MOBİL CBS TABANLI TAŞINMAZ DEĞERLEME KARAR DESTEK UYGULAMASI GELİŞTİRİLMESİ

DEVELOPMENT OF GIS BASED DECISION SUPPORT MOBILE APPLICATION FOR REAL ESTATE VALUATION

TOLGAHAN ÖZDEN

DR. ÖĞR. ÜYESİ MURAT DURMAZ

Tez Danışmanı

Hacettepe Üniversitesi

Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin

Geomatik Mühendisliği Anabilim Dalı için Öngördüğü

YÜKSEK LİSANS TEZİ olarak hazırlanmıştır.

2022

Biricik oğlum Ahmet Akif ÖZDEN'e

ÖZET

MOBİL CBS TABANLI TAŞINMAZ DEĞERLEME KARAR DESTEK UYGULAMASI GELİŞTİRİLMESİ

Tolgahan ÖZDEN

Yüksek Lisans, Geomatik Mühendisliği Bölümü

Tez Danışmanı: DR. ÖĞR. ÜYESİ MURAT DURMAZ

Ocak 2022, 119 sayfa

Devlet kurumları için vergilendirme, bankacılık ve sigortacılık sektörleri için kredilendirme, yatırımcılar için ise alım satım değerlerinin belirlemesi açısından taşınmaz değerlemesi önem arz etmektedir. Türkiye ve diğer pek çok ülkede taşınmaz değerlemesi lisanslı değerleme uzmanları tarafından yapılmaktadır. Değerleme süreci talep ile başlamakta değerleme uzmanının değer tespiti ve denetmen tarafından kontrol edilmesiyle sonlanmaktadır. Günümüzde bilgi teknolojilerin sektöre girmesi ile birlikte bu yöntemler otomatik hale getirilmeye çalışılmaktadır.

Bu çalışmada değerleme uzmanının piyasa araştırması yönteminde kullanılan emsal tespit sürecini bir mobil CBS uygulaması ile gerçekleştirerek sürecin süresinin kısaltılması, bununla birlikte nihai rapor öncesi denetmen tarafından yapılan değer kontrolünün kolaylaştırılması hedeflenmektedir. Uygulama taşınmaz değer tespiti için daha önce yapılan değerlendirmeleri konum ve değere esas diğer bilgiler ile eşlenik olarak bir coğrafi veri tabanında saklamaktadır. Değerlendirilecek her yeni taşınmaz için bazı konumsal (şehir merkezine, hastanelere uzaklık v.b.) öznitelikler geliştirilen yazılım tarafından otomatik hesaplanmaktadır. Emsal değerlemesinin otomatik olarak hesaplanmasını sağlayacak makine öğrenmesi yöntemlerinin entegrasyonunu

kolaylaştırmak amacıyla genişletilebilir bir mimari benimsenmiş ve başlangıçta Ters Uzaklık Ağırlıklandırması (IDW), En Küçük Kareler Yöntemi (OLS) ve Uzamsal Gecikmeli Regresyon (SLR) yöntemleri entegre edilerek test edilmiştir. Kesinleşmiş her değerin bir sonraki emsal tahmini için kullanılabilmesi hedeflenerek kendi kendine öğrenebilen bölgesel bir değerleme yaklaşım geliştirilmiş. Geliştirilen yaklaşım ve yazılım Ankara ili Mamak ilçesine ait veriler aracılığı ile test edilerek sonuçlar tartışılmıştır.

Anahtar Kelimeler: Taşınmaz Değerleme, Mobil CBS, IDW, OLS, SLR

ABSTRACT

DEVELOPMENT OF GIS BASED DECISION SUPPORT MOBILE APPLICATION FOR REAL ESTATE VALUATION

Tolgahan ÖZDEN

Master, Department of Geomatics Engineering

Supervisor: DR. ÖĞR. ÜYESİ MURAT DURMAZ

January 2022, 119 pages

Real estate valuation is important in terms of taxation for government institutions, crediting and insurance sectors and determination of trading values for investors. Real estate valuation is carried out by licensed appraisers in Turkey and many other countries. The appraisal process starts with a request and ends with the expert's valuation and control by the auditor. With the advancements in information technologies these processes are being automated.

In this study, it is aimed to reduce the time required by an expert for performing the precedent determination process in the market research by providing a mobile GIS platform in order to facilitate the value control made by the auditor before the final report. The application stores the previous evaluations for the estimation of real estate value in a geodatabase in conjunction with location and other important features. For each new real estate to be evaluated, some spatial attributes (distance to city center, hospitals, etc.) are automatically calculated by the developed software. An extensible architecture has been adopted to facilitate the integration of machine learning methods that will enable the automatic prediction of real estate valuation and is initially tested by integrating Inverse Distance Weighted (IDW), Ordinary Least Square (OLS) and Spatially Lagged Regration (SLR) methods. A self-learning regional valuation approach has been developed with the

aim that each finalized value can be used for the next estimate. The developed approach and software is tested with the data of Mamak district in Ankara province and the results are discussed.

Keywords: Real Estate Valuation, Mobile GIS, IDW, OLS, SLR

TEŞEKKÜR

Bu tezin oluşturulması sürecinde bilgi ve tecrübesiyle her zaman yol gösteren ve yardımlarını esirgemeyen danışman hocam Dr. Öğr. Üyesi Murat DURMAZ'a, değerli yorumlarıyla teze katkı sağlayan jüri üyeleri Prof. Dr. Zuhal AKYÜREK, Prof. Dr. Ali Özgün OK, Doç. Dr. Sultan KOCAMAN GÖKÇEOĞLU ve Doç. Dr. Fatih NAR'a teşekkür ederim.

Hayatımın her döneminde bana güvenen ve her zaman yanımda olan sevgili eşim Gözde ÖZDEN'e ve ihtiyacım olduğunda bilgi ve düşüncelerini benimle paylaşan değerli meslektaşlarım Hüseyin ÇALIŞ ve Seçkin YILMAZER'e teşekkürlerimi sunarım.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

SİMGELER VE KISALTMALAR

Kısaltmalar

CBS	Coğrafi Bilgi Sistemleri
TDUB	Türkiye Değerleme Uzmanları Birliği
TSPB	Türkiye Sermaye Piyasaları Birliği
UDSK	Uluslararası Değerleme Standartları Konseyi
UDS	Uluslararası Değerleme Standartları
TÜİK	Türkiye İstatistik Kurumu
TKGM	Tapu ve Kadastro Genel Müdürlüğü
OGC	Open Geospatial Consertium
WMS	Web Mapping Service
KML	Keyhole Markup Language
API	Application Programming Interface
SDK	Software Development Kid

1. GİRİŞ

Menkul, değeri olan tüm malları ifade etmektedir. Gayrimenkul ise bir yerden diğer bir yere taşınması mümkün olmayan ve değeri olan varlıklara verilen isimdir. Gayrimenkul bir diğer ifade ile taşınmaz değerlemeye ilk olarak tarımsal arazilerin vergilendirilmesi amacıyla ihtiyaç duyulmuş daha sonra alım-satım, kiralama, kredilendirme, sigortalama, kamulaştırma gibi işlemler için genişlemiştir [1]. Taşınmaz bir varlığın değerinin belirlenmesi işi ülkemizde ve diğer birçok ülkede değerleme uzmanları ve devletlerin ilgili kurumları tarafından yapılmaktadır. Değerleme uzmanı bir taşınmazın değerini tüm hak ve faydalarının objektif bir şekilde ulusal ve uluslararası kabul görmüş standartlara uygun şekilde belirleyen kişidir [2].

Günümüzde taşınmaz önemli bir yatırım aracı olarak kullanılmaktadır. Tapu ve Kadastro Genel Müdürlüğü verilerine göre Türkiye'de 2021 yılı Ocak- Eylül ayları arasında 2 milyonun üzerinde gayrimenkul satışı gerçekleşmiştir [3]. Bu satışlardan yaklaşık 16 milyar TL tapu harcı toplanmıştır. Gayrimenkul Yatırımcıları Derneği (GYODER) tarafından hazırlanan Türkiye Gayrimenkul Sektörü 2021 2. Çeyrek Raporuna göre ise 2021 Mayıs ayının sonunda konut kredisi hacmi 275,1 milyar TL olarak gerçekleşmiştir [4]. Bu veriler taşınmaz sektöründeki işlem hacminin büyüklüğünü göstermektedir. Taşınmaz değerinin doğru tespit edilmesi devlet kurumları için vergilendirme, bankacılık ve sigortacılık sektörleri için kredilendirme, yatırımcılar için ise alım satım değerlerinin belirlemesi açısından önem arz etmektedir.

Taşınmaz değerlemesi için çeşitli yöntemler kullanılmaktadır. 2017 yılında yayımlanan Sermaye Piyasasında Değerleme Standartları Hakkında tebliğine göre, değerleme uzmanları Türkiye Değerleme Uzmanları Birliği ve Türkiye Sermaye Piyasaları Birliği tarafından yayımlanan uluslararası standartlara uymak zorundadır [5]. Standardın UDS 400 Taşınmaz Mülkiyet Hakları başlığı altında değerleme yaklaşım ve yöntemleri açıklanmıştır. Bu yöntemler emsal karşılaştırma, maliyet ve gelir temelli yaklaşımlardır [6]. Emsal karşılaştırma yöntemi değeri belirlenecek taşınmaza özellik olarak benzer ve konum olarak yakın taşınmazların değerlerini baz alarak yapılmaktadır [7]. Maliyet tabanlı yöntem ise genellikle yapıların değerlemesinde kullanılır. Değerlenecek yapının değeri, mevcut inşaat maliyeti belirlenip, yıpranma payı eksik imalatı gibi olumsuz değerler çıkarılarak elde edilir [2,8]. Gelir temelli yöntem de ise

taşınmazın değeri kalan ekonomik ömründe yatırımcıya kazandıracağı net gelir üzerinden hesaplanır [2]. Bu yöntemler, bilgi teknolojilerinin taşınmaz değerlemesinde kullanılmaya başlanmasıyla birlikte bilgisayar ortamında otomatik çalışacak şekilde uygulanmaya çalışılmaktadır. Taşınmaz değerlemesi uygulama alanına göre, toplu değerleme ve tekil değerleme olmak üzere iki ana başlık altında incelenmektedir.

Tekil değerleme, taşınmazın belirli tarihteki satış değerinin tespitidir[9]. Ülkemizde SPK ve BDDK mevzuatı kapsamında yapılan değerleme uygulamaları tekil değerleme olarak adlandırılabilir[10]. Tekil değerleme yukarıda bahsedilen emsal karşılaştırma, maliyet ve gelir tabanlı olarak veya bu yöntemlerin beraber kullanılması şeklinde yapılmaktadır. Toplu değerleme ise çok sayıda taşınmazın belirli bir tarihteki değerlerinin standartlaştırılmış süreçlerle ve istatistiksel testlerle belirlenmesidir[8]. Toplu değerleme için kullanılan yöntemler temelde tekil değerleme için kullanılan yöntemlerle benzerdir. Emsal karşılaştırma, maliyet ve gelir esaslı yöntemler, değerlemesi yapılacak bölgeye uygun şekilde modellenerek toplu değerleme için kullanılır. Bilgi teknolojilerinin taşınmaz değerlemesi için kullanılmaya başlanması ile birlikte bahsedilen bu geleneksel yöntemlere istatistiki yöntemler ve modern değerleme yöntemleri de eklenmiştir. İstatistiki yöntemler nominal, hedonik ve çoklu regresyon değerleme yöntemleri olarak sınıflandırılabilir. Modern yöntemler de ise yapay sinir ağları, bulanık mantık veya makina öğrenmesi kullanılmaktadır[11]. Bu yöntemlerle yapılmış birçok çalışma bulunmaktadır.

