu katmanda oluşturulur. Bu katmandaki oluşturulan bu işleme "migration" adı verilir (Şekil 4.4).

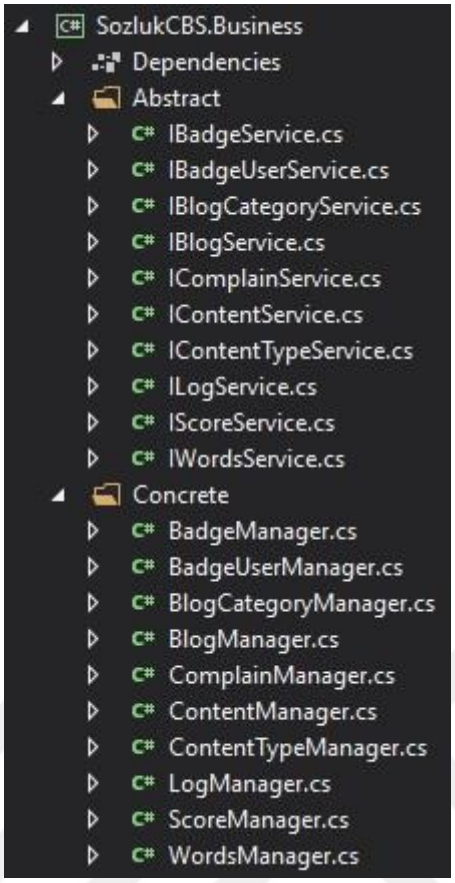


Şekil 4.3 *Veri erişim katmanı*



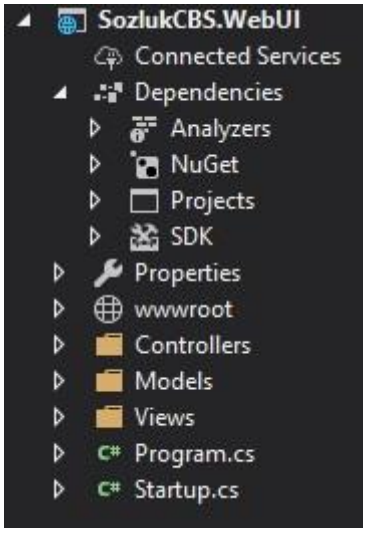
Şekil 4.4 *Migrations*

Altyapıyı oluşturan son katman olan iş katmanı kullanıcı ara yüzlerinin ulaştığı katmandır. İş katmanında çift kademeli iş bulunmaktadır ilk kademede kullanıcıların arayüz üzerinden veritabanına ulaşabilmesi için gereken fonksiyon servisleri tanımlanır. İkinci kademede ise bu fonksiyonları yöneten yönetici kademesi yer alır (Şekil 4.5).



Şekil 4.5 *İş katmanı*

Bir programın ana altyapısını oluşturan katmanlar yukarıda incelenen dört katmandan oluşmaktadır. Son katman olan web arayüzü katmanı ise isteğe bağlı olarak farklı ortamlarda kullanılabilir. Bir arayüz olarak internet sayfası, mobil uygulama veya masaüstü bir program kullanılabilir. Çalışmada kullanılan platform bir kitle kaynak uygulaması olduğundan internet sayfası olarak kullanılmıştır. İnternet sayfası içeriği olarak kullanılan web arayüzü katmanında MVC modeline uygun olarak tasarlanmış ve nesne tabanlı bir uygulama ile programlanmıştır.



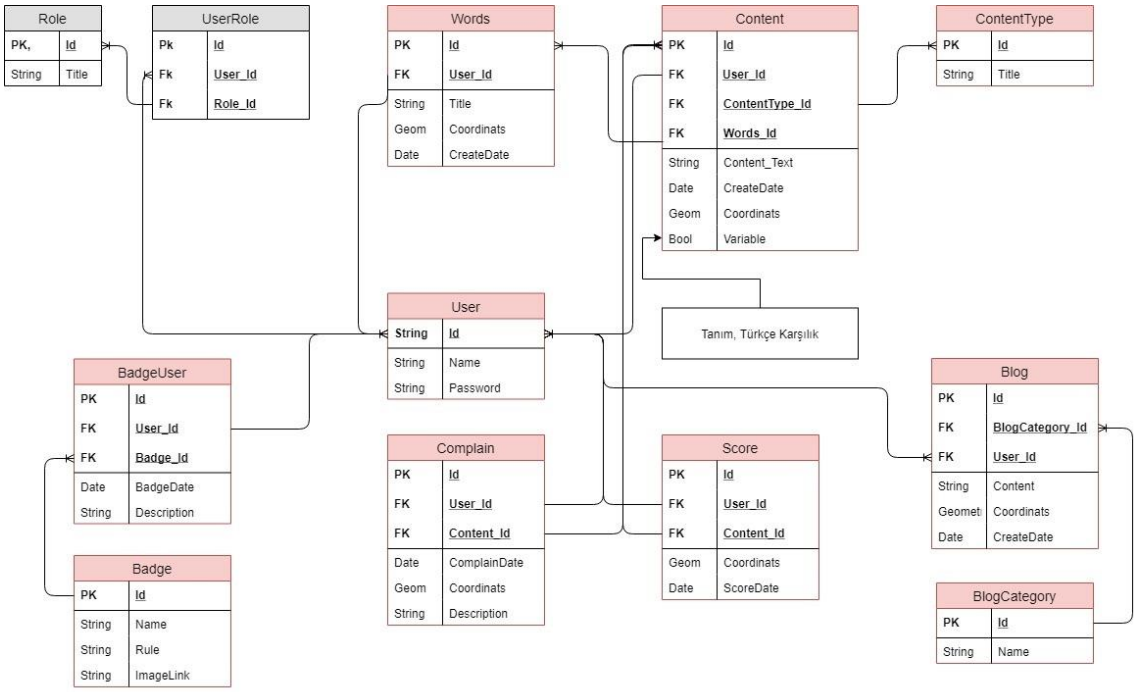
Şekil 4.6 *Web arayüzü katmanı*

4.2. Veritabanı Yapısı

Programlama için yapılacak ilk işlem veritabanı yapısının senaryolaştırılmasıdır. Senaryo hazırlandıktan sonra bu senaryoya uygun olarak veritabanı diyagramı çizilir. Çizilen diyagram üzerinden veritabanı tasarımının kullanılabilirliği ve uygunluğu denetlenebilir olmaktadır. Diyagramlar üzerinde incelenen veritabanı yine diyagramlar üzerinden düzeltilerek yaratılmaktadır. Diyagramların en büyük kullanım amacı projeye başlanmadan önce hataların en aza indirilmesidir. Araştırmada uygulanan program için veritabanı diyagramı oluşturulmuştur (Şekil 4.7). Diyagrama göre tablolar;

- **Role,** yetkilerin tutulduğu tablodur.
- **UserRole,** kullanıcı yetkilerini barındıran tablodur.
- **User,** kullanıcıların özlük bilgilerini ve hesap bilgilerini barındıran tablodur.
- **Score,** kullanıcıların yaptıkları işleme göre kazandıkları puanları barındıran tablodur.
- **Badge,** yapılacak işlemlere göre aşamaların belirlendiği rozetleri barındıran tablodur.
- **BadgeUser,** kullanıcıların kazandıkları rozetleri barındıran tablodur.
- **Words,** genel içerikte bulunan CBS kelimelerini barındıran tablodur.
- **Content,** CBS kelimelerinin tanım ve Türkçe karşılıklarını barındıran tablodur.

- **ContentType,** CBS kelimelerinin tanım ve Türkçe karşılıklarının kullanıcı tarafından oluşturulan türünü barındıran tablodur.
- **Complain,** kullanıcıların bir içerik için şikâyetlerini barındıran tablodur.
- **Blog,** belirli bir seviyeye gelen kullanıcıların oluşturabildikleri blog içeriklerini barındıran tablodur.
- **BlogCategory,** blog içeriklerinin kategorilerini barındıran tablodur.



Şekil 4.7 *CBS-Sözlük veritabanı diyagramı*

4.2.1. Katmanların oluşturulması

Bu bölümde katmanlardaki sınıfların kodları yer almaktadır.

4.2.1.1. *Ana katmanın oluşturulması*

Ana katman içerisinde iki genel klasör bulunmaktadır. İlk klasör içerisinde varlıkların ulaştığı ve üzerindeki işlemlerin tanımlandığı genel sınıflar yer almaktadır. İkinci klasör içerisinde varlıkların türediği bir "interface" yer almaktadır. İlk klasör içerisinde varlık üzerindeki ana işlemlerin tanımlandığı bir "IEntityRepository" yer almaktadır. Programlama içerisinde "EntityFramework" kullanıldığı için genel fonksiyonların tanımlandığı "EfEntityRepositoryBase" bulunmaktadır. İkinci klasör içerisinde bulunan varlıkların türediği "IEntity" bulunmaktadır.

IEntityRepository:

```
namespace SozlukCBS.Core.DataAccess
{
    public interface IEntityRepository<T> where T : class, IEntity, new()
    {
        T Get(Expression<Func<T, bool>> filter = null);
        List<T> GetList(Expression<Func<T, bool>> filter = null);
        void Add(T entity);
        void Update(T entity);
        void Delete(T entity);
    }
}
```

IEntity

```
namespace SozlukCBS.Core.Entities
{
    public interface IEntity
    {
    }
}
```

EfEntityRepositoryBase

```
namespace SozlukCBS.Core.DataAccess.EntityFramework
{
   public class EfEntityRepositoryBase<TEntity, TContext> : IEntityRepository<TEntity>
      where TEntity : class, IEntity, new()
      where TContext : DbContext, new()
  {
      public TEntity Get(Expression<Func<TEntity, bool>> filter)
      {
        using (var context = new TContext())
        {
          return context.Set<TEntity>().SingleOrDefault(filter);
        }
      }
      public List<TEntity> GetList(Expression<Func<TEntity, bool>> filter = null)
      {
         using (var context = new TContext())
         {
            return filter == null
               ? context.Set<TEntity>().ToList()
               : context.Set<TEntity>().Where(filter).ToList();
         }
       }
      public void Add(TEntity entity)
      {
         using (var context = new TContext())
         {
            var addedEntity = context.Entry(entity);
            addedEntity.State = EntityState.Added;
            context.SaveChanges();
         }
      }
      public void Update(TEntity entity)
      {
         using (var context = new TContext())
         {
            var updatedEntity = context.Entry(entity);
            updatedEntity.State = EntityState.Modified;
            context.SaveChanges();
         }
      }
      public void Delete(TEntity entity)
      {
         using (var context = new TContext())
         {
            var deleteEntity = context.Entry(entity);
            deleteEntity.State = EntityState.Deleted;
            context.SaveChanges();
         }
      }
   }
}
```

4.2.1.2. *Varlık katmanının oluşturulması*

Varlık katmanı içerisinde veritabanında bulunan tabloları oluşturan varlık sınıfları yer almaktadır. Bu sınıfların birbirleri ile olan bağlantıları tanımlanmaktadır. .Net yapısı içerisinde yer alan önceden oluşturulmuş paketlerin yer aldığı bir paket yöneticisi bulunmaktadır. Paket yöneticisi içerisinden Microsoft tarafından geliştirilen "Identity" paketi kullanılarak kullanıcı işlemleri ve bu kullanıcıların veritabanı yapıları oluşturulmuştur. Bu nedenle Varlık katmanında "Role" ve "RoleUser" varlıklarının oluşturulmasına gerek yoktur. "User" varlığının içerisine "Identity" paketi içerisinde var olmayan bir sütun eklemek istediğimiz için "IdentityUser" sınıfı içerisinden türetilen bir "User" varlığı bulunmalıdır. "User" varlığı içerisinde bu varlık ile bağlantısı olan diğer varlıkların ulaşabilmesi için birer koleksiyon tanımlanmıştır.