Amerika Birleşik Devletleri'nin Virginia eyaletinde yapılan bir çalışmada makina öğrenmesi ile konut değerleri tahmin edilmeye çalışılmıştır. Konutun fiziksel özelliklerini (oda sayısı, alanı, ısıtma sistemi, otopark alanı vb.) içeren bir veri seti ile C4.5, RIPPER, Naive Bayesian ve AdaBoost makina öğrenme algoritmaları kullanılmış, hangi yöntemin daha iyi sonuç verdiği tartışılmış, en iyi sonucun o bölge için RIPPER algoritması ile elde edildiği tespit edilmiştir. Değerlenen konutların konumları dikkate alınmamıştır[12]. İtalya'nın Taranto şehrinde yapılan bir diğer çalışmada yapay sinir ağları kullanılmıştır. Veri setinin içinde konutun fiziksel özelliklerinin yanı sıra konut etrafında ölçülmüş hava kirliliği verisi de eklenmiştir. Çalışma sonucunda konutun fiziksel özelliklerinin yanı sıra hava kirliliği gibi çevresel etkilerin de konut fiyatlarını etkilediğinden bahsedilmiştir [13]. Mehmet Emin Tabar tarafından hazırlanan yüksek lisans tezinde Samsun ili için yapay sinir ağları ve bulanık mantık kullanılarak taşınmaz değerleme modeli oluşturulmuştur.

Her iki yöntemin tahmin doğruluk oranının yüksek olduğu sonucuna varılmıştır[14]. Yapılan bu çalışmalarda değerlenecek taşınmazın konum bilgisi dikkate alınmamıştır. Diğer yöntemlerle karşılaştırıldığında, taşınmaz değerlemesinde jeoistatistiksel teknikler nadiren uygulanmaktadır[15].

Konum bilgisinin dikkate alındığı ABD Dallas eyaletinde yapılan bir çalışmada hedonik yöntemlerle değere en çok etki eden konut özellikleri tespit edilmiş sonrasında kriging yöntemiyle diğer konutlarla olan mesafesi ilişkilendirilerek değer tahmininin değişimi izlenmiştir[16]. Polonya'da yapılan bir diğer çalışmada konum bilgisinin dikkate alındığı ve alınmadığı durumlarda konut değerinin tahmininin nasıl değiştiği tartışılmış, konum bilgisinin tahmine olumlu katkı yaptığı sonucuna varılmıştır[17]. İngiltere'de yapılan konum ve uzaklık bazlı otomatik değerleme çalışmasında ise geleneksel kriging yöntemindeki Öklid mesafe ölçümü yerine seyahat uzaklığı ve seyahat zamanı dikkate alınmıştır. Yapılan karşılaştırmada Öklid mesafesinin kullanılmasının en kötü sonucu ürettiği sonucuna varılmıştır[18]. Yapay sinir ağları ve kriging istatistiksel yöntemini kullanan bir çalışmada her iki yönteminde artılarından bahsedilmiştir. Yapay sinir ağlarının var olan yapılar için daha doğru tahmin yaptığı, kriging metodunun ise daha çok boş arazilerdeki şehir planlaması için uygun olduğu sonucuna varılmıştır[19]. Konum bilgisi ve belirli yerlere uzaklığın, taşınmaz değerlemesinde kullanılmasının Todler'in coğrafyanın temel yasası olarak savunduğu "her şey diğer her şeyle ilgilidir, ama yakın şeyler uzak şeylerden daha fazla ilişkilidir" tezi etkili olmuştur[20]. 2007 yılında yapılan bir başka çalışmada kriging ve co-kriging metodu toplu değerleme için kullanılmış ve değer haritaları çıkarılmıştır. Çalışmanın yapıldığı yıllarda çok yaygın olmayan jeoistatistiksel yöntemlerin konut değerlemesinde kullanılması önerilmiştir[21]. Piyasa değeri karşılaştırması yöntemine mekânsal bileşenleri de ekleyerek yapılan bir başka çalışmada ise kriging, cokriging ve ters mesafe ağırlıklandırması (Inverse Distance Weighted) yöntemleri karşılaştırılmıştır. Co-kriging metodunun emsal karşılaştırılması yöntemiyle birlikte kullanılmasının konut değer tahminini artırdığı tespit edilmiştir[22].

Yukarıda bahsedilen çalışmaların birçoğu toplu değerleme için kullanılmaktadır. Çalışma çıktıları bölgesel bazda değer haritalarının çıkarılmasına yöneliktir. Toplu değerleme devletlerin şehir planları yapmaları için veya bölgesel emlak vergilendirme değerlerinin tespiti için kullanılabilmektedir. Taşınmaz bazlı kesin değerlere ihtiyaç duyulmadığı alanlarda toplu değerleme işe yaramaktadır. Fakat kredilendirme veya

sigortacılık gibi ipotekli işlemlerde taşınmazın değerleme uzmanı tarafından raporlaştırılmış ve imzalanmış değerine ihtiyaç duyulmaktadır. Bahsedilen sektörlerin işlem hacminin büyüklüğü göz önüne alındığında tekil değerlemenin önemi anlaşılmaktadır. İster geleneksel yöntemler ister modern yöntemler isterse de istatistiki yöntemler kullanılsın otomatik değerleme yöntemleri tahmin üretmektedir. Değerleme uzmanının imzasının gerektiği her taşınmaz değerlemesinde tahmin değil gerçek değerin belirlenmesi gerekmektedir. Ekonomik değişimlerin hızlı olduğu ülkeler için taşınmaz değerinin doğruluğu kadar değerleme süresi de önemlidir. İpotekli kredilendirme işlemlerinde krediyi verecek kurum değerleme sürecini beklemektedir. Bu bekleme süresinin uzaması kredi bekleyen yatırımcı için fırsatların kaçmasına sebep olabilmektedir.

1.1. Motivasyon

1985-2017 yılları arasında taşınmaz değerleme konusuyla ilgili 227 doktora ve yüksek lisans tezini inceleyen bir çalışmada taşınmaz değerine ulaşmaya çalışılırken en çok kullanılan kriterlerin bazılarının alan, yaş, oda sayısı, balkon sayısı, banyo sayısı, sosyal donatılara uzaklık, bölgedeki nüfus, kredi faiz oranları, inşaat maliyetleri olduğu tespit edilmiştir[23]. Bu kriterlerin yanında coğrafi konumu, çevresindeki taşınmazların değerleri (emsal değerleri), cephesi, müştemilatları, eklentileri, ağaç ve duvar gibi değeri etkileyen tüm kriterler göz önünde bulundurulur. Günümüzde bu işlem uzmanların taşınmazın yerinde incelenmesiyle gerçekleştirilmektedir.

Değerleme uzmanı taşınmazın bulunduğu konuma giderek tespitlerde bulunmakta ve emsal araştırması yapmaktadır. Emsal araştırması çevrede bulunan emlakçılardan bilgi alma emlak sitelerini araştırma ve komşu taşınmaz sahipleriyle görüşmeler yapma şeklinde gerçekleşmektedir. Piyasa araştırması şeklinde geçen bu süreç değerleme süresini uzatmakta ve maliyetini artırmaktadır. Geliştirilecek mobil uygulama ile daha önce değeri belirlenmiş ve doğruluğu bilinen verilerle emsal değeri hesaplanacaktır. Bu sayede değerleme uzmanının sahada geçireceği sürenin azaltılacağı düşünülmektedir.

Günümüzde değerleme firmaları değerleme sonuçlarının kontrolü için denetim mekanizmaları oluşturmak zorundadır. Kredilendirme, vergilendirme ve yatırım için

büyük önem taşıyan taşınmaz değeri değerleme uzmanlarının tespitinden sonra bir başka uzman veya denetmen tarafından kontrol edilmektedir.

Bu çalışmada değerleme uzmanlarının sahada kullanımına sunulacak ve mobil cihazlarda çalışabilecek bir WEB-GIS tabanlı karar destek uygulaması geliştirilecektir. Bu uygulama ile bölgede daha önce değeri tespit edilmiş taşınmaz değer verileri eğitilerek harita üzerinde seçilen bir taşınmaz için ortalama emsal değeri ile ilintili doğruluk oranı değerleme uzmanına bir mobil ara yüz üzerinden sunulacaktır. Ayrıca değerlendirilecek taşınmazın yakınında daha önce yapılan diğer normalize edilmiş değerlendirmeler harita üzerinde öznitelikleri ile birlikte gösterilerek uzmanın durumsal farkındalığını arttırmak amaç edinilmiştir. Geliştirilecek uygulama ile piyasa araştırma süresinin kısaltılması ve belirlenmiş değerin kontrol sürecinin kolaylaştırılması hedeflenmektedir.

1.2. Hedefler

Çalışma kapsamında mobil ortamda değerleme uzmanlarının kullanabileceği bir uygulama geliştirilmesi hedeflenmiştir. Geliştirilecek uygulama kapsamında yerel jeoistatistik yöntemler yanında makine öğrenmesi gibi modern yöntemlerin de kullanılabileceği genişletilebilir bir bölgesel tekil değer tahmini yaklaşımı geliştirilmesi hedeflenmektedir. Aynı uygulama üzerinden harita altlığı ile bu bilgilerin görselleştirilebilmesi, uygulama üzerinden kaydedilen doğrulanmış taşınmaz değerlerinin bir sonraki tahminler için kullanılmak üzere değer tespit yöntemlerine beslenmesi ve böylece kendi kendine öğrenebilir bir bölgesel tekil değerleme yaklaşımı geliştirilmesi amaçlanmıştır.

1.3. Yöntem

Tez kapsamında mobil CBS tabanlı bir uygulama geliştirilecektir. Uygulamanın temel amacı taşınmaz değerleme sürecinin mobil destekli bir uygulama üzerinden yönetilmesi ve otomatik emsal tahmini üreterek karar desteği vermesidir. Bu kapsamda önce uygulamanın kullanacağı veri tabanı yönetim sistemi belirlenecektir. Daha sonra uygulamanın geliştirileceği teknoloji seçilip uygulama geliştirme ortamı hazırlanacaktır.

Emsal tahmin yöntemleri test verisinin bulunduğu bölge üzerinde uygulanarak test edilecektir.

1.4. Tez içeriği

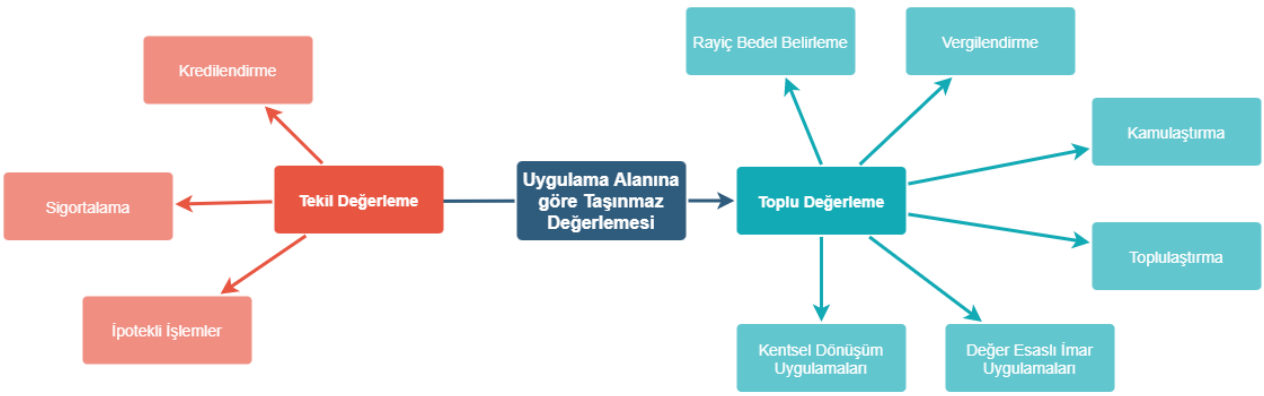
Çalışmanın hedef, yöntem ve içeriği Giriş bölümünde anlatılmaktadır. İkinci bölümde taşınmaz değerleme yöntemleri ve bilgi teknolojilerinin kullanıldığı gelişmiş yöntemler listelenmektedir. Üçüncü bölümde günümüzde yaygın olarak kullanılan mobil web teknolojileri açıklanmaktadır. Yöntem bölümünde seçilen veri tabanı ve uygulama geliştirme teknolojileri açıklanarak tasarımsal kararlar ve çalışma ortamının kurulum aşamalarından bahsedilmektedir. Beşinci bölümde uygulamanın geliştirme aşamaları, veri tabanının oluşturulması ve test verisinin düzenlenmesinden bahsedilip, sunucu-istemci uygulama geliştirme adımları yazılan kod örnekleriyle birlikte açıklanmaktadır. Sonuç bölümünde uygulamanın ürettiği sonuçlar ile karar destek sistemine daha fazla katkı sağlayabilmesi için gelecekte yapılabilecek işlerden bahsedilmektedir.

2. TAŞINMAZ DEĞERLEME YÖNTEMLERİ

Gayrimenkul sektörünün ekonomik hacminin artması ve gayrimenkullerin yatırım aracı olarak kullanılması ile birlikte taşınmaz değerleme konusu her geçen sene önemini daha da arttırmaktadır. Taşınmaz değerleme süreci uygulama alanına göre tekil ve toplu değerleme olarak iki ana başlık altında incelenebilir. Tekil değerleme daha çok kredilendirme ve sigortalama gerektiren ipotekli işlemlerde kullanılırken, toplu değerleme vergilendirme, kamulaştırma veya emlak rayiç bedeli belirleme gibi bölgesel ortalama değer gerektiren konularda kullanılmaktadır. Uygulama şekline göre ise literatürde birçok sınıflama yapılmaktadır. Bu çalışmada, uygulama şekline göre taşınmaz değerlemesi, geleneksel ve gelişmiş değerleme olarak sınıflandırılacaktır. Geleneksel yöntemler taşımaz değerlemenin temelini oluşturmaktadır. Kendi içinde piyasa araştırması, gelir ve maliyet temelli olarak 3 başlık altında toplanmaktadır. Gelişmiş yöntemler ise bilgi teknolojilerinin gelişmesiyle genelde bilgisayar ortamında geliştirilmiş yöntemlerdir. Geleneksel yöntemleri otomatik sistemlerle daha hızlı ve daha güvenilir hale getirmek için geliştirilmişlerdir. Yapay Sinir Ağları, Çoklu Regresyon, Bulanık Mantık, Kriging, Co-kriging gelişmiş taşınmaz değerleme yöntemlerinden bazılarıdır.

2.1. Uygulama Alanına Göre Taşınmaz Değerleme Yöntemleri

Uygulama alanlarına göre taşınmaz değerleme yöntemleri Şekil 2.1'de özetlenmiştir. Uygulama alanına göre Toplu ve Tekil değerleme olarak iki ana başlık altında ele alınan taşınmaz değerlemesi bu bölümde ayrı başlıklar altında ayrı ayrı detaylandırılmaktadır.



Şekil 2.1. Uygulama Alanına Göre Taşınmaz Değerleme Yöntemleri.

2.1.1. Tekil Değerleme

Türkiye İstatistik Kurumu (TÜİK) istatistikleri incelendiğinde Türkiye'de 2021 yılı Ocak-Eylül arası dönemde 147 bin konut satışının gerçekleştiği görülmektedir. Bu satışların yaklaşık 30 bini ipotekli olarak gerçekleşmiştir [24]. Ülkemizde ipotekli satışlar genel itibari ile banka kredisi ile el değiştirmektedir. Aynı zamanda bu konutların birçoğu sigortalanmaktadır. Toplam konut satışının %20 sinin ipotekli olması bu 30 bin satış için ayrı ayrı tekil değerleme yapıldığı anlamına gelmektedir. Tekil değerleme işlemi, değerleme uzmanının sahaya gidip değerlenecek taşınmazı yerinde inceleyerek geleneksel değerleme yöntemleri yardımıyla yapılmaktadır. Genel olarak değerleme uzmanı tarafından tespit edilen değer bir denetmen tarafından kontrol edilerek nihai sonuç elde edilmekte ve raporlaştırılmaktadır.

2.1.2. Toplu Değerleme

Tekil değerleme uygulamasının zor olduğu emlak vergisi, toplulaştırma, yeniden iskân, kentsel dönüşüm, değer esaslı imar uygulamaları, arazi toplulaştırma vergilendirilme, rayiç bedel belirleme, kamulaştırma gibi konularda bölgesel toplu değerlemenin yapılması zorunluluk haline gelmektedir[25]. Bahsedilen bu konularda değerlemenin aynı anda yapılması gerektiğinden tek tek değerleme yapmanın hem iş gücü hem de zaman açısından mümkün olmadığı görülmektedir. Türkiye'de Tapu ve Kadastro Genel Müdürlüğü (TKGM) altında faaliyet gösteren Taşınmaz Değerleme Dairesi Başkanlığı bulunmaktadır. 2019 yılında kurulan Başkanlık'ın birim ve görevleri arasında "Taşınmazların toplu değerleme yöntemleriyle değerini belirlemek, değer bilgi merkezini kurmak, yönetmek ve değer haritalarının üretilmesi ile güncel tutulmasını sağlamak" bulunmaktadır[26]. Bu kapsamda TKGM toplu değerleme üzerine çalışmalar yürütmekte ve standartların oluşması için çalışmaktadır.

2.2. Uygulama Şekline Göre Taşınmaz Değerleme Yöntemleri

Taşınmaz değerlemesi sahada uygulanırken belirli yöntemler kullanılmaktadır. Uluslararası Değerleme Standartlarına göre taşınmaz değerlemesinde 3 tür değer

yaklaşımı bulunmaktadır: Pazar yaklaşımı, gelir yaklaşımı ve maliyet yaklaşımı[6]. Bu yaklaşımlar taşınmaz değerlemesinin temelini oluşturmakta ve geleneksel değerleme yöntemleri olarak sınıflandırılmaktadır. Gelişmiş yöntemler veya modern yöntemler ise geleneksel yöntemleri otomatikleştiren ve bilgi teknolojileri sayesinde kolaylaştıran yöntemlerdir. Gelişmiş yöntemler literatürde çok çeşitli dallara ayrılsa da, bu tezde yaygın kullanılan yöntemler anlatılacaktır. Bu yöntemlerin hepsi hem tekil hem de toplu değerleme için kullanılabilmektedir.

2.2.1. Geleneksel Yöntemler

Geleneksel değerleme yöntemleri piyasa, gelir ve maliyet yaklaşımı olmak üzere üç başlıkta sınıflandırılmaktadır. Bu yaklaşımlar tek tek kullanılabildiği gibi bir veya birkaçı beraber de kullanılabilmektedir.

Şekil 2.2. Geleneksel Değerleme Yöntemleri.

2.2.1.1. Piyasa Yaklaşımı

Piyasa yaklaşımı yöntemi karşılaştırma veya emsal yöntemi olarak da adlandırılmaktadır. Değerlemeye konu olan taşınmaza emsal olabilecek taşınmazların güncel alım satım değerinin araştırılmasına dayanmaktadır[27]. Bu araştırma emlak alım satımının yapıldığı web sitelerini inceleyerek, taşınmaz bölgesinde bulunan emlakçılardan ve komşu taşınmazların sahiplerinden alınan bilgilerle yapılmaktadır. Genellikle üzerinde yapı bulunmayan arsa ve tarlalar için kullanılmaktadır. Üzerinde yapı

bulunan taşınmazlar için ise uygulanacak diğer yöntemlerle birlikte kullanılmakta ve değerleme uzmanına yardımcı olmaktadır.

2.2.1.2. Gelir Yaklaşımı

Gelir yaklaşımı yöntemi genellikle gelir getiren taşınmazlar üzerinde uygulanmaktadır. Taşınmazın kira getirisi, tarla ise yetiştirilen ürün miktarı gibi tüm gelir getiren kalemler detaylı bir şekilde incelenir. Yıllık olarak hesaplanan bu gelirden yıllık giderler düşülerek net gelir elde edilir. Güncel kazancın hesap edilmesinden sonra enflasyon ve diğer ekonomik kriterler göz önünde bulundurularak gelecekteki muhtemel kazancı hesaplanır. Bu yöntemin uygulanmasında özellikle gelecekteki muhtemel gelirleri hesaplanması kritik öneme sahiptir. Kalan kullanım süresi, kullanılırken ki yıpranma veya amortisman giderlerinin hesaba dahil edilmesi gerekmektedir. Genellikle tarım alanları, ticari arsa ve yapılarda gelir yaklaşımı ile değerleme yapılmaktadır[10,28].

2.2.1.3. Maliyet Yaklaşımı

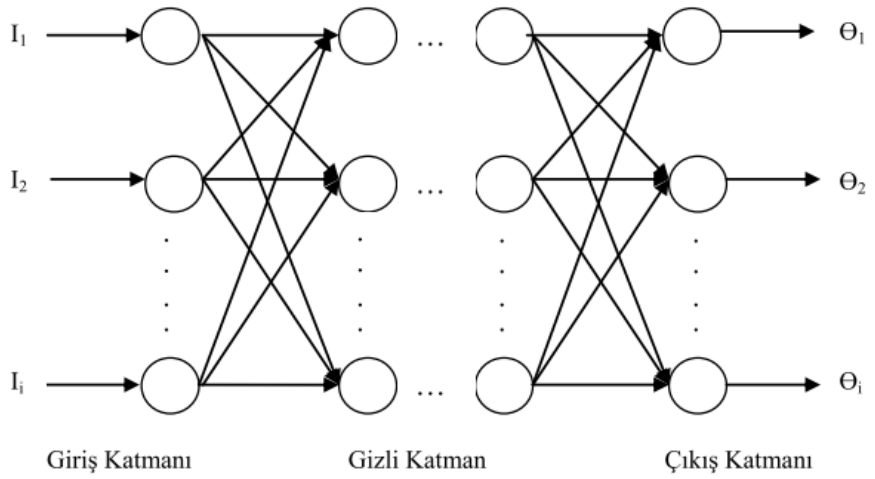
Maliyet yaklaşımı yöntemi özellikle üzerinde yapı bulunan taşınmaz değerlerini hesaplamak için kullanılmaktadır. Öncelikle değerlenecek taşınmaz üzerindeki yapının yeniden yapım maliyeti hesaplanır. Günümüz şartlarında yapının tekrar yapım maliyetinden yapının yaşından dolayı yıpranması, eksik imalatları gibi değer düşürücü özellikleri düşülür. Yapının değeri bu yaklaşımla hesaplandıktan sonra, yapının arsa hakkı bedelinin de hesaplanması gerekmektedir. Genellikle piyasa yaklaşımıyla üzerinde yapı olmadığında arsasının edeceği değer de hesaplanarak toplam taşınmaz değerine ulaşılmaktadır[29].

2.2.2. Gelişmiş Yöntemler

Gelişmiş değerleme yöntemleri bilgi teknolojilerinin taşınmaz değerlemesinde kullanılmaya başlamasıyla yaygınlaşmıştır. Makine öğrenmesi, istatistik algoritmaları, derin öğrenme algoritmaları ve yapay zekâ uygulamaları gelişmiş taşınmaz değerleme yöntemlerinde kullanılmaktadır. Temel olarak geleneksel yöntemlerin otomatik ve hızlı bir şekilde uygulamasını sağlayacak çalışmalar yapılmaktadır.

2.2.2.1. Yapay Sinir Ağları

Yapay sinir ağları insan beyninin tecrübe etme, öğrenme, bilgiyi kendi kendine yorumlama gibi fonksiyonlarını bilgisayar tabanlı modellemek üzere geliştirilmiştir. İnsan sinir sistemi birbirine bağlı ve iletişim halinde olan nöronlardan oluşmaktadır. Yapay sinir ağları da insan sinir sisteminden esinlenerek tasarlanmıştır. Ağırlıklandırılmış bağlantılarla birbiriyle iletişim halinde olan basit işlemciler içermektedir [30]. Geleneksel yazılım algoritmalarının aksine öğrenebilen bir yapıya sahiptirler. Yapay sinir ağları yapılması istenen işin örnekleri tarafından eğitilirler. Dış dünyayla bağlantılı girdi ve çıktı katmanları bulunmaktadır. Ortada ise karar veren ve problemi çözen gizli katman veya katmanlar bulunmaktadır [31].



Şekil 2.3. Yapay Sinir Ağı Modeli [32].

Şekil 2.3'te yapay sinir ağı katmanları görünmektedir. I giriş parametrelerini θ is çıkış değerlerini göstermektedir.

2.2.2.2. Çoklu Regresyon Analizi Yöntemi

Çoklu regresyon analizi birkaç bağımsız değişkenin değerinden bağımlı bir değişkenin değerini tahmin etme yöntemidir[33]. Regresyon analizi finans, ekonomi, tıp gibi birçok alanda kullanılan istatistiki bir yöntemdir. Taşınmaz değerleme sistemlerinde özellikle maliyet yaklaşımlı yöntemde bağımsız değişkenler (alan, oda sayısı, kat, cephe

vb.) çok fazladır. Bu bağımsız değişenlerden bağımlı değişken olan taşınmaz değerine ulaşılmaya çalışılmaktadır[34].