User:

```
namespace SozlukCBS.Entities.Concrete
{
    public class User : IdentityUser
    {
        public string Name { get; set; }

        public virtual ICollection<BadgeUser> BadgeUser { get; set; }
        public virtual ICollection<Blog> Blog { get; set; }
        public virtual ICollection<Complain> Complain { get; set; }
        public virtual ICollection<Content> Content { get; set; }
        public virtual ICollection<Score> Score { get; set; }
        public virtual ICollection<Words> Words { get; set; }
    }
}
```

Score:

```
namespace SozlukCBS.Entities.Concrete
{
    [Table("Score", Schema = "public")]
    public class Score : IEntity
    {
        [Key, DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Display(Name = "YerX")]
        public double CoordX { get; set; }

        [Display(Name = "YerY")]
        public double CoordY { get; set; }

        [Display(Name = "Oluşturulma Zamanı")]
        public DateTime ScoreDate { get; set; }

        [Display(Name = "Oluşturan")]
        public string User_Id { get; set; }
        [ForeignKey("User_Id"), Display(Name = "Oluşturan")]
        public virtual User CreateUser { get; set; }

        [Display(Name = "İçerik")]
        public int Content_Id { get; set; }
        [ForeignKey("Content_Id"), Display(Name = "İçerik")]
        public virtual Content Content { get; set; }
    }
}
```

Badge:

```
namespace SozlukCBS.Entities.Concrete
{
    [Table("Badge", Schema = "public")]
    public class Badge : IEntity
    {
        [Key, DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Display(Name = "Rozet")]
        public string Name { get; set; }

        [Display(Name = "Fotoğraf")]
        public string ImageLink { get; set; }

        [Display(Name = "Kural")]
        public string Rule { get; set; }

        public virtual ICollection<BadgeUser> BadgeUser { get; set; }
    }
}
```

BadgeUser:

```
namespace SozlukCBS.Entities.Concrete
{
    [Table("BadgeUser", Schema = "public")]
    public class BadgeUser : IEntity
    {
        [Key, DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Display(Name = "Oluşturan")]
        public string User_Id { get; set; }
        [ForeignKey("User_Id"), Display(Name = "Oluşturan")]
        public virtual User CreateUser { get; set; }

        [Display(Name = "Rozet")]
        public int Badge_Id { get; set; }
        [ForeignKey("Badge_Id"), Display(Name = "Rozet")]
        public virtual Badge Badge { get; set; }

        [Display(Name = "Açıklama")]
        public string Description { get; set; }

        [Display(Name = "Oluşturulma Zamanı")]
        public DateTime BadgeDate { get; set; }
    }
}
```

Words:

```
namespace SozlukCBS.Entities.Concrete
{
    [Table("Words", Schema = "public")]
    public class Words : IEntity
    {
        [Key, DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Display(Name = "Kelime")]
        public string Title { get; set; }

        [Display(Name = "YerX")]
        public double CoordX { get; set; }

        [Display(Name = "YerY")]
        public double CoordY { get; set; }

        [Display(Name = "Oluşturulma Zamanı")]
        public DateTime CreateDate { get; set; }

        [Display(Name = "Oluşturan")]
        public string User_Id { get; set; }
        [ForeignKey("User_Id"), Display(Name = "Oluşturan")]
        public virtual User CreateUser { get; set; }

        public virtual ICollection<Content> Content { get; set; }
    }
}
```

Content:

```
namespace SozlukCBS.Entities.Concrete
{
    [Table("Content", Schema = "public")]
    public class Content : IEntity
    {
        [Key, DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Display(Name = "İçerik")]
        public string Text { get; set; }

        [Display(Name = "YerX")]
        public double CoordX { get; set; }

        [Display(Name = "YerY")]
        public double CoordY { get; set; }

        [Display(Name = "Oluşturulma Zamanı")]
        public DateTime CreateDate { get; set; }

        [Display(Name = "Tür")]
        public bool Variable { get; set; }

        [Display(Name = "Oluşturan")]
        public string User_Id { get; set; }
        [ForeignKey("User_Id"), Display(Name = "Oluşturan")]
        public virtual User CreateUser { get; set; }

        [Display(Name = "İçerik")]
        public int ContentType_Id { get; set; }
        [ForeignKey("ContentType_Id"), Display(Name = "İçerik")]
        public virtual ContentType ContentType { get; set; }

        [Display(Name = "Kelime")]
        public int Words_Id { get; set; }
        [ForeignKey("Words_Id"), Display(Name = "Kelime")]
        public virtual Words Words { get; set; }

        public virtual ICollection<Complain> Complain { get; set; }
        public virtual ICollection<Score> Score { get; set; }
    }
}
```

ContentType:

```
namespace SozlukCBS.Entities.Concrete
{
    [Table("ContentType", Schema = "public")]
    public class ContentType : IEntity
    {
        [Key, DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Display(Name = "Tür")]
        public string Title { get; set; }

        public virtual ICollection<Content> Content { get; set; }
    }
}
```

Complain:

```
namespace SozlukCBS.Entities.Concrete
{
    [Table("Complain", Schema = "public")]
    public class Complain : IEntity
    {
        [Key, DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Display(Name = "Açıklama")]
        public string Description { get; set; }

        [Display(Name = "YerX")]
        public double CoordX { get; set; }

        [Display(Name = "YerY")]
        public double CoordY { get; set; }

        [Display(Name = "Oluşturulma Zamanı")]
        public DateTime CreateDate { get; set; }

        [Display(Name = "Oluşturan")]
        public string User_Id { get; set; }
        [ForeignKey("User_Id"), Display(Name = "Oluşturan")]
        public virtual User CreateUser { get; set; }

        [Display(Name = "İçerik")]
        public int Content_Id { get; set; }
        [ForeignKey("Content_Id"), Display(Name = "İçerik")]
        public virtual Content Content { get; set; }
    }
}
```

Blog:

```
namespace SozlukCBS.Entities.Concrete
{
    [Table("Blog", Schema = "public")]
    public class Blog : IEntity
    {
        [Key, DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Display(Name = "İçerik")]
        public string Content { get; set; }

        [Display(Name = "YerX")]
        public double CoordX { get; set; }

        [Display(Name = "YerY")]
        public double CoordY { get; set; }

        [Display(Name = "Oluşturulma Zamanı")]
        public DateTime CreateDate { get; set; }

        [Display(Name = "Kategori")]
        public int BlogCategory_Id { get; set; }
        [ForeignKey("BlogCategory_Id"), Display(Name = "Kategori")]
        public virtual BlogCategory BlogCategory { get; set; }

        [Display(Name = "Oluşturan")]
        public string User_Id { get; set; }
        [ForeignKey("User_Id"), Display(Name = "Oluşturan")]
        public virtual User CreateUser { get; set; }

    }
}
```

BlogCategory:

```
namespace SozlukCBS.Entities.Concrete
{
    [Table("BlogCategory", Schema = "public")]
    public class BlogCategory : IEntity
    {
        [Key, DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id { get; set; }

        [Display(Name = "Kategori")]
        public string Name { get; set; }

        public virtual ICollection<Blog> Blog { get; set; }
    }
}
```

4.2.1.3. *Veri erişim katmanının oluşturulması*

Veri erişim katmanı içerisinde iki ana klasör bulunmaktadır. İlk klasör içerisinde Ana katmandaki "IEntityRepository" sınıfından türetilen "interfaceler" yer almaktadır. Bu "interfaceler" içerisinde ana katmanda olmayan fonksiyonları tanımlayarak özel fonksiyonlar yaratılabilmektedir. İkinci klasör içerisinde program içerisinde "EntityFramework" kullanıldığından bu yapıya özgü fonksiyonların oluşturulabildiği sınıflar yer almaktadır. Bu sınıflar içerisine ilk klasör içerisinde bulunan "interfacelerde" tanımlanan fonksiyonların çalışma prensipleri bulunmaktadır.

Veri erişim katmanında bulunan bir diğer sınıf ise veritabanı bağlantısını sağlayan Bağlam (Context) sınıflarıdır. Bu bağlam sınıfları içerisinde veritabanına bağlantı sağlayan bağlantı cümleciği yer almaktadır. Cümlecik veritabanın yerinden, bağlantı portundan, adından, kullanıcı adından ve şifresinden oluşmaktadır. Çalışmadaki program için "Identity" paketi kullanıldığından iki adet bağlam bulunmaktadır. İlk bağlam Varlık katmanında oluşturulan tabloların bağlantısını sağlarken ikinci bağlam "Identity" paketi tarafından oluşturulan tabloların bağlantısını sağlamaktadır.

.Net içerisinde bulunan bir diğer özellik olarak varlık sınıfı ve bu sınıfın veritabanı bağlantı özelliklerini içeren veri erişim katmanı doğru olarak yazıldıktan sonra kullanılan veritabanından bağımsız olarak tabloların ve veritabanı yapılarının otomatik oluşumunu sağlayan "CodeFirst" bulunmaktadır. Bu özellikte

add-migration –context bağlam_sınıfının_adı

şeklinde bir kod ile veri erişim katmanına veritabanını oluşturan SQL cümlecikleri oluşmaktadır. Oluşan bu cümlecikler

update-database –context bağlam_sınıfının_adı

kodu ile veritabanında çalıştırılmaktadır. Oluşan bu "migrationlar" veri erişim katmanı içerisinde saklanmaktadır.

Veri erişim katmanındaki "interfaceler" aşağıdaki gibidir:

IBadgeDal:

```
namespace SozlukCBS.DataAccess.Abstract
{
   public interface IBadgeDal : IEntityRepository<Badge>
   {
   }
}
```

IBadgeUserDal

```
namespace SozlukCBS.DataAccess.Abstract
{
   public interface IBadgeUserDal : IEntityRepository<BadgeUser>
   {
      IEnumerable<BadgeUser> BadgewithUser(string userId);
   }
}
```

IBlogCategoryDal

```
namespace SozlukCBS.DataAccess.Abstract
{
   public interface IBlogCategoryDal : IEntityRepository<BlogCategory>
   {
   }
}
```

IBlogDal

```
namespace SozlukCBS.DataAccess.Abstract
{
   public interface IBlogDal : IEntityRepository<Blog>
   {
      IEnumerable<Blog> BlogFull();
      IEnumerable<Blog> BlogwithUser(string userId);
      IEnumerable<Blog> BlogId(int Id);
   }
}
```

IComplainDal

```
namespace SozlukCBS.DataAccess.Abstract
{
   public interface IComplainDal : IEntityRepository<Complain>
   {
   }
}
```

IContentDal

```
namespace SozlukCBS.DataAccess.Abstract
{
   public interface IContentDal : IEntityRepository<Content>
   {
      IEnumerable<Content> ContentWord(int wordId);
      IEnumerable<Content> ContentwUser(string userId);
      IEnumerable<Content> ContentFull();
   }
}
```

IContentTypeDal

```
namespace SozlukCBS.DataAccess.Abstract
{
   public interface IContentTypeDal : IEntityRepository<ContentType>
   {
   }
}
```

IScoreDal

```
namespace SozlukCBS.DataAccess.Abstract
{
   public interface IScoreDal : IEntityRepository<Score>
   {
   }
}
```

IWordsDal

```
namespace SozlukCBS.DataAccess.Abstract
{
   public interface IWordsDal : IEntityRepository<Words>
   {
      IEnumerable<Words> GetwInd(char Ind);
      IEnumerable<Words> GetFulls();
   }
}
```

Veri erişim katmanındaki “EntityFramework” sınıfları aşağıdaki gibidir:

EfBadgeDal

```
namespace SozlukCBS.DataAccess.Concrete.EntityFramework
{
   public class EfBadgeDal : EfEntityRepositoryBase<Badge, SozlukContext>, IBadgeDal
   {
   }
}
```

EfBadgeUserDal

```
namespace SozlukCBS.DataAccess.Concrete.EntityFramework
{
   Public class EfBadgeUserDal : EfEntityRepositoryBase<BadgeUser, SozlukContext>,
IBadgeUserDal
   {
      public IEnumerable<BadgeUser> BadgewithUser(string userId)
      {
         using (var context = new SozlukContext())
         {
            return context.BadgeUser
               .Where(x=>x.User_Id==userId)
               .Select(x => x)
               .Include(x => x.Badge)
               .Include(x => x.CreateUser)
               .ToList();
         }
      }
   }
}
```

EfBlogCategoryDal

```
namespace SozlukCBS.DataAccess.Concrete.EntityFramework
{
   public class EfBlogCategoryDal : EfEntityRepositoryBase<BlogCategory, SozlukContext>,
IBlogCategoryDal
   {
   }
}
```

EfBlogDal

```
namespace SozlukCBS.DataAccess.Concrete.EntityFramework
{
    public class EfBlogDal : EfEntityRepositoryBase<Blog, SozlukContext>, IBlogDal
    {
        public IEnumerable<Blog> BlogFull()
        {
            using (var context = new SozlukContext())
            {
                return context.Blog
                    .Select(x => x)
                    .Include(x => x.BlogCategory)
                    .Include(x => x.CreateUser)
                    .ToList();
            }
        }

        public IEnumerable<Blog> BlogwithUser(string userId)
        {
            using (var context = new SozlukContext())
            {
                return context.Blog
                    .Where(x=>x.User_Id==userId)
                    .Select(x => x)
                    .Include(x => x.BlogCategory)
                    .Include(x => x.CreateUser)
                    .ToList();
            }
        }

        public IEnumerable<Blog> BlogId(int Id)
        {
            using (var context = new SozlukContext())
            {
                return context.Blog
                    .Where(x => x.Id == Id)
                    .Select(x => x)
                    .Include(x => x.BlogCategory)
                    .Include(x => x.CreateUser)
                    .ToList();
            }
        }
    }
}
```

EfComplainDal

```
namespace SozlukCBS.DataAccess.Concrete.EntityFramework
{
    public class EfComplainDal : EfEntityRepositoryBase<Complain, SozlukContext>, IComplainDal
    {
    }
}
```

EfContentDal

```
namespace SozlukCBS.DataAccess.Concrete.EntityFramework
{
   public class EfContentDal : EfEntityRepositoryBase<Content, SozlukContext>, IContentDal
   {
      public IEnumerable<Content> ContentWord(int wordId)

      {
         using (var context = new SozlukContext())
         {
            return context.Content
               .Select(x => x)
               .Where(x => x.Words_Id == wordId)
               .Include(x => x.Words)
               .Include(x => x.CreateUser)
               .ToList();
         }
      }

      public IEnumerable<Content> ContentwUser(string userId)

      {
         using (var context = new SozlukContext())
         {
            return context.Content
               .Select(x => x)
               .Where(x => x.User_Id == userId)
               .Include(x => x.Words)
               .Include(x => x.CreateUser)
               .ToList();
         }
      }

      public IEnumerable<Content> ContentFull()

      {
         using (var context = new SozlukContext())
         {
            return context.Content
               .Select(x => x)
               .Include(x => x.Words)
               .Include(x => x.CreateUser)
               .ToList();
         }
      }
   }
}
```

EfContentTypeDal

```
namespace SozlukCBS.DataAccess.Concrete.EntityFramework
{
   public class EfContentTypeDal : EfEntityRepositoryBase<ContentType, SozlukContext>, IContentTypeDal
   {
   }
}
```

EfScoreDal

```
namespace SozlukCBS.DataAccess.Concrete.EntityFramework
{
   public class EfScoreDal : EfEntityRepositoryBase<Score, SozlukContext>, IScoreDal
   {
   }
}
```

EfWordsDal

```
namespace SozlukCBS.DataAccess.Concrete.EntityFramework
{
   public class EfWordsDal : EfEntityRepositoryBase<Words, SozlukContext>, IWordsDal
   {
      public IEnumerable<Words> GetwInd(char Ind)
      {
         using (var context = new SozlukContext())
         {
            return context.Words
               .Select(x => x)
               .Where(x=>x.Title.StartsWith(Ind))
               .Include(x=>x.Content)
               .ToList();
         }
      }

      public IEnumerable<Words> GetFulls()
      {
         using (var context = new SozlukContext())
         {
            return context.Words
               .Select(x => x)
               .Include(x => x.Content)
               .ToList();
         }
      }
   }
}
```

Veri erişim katmanındaki bağlam sınıfları aşağıdaki gibidir:

SozlukContext

```
namespace SozlukCBS.DataAccess.Concrete.EntityFramework
{
   public class SozlukContext:DbContext
   {
      protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
      {

optionsBuilder.UseNpgsql("Host=host_adresi;Database=veritabanı_adı;Username=kullanıcı_adı;Pass
word=şifre",
            o => o.UseNetTopologySuite());
      }
      protected override void OnModelCreating(ModelBuilder builder)
      {
         builder.HasPostgresExtension("postgis");
      }
      public DbSet<Badge> Badge { get; set; }
      public DbSet<BadgeUser> BadgeUser { get; set; }
      public DbSet<Blog> Blog { get; set; }
      public DbSet<BlogCategory> BlogCategory { get; set; }
      public DbSet<Complain> Complain { get; set; }
      public DbSet<Content> Content { get; set; }
      public DbSet<ContentType> ContentType { get; set; }
      public DbSet<Log> Log { get; set; }
      public DbSet<Score> Score { get; set; }
      public DbSet<Words> Words { get; set; }
   }
}
```

CustomIdentityDbContext

```
namespace SozlukCBS.DataAccess.Concrete.EntityFramework
{
   public class CustomIdentityDbContext : IdentityDbContext<User>
   {
      public CustomIdentityDbContext(DbContextOptions<CustomIdentityDbContext> options) :
base(options)
      {
      }
      protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
      {
optionsBuilder.UseNpgsql("Host=host_adresi;Database=veritabanı_adı;Username=kullanıcı_adı;Pass
word=şifre",
            o => o.UseNetTopologySuite());
      }
      protected override void OnModelCreating(ModelBuilder builder)
      {
         base.OnModelCreating(builder);
         builder.Entity<User>().ToTable("User");
         builder.Entity<IdentityRole>().ToTable("Role");
      }
   }
}
```

4.2.1.4. *İş katmanının oluşturulması*

İş katmanı içerisinde iki ana klasör yer almaktadır. İlk klasör içerisinde kullanıcı arayüzünden ulaşabildiğimiz sanal "interfaceler" yer almaktadır. Bu "interfaceler" kullanıcı arayüzünde yapılacak işlemlerin tanımlandığı servislerden oluşmaktadır. Servis katmanı kullanıcıların ulaşabildiği katmandır. Bu yüzden bu katman içerisinde veritabanı bağlantıları bulanmamaktadır. İş katmanını oluşturan bir diğer klasörde ise servislerde tanımlı fonksiyonların yapacağı işlerin tanımlandığı yönetici sınıfları yer almaktadır. Servislere ulaşabilen kullanıcılar katmandaki yöneticilere ulaşamamaktadırlar. Yönetici sınıfları fonksiyonların belirttiği doğrultuda gelen verileri veri erişim katmanına göndermektedirler.

İş katmanındaki servisler aşağıdaki gibidir:

IBadgeService

```
namespace SozlukCBS.Business.Abstract
{
   public interface IBadgeService
   {
      List<Badge> GetAll();
      void Add(Badge badge);
      void Update(Badge badge);
      void Delete(int badgeId);
      Badge GetById(int badgeId);
   }
}
```

IBadgeUserService

```
namespace SozlukCBS.Business.Abstract
{
   public interface IBadgeUserService
   {
      List<BadgeUser> GetAll();
      void Add(BadgeUser badgeuser);
      void Update(BadgeUser badgeuser);
      void Delete(int badgeuserId);
      BadgeUser GetById(int badgeuserId);
      IEnumerable<BadgeUser> BadgebyUser(string userId);
   }
}
```

IBlogCategoryService

```
namespace SozlukCBS.Business.Abstract
{
   public interface IBlogCategoryService
   {
      List<BlogCategory> GetAll();
      void Add(BlogCategory blogcategory);
      void Update(BlogCategory blogcategory);
      void Delete(int blogcategoryId);
      BlogCategory GetById(int blogcategoryId);
   }
}
```

IBlogService

```
namespace SozlukCBS.Business.Abstract
{
    public interface IBlogService
    {
        List<Blog> GetAll();
        List<Blog> GetByCategory(int blogcategoryId);
        void Add(Blog blog);
        void Update(Blog blog);
        void Delete(int blogId);
        Blog GetById(int blogId);
        IEnumerable<Blog> BlogwUser(string userId);
        IEnumerable<Blog> BlogbyId(int Id);
        IEnumerable<Blog> BlogFulls();
    }
}
```

IComplainService

```
namespace SozlukCBS.Business.Abstract
{
    public interface IComplainService
    {
        List<Complain> GetAll();
        void Add(Complain complain);
        void Update(Complain complain);
        void Delete(int complainId);
        Complain GetById(int complainId);
    }
}
```