2.2.2.3. Bulanık Mantık

Bulanık mantık kavramı ilk defa 1965 yılında belirsiz ifadelerin matematiksel olarak ifadesi olarak Zadeh tarafından ortaya atılmıştır. Buna göre bir küme içindeki değerlerin mutlak bir değeri olmadığı belirli bir aralıkta olabileceği ifade edilmektedir. Klasik küme teorisinde bir değer ya kümenin elemanıdır ya da değildir. Bulanık mantıkta ise bu değerin kesin olarak bir kümeye dahil olup olmaması değil ne kadar dahil olduğu ya da olmadığı önemlidir. Mühendislik veya diğer alanlarda bulanık mantığın kullanılma amacı kesin olmayan verilerden tutarlı sonuçlar elde etmektir[27,35]. Taşınmaz değerlemede kesinliği bilinmeyen kriterlerden değer üretmek için kullanılabilmektedir.

2.2.2.4. Random Forest (Rastgele Orman)

Random Forest yöntemi 2001 yılında Braiman tarafından geliştirilmiş ve oldukça başarılı olmuş bir sınıflama algoritmasıdır. Temel olarak birden fazla karar ağacını (decision tree) birleştirilmesi şeklinde düşünülebilmektedir. Çok büyük veri setleri kullanılan uygulamalarda oldukça iyi sonuçlar verdiği görülmüştür. Karar ağaçlarının en büyük problemlerinden biri girdi verilerinin çok fazla olması sonucu veriyi ezberleyebilmesidir (overfitting). Bu durum tahmin değerinin yeni girdiler için tutarsız olmasına sebep vermektedir[35,36].

2.2.2.5. Kriging Yöntemi

Kriging yöntemi 1963 yılında Matheron tarından geliştirilmiş ve geleneksel haritalama yöntemleri yerine bilgisayar tabanlı istatistiki tahmin sunan bir modeldir. Kriging yöntemi temel olarak bulunmak istenen değeri etrafındaki yakınlık uzaklığa göre ağırlıklandırılmış değerler üstünden tahmin eder. Bu tahminle birlikte tahminin güvenilirliği ile ilgili bilgi de vermektedir[37,38]. Taşınmaz değerlemesinde üzerinde yapı bulunmayan taşınmazlar için emsal karşılaştırmasının konum bazlı yapması sebebiyle etkili bir yöntemdir. Üzerinde yapı bulunan taşınmazlar için ise arsa pay değerlerini tahmin etmek ve yapısal özelliklerin dikkate alındığı gelişmiş değerleme yöntemleriyle beraber kullanılarak değer tahmin doğruluğunu artırabilmektedir.

2.3. Karar Destek Sistemler

Günümüz kurum ve şirketlerinin başarısı karar verme süreçlerinin doğru yönetilmesiyle doğrudan ilişkilidir. Günümüz piyasalarındaki rekabet göz önüne alındığında, karar vericiler riskleri ve fırsatları doğru analiz edip kurum veya şirketleri için en uygun stratejiyi belirlemelidirler. Karar verme süreci ile ilgili yapılan araştırmalarda sezgisel yöntemlerin birçok dezavantajı olduğu görülmüştür. Sezgisel kararlar kişiye bağlı olduğu için kişinin özelliklerden çok fazla etkilenmekte ve kişiden kişiye farklılık gösterebilmektedir. Bu sebeple bilgi teknolojileri yardımı ile işletmeler ürettikleri ve kullandıkları veriyi saklayarak kurum hafızası oluşturmakta ve bu veriler ışığında karar destek sistemleri oluşturmaktadırlar. Ham ve işlenmemiş bu verileri karar vericiler için anlamlı ve yararlı hale getiren sistemler karar destek sistemleri şeklinde tanımlanmaktadır. [39] Karar destek sistemleri literatürde çok farklı yöntemlerle yapılabiliyor olsa da genel olarak 3 ana başlık altında karar vericilere destek olmaktadırlar.

Yöntemlerden ilki istatistik üretilmesidir. Verilecek karar ile alakalı toplanan verilerden istatistik üretilmesine dayanmaktadır. Konu ile ilgili daha önce yapılmış işlemler veya konuya etkisi olan durumların yoğunlukları ile ilgili istatistik üretilerek karar vericilerin bilgisine sunulmaktadır. Karar vericilerin bu istatistikler ışığında kararlarını daha doğru şekilde vermesini amaçlamaktadır.

Günümüzde yapay zekâ ve bilgisayar öğrenmesi algoritmaları karar destek sistemleri için de kullanılmaya başlanmıştır. Bu yöntemler ile karar verilmeden önce olası sonuçlar, bilgisayar ve geliştirilen uygulamalar sayesinde bir modelle tanımlanarak tahmin yaptırılabilmektedir. Bu sayede bir insanın günlerini alabilecek karar sonrası oluşabilecek durumların analizi, bilgisayar ortamında hızlı bir şekilde belirlenebilmekte ve karar vericinin önüne sunulmaktadır.

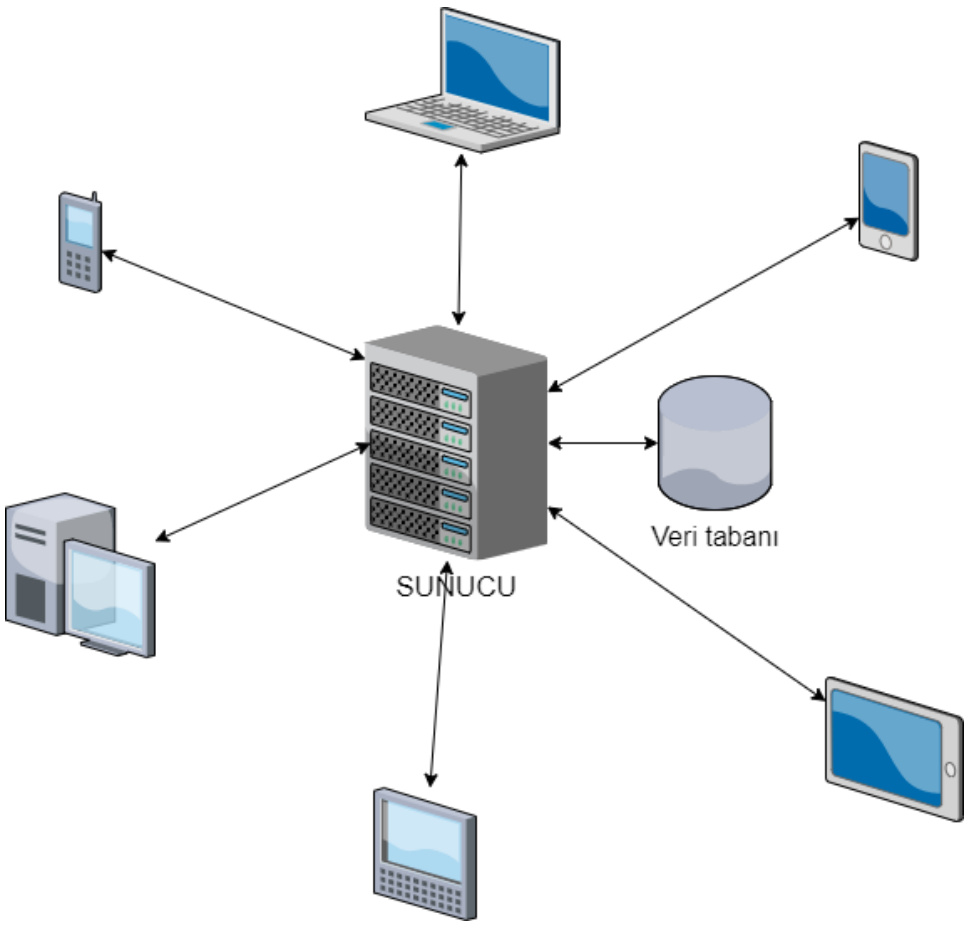
Karar destek sistemleri, karar verme öncesi yardımcı olabileceği gibi verilen karar sonrası çıkan sonucun aykırılık kontrolünü de yapabilmektedir. Daha önceki sonuçlara göre olması gereken değer aralığını tahmin ederek kontrol esnasında bu aralığın dışında üretilmiş aykırı değerleri tespit edebilmekte ve karar vericiyi uyarabilmektedir.

Bu çalışmada taşınmaz değerleme konusunda yardımcı olması için bir karar destek uygulaması geliştirilecektir. Değerleme uzmanına taşınmaz değerinin tespiti için yakınında bulunan taşınmaz özelliklerinin istatistiklerini sunacaktır. Emsal tahmin

algoritmaları ile seçilen konumdaki birim fiyat değerini tahmin edip değerleme uzmanın kullanımına sunacaktır. Ayrıca denetmenler için değerleme uzmanının belirlediği değerin kontrolünde emsal tahmin değerini de göstererek aykırılık kontrolüne yardımcı olacaktır.

3. MOBİL WEB CBS TEKNOLOJİLERİ

Bilgisayar ve internet teknolojilerinin gelişmesi ile web mimarileri hayatımızda önemli bir yer tutmaktadır. Web teknolojilerinin sadece bilgisayarlarla sınırlı olmayıp mobil cihazlarda da çalışıyor olması popülerliğini artırmıştır[39]. Bu süreçte web ve mobil uygulama ihtiyacı artmış, insanların bilgiye her yerden ulaşabilme alışkanlığı oluşmuştur. Tek bir bilgisayar üzerinden işlem yapan uygulamalar yerine istemci (client) sunucu (server) mimarileri tercih edilir duruma gelmiştir. Sunucu istemci mimarisi, geliştiren uygulama fonksiyonlarını istemci ve sunucu tarafına dağıtarak uygulama geliştirme süresini önemli ölçüde azaltmıştır. Bu mimaride geliştirilmiş çoğu uygulamada veri işleme süreçleri sunucu tarafında yapılırken, işlenmiş veri kullanıcıya istemci tarafında sunulmakta ve bu durum genelde uygulama performansını ve bakım kolaylığını artırmaktadır[41]. Sunucu tarafında geliştirilen uygulama birden çok istemciye aynı anda hizmet verebilmektedir.

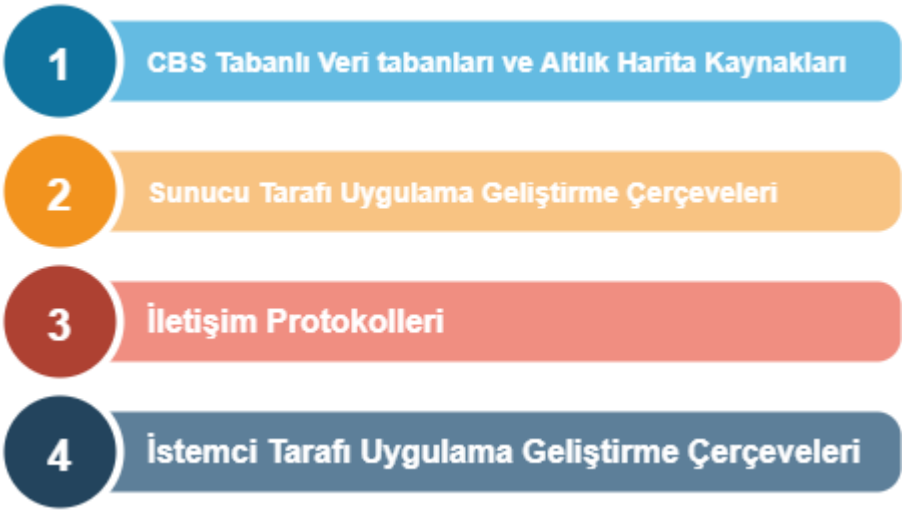


Şekil 3.1. Sunucu İstemci Mimarisi

Mobil ve web teknolojileri uygulama geliştirme bileşenleri genel anlamda veri tabanı, sunucu tarafı uygulama geliştirme çerçevesi (framework), sunucu istemci arası iletişim protokolü ve istemci uygulaması geliştirme çerçevesinden oluşmaktadır. Bu bölümde yaygın kullanılan mobil ve web teknoloji bileşenleri anlatılmaktadır.