IContentService

```
namespace SozlukCBS.Business.Abstract
{
    public interface IContentService
    {
        List<Content> GetAll();
        void Add(Content content);
        void Update(Content content);
        void Delete(int contentId);
        Content GetById(int contentId);
        IEnumerable<Content> ContentbybWord(int wordId);
        IEnumerable<Content> ContentbybUser(string userId);
        IEnumerable<Content> ContentFulls();
    }
}
```

IContentTypeService

```
namespace SozlukCBS.Business.Abstract
{
    public interface IContentTypeService
    {
        List<ContentType> GetAll();
        void Add(ContentType contenttype);
        void Update(ContentType contenttype);
        void Delete(int contenttypeId);
        ContentType GetById(int contenttypeId);
    }
}
```

IScoreService

```
namespace SozlukCBS.Business.Abstract
{
    public interface IScoreService
    {
        List<Score> GetAll();
        void Add(Score score);
        void Update(Score score);
        void Delete(int scoreId);
        Score GetById(int scoreId);
    }
}
```

IWordsService

```
namespace SozlukCBS.Business.Abstract
{
    public interface IWordsService
    {
        List<Words> GetAll();
        void Add(Words words);
        void Update(Words words);
        void Delete(int wordsId);
        Words GetById(int wordsId);
        IEnumerable<Words> GetbyInd(char Ind);
        IEnumerable<Words> GetFull();
    }
}
```

İş katmanındaki yönetici sınıfları aşağıdaki gibidir:

BadgeManager

```
namespace SozlukCBS.Business.Concrete
{
    public class BadgeManager : IBadgeService
    {
        private IBadgeDal _badgeDal;
        public BadgeManager(IBadgeDal badgeDal)
        {
            _badgeDal = badgeDal;
        }
        public List<Badge> GetAll()
        {
            return _badgeDal.GetList();
        }
        public void Add(Badge badge)
        {
            _badgeDal.Add(badge);
        }
        public void Update(Badge badge)
        {
            _badgeDal.Update(badge);
        }
        public void Delete(int badgeId)
        {
            _badgeDal.Delete(new Badge { Id = badgeId });
        }
        public Badge GetById(int badgeId)
        {
            return _badgeDal.Get(p => p.Id == badgeId);
        }
    }
}
```

BadgeUserManager

```
namespace SozlukCBS.Business.Concrete
{
    public class BadgeUserManager : IBadgeUserService
    {
        private IBadgeUserDal _badgeuserDal;
        public BadgeUserManager(IBadgeUserDal badgeuserDal)
        {
            _badgeuserDal = badgeuserDal;
        }
        public List<BadgeUser> GetAll()
        {
            return _badgeuserDal.GetList();
        }
        public void Add(BadgeUser badgeuser)
        {
            _badgeuserDal.Add(badgeuser);
        }
        public void Update(BadgeUser badgeuser)
        {
            _badgeuserDal.Update(badgeuser);
        }
        public void Delete(int badgeuserId)
        {
            _badgeuserDal.Delete(new BadgeUser { Id = badgeuserId });
        }
        public BadgeUser GetById(int badgeuserId)
        {
            return _badgeuserDal.Get(p => p.Id == badgeuserId);
        }
        public IEnumerable<BadgeUser> BadgebyUser(string userId)
        {
            return _badgeuserDal.BadgewithUser(userId);
        }
    }
}
```

BlogCategoryManager

```
namespace SozlukCBS.Business.Concrete
{
    public class BlogCategoryManager : IBlogCategoryService
    {
        private IBlogCategoryDal _blogcategoryDal;
        public BlogCategoryManager(IBlogCategoryDal blogcategoryDal)
        {
            _blogcategoryDal = blogcategoryDal;
        }
        public List<BlogCategory> GetAll()
        {
            return _blogcategoryDal.GetList();
        }
        public void Add(BlogCategory blogcategory)
        {
            _blogcategoryDal.Add(blogcategory);
        }
        public void Update(BlogCategory blogcategory)
        {
            _blogcategoryDal.Update(blogcategory);
        }
        public void Delete(int blogcategoryId)
        {
            _blogcategoryDal.Delete(new BlogCategory { Id = blogcategoryId });
        }
        public BlogCategory GetById(int blogcategoryId)
        {
            return _blogcategoryDal.Get(p => p.Id == blogcategoryId);
        }
    }
}
```

BlogManager

```
namespace SozlukCBS.Business.Concrete
{
    public class BlogManager : IBlogService
    {
        private IBlogDal _blogDal;

        public BlogManager(IBlogDal blogDal)
        {
            _blogDal = blogDal;
        }
        public List<Blog> GetAll()
        {
            return _blogDal.GetList();
        }
        public List<Blog> GetByCategory(int blogId)
        {
            return _blogDal.GetList(p => p.BlogCategory_Id == blogId || blogId == 0);
        }
        public void Add(Blog blog)
        {
            _blogDal.Add(blog);
        }
        public void Update(Blog blog)
        {
            _blogDal.Update(blog);
        }
        public void Delete(int blogId)
        {
            _blogDal.Delete(new Blog { Id = blogId });
        }
        public Blog GetById(int blogId)
        {
            return _blogDal.Get(p => p.Id == blogId);
        }
        public IEnumerable<Blog> BlogwUser(string userId)
        {
            return _blogDal.BlogwithUser(userId);
        }
        public IEnumerable<Blog> BlogFulls()
        {
            return _blogDal.BlogFull();
        }
        public IEnumerable<Blog> BlogbyId(int Id)
        {
            return _blogDal.BlogId(Id);
        }
    }
}
```

ComplainManager

```
namespace SozlukCBS.Business.Concrete
{
    public class ComplainManager : IComplainService
    {
        private IComplainDal _complainDal;
        public ComplainManager(IComplainDal complainDal)
        {
            _complainDal = complainDal;
        }
        public List<Complain> GetAll()
        {
            return _complainDal.GetList();
        }
        public void Add(Complain complain)
        {
            _complainDal.Add(complain);
        }
        public void Update(Complain complain)
        {
            _complainDal.Update(complain);
        }
        public void Delete(int complainId)
        {
            _complainDal.Delete(new Complain { Id = complainId });
        }
        public Complain GetById(int complainId)
        {
            return _complainDal.Get(p => p.Id == complainId);
        }
    }
}
```

ContentManager

```
namespace SozlukCBS.Business.Concrete
{
    public class ContentManager : IContentService
    {
        private IContentDal _contentDal;

        public ContentManager(IContentDal contentDal)
        {
            _contentDal = contentDal;
        }
        public List<Content> GetAll()
        {
            return _contentDal.GetList();
        }
        public List<Content> GetByCategory(int contentId)
        {
            return _contentDal.GetList(p => p.ContentType_Id == contentId || contentId == 0);
        }
        public void Add(Content content)
        {
            _contentDal.Add(content);
        }
        public void Update(Content content)
        {
            _contentDal.Update(content);
        }
        public void Delete(int contentId)
        {
            _contentDal.Delete(new Content { Id = contentId });
        }
        public Content GetById(int contentId)
        {
            return _contentDal.Get(p => p.Id == contentId);
        }
        public IEnumerable<Content> ContentbybWord(int wordId)
        {
            return _contentDal.ContentWord(wordId);
        }
        public IEnumerable<Content> ContentFulls()
        {
            return _contentDal.ContentFull();

        }
        public IEnumerable<Content> ContentbybUser(string userId)
        {
            return _contentDal.ContentwUser(userId);
        }
    }
}
```

ContentTypeManager

```
namespace SozlukCBS.Business.Concrete
{
    public class ContentTypeManager : IContentTypeService
    {
        private IContentTypeDal _contenttypeDal;
        public ContentTypeManager(IContentTypeDal contenttypeDal)
        {
            _contenttypeDal = contenttypeDal;
        }
        public List<ContentType> GetAll()
        {
            return _contenttypeDal.GetList();
        }
        public void Add(ContentType contenttype)
        {
            _contenttypeDal.Add(contenttype);
        }
        public void Update(ContentType contenttype)
        {
            _contenttypeDal.Update(contenttype);
        }
        public void Delete(int contenttypeId)
        {
            _contenttypeDal.Delete(new ContentType { Id = contenttypeId });
        }
        public ContentType GetById(int contenttypeId)
        {
            return _contenttypeDal.Get(p => p.Id == contenttypeId);
        }
    }
}
```

ScoreManager

```
namespace SozlukCBS.Business.Concrete
{
    public class ScoreManager : IScoreService
    {
        private IScoreDal _scoreDal;
        public ScoreManager(IScoreDal scoreDal)
        {
            _scoreDal = scoreDal;
        }
        public List<Score> GetAll()
        {
            return _scoreDal.GetList();
        }
        public void Add(Score score)
        {
            _scoreDal.Add(score);
        }
        public void Update(Score score)
        {
            _scoreDal.Update(score);
        }
        public void Delete(int scoreId)
        {
            _scoreDal.Delete(new Score { Id = scoreId });
        }
        public Score GetById(int scoreId)
        {
            return _scoreDal.Get(p => p.Id == scoreId);
        }
    }
}
```

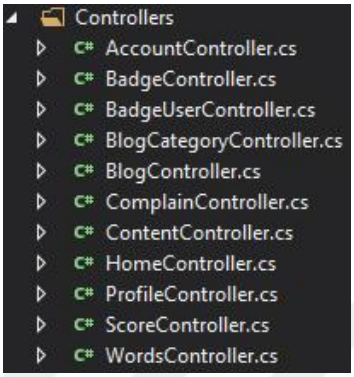
WordsManager

```
namespace SozlukCBS.Business.Concrete
{
    public class WordsManager : IWordsService
    {
        private IWordsDal _wordsDal;
        public WordsManager(IWordsDal wordsDal)
        {
            _wordsDal = wordsDal;
        }
        public List<Words> GetAll()
        {
            return _wordsDal.GetList();
        }
        public void Add(Words words)
        {
            _wordsDal.Add(words);
        }
        public void Update(Words words)
        {
            _wordsDal.Update(words);
        }
        public void Delete(int wordsId)
        {
            _wordsDal.Delete(new Words { Id = wordsId });
        }
        public Words GetById(int wordsId)
        {
            return _wordsDal.Get(p => p.Id == wordsId);
        }
        public IEnumerable<Words> GetbyInd(char Ind)
        {
            return _wordsDal.GetwInd(Ind);
        }
        public IEnumerable<Words> GetFull()
        {
            return _wordsDal.GetFulls();
        }
    }
}
```

4.2.1.5. *Web arayüzü katmanının oluşturulması*

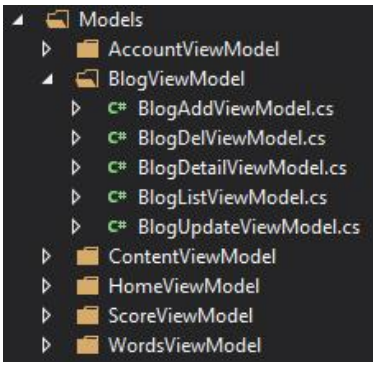
Web arayüzü katmanı uygulamanın son ve görünen katmanıdır. Bu katman kullanıcının direkt olarak etkileşime geçtiği tasarımın yer aldığı kısımdır. Web arayüzü katmanı genel olarak dört temel klasör yapısında oluşturulmaktadır. Programlamada MVC modeli kullanıldığından öncelikle "Models", "Controllers", "Views" klasörlerinden ve son olarak tasarımın temel dosyalarının yer aldığı "wwwroot" klasöründen oluşmaktadır. Ayrıca bu katmanda programın ilk açılışta çalışmasını sağlayacak "Startup" sınıfı da bulunmaktadır.

“Controller” klasörü içerisinde kullanıcı tarafından kullanılacak sayfaların servislere ulaşması için gereken fonksiyon ve işlemler bulunmaktadır. Buradaki kontroller içerisinde kullanıcının arayüzde bulunan formlarda yapacağı işlemler tanımlanmaktadır.



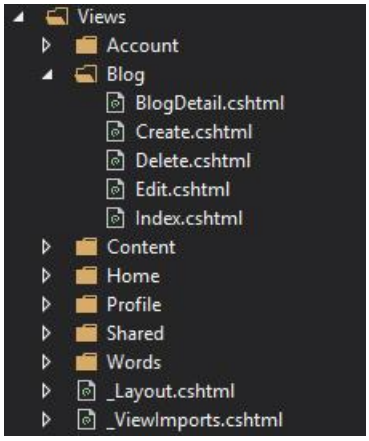
Şekil 4.8 *Controller klasörü ekran görüntüsü*

“Models” klasörü içerisinde kullanıcı arayüzü içerisinde görüntülenecek ve formlardaki dönüşlerde kullanılacak nesne ve varlıkların bir tek model olarak yönlendirilmesini sağlayan görüntüleme modelleri yer almaktadır.



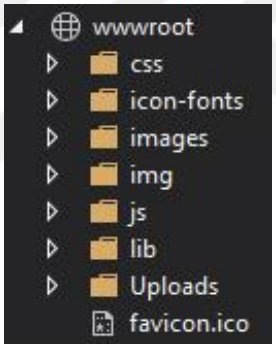
Şekil 4.9 *Models klasörü ekran görüntüsü*

“Views” klasörü içerisinde “controllers” içerisinde yer alan kontrollerin isimlerine uygun olarak kullanıcının görüntülediği tasarım sayfaları bulunmaktadır.



Şekil 4.10 *Views klasörü ekran görüntüsü*

Son olarak "wwwroot" klasörü içerisinde tasarımın yapıtaşları olan stil dosyaları (css) ve "Javascript" dosyaları yer almaktadır. Ayrıca kullanıcılar tarafından yüklenecek fotoğraf ve resimler bu klasör altına gönderilmektedir.



Şekil 4.11 *wwwroot klasörü ekran görüntüsü*

Programın tasarım kısmı için colorib tarafından tasarlanılan ve sunulan "Arcade" teması kullanılmıştır (http-40). Bu tema içerisindeki temel dosyalar "wwwroot" klasörü içerisine yüklenmiştir.

4.3. Kullanımı

Uygulamanın kullanımı diğer sözlük uygulamaları gibi basit ve kullanıcı dostu olarak tasarlanmıştır. Uygulamaya üye olmadan da erişilebilmektedir. Üye olmayan kullanıcılar sadece içerikleri okuyarak oylama yapabilmektedir. Yapılan oylamalar ile üyelik sisteminde yükselme sağlayamazlar. Sürekli bir kullanıcı olmak için kullanıcıların üye olması gerekmektedir.

Uygulama ana sayfasında kelimeler için bir “fihrist”, “hakkında en çok içerik girilen kelimeler” ve “blog yazıları” bulunmaktadır. Ayrıca ana sayfanın sağ üst kısmında “üye girişi” ve “yeni üyelik” kısmı bulunmaktadır. Ana sayfa içerisinde bulunan “fihrist”, sayfanın bir sözlük sayfası olduğunu vurgulamak ve insanların sözlük kullanma alışkanlıklarında bulunmasından dolayı bulunmaktadır (Şekil 4.12). “Üyelik girişi” kullanıcılara içeriğe katkıda bulunmalarını sağlayabilmek ve kullanıcıları kitle-kaynaklı bir uygulamanın adımlarında belirtildiği gibi ödüllendirebilmek adına uygulanmaktadır. Ana sayfa içerisinde bulunan bir diğer bölüm olan “en çok içerik girilen kelimeler” ‘den oluşan bölümdür. Bu bölüm kullanıcı alışkanlığı gereği gündemi takip edebilme ve en çok yorum yapılana merak uyandırma amacı taşımaktadır (Şekil 4.13). Aynı mantıkla yapılan “blog yazıları” bölümü, kullanıcılara yazılan en son yazıları görüntüleme imkânı sunmaktadır (Şekil 4.14). Ana sayfanın en son bölümü olan “özet bölümü” ise tüm uygulamanın özetini rakamsal olarak sunmaktadır (Şekil 4.15).

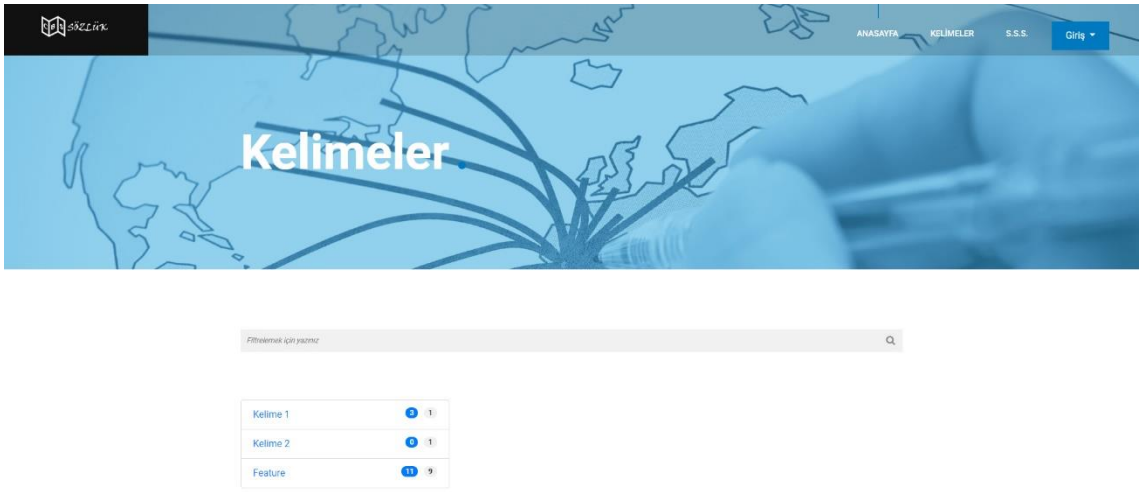
Şekil 4.12 *Anasayfanın en üst bölümünün ekran görüntüsü*

Şekil 4.13 *Anasayfadaki en çok içerik girilen kelimeler bölümü ekran görüntüsü*

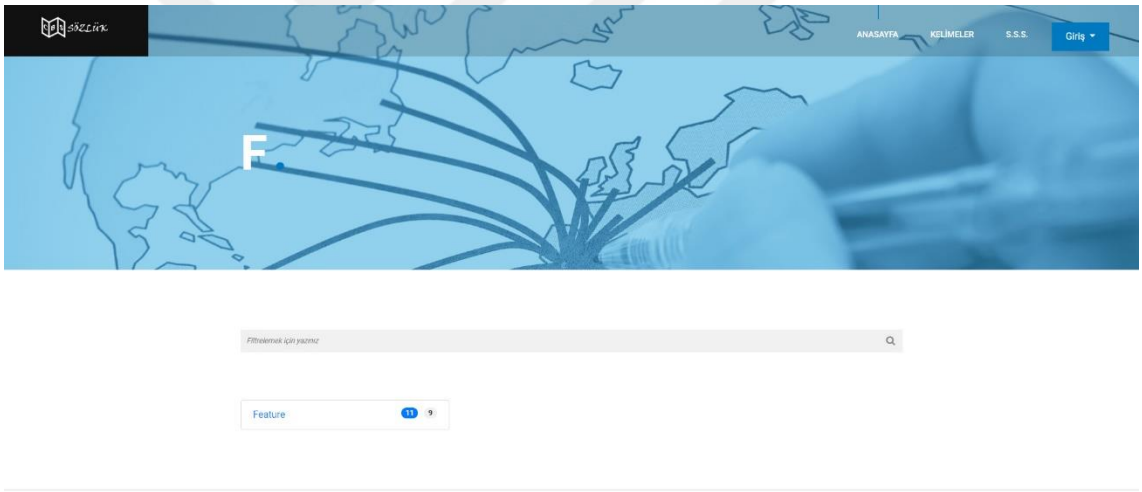
Şekil 4.14 *Anasayfadaki blog yazıları bölümü ekran görüntüsü*

Şekil 4.15 *Anasayfadaki özet bölümü ekran görüntüsü*

Uygulamanın ana başlığı olan “kelimeler” kısmına ana menü içerisinde bulunan “kelimeler” sekmesinden ve “fihrist” üzerinden ulaşılabilmektedir. “Kelimeler” sekmesinden ulaşıldığında sistemde bulunan tüm kelimelere erişilebilmekte, “fihrist” üzerinden ulaşıldığında ise sadece seçilen harf ile başlayan kelimelere ulaşılabilmektedir (Şekil 4.16, Şekil 4.17). “Kelimeler” bölümüne ulaşıldığında kelimelerin bulunduğu listenin üst kısmından bulunan kutucuk ile kelimeler filtrelenebilmektedir. Kelimeler listesinde kelimelerin yanlarında bulunan sayılar kelime içeriklerindeki tanım ve Türkçe karşılık sayılarını göstermektedir. Listede bulunan kelimelerin içeriklerine bu bölümden erişilmektedir.