3.1. Mobil-Web CBS Tabanlı Uygulama Geliştirme Bileşenleri

Coğrafi Bilgi Sistemleri fiziksel dünyaya ait veriyi depolayan, dönüştüren, gösteren ve mekânsal veriyi karar desteğine dahil eden araçlar bütünü olarak tanımlanabilir[41]. Taşınmaz değerlemesinin önemli unsurlarından biri de mekânsal veridir. Taşınmazın bulunduğu konum değerini önemli ölçüde etkilemektedir. Çağımızda değerlemenin hızlıca yapılması taşınmaz yatırımcıları için zorunluluk haline gelmiştir. Bu sebeple CBS tabanlı uygulamalar değerleme uzmanlarının sahada mobil cihazlarla kolayca erişebileceği ve işlem yapabileceği şekilde geliştirilmektedir. Verinin depolandığı ve işlendiği, harita altlıklarının bulunduğu veri tabanı katmanı, sunucu uygulamasının geliştirildiği uygulama geliştirme çerçevesi, istemci uygulamasının geliştirildiği uygulama çerçevesi ve sunucu ile istemci arasındaki iletişim protokolleri CBS tabanlı mobil ve web uygulaması bileşenlerinin dört ana başlığını oluşturmaktadır.



Şekil 3.2. Mobil-Web CBS Tabanlı Uygulama Geliştirme Bileşenleri

3.1.1. Konum Destekli Veri tabanları ve Altlık Harita Kaynakları

2020 yılı itibariyle dünyadaki her insan için saniyede yaklaşık 1.7 MB veri üretildiği tahmin edilmektedir[43]. Bu verinin önemli bir bölümü de konumsal veridir. Coğrafi veri günümüzde popülerliğini artırmakta, insanlara aradığı yeri bulma, trafik durumunu öğrenme gibi günlük hayatı kolaylaştırıcı hizmetler sunabilmektedir. Bununla birlikte coğrafi verinin kullanımının artması bu verinin ekonomik yatırımlar için altlık oluşturmasına olanak sağlamıştır. Haritalar coğrafi bilgiyi görselleştirerek kullanıcıya aktarmanın en doğal yoludur[44]. Bu bölümde coğrafi veriyi depolamak, işlemek ve sunmak için yaygın olarak kullanılan ilişkisel coğrafi veri tabanları (MySQL, PostgreSQL, MSSQL) ile uygulamalar için harita altlıkları (OpenStreetMap, GoogleMaps, Yandex.Maps, Bing Maps) hakkında güncel bilgiler listelenmektedir.

3.1.1.1. MySQL

MySQL Oracle firması tarafından desteklenen açık kaynaklı ilişkisel bir veri tabanı yönetim sistemidir. İlişkisel veri tabanları veriyi tek bir yerde depolamak yerine veriyi tutarlılığı da sağlayarak ayrı ayrı tablolarda saklamaktadır. MySQL mantıksal modeli de bu ilişkileri sağlamak üzere tablo (table), görüntü (view), satır (row), sütun (column) gibi bileşenler ile sunmaktadır. Farklı tablolar ve farklı veri alanları arasında bire bir, bire çok, benzersiz, gerekli veya isteğe bağlı gibi ilişkileri yöneten kurallar tanımlanabilmektedir. MySQL veri tabanı yönetim sistemi bu kuralları uygulayabilmekte, bu sayede tutarsız, yinelenen, yetim, güncel olmayan veya eksik veri problemlerini engelleyebilmektedir. Temel olarak veri tabanı işlemleri için yaygın olarak kullanılan yapısal sorgu dilini SQL (Structured Query Language) kullanmaktadır[45].

MySQL veri tabanı veriyi tanımlamak, yönetmek, sunmak ve ilişkilendirmek için model odaklı bir sistem sunmaktadır. Bu modeller veri öğelerini temsil etmek ve görselleştirmek için iyi araçlardır. Model odaklı veri yönetiminin meta-veri yönetimini standartlaştırması, hızlı uygulama geliştirilmesine olanak sağlaması, değişiklik ve performans yönetimini kolaylaştırması, iletişim ve raporlamayı basitleştirmesi gibi avantajları bulunmaktadır. MySQL geliştiricileri veri tabanı model odaklı bu yaklaşımı kullanıcıların daha kolay gerçekleştirebilmesi için MySQL WorkBench tasarım aracını geliştirmiştir. Grafik ara yüze sahip bu araç sayesinde veri tabanı tasarımı, yedekleme,

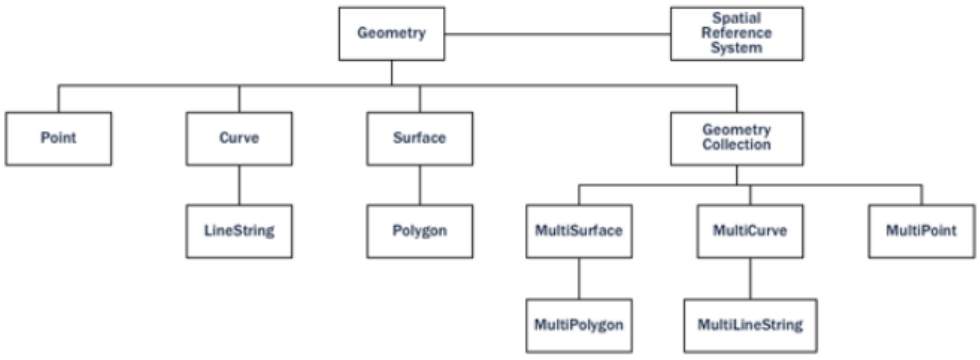
bakım, değişiklik yönetimi, raporlama gibi konular daha basit şekilde çözülebilmektedir[46]

MySQL veri tabanı mekânsal (spatial) veriyi de yönetebilmektedir. MySQL içeriğindeki "OpenGIS Geometry" modeli sayesinde Open Geospatial Consortium (OGC)'nin sunduğu "SQL with Geometry Types" standartlarındaki veri tiplerini modelleyebilmektedir. Temel veri tipleri GEOMETRY, POINT, LINESTRING ve POLYGON'dur. Bu veri tiplerini tekil ya da çoklu olarak saklayabilmekte ve işlem yapabilmektedir[45].

3.1.1.2. PostgreSQL

PostgreSQL'in temeli California Üniversitesinde geliştirilmiş nesne tabanlı ilişkisel bir veri saklama sistemi olan POSTGRES projesiyle atılmıştır. 1986 yılında başlayan POSTGRES projesine Andrew Yu ve Jolly Chen tarafından 1994 yılında SQL dili işleme özelliği eklenerek Postgres95 adını almıştır. Proje 1996 yılında ise isim değiştirerek PostgreSQL olarak devam etmektedir. PostgreSQL'in desteklediği doğal veri tipleri oldukça fazladır. Veri saklamak için gereken tüm veri tiplerini barındırmakla beraber CREATE TYPE komutuyla yeni veri tipi oluşturmaya da izin vermektedir[45].

Genel veri tabanı işlemlerini gerçekleştirebilen PostgreSQL üzerine kurulan PostGIS eklentisi ile PostgreSQL veri tabanını mekânsal veriyi de işleyebilecek duruma getirmektedir. Mekânsal veriyi işleyebilmek için PostgreSQL'e mekânsal veri tipi, mekânsal indeks ve mekânsal fonksiyon özelliklerini eklemiştir. Desteklenen mekânsal veri tipleri hiyerarşik olarak Şekil 3.3'te verilmiştir.



Şekil 3.3. Geometrik Hiyerarşi [48]

İndeksler veri tabanlarında veriye hızlı erişmeyi sağlamaktadır. Geleneksel veri tabanlarında indeks yapısı string, date, numeric gibi veri tipleri üzerinde B-tree algoritmasıyla oluşturulur. Şekil 3.3.'de görüldüğü üzere mekânsal veriler çok boyutlu olabilmektedir. Bu sebeple hiyerarşik ağaç yapısıyla mekânsal veri üstünde indeks oluşturmak mümkün olmamaktadır. Büyük boyutta mekânsal veri üzerinde sorgulama yapıldığında, indeks kullanımı sorgu performansını doğrudan etkilemektedir[48,49]. PostGIS mekânsal indekslemeyi çevreleyen kutular (bounding box) yöntemiyle gerçekleştirmektedir. Problemi kesin değerle çözmek yerine yaklaşık bir değer bulmak adına çevreleyen kutuların kaç kenarının poligonun içinde kaldığını bularak bir çözüm üretilmiştir. PostGIS bu çözüm için tasarlanmış Q-tree yöntemini kullanmaktadır[48].

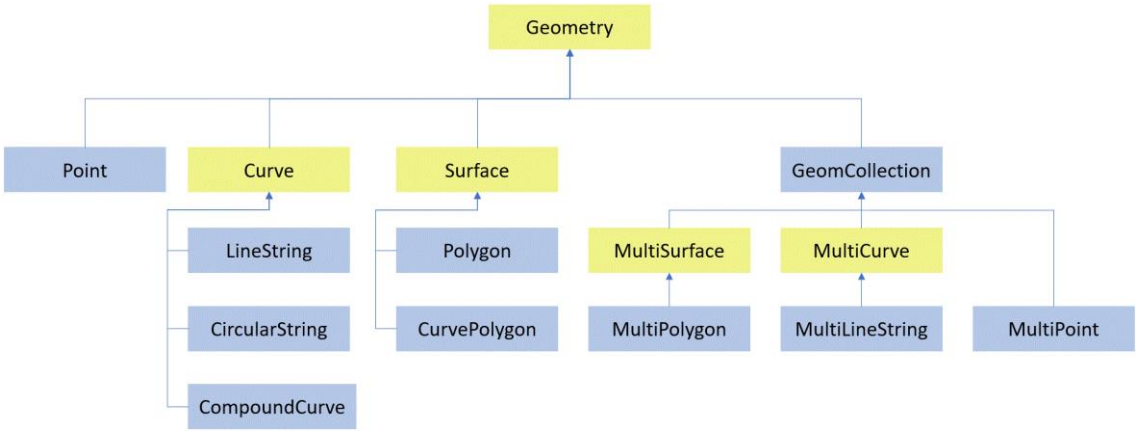
PostGIS genel kabul görmüş tüm mekânsal fonksiyonları desteklemektedir. Bu fonksiyonlar geometrik veri dönüşümlerini, geometrik işlemler (orta nokta, yol üstündeki nokta sayısı, kapalı alan olup olmadığı vb.), karşılaştırmalar, mekânsal veri yönetimi fonksiyonları ve var olan geometriden yeni geometriler türetme olarak sınıflandırılmaktadır [46,48].

3.1.1.3. MSSQL

MSSQL, Microsoft firması tarafından büyük çaptaki veriyi yönetmek amacıyla geliştirilmiş bir veri tabanı yönetim sistemidir. Ücretli bir uygulama olan MSSQL'in Express versiyonu bazı limitlerle birlikte ücretsiz olarak kullanılabilmektedir. MSSQL 2019 versiyonuyla birlikte artık sadece Windows işletim sistemli bilgisayarda değil Linux işletim sistemli bilgisayarlarla da çalışabilmektedir. Yine Microsoft'un bulut sistemi olan Azure üzerinden de hizmet verebilmektedir. Makine öğrenmesini desteklemekte R, Java, Microsof.NET, Python gibi dillerle sunucu tarafında makine öğrenmesi modelleri uygulanabilmektedir. Rapor servisleri ile barındırdığı verilerden Power BI ve Mobil BI uygulamalarını destekleyebilmektedir[50].

MSSQL yaygın kullanılan veri tiplerinin yanında mekânsal veri tiplerini de desteklemektedir. Mekânsal veriyi, geometri (geometry) ve coğrafi (geography) veri tipleri altında saklayabilmektedir. Geometri veri tipi nesnelerin şekillerini temsil ederken coğrafi veri tipi koordinatlı konumu temsil etmektedir. Coğrafi veri tipi OGC ve SQL

MM (ISO standardı) ile uyumlu çalışabilmektedir. Ayrıca GPS enlem boylamı gibi elipsoidal veri barındıran coğrafi veriyi de desteklemektedir[51].



Şekil 3.4. MSSQL Geometrik Hiyerarşi [52]

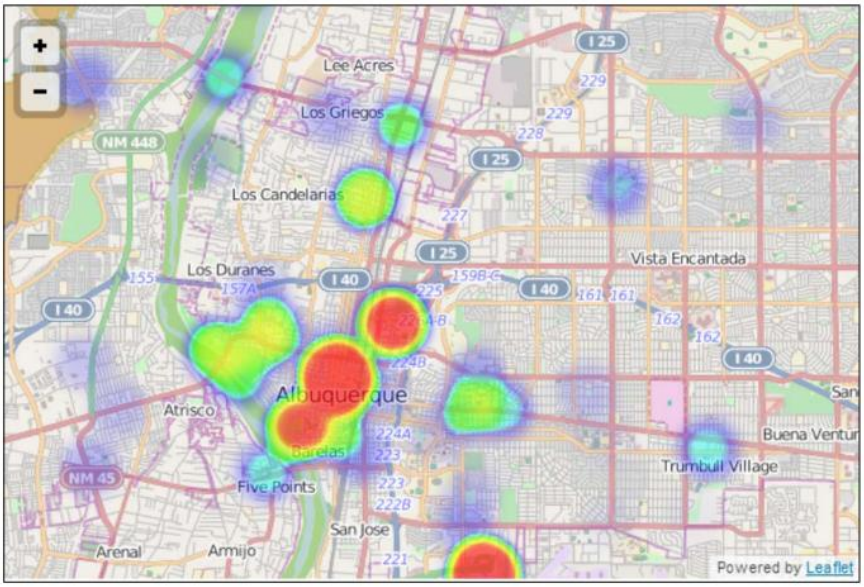
Şekil 3.4. geometri ve coğrafi veri tiplerinin dayandığı geometri hiyerarşisini göstermektedir. Tanımlanabilir geometri ve coğrafya türleri mavi renkle gösterilmiştir. Coğrafi ve geometri tipleri genel olarak yakın özellikler gösterse de birkaç noktada farklılık içermektedir. Örnek olarak iki geometri tipindeki verinin birbirine uzaklığı bir doğru oluştururken coğrafi tipinden iki verinin birbirine uzaklığı bir eğri oluşturmaktadır[51].

3.1.1.4. OpenStreetMap

OpenStreetMap tüm dünyanın ücretsiz ulaşabildiği, gönüllüler tarafından büyük ölçüde sıfırdan oluşturulan ve düzenlenebilen açık içerik lisansı (open-content license) ile yayımlanan haritadır. Yeryüzündeki fiziksel özellikleri (yol, bina vb.) kendi veri yapısı (nodes, ways, relations) içinde etiketleme (tag) yöntemiyle çalışmaktadır. Her bir etiket coğrafi bir özelliği tanımlamaktadır. Etiketleme işlemi haritayı kullanan ve geliştiren katılımcılar tarafından tanımlanmaktadır[53]. Açık kaynak kodlu olması ve gönüllülük esaslı geliştirilmesi sebebiyle OpenStreetMap hem akademik çalışmalarda hem de eğitim amaçlı olarak sıkça kullanılmaktadır.

Leaflet, OpenStreetMap harita altlığıyla çalışan bir javascript kütüphanesidir. Mobil uygulamalar için de uyumlu olan Leaflet kütüphanesi uygulama geliştiricilerinin

ihtiyaç duyabileceği çoğu özelliği barındırmaktadır. Boyutunun çok küçük olması (son versiyonu 39KB) ve yardımcı kaynak sayısının fazla olması sebebiyle uygulama geliştiriciler tarafından harita tabanlı uygulamalar için fazlaca tercih edilmektedir. Ayrıca OGC-WMS (Web Map Service) servislerini de desteklemektedir[54]. Leaflet ayrıca TMS (Tiled Map Service) standartlarını da destekleyebilmektedir. TMS servisler ise harita taban altlıklarını karolar halinde sunmaktadır. Leaflet harita altlığı ve bu harita üzerine WMS servislerle katmanlar ekleyebilme imkanının yanı sıra kendi katmanlarınızı da oluşturmanıza izin vermektedir. Yarattığınız katman üzerine nokta (point), doğru (line), poligon (polygon) ve bu veri tiplerinden türetilen kare (rectangle), daire (circle), çoklu doğru (multypolylines) veya çoklu poligon (multipolygon) çizilmesine imkân vermektedir. Bununla birlikte Leaflet kütüphanesiyle yoğunluk haritaları (heat map) oluşturulabilmektedir[55]. Örnek bir yoğunluk haritası görünümü şekil 3.6 da görülmektedir.



Şekil 3.5. Leaflet örnek yoğunluk haritası [55]

3.1.1.5. Google Maps

Google Maps platformu Google tarafından hizmete sunulan ve belirli limitlere kadar ücretsiz olarak hizmet veren bir harita uygulamasındır. 2005 yılında hizmete girdiğinden beri uygulama geliştiricilerin sıklıkla kullandığı ve kullanıcıların mekânsal ve kültürel veriye kolayca ulaşabilmesini sağlamıştır. Özelleştirebilir haritalar sayesinde

eğitimcilere coğrafi bilgi için kullanılabilir bir kaynak sunmaktadır. Ayrıca KML (Keyhole Markup Language) dosyalarına destek vererek üretilen projelerin kolayca paylaşmasını sağlamaktadır[56].

Google haritalar coğrafi bilgi sistemleri işlemlerine altlık oluşturmanın yanı sıra sağladığı uygulama programlama ara yüzleri (API) sayesinde harita tabanlı uygulama geliştiricilerine kolaylık sağlamaktadır. Google harita API'leri, haritalar (maps), rotalar (routes) ve yerler (places) kategorileri altında toplanmaktadır[56].

Çizelge 3.1. Google Haritalar (Maps) Api Listesi.

Api	Açıklama
Maps JavaScript API	Özelleştirilmiş haritaları mobil ve web uygulamaları üzerinden sunmak için kullanılmaktadır. Haritalar roadmap, satellite, hybrid ve terrain harita tiplerinde sunulabilmektedir.
Maps SDK for Android	Android tabanlı mobil cihazlar ve giyilebilir teknolojilerde harita kullanılmasını sağlamaktadır. Harita verileri üzerinde işlem yapma, katman (layer), işaretçi (marker), poligon (polygon) ekleme gibi fonksiyonlar içermektedir. Android cihazlar için kullanılan Java ve Kotlin programlama dillerini desteklemektedir.
Maps SDK for iOS	IOS işletim sistemine sahip cihazlar için harita hizmeti sunmaktadır. SDK, Google harita sunucusuna bağlanarak, harita gösterme (display), tıklama (click), sürükle-bırak (drag-drop) gibi hareket algılama ve Android SDK da olduğu gibi katman (layer), işaretçi (marker), poligon (polygon) ekleme gibi fonksiyonlar içermektedir.
Maps Static API	Web sitelerine Javascript ya da diğer dinamik web yükleme araçlarına bağımlı olmadan harita resimleri sağlamaktadır. Static Map API servisi HTTP protokolü üzerinden istenen harita bölgesinin resmini içeren bağlantı adresini göndermekte ve istenen harita resmini web sayfası içinden kolayca gösterilmesini sağlamaktadır.
Street View Static API	Web sayfalarına Javascript kullanmadan istenen nokta için sokak görünümü panoraması veya küçük resim yerleştirilmesine olanak

	tanır. Standart http protokolü üzerinden iletişim kuran API istenen nokta için statik görüntü içeren bağlantı adresini döndürmektedir.
Maps Embed API	Javascript kullanmadan web sayfalarına etkileşimli harita yerleştirilmesi sağlamaktadır. Standart http protokolü kullanılarak istemde bulunan web uygulamalarına istenen özellikte harita bağlantısı sunmaktadır.

Çizelge 3.2. Google Rotalar (Routes) Api Listesi.

API	Açıklama
Directions API	Directions API, HTTP isteği kullanarak konumlar arasında JSON veya XML biçimli yönergeleri döndürmek için kullanan bir web hizmetidir. Toplu taşıma, araba kullanma, yürüme veya bisiklete binme gibi çeşitli ulaşım yöntemleri için yol tarifi sunmaktadır.
Distance Matrix API	Başlangıç ve varış noktası gönderilen konumlar için seyahat mesafesi ve süresi sağlayan bir hizmettir. Dönen değer Başlangıç ve bitiş noktaları arasında önerilen rotaya dayalı bilgileri döndürür ve her bir çift için süre ve mesafe değerlerini içeren satırlardan oluşmaktadır.
Roads API	Roads API, bir aracın seyahat ettiği yolları tanımlamakta ve bu yollar hakkında hız sınırları gibi ek veriler sağlamaktadır.

Çizelge 3.3. Google Yerler (Places) Api Listesi.

API	Açıklama
Places API	Places API, HTTP isteklerini kullanarak yerler hakkında bilgi döndüren bir hizmettir. Bu API içinde kuruluşlar, coğrafi konumlar veya önemli mekanlar tanımlanmaktadır.
Places SDK for Android Places SDK for iOS	Her iki API mobil kullanıcı cihazının yakınındaki diğer yerlere duyarlı uygulamalar oluşturmanıza olanak tanımaktadır.
Geocoding API	Geocoding API sayesinde metin tipindeki adresleri coğrafi koordinatlara veya coğrafi koordinatları metin tipindeki adrese

	çevrilmesini sağlayabilmektedir. Harita üzerinde aranan adres noktasını gösterebilmektedir.
Geolocation API	Kullanıcı cihazının iletişim kurabildiği baz istasyonları ve WIFI hatlarını kullanarak konum ve konumun bulunabileceği yaklaşık yeri bulmak için kullanılmaktadır.
Time Zone API	Dünya yüzeyindeki konumlar için saat dilimlerini ve bu saat dilimlerinin UTC'den farklarını hesaplayabilmektedir.
Elevation API	Konum bilgisinden konumun yükseklik bilgisini sorgulamak için kullanılmaktadır. Bununla birlikte seyahat edilen rota boyunca yükseklik değişimlerinin hesaplanmasına olanak vermektedir. Ayrıca okyanus ve denizler gibi negatif yüksekliğe sahip bilgileri de içermektedir.

3.1.1.6. Yandex.Maps

Yandex haritalar aynı Google haritalar gibi HTTP protokolü ve Javascript kütüphaneleri kullanılarak harita servisi sunmaktadır. Belirli sorgu limitleri aşılmadığı sürece ücretsiz olarak hizmet vermektedir. Yandex, haritaları etkin kullanabilmek için JavaScript API, Mapkit, Geocoder, Places API, Static API ve Distance Matrix geliştirilmiştir. JavaScript API web sayfaları içinde haritalarla çalışmayı mümkün kılmaktadır. Rota oluşturma, sokak görüntülerini gösterebilme, yer arayabilme gibi mekânsal özellikleri barındırmaktadır. Ayrıca açık kaynak kodlu eklenti geliştirilebilmektedir[58]. Yaygın kullanılan eklentilerden bir tanesi alan hesaplama eklentisidir (Area calculation plugin). Eklenti ymaps.GeoObject, ymaps.Polygon, ymaps.Rectangle, ymaps.Circle, GeoJson Polygon, GeoJson Rectangle, GeoJson Circle tipindeki nesneleri girdi alarak alanlarını hesaplayabilmektedir[58]. Bir başka eklenti ise yoğunluk haritaları oluşturmak için geliştirilmiştir. (Yandex Maps API Heatmap Module). Bu eklenti sayesinde istenilen mekânsal veriler yoğunluğuna göre renklendirip harita üzerinde sunulabilmektedir[60].

Places Api, Static Api, Distance Matrix ve Geocoder Google haritalarda olduğu gibi yer arama, iki nokta arası rota oluşturma, seyahat süresi ve mesafesi, adresten koordinata, koordinatta adrese dönüşüm gibi fonksiyonları sunan servislerdir. Google haritalardan farklı olarak mobil cihazlar için geliştirilen servis MapKit adıyla tek çatı

altında toplanmıştır. MapKit Android veya iOS işletim sistemi için mobil uygulamalarda Yandex harita servislerinin kullanılmasını sağlamaktadır. JavaScript Api'sinde web için kullanılan tüm fonksiyonları mobil uygulamalarda da kullanmak üzere geliştirilmiştir.

3.1.1.7. Bing Maps

Bing haritaları Microsoft tarafından geliştirilmiş Google ve Yandex haritalarında olduğu gibi harita altlığı sunan bir servistir. Servisler HTTP istekleriyle çalışmakta ve genel kullanılan mekânsal fonksiyonların tümünü desteklemektedir. Web tabanlı uygulamalarda JavaScript ve TypeScript kullanılarak haritaların gösterilmesi ve mekânsal fonksiyonların kolay şekilde geliştirilmesi için Bing Maps V8 Web Control aracını geliştirmiştir. Bu aracın öne çıkan özellikleri GeoJson ve GeoXML'i destekleyebilmesi, mobil cihazlarda bulunan web tarayıcılarında kullanım kolaylığı sağlaması için dokunmatik desteği, 360 derece sokak görüntüsü sağlayabilmesi, şehir sınırlarına kolayca ulaşılabilmesi, gerçek zamanlı trafik verisini alabilmesi ve bu veri ışığında seyahat rotaları çizebilmesi olarak gösterilmektedir[61].