Şekil 4.16 *Kelimeler bölümünün menüden ulaşıldığındaki ekran görüntüsü*

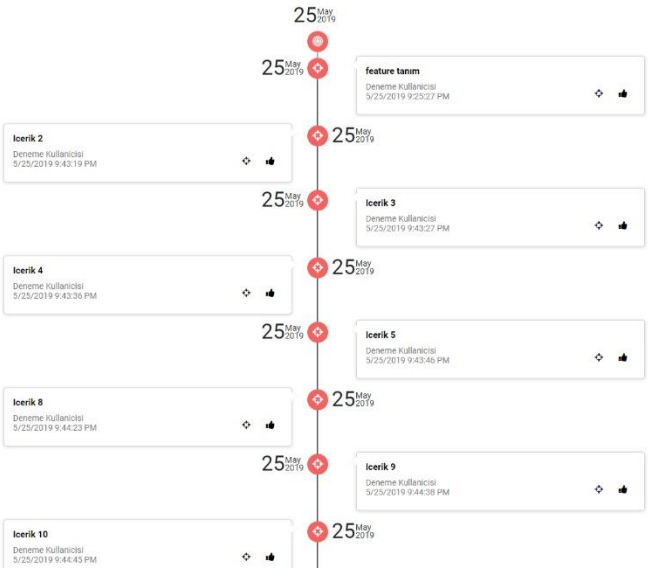


Şekil 4.17 *Kelimeler bölümüne fihristten ulaşıldığındaki ekran görüntüsü*

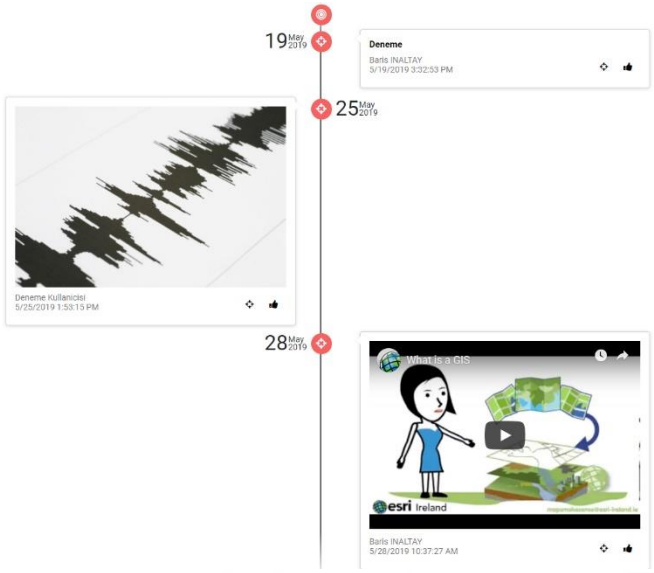
"Kelime içerikleri" bölümünde, kelime için iki ana başlık bulunmaktadır. İlk başlık "tanım" kısmı ikinci başlık ise "Türkçe karşılık" kısmından oluşmaktadır (Şekil 4.18). Kelime için girilebilecek içerikler sadece yazı ile sınırlandırılmamıştır (Şekil 4.19). Bir kelimenin tanımının veya Türkçe karşılığının bir resim ya da bir video ile de anlatılabileceği düşünülerek içerik girişlerinde bu ayrıma da yer verilmiştir (Şekil 4.20). İçerik girişleri için içeriklerin en altında bulunan bölümden girmek istenilen içerik türüne göre seçim yapılarak giriş gerçekleştirilmektedir (Şekil 4.21). İçerik girişlerini

kolaylaştırmak açısından, içerik girişleri formu yazı için bir metin kutusu, resim için bir doya yöneticisi ve video için de yerleştirme kodu eklenebilecek bir metin kutusundan oluşmaktadır (Şekil 4.22, Şekil 4.23, Şekil 4.24). İçeriklerin sağ alt kısmında bulunan beğeni simgesi ile beğenilen içeriğe oy verilebilmektedir.

Şekil 4.18 *Kelime içerik bölümü tanım ve Türkçe karşılık ayrımı ekran görüntüsü*

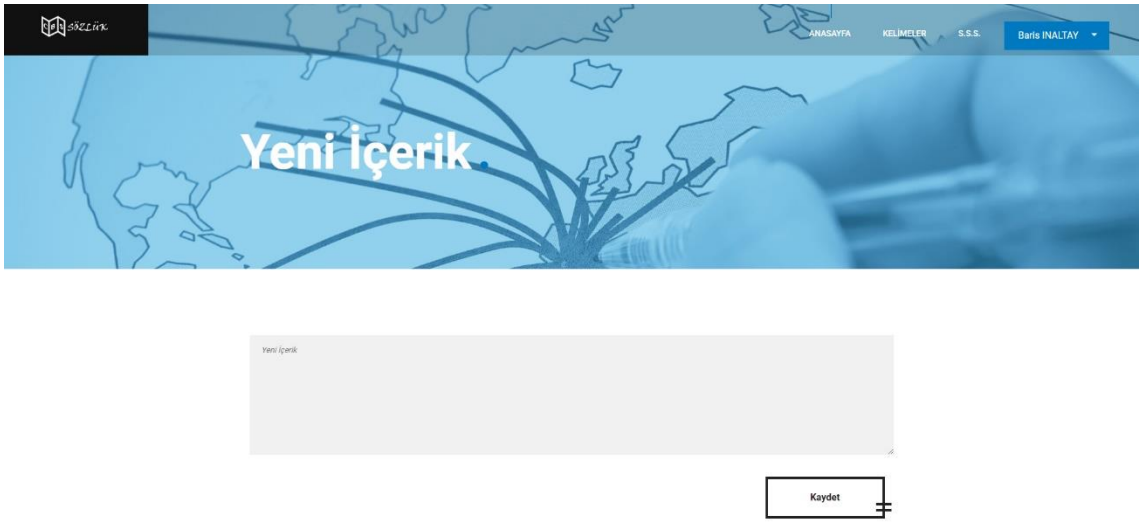


Şekil 4.19 *Kelime içerik bölümü ekran görüntüsü*

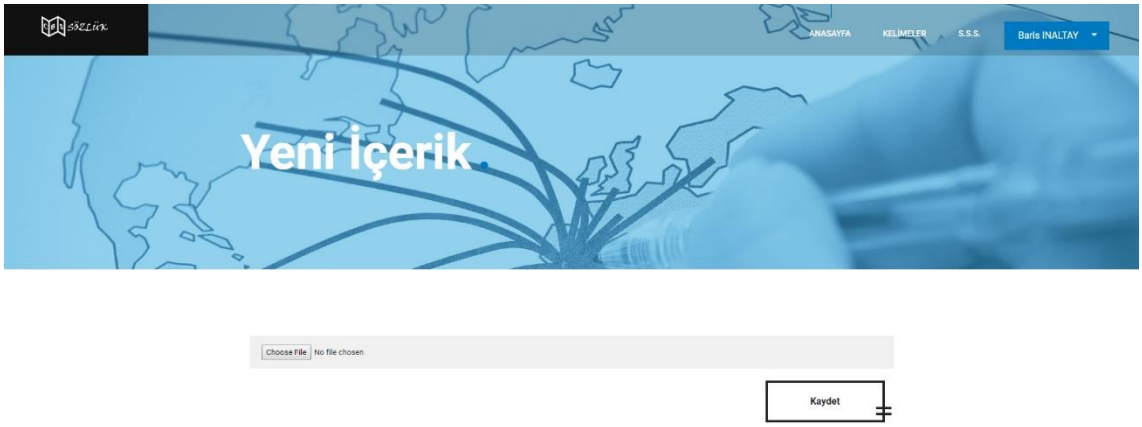


Şekil 4.20 *Kelime içeriklerinde resim ve video ekran görüntüsü*

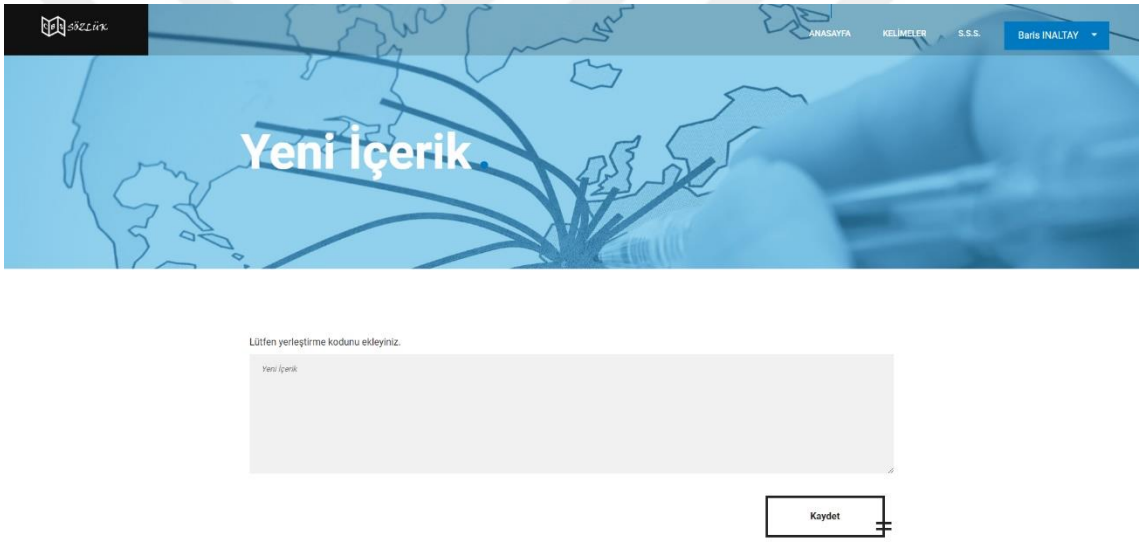
Şekil 4.21 *İçerik ekleme bölümü ekran görüntüsü*



Şekil 4.22 *Yazı ile içerik oluşturma ekran görüntüsü*



Şekil 4.23 *Resim ile içerik oluşturma ekran görüntüsü*



Şekil 4.24 *Video ile içerik oluşturma ekran görüntüsü*

Oylama işlemi üyelik olmadan yapılabilmektedir. Fakat üyelik sistemine dâhil olunduğunda oylama işlemlerinden kazanılan puanlar sistemde üyelik seviyesinin yükselmesini ve farklı rozetler kazanılmasını sağlamaktadır. Kitle-kaynaklı uygulamalar içerisinde projeye üye olan kullanıcıların sisteme bağlı kalmalarını sağlamak ve ödüllendirebilmek adına farklı işlemler uygulanmaktadır. Kitle fonlaması uygulamalarında kazanılan kardan elde edilen gelirin bir kısmı verilirken ücretsiz işlemlerde ise üyelik sisteminde yükselme ya da rozet kazanma gibi hedefler belirlenmektedir. Üyelerin belirlenen hedeflere ulaşması için yapacakları sistemi geliştirmeyi sağlamaktadır. Bu araştırmalar sonucu yazılan "CBSSözlük" uygulamasında

belirlenen hedefler hem rozetleri hem de puanlama ile üyelik seviyesinde yükselmeyi içermektedir.

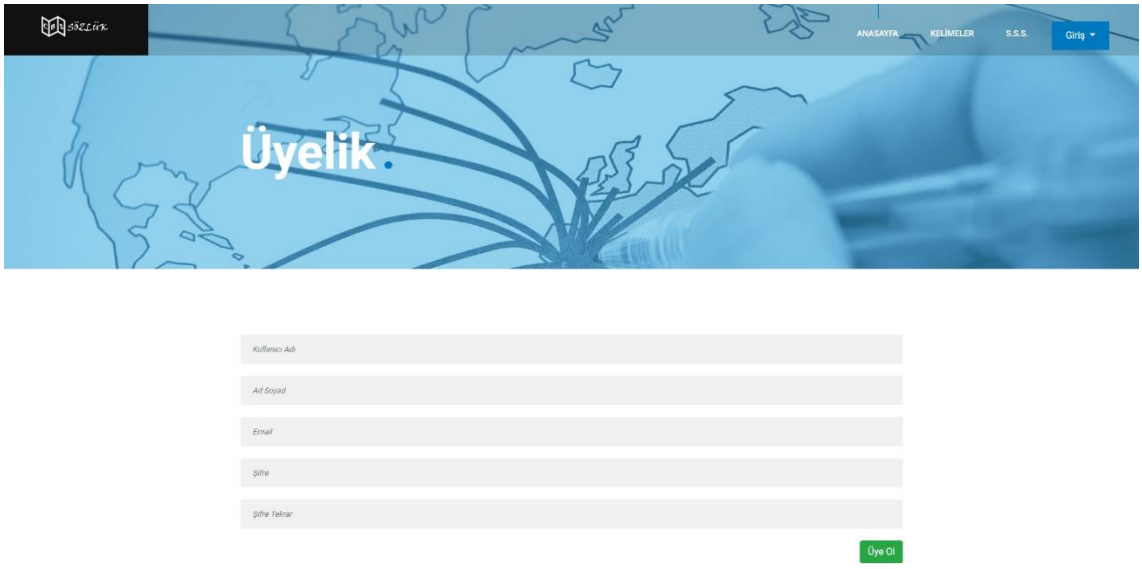
Üye olabilmek için menüde bulunan "giriş" kısmından "Yeni Üyelik" sekmesi ile üye olunabilmektedir (Şekil 4.25). Yine aynı sekme üzerinden üye olunan kullanıcı bilgileri ile giriş yapılabilmektedir (Şekil 4.26). Üyelik sistemindeki seviyeler,

- Yeni Üye
- Yazar
- İçerik Üretici
- Master

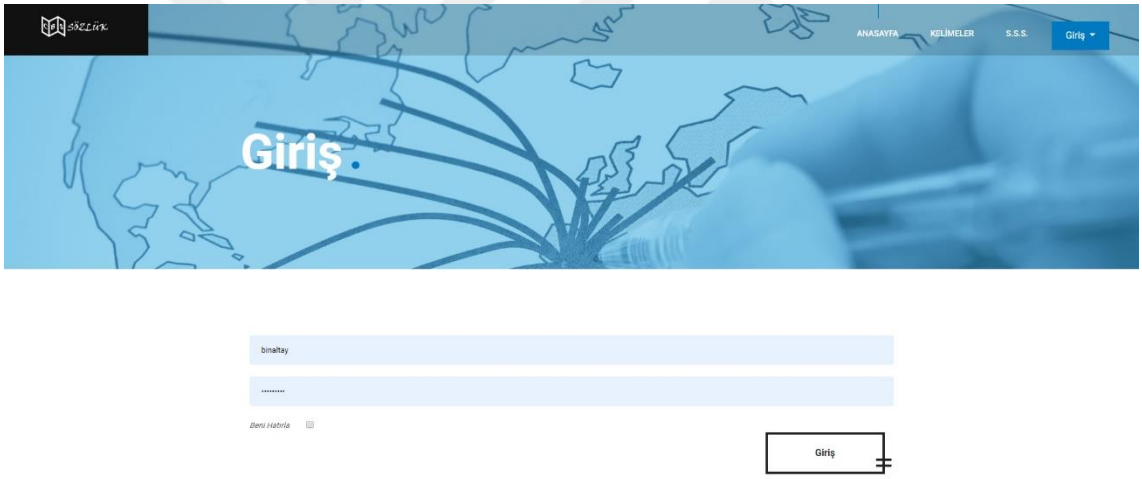
seviyeleridir. İlk üye olunduğunda üye otomatik olarak yeni üye seviyesindedir. Yeni üye seviyesindeki kullanıcılar içeriklere oy verebilmekte ve içerik girebilmektedir. Yeni üyeler toplamda 60 puan topladıklarında yazar seviyesine yükselmektedir. Yazar olan üyeler puanlama ve içerik girebilmek haricinde yeni kelime başlıkları açabilirler. Yazar seviyesindeki üyeler ise 100 puana ulaştıklarında içerik üretici seviyesine terfi ederek blog yazıları yazabilmektedirler. Son seviye olan master seviyesine ulaşabilmek için ise üyelerin 200 puan toplamaları gerekmektedir. Puanlama sistemi, içerikleri oylama kısmına katkıda bulunanlara 1 puan, içerik girenlere 2 puan blog yazısı yazanlara 5 puan şeklinde oluşturulmuştur. Aynı şekilde üyeler yaptıkları işlemler üzerinden rozet kazanabilmektedirler. Sisteme dâhil olan rozetler,

- Okuyucu, 20 adet içerik oylandığında kazanılmaktadır.
- Yazar, 20 adet içerik girildiğinde kazanılmaktadır.
- Blogger, 5 adet blog yazısı yazıldığında kazanılmaktadır.

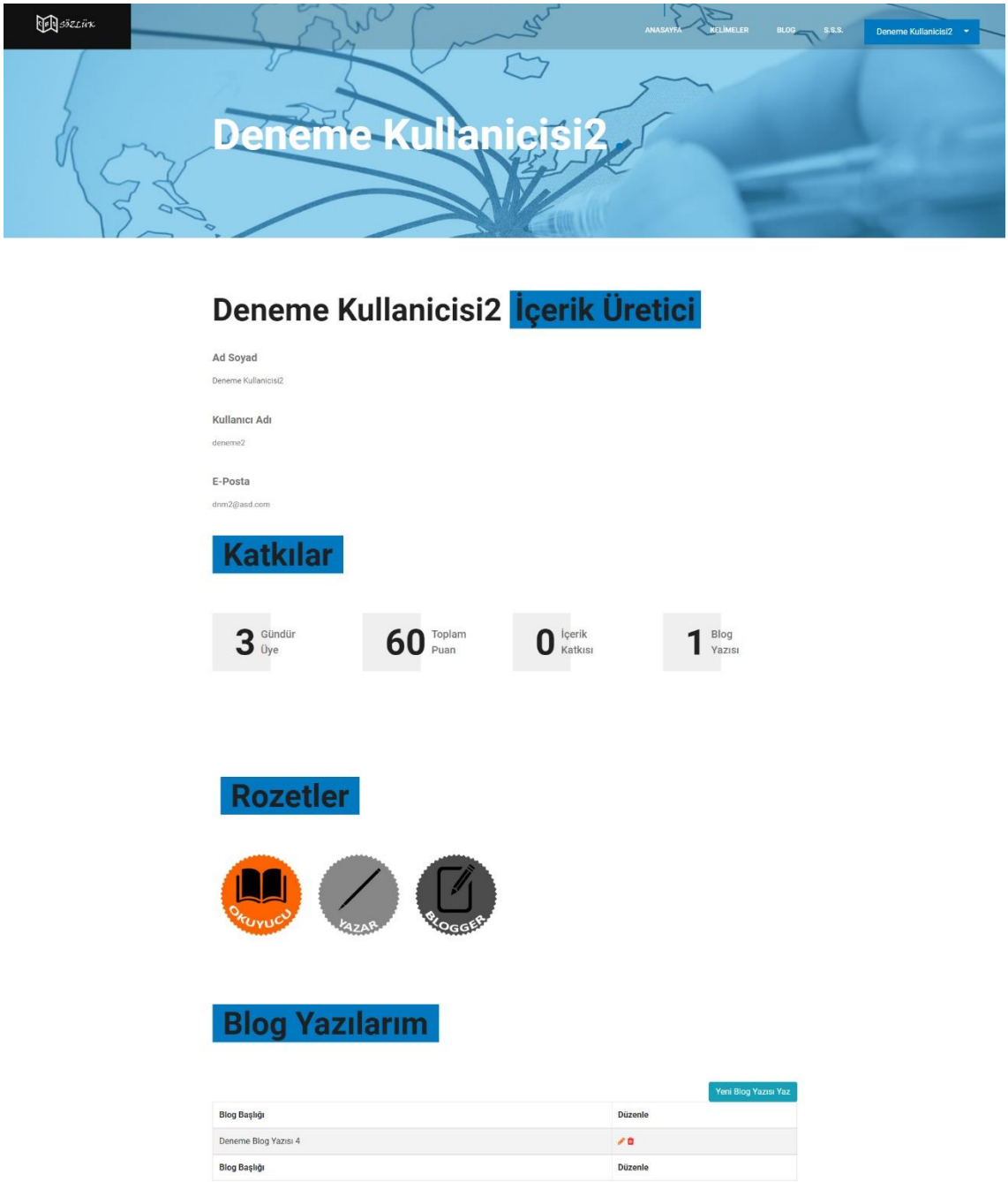
Üyeler yaptıkları tüm işlemlerin bir özetini, üyelik bilgilerini ve blog yazılarını kullanıcılarının "profil" bölümünde görüntüleyebilmektedirler. "Profil" bölümünde kullanıcının üyelik bilgileri, uygulamaya yaptığı katkıların sayısal özetini, kazandığı rozetleri ve blog yazıları kısmı bulunmaktadır (Şekil 4.27). "Blog yazıları" kısmında kullanıcının yazdığı yazıları düzenleyebildiği ve silebildiği bir liste bulunmaktadır.



Şekil 4.25 *Üyelik bölümü ekran görüntüsü*



Şekil 4.26 *Giriş bölümü ekran görüntüsü*



Şekil 4.27 *Profil sayfası ekran görüntüsü*

5. TARTIŞMA VE ÖNERİLER

Çalışma sonucunda ortaya konulan program yayınlanmamış olduğu için yerel kullanıcı görüşlerine başvurulmuştur. Yerel kullanıcı görüşleri doğrultusunda programda yapılabilecek değişiklikler ortaya çıkmıştır. Kullanıcı görüşlerine göre programın üyelik seviyesinde yükselme sisteminin, girilen içeriklerin beğenilerine göre yapılması önerilmiştir. Yazılımın bilimsel bir literatür çıktısı alabilmesi için uygulamaya ek olarak uzman kişilerce kullanılan başka bir arayüz ile filtrelenebilecek hale getirilebilir. Daha sonraki aşamalar için kullanıcı üyeliklerinden en yüksek seviyeye ulaşanlar uzman arayüzünde yetkilendirilebilir. Uzman kullanıcılar, diğer üyelerden en fazla oyu alan içerikler arasından en uygun olanı belirleyerek akademik olarak kullanılabilecek hale getireceklerdir.

Yazılım sadece CBS üzerine yazılmış olsa da diğer bilim alanlarına mensup uzmanlardan gelen önerilere göre farklı alanlarda da gereklilik olduğu yönündedir. Yazılımın içerik ve yönetim kısımlarında değişiklikler yapılarak diğer bilim alanları için uygulanabilir hale getirilebilir.

Sonuç olarak yazılım, kullanıma açıldıktan sonra yenilenmeye ve gelişmeye açık olabilecek bir biçimde yazılmıştır. Gerekli görüldüğünde işleyiş ve yönetim değişiklikleri ile kitlenin kullanımını kolaylaştıracak ya da kitleyi yönlendirebilecek şekilde güncellemeler yapılabilecektir. Yapılacak değişiklikler sayesinde de kitleden alınan içerikler akademik kullanım açısından daha doğru ve güvenilir olacaktır.