Bing haritaları iOS ve Android cihazlar için Maps SDK for Android and iOS'u yayımlamıştır. Üç boyutlu harita motoru sayesinde harita üzerindeki temel nokta, doğru ve poligon gibi geometrileri gösterebilmenin yanında üç boyutlu binaları ve yükseltilmiş dünya haritası gösterimini desteklemektedir[61]. MapsSDK-Unity, Microsoft tarafından geliştirilmeye devam eden Unity tabanlı uygulama geliştiriciler için dünyanın üç boyutlu modelini oluşturmaya çalışan bir projedir. Proje hala geliştirilmeye devam ediliyor olup üç boyutlu haritaları yine bir Microsoft sanal gerçeklik projesi olan HoloLens gibi araçlarda da kullanmayı hedeflemektedir[63].

Bing haritalarının bir diğer projesi ise Lojistik ve Filo Yönetimi'dir. Bing haritaları kullanarak filo araçlarının takibini (Fleet Tracker) ve kamyon tır gibi büyük araçların taşımacılık için kullanacakları güzergahları (Truck Routing API) belirlemek üzere geliştirilmiştir. Proje kapsamında geliştirilen Isochrone API ise belirli bir noktadan verilen zaman veya mesafe ölçüsüne göre gidilebilecek yerleri belirlemek için geliştirilmiştir.

3.1.2. Sunucu Tarafı Uygulama Geliştirme Çerçeveleri

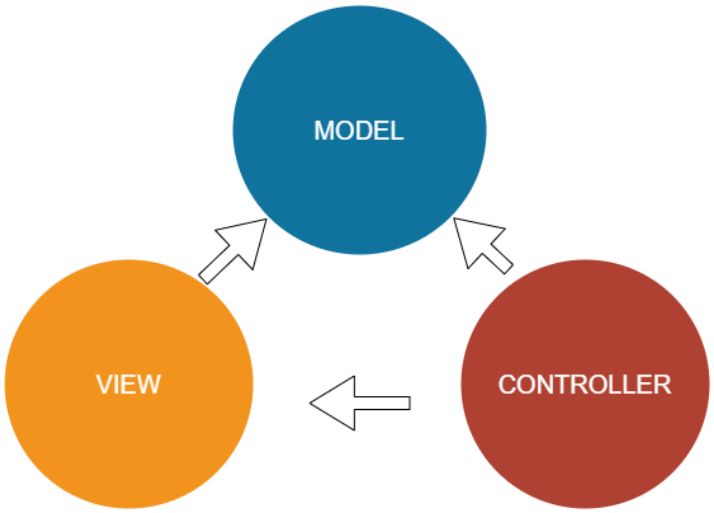
Uygulama geliştirme çerçeveleri (framework) tercih edilen yazılım dili açısından farklılık göstermektedir. Kullanılan yazılım diline göre geliştirme araçları ve yazılım mimarileri de farklılıklar göstermektedir. Bu bölümde kullandıkları yazılım dillerinin göre yaygın olarak kullanılan ASP.NET Core, Spring Framework, Php Frameworks, DJango ve GeoDJango sunucu uygulama geliştirme çerçeveleri anlatılacaktır.

3.1.2.1. ASP.NET Core

.NET platformu Microsoft firması tarafından sunulan ve birçok farklı tipte uygulama geliştirmeye imkân veren yazılım kütüphaneleri ve dillerinin bütünüdür. ASP.NET ise .NET platformunun web uygulamaları için özelleştirilmiş halidir. .NET platformuna web isteklerini C# ve F# yazılım dilleriyle gerçekleştirme, oturum yönetim sistemi, MVC (Model-View-Controller) yazılım deseni ve çeşitli web eklentileri eklenerek ASP.NET oluşturulmuştur. ASP.NET Core ise 2016 yılında yine Microsoft tarafından sunulmuş ASP.NET'in açık kaynak kodlu versiyonudur. Önceki ASP.NET versiyonlarının sadece Windows işletim sisteminde çalışabiliyor olmasının aksine Core versiyonu Linux, macOS ve Windows işletim sisteminde çalışabilmektedir[63].

ASP.NET Core yaygın olarak kullanılan yazılım deseni MVC'yi desteklemektedir. Web uygulamaları ya da web API'leri MVC desenini temel alarak geliştirilebilmektedir. MVC deseninde temel amaç kullanıcı ara yüzleri ile bu ara yüzde sunulan verilerin ve yönetimin ayrılmasıdır[64]. Bir uygulamanın temel işleyişi girdi, girdiye uygulanan işlemler ve çıktı şeklinde sınıflandırılabilmektedir. Uygulama içinde veriyle uğraşan katman Model bölümüdür. Model katmanı veri işlemlerini diğer katmanlardan ayırarak birden fazla farklı giriş ve çıkış katmanına aynı anda hizmet verilmesine olanak sağlamaktadır[65]. Controller katmanı kullanıcıdan giriş değerlerini alan, Model ve View katmanları arasında köprü görevini üstlenen bölümdür. View katmanı ise uygulamanın çıktı olarak ürettiği her şeyin sunumu ile ilgilenmektedir. Controller aracılığı ile Model ile iletişim kurarak işlenen verileri kullanıcıya sunmaktadır [67]. ASP.NET Core MVC yazılım mimarisi desenini test edilebilirliğini arttırdığı için tercih etmektedir. Her bileşen birbirinden bağımsız şekilde test edilmesi hata yakalama ve giderme için uygun ortamı da oluşturmaktadır [64].

ASP.NET Core'un bir diğer özelliği Razor sözdizimini (syntax) kullanabiliyor olmasıdır. Razor sözdizimi, web uygulamalarında View katmanında kullanıcıya sunulan ve HTML temelli web sayfaları içinde .NET kodlarını kullanılmasını sağlamaktadır. Örnek olarak ** @Username ** kod parçası incelendiğinde HTML dilinde paragraf için kullanılan bir bileşendir. Paragraf içinde yazılacak değer normal bir html sayfasında bir bileşen kullanılarak yapılabilirken, Razor sözdiziminde @ işareti ile .Net değişkenleri doğrudan kullanılabilmektedir. ASP.NET Core'un dikkat çeken yeteneklerinden bir tanesi de Model bağlama (binding) özelliğidir. Razor sayfaları ve Controller sayfaları HTTP istekleri ile çalışmaktadır. Bu istekler sonucunda dönen değerler bir modele değişken isimleri aynı olmak kaydıyla doğrudan bağlanabilmektedir. Bu sayede dönüş değerlerinin Controller veya Razor tarafında teker teker eşitlenmesine gerek kalmamaktadır. Aynı durum tam tersi şekilde istek değerleri içinde çalışmaktadır [64].



Şekil 3.6. MVC (Model-View-Controler) Yazılım Deseni

3.1.2.2. Spring Framework

Güvenli yazılım geliştirici platformu olan Snky'nin yayımladığı "JVM Ecosystem Report 2021" raporuna göre Spring dünyada en fazla kullanılan Java uygulama geliştirme çerçevesidir [67]. Spring çerçevesi Java dilinde uygulama geliştirmeyi kolaylaştırmakta, hızlandırmakta ve daha güvenli hale getirmektedir [68]. Spring çerçevesinin öne çıkan yeteneklerinden bir tanesi mikro servislerdir. Mikro servis mimarisi uygulama kodunun

birbirinden bağımsız küçük ve yönetilebilir parçalara ayrılması şeklinde uygulanmaktadır. Küçük ölçekli ve bağımsız olması bakım ve hata yakalamayı kolaylaştırmakta böylece üretkenliği artırmaktadır [68].

Web uygulamaları Spring çerçevesinde hızlıca geliştirilebilmektedir. Uygulama geliştiricilerine uygulamada kullanacakları verilere erişim için çeşitli imkanlar sunmaktadır. İlişkisel ve ilişkisel olmayan veri tabanlarına ve bulut tabanlı verilere erişimi desteklemektedir. Bir diğer dikkat çekici yeteneği ise sunucusuz (serverless) uygulama geliştirme yeteneğidir. Kaynak tahsisi, güvenlik, ölçeklendirme, işletim sistemi seçimi gibi sunucu tarafı işlemleri yerine daha çok uygulama odaklanılmasını sağlamaya çalışmaktadır. Olay tabanlı tetikleyicilerle çalışmakta, bu sayede bulut sistemleri kaynaklarının sadece tetiklendiğinde harcanmasını sağlayarak maliyetleri düşürebilmektedir [69]. Spring çerçevesi toplu (batch) işlemleri verimli bir şekilde gerçekleştirebilmektedir. Toplu işlemler sınırlı ve çok sayıda veri işleminin kesintisiz ve dış müdahalelere kapalı şekilde yapılabilmesidir. Spring çerçevesi bu işlemleri zamanlayarak ve yeteri kadar kaynak ayarlamasını otomatik yaparak çözebilmektedir [69].

3.1.2.3. Php Çerçeveleri

Php web uygulamaları için yaygın kullanılan programlama dillerinden biridir. Ücretsiz olarak hizmet veren ve açık kaynak kodlu Php programlama dili web uygulamaları için geniş bir kütüphaneye sahiptir. Php dilinin önemli özelliklerinden biri basit metin işleyiciler içinde bile kod yazılmasına izin vermesidir. Kodları Php motoru tarafından çalıştırıldığı için derlenmesine gerek duyulmamakta bu sebeple herhangi bir yazılım geliştirme ortamına doğrudan ihtiyaç duymamaktadır. Php dilinin kullanıldığı üç ana alan bulunmaktadır. Bunlardan ilki sunucu tarafı programlamadır. Php'nin temelini oluşturan sunucu tarafı programlamayı çalıştırabilmek için Php motoru, HTML sunucusu ve bir web tarayıcısına ihtiyaç duyulmaktadır. Bir diğer kullanım şekli ise komut satırı uygulamalarıdır. Komut satırı uygulaması sayesinde bir Php uygulaması hiçbir sunucu ya da tarayıcıya ihtiyaç duymadan sadece Php motoru ile çalıştırılabilmektedir. Son olarak Php ile aynı zamanda masaüstü uygulamalar geliştirilebilmektedir. Php-GTK eklentisi masaüstü uygulama geliştirmeyi mümkün kılmaktadır [69]. Php ile uygulama geliştirmeyi kolaylaştırmak açısından bazı uygulama çerçeveleri geliştirilmiştir. Laravel

ve CodeIgneter yaygın kullanılan Php çerçevelerinden bazılarıdır. Laravel Php ekosistemi kullanıcılara esnek, genişletilebilir ve kolay yazılım geliştirmeyi hedeflemektedir. Kullandığı Docker teknolojisi sayesinde tüm işletim sistemlerinde uygulama geliştirilebilmektedir. Docker bir işletim sistemi üzerinde sanallaştırılmış olarak farklı bir işletim sistemi çalıştırmak ve uygulamaları bağımlılıkları ile birlikte paketlemek için kullanılmaktadır. Laravel, Sail modülü ile kolayca Docker'a bağlanmakta, bu sayede uygulamayı çalıştıracağınız sunucu özelliklerini geliştirici bilgisayarında test edilmesine imkân vermektedir. MySQL, PostgreSQL, MSSQL ve SQLite veri tabanlarını desteklemektedir. Bu veri tabanları üzerinde ham SQL sorgusu çalıştırmanın yanı sıra Eloquent ORM (object-relational mapper) aracı sayesinde ilişkisel veritabanı tablolarını modeller ile eşleştirip bu modeller sayesinde kolayca ve güvenli bir şekilde veri tabanı işlemlerinin yapılabilmesini sağlamaktadır. Ayrıca içinde bulunan Query Builder sayesinde veri tabanı sorguları doğrudan SQL kodu yazılmadan oluşturulabilmektedir. Web uygulamalarının en temel prensiplerinden biri oturum (session) yönetimidir. Kullanıcıların sisteme girişleri ve sistemde kalmalarını kontrol etmek için kullanılırlar. Laravel içinde bulunan Auth ve Session modülleri sayesinde oturum yönetimini güvenli ve kolayca yapılacak hale getirilmiştir [71]. Yaygın kullanılan Php çerçevelerinden diğer ise CodeIgneter'dır. Boyut olarak çok küçük olan bu çerçeve (12MB) performansı yüksek ve güvenli yazılım geliştirmeye olanak sağlamaktadır. Veri tabanı işlemleri, çoklu dil desteği, MVC yazılım desteği ve kullanılan kütüphaneler gibi bileşenlere erişim kolaylaştırılmıştır [71].

3.1.2.4. DJango ve GeoDJango

Django yüksek seviyeli Python programlama dili ile geliştirilmiş bir web uygulama geliştirme çerçevesidir. Django web uygulaması geliştirmek için gerekli tüm yeteneklere sahiptir. Uygulama verilerini yönetmek için soyut bir model katmanı bulunmaktadır. Bu katmanda verileri temsil etmesi için nesne tabanlı modeller tanımlanabilmekte, modeller üstünde işlem yapabilmek için gelişmiş sorgular çalıştırılabilmektedir. PostgreSQL, MariaDB, MySQL, Oracle ve SQLite veri tabanlarını desteklemektedir. Veri tabanı işlemleri transaction kontrollü sayesinde, işlemlerin yarıda kalması ya da hata durumlarında veri tutarlılığı koruyacak şekilde eski haline geri getirilebilmektedir. View katmanı ise kullanıcı isteklerini karşılayan bölümdür. İçinde bulunan Middleware bileşeni sayesinde kullanıcıların istekleri özel fonksiyonlara

bölünmüştür. Bu fonksiyonlar kullanıcı isteklerine cevap vermek için geliştirilmiştir. Örnek olarak AuthenticationMiddleware bileşeni kullanıcı oturum yönetimini sağlamaktadır. Bunun gibi dosya yükleme istekleri veya pdf dosyası oluşturma gibi kullanıcı etkileşimi olan bileşenler de içermektedir. Uygulama geliştiriciler için temel form bileşenlerini yaratmak için Form API geliştirmiştir. Bu sayede HTML formları veri modelini de destekleyecek şekilde kolayca oluşturulabilmekte ve form başvuruları (submit) GET ve POST HTTP istekleri ile karşılanabilmektedir. Django'nun en dikkat çekici ve güçlü özelliklerinden biri otomatik yönetici paneli (admin site) oluşturabilmesidir. Bu sayede istenilen modellerin yönetici tarafından güncellenebilir olması uygulama çatısı tarafından otomatik olarak sağlanabilmektedir. Bu yeteneklerin yanında güvenlik (security), globalleştirme-yerelleştirme (internationalization - localization), performans ve optimizasyon (performance-optimization) için geliştirilmiş bileşenleri bulunmaktadır [72].

DJango çerçevesinde CBS tabanlı web uygulaması geliştirilebilmesi için GeoDjango modülünü kullanılabilir. Bu modül Django çerçevesini dünya çapında hizmet verebilecek bir coğrafi web çerçevesine dönüştürmektedir. Veri tabanı olarak mekânsal veri saklayabilen çoğu veri tabanlarını desteklemektedir. Coğrafi sınıflar oluşturulabilmekte ve bu sınıflar üzerinden coğrafi veri tabanı tabloları otomatik olarak üretilebilmektedir. Oluşturulan bu tablolar üstünde coğrafi veri tabanı işlemleri nesne tabanlı olarak yapılabilmektedir. Çizelge 3.4'te GeoDjango çerçevesi ile veri tabanında yapılabilecek önemli bazı işlemlerin listesi ve açıklamaları verilmiştir [72].

Çizelge 3.4. GeoDjango Veri Tabanı İşlem Listesi.

Fonksiyon İsmi	**Açıklama**
Area	Coğrafi bölgenin alanını hesaplamaktadır.
AsGeoJSON	Coğrafi veriyi GeoJson formatına dönüştürmektedir.
AsGML	Coğrafi veriyi Geographic Markup Language (GML) formatına dönüştürmektedir.
AsSVG	Coğrafi veriyi Scalable Vector Graphics (SVG) formatına dönüştürmektedir.
AsWKB	Coğrafi veriyi Well-known binary (WKB) formatına dönüştürmektedir.
AsWKT	Coğrafi veriyi Well-known text (WKT) formatına dönüştürmektedir.

Azimuth	Radyan cinsinden azimuth açısını hesaplamaktadır.
BoundingCircle	Coğrafi veriyi kapsayan en küçük daireyi bulmaktadır.
Centroid	Coğrafi verinin merkez noktasını bulmaktadır.
Difference	İki coğrafi veri arasındaki farkları bulmaktadır.
Distance	İki coğrafi veri arasındaki mesafeyi hesaplamaktadır.
Envelope	Coğrafi veriyi kapsayan en küçük kareyi bulmaktadır.
Intersection	İki coğrafi verinin kesişimlerini bulmaktadır.
Reverse	Coğrafi verinin koordinatlarını ters çevirip döndürmektedir.
Scale	Coğrafi veriyi istenilen miktarda ölçeklendirebilmektedir.
Union	İki coğrafi verinin birleşimlerini bulmaktadır.

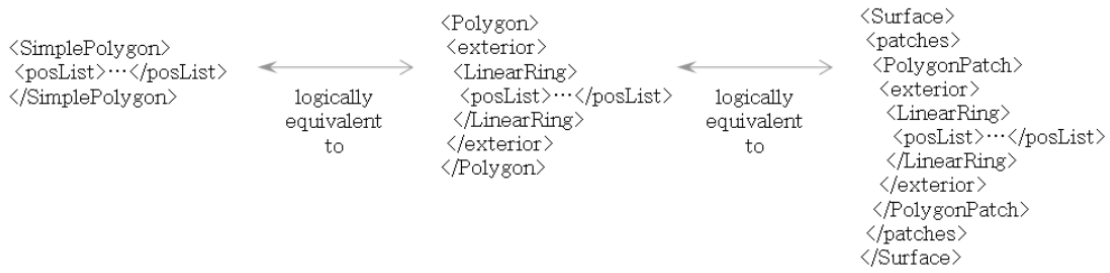
3.1.3. İletişim Protokolleri

Sunucu-istemci mimarisinin önemli bileşenlerinden biri de iletişim protokolleridir. Bu protokoller isteklerin ve cevapların doğru, güvenilir ve hızlı şekilde iletilmesini amaçlamaktadır. Uygulama geliştirmede iletişim sunucudan istemciye ya da tersi şeklinde olabilmektedir. Bu iletişim senkron ve asenkron olarak iki ana kategoriye ayrılabilir. Senkron iletişimde uygulama isteğin cevaplanması beklenmekte, cevap aldıktan sonra işlemine devam etmektedir. Cevabın uygulamanın geri kalanını etkilediği durumlarda kullanılmaktadır. Asenkron iletişimde ise uygulama isteğin cevaplanmasını beklemeden bir sonraki görevi yapmaya devam etmektedir. Cevap daha sonra bir tetikleyici olay ile uygulamaya aktarılmaktadır. İsteğin başka bir işlemin yapılması için gerekli olmadığı durumlarda kullanılmaktadır. Bu bölümde yaygın kullanılan iletişim yöntemleri ve veri yapıları listelenmektedir.

3.1.3.1. Xml ve Gml

XML (Extensible Markup Language) genişletilebilir işaretleme dili anlamına gelmektedir. Yapılandırılmış bilgileri temsil etmek üzere basit metin tabanlı bir biçim

sunmaktadır. Web uygulamalarında kullanılmak üzere ISO 8879 (SGML) standardından türetilmiştir. Dünya çapında yaygın bir kullanıma sahip olan XML uygulamalar arası iletişimi ve işbirliğini kolaylaştırmaktadır [73,74]. Uygulamalar birbirleri arasında bu standarda uygun şekilde oluşturulmuş XML formatındaki verilerle kolaylıkla iletişim kurabilmektedirler. GML (Geography Markup Language) ise OGC'nin geliştirdiği XML kurallarını kullanarak coğrafi özellikleri temsil eden bir işaretleme dilidir. OGC'nin basit poligonlar için standartlaştırdığı GML formatı Şekil 3.7 de gösterilmektedir. Bir poligonu tanımlamak için en az üç nokta gereklidir [75]. XML ve GML çözümlemeyi gerçekleştirebilen tüm yazılım dilleri bu standartlarla kendi programlama dilinden bağımsız şekilde iletişim kurabilmektedirler.

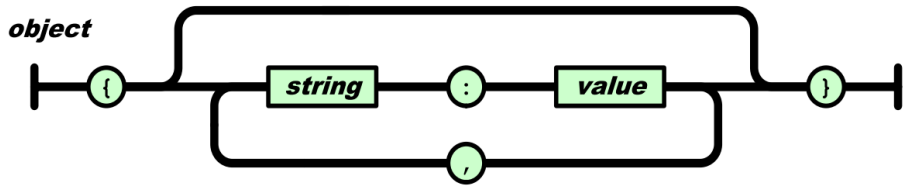


Şekil 3.7. Poligonlar için örnek GML formatı [75]

3.1.3.2. Json ve GeoJson

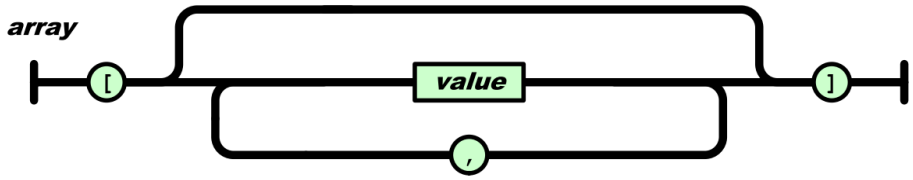
Json (JavaScript Object Notation) ECMA-404 (JSON Data Interchange Format) standardında tanımlanmış bir veri değişim formatıdır. Okunup yazılabilmesinin kolay olması sebebiyle uygulamalar arası iletişimde sıkça kullanılmaktadır. Programlama dillerinden bağımsız olmasından dolayı Json formatını çözebilen her dil tarafından işlenebilmektedir. Json formatı iki yapı üzerine kurulmuştur.

İsim-Değer Çifti Koleksiyonu: Programlama dillerinde object, record, struct, dictionary, hash table, keyed list şeklinde tanımlanan değerler için kullanılmaktadır. Bir nesne (object), isim-değer çiftlerinin sırasız birleşiminden oluşur. Nesne tanımlaması, "{" ile başlar ve "}" ile biter. Her isimden sonra " : " gelir ve isim-değer çiftleri "," işareti ile ayrılır [76].

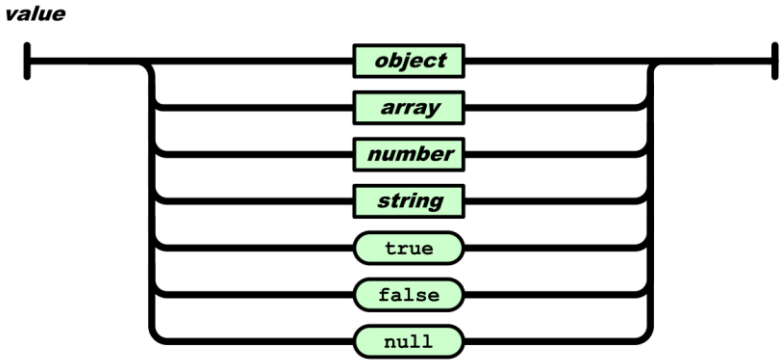


Şekil 3.8. ECMA Json Nesne (Object) Standardı [77]

Sıralı Değer Listesi: Programlama dillerinde tanımlanan array, vector, list veya sequence tipindeki değerleri tanımlamak için kullanılmaktadır. Bir sıralı değer listesi "[" ile başlar ve "]" ile biter. Değerler birbirlerinden "," işareti ile ayrılır [76]. Şekil 3.9 ve 3.10 da görülen nesne ve dizi standartlarının içinde bulunan değerler (value) object, array, number, string , true, false, veya null olabilmektedir.



Şekil 3.9. ECMA Json Dizi (Array) Standardı [77]



Şekil 3.10. ECMA Json Değer (Value) Standardı [77]

Json dili bir önceki kısımda anlatılan XML diline benzemektedir. İkisi de veri değişimi için kullanılabilmektedir. Json XML'e göre daha veri odaklıdır. Bu durum Json'ı yapısal verilerde kullanılmasını mümkün kılarken XML hem veri hem de doküman odaklı olabilmektedir. Bununla birlikte Json bir işaretçi dili (markup language) değildir. Sadece verinin sunumuyla ilgilenir. Json veri tipleri önceden tanımlı (object, string, number vb