KAYNAKÇA

Anbaroğlu, B. (2017). Gönüllü Coğrafi Bilgi: Mekânsal Bilişim Çalışmalarına Web 2.0 Devrinde Yeni Bir Yaklaşım. *Harita Dergisi*, 158,1-9.

Abdulnasyrov, A. et al. (2015). Crowd-sourced automated vocabulary learning system. U.S Patent, No:9,208,144 B1.

Brabham, D. C. (2008). Crowdsourcing as a Model for Problem Solving: An Introduction and Cases. *Convergence: The International Journal of Research into New Media Technologies,* 14 (1), 75-90.

Bücheler, T., Füchslin, R. M., Pfeifer, R., Sieg, J. H. (2010). Crowdsourcing, Open Innovation and Collective Intelligence in the scientific method: a research agenda and operational framework. *In: Artificial Life XII -- Twelfth International Conference on the Synthesis and Simulation of Living Systems*, Odense, Denmark, 19 August 2010 - 23 August 2010, 679-686.

Cartwright, M. and Pardo, B. A.2013, SocialEQ: Crowdsourcing an Equalization Descriptor Map. in *Proceedings of the International Society of Music Information Retrieval Conference.* ISMIR 2013, 11/1/13.

Grier, David Alan. 2011. Not for All Markets. *Computer* 44(5): 6-8

Howe, J. 2006. The Rise of Crowdsourcing. *Wired*, June.

Howe, J. (2008). *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business 1*. NY, USA. Crown Publishing Group New York.

International Journal of Humanities and Arts Computing 8(1):46-64 · April 2014 DOI: 10.3366/ijhac.2014.0119.

Kietzmann, J. H. (2017). Crowdsourcing: A revised definition and introduction to new research. *Business horizons*, 60(2), 151-153.

Liu, Y. (2016). Chapter 8 Appeal to the public: Lessons from the early history of the Oxford English Dictionary. *Digital Studies/le Champ Numérique*, 6

MacLean DL, Heer J. (2013). Identifying medical terms in patient-authored text: a crowdsourcing-based approach. *J Am Med Inform Assoc*, 20,1120–1127.

Tün, M , Pekkan, E , Mutlu, S . (2018). Depremlerde Gözlenen Etkilerin Gönüllü Katılımıyla Hızlı Bir Şekilde Toplanması. *Eskişehir Teknik Üniversitesi Bilim ve Teknoloji Dergisi B - Teorik Bilimler*, 6 (-), 73-86.

Uyguçgil, H. (2011). Coğrafi Bilgi Sistemlerine İlişkin Temel Kavramlar. A. Çabuk (Editör), *Coğrafi Bilgi Sistemlerine Giriş içinde* (s. 141). Eskişehir: Anadolu Üniversitesi Açıköğretim Yayınları.

http-1: https://datareportal.com/reports/digital-2019-turkey?rq=Turkey (Erişim Tarihi: 10.04.2019)

http-2: https://www.slideshare.net/pavan123/discover-the-power-of-crowdsourcing (Erişim Tarihi: 14.04.2019)

http-3: https://www.matematiksel.org/denizlerin-zaman-makinesini-yapan-adam-john-harrison/ (Erişim Tarihi: 10.04.2019)

http-4: https://www.theguardian.com/science/2014/may/18/true-sea-shanty-story-behind-longitude-prize-john-harrison (Erişim Tarihi: 10.04.2019)

http-5: https://www.ideaconnection.com/open-innovation-success/Crowdsourcing-the-Oxford-English-Dictionary-00750.html (Erişim Tarihi: 10.04.2019)

http-6: https://americanhistory.si.edu/blog/2014/05/mr-peanut-and-antonio-gentile-a-trademark-that-defined-a-life.html (Erişim Tarihi: 10.04.2019)

http-7: https://www.sydneyoperahouse.com/our-story/sydney-opera-house-history/the-competition.html (Erişim Tarihi: 12.04.2019)

http-8: https://www.infovis.info/index.php?words=dupin (Erişim Tarihi: 12.04.2019)

http-9: http://www.acikbilim.com/2013/04/dosyalar/bir-seyler-biliyorsun-john-snow.html (Erişim Tarihi: 12.04.2019)

http-10: http://www.acikders.org.tr/mod/resource/view.php?id=406 (Erişim Tarihi: 12.04.2019)

http-11: https://www.ou.edu/class/webstudy/fehler/E3/go/location_sketch.jpg (Erişim Tarihi: 12.04.2019)

http-12: https://www.esri.com (Erişim Tarihi: 12.04.2019)

http-13: http://proceedings.esri.com/library/userconf/europroc98/proc/images/idp35-fig1.jpg (Erişim Tarihi: 12.04.2019)

http-14: https://static1.squarespace.com/static/576ee04d20099e977d0cab21/5771cb10893fc03e67149f7e/5771cc22bebafb593db8795f/1467075644823/Submit_Phase_Doritos_CTSB-2.png?format=1000w (Erişim Tarihi: 01.05.2019)

http-15: http://cyrillouis.com/hi/wp-content/uploads/2017/02/myburger_03_moodboard.jpg (Erişim Tarihi: 01.05.2019)

http-16: https://www.campaignlive.co.uk/article/mcdonalds-rolls-crowdsourced-burgers/1317052 (Erişim Tarihi: 01.05.2019)

http-17: https://contributors.figure-eight.work/register (Erişim Tarihi: 01.05.2019)

http-18: https://m.media-amazon.com/images/M/MV5BMjE4MDQ2NTM3Nl5BMl5BanBnXkFtZTcwMDQ1MDY1NQ@@._V1_SY1000_CR0,0,675,1000_AL_.jpg (Erişim Tarihi: 04.05.2019)

http-19: https://m.media-amazon.com/images/M/MV5BMTQ4MDc0Mjg4OV5BMl5BanBnXkFtZTgwODk3NjYyMTE@._V1_SY1000_CR0,0,674,1000_AL_.jpg (Erişim Tarihi: 04.05.2019)

http-20: https://m.media-amazon.com/images/M/MV5BZDVhNDQxYzktMjUyMS00Njk0LWJjMDAtZWM3M2ZkZGI1M2Q2XkEyXkFqcGdeQXVyOTc0NjcxNzM@._V1_SY1000_CR0,0,658,1000_AL_.jpg (Erişim Tarihi: 04.05.2019)

http-21: https://studios.amazon.com/storybuilder (Erişim Tarihi: 04.05.2019)

http-22: http://www.wikizero.biz/index.php?q=aHR0cHM6Ly91cGxvYWQud2lraW1lZGlhLm9yZy93aWtpcGVkaWEvY29tbW9ucy9mL2ZmL1NldGlhdGhvbWV2ZXJzaW9uM3BvaW50MDcuSlBH (Erişim Tarihi: 04.05.2019)

http-23: https://www.esa.int/gsp/ACT/images/projects/img_0765.png (Erişim Tarihi: 04.05.2019)

http-24: https://www.parrot.com/files/s3fs-public/styles/product_teaser_display/public/ps/3030-large-parrot-3030jpg.jpg?itok=xdznAVPT (Erişim Tarihi: 04.05.2019)

http-25: https://images-i.jpimedia.uk/imagefetch/c_fill,f_auto,q_auto:eco,w_968/https://inews.co.uk/wp-content/uploads/2019/03/masterchef.png (Erişim Tarihi: 06.05.2019)

http-26: https://i.ytimg.com/vi/j9aDrZoMvyU/hqdefault.jpg (Erişim Tarihi: 06.05.2019)

http-27: https://images.techhive.com/images/article/2017/04/wow-5k-stormwind-100716784-orig.jpg (Erişim Tarihi: 06.05.2019)

http-28: https://fold.it/portal/files/newcompetition.png (Erişim Tarihi: 06.05.2019)

http-29: https://s3.amazonaws.com/eterna/labs/news_blog_images/Full+moving+switch+example.png (Erişim Tarihi: 06.05.2019)

http-30: https://ideas.lego.com (Erişim Tarihi: 06.05.2019)

http-31: https://www.designcrowd.com (Erişim Tarihi: 06.05.2019)

http-32: https://www.kickstarter.com/ (Erişim Tarihi: 06.05.2019)

http-33: https://www.indiegogo.com/ (Erişim Tarihi: 06.05.2019)

http-34: https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcShNl-vEM9l9RlS9M7nyOJHm4LUq0P9Ba8G-4VmdUE1iNsp8DWt (Erişim Tarihi: 06.05.2019)

http-35: http://3.bp.blogspot.com/_-2Rsm6MfCOs/StYVZ3Qv79I/AAAAAAAABEg/IpFWETFK4Yk/s400/90urkcell.jpg (Erişim Tarihi: 06.05.2019)

http-36: https://img.acunn.com/uploads/tema/2019/04/04/survivortop-16633487985ca5e91e826df.jpg (Erişim Tarihi: 06.05.2019)

http-37: https://www.airbnb.com (Erişim Tarihi: 27.05.2019)

http-38: https://www.wikizero.biz/index.php?q=aHR0cHM6Ly91cGxvYWQud2lraW1lZGlhLm9yZy93aWtpcGVkaWEvY29tbW9ucy90aHVtYi9kL2RjL1ViZXJfcmlkZV9Cb2dvdGFfJTI4MTAyNzc4NjQ2NjYlMjkuanBnLzgwMHB4LVViZXJfcmlkZV9Cb2dvdGFfJTI4MTAyNzc4NjQ2NjYlMjkuanBn (Erişim Tarihi: 27.05.2019)

http-39: http://i2.res.24o.it/images2010/Editrice/ILSOLE24ORE/NOVA24/2016/05/20/Nova24/ImmaginiWeb/nova.jpg (Erişim Tarihi: 27.05.2019)

http-40: https://colorlib.com/download/457/ (Erişim Tarihi: 19.04.2019)

ÖZGEÇMİŞ

Adı-Soyadı : Barış İNALTAY

Yabancı Dil : İngilizce

Doğum Yeri ve Yılı : Eskişehir / 1989

E-posta : b.inaltay@gmail.com

Eğitim ve Mesleki Geçmişi:

- 2008 - 2014, Lisans, Eskişehir Osmangazi Üniversitesi, Fen - Edebiyat Fakültesi, Matematik Bölümü
- 2013 - 2017, Anadolu Üniversitesi, Yer ve Uzay Bilimleri Enstitüsü, Araştırmacı
- 2017 – Devam ediyor, Eskişehir Tepebaşı Belediyesi, Bilgi İşlem Müdürlüğü, Yazılımcı

Yayınları ve Bilimsel Faaliyetleri:

- YÜNCÜ HİLMİ RAFET, ERŞEN GÖKHAN, METİN TAKİ CAN, İNALTAY BARIŞ, UYGUÇGİL HAKAN, ÇABUK ALPER, “Developing farm products routes with the purpose of farm tourism and gastro tourism in Karaburun using Geographic Information Systems” II. Uluslararası Gastronomi Turizmi Kongresi 08-10 Aralık 2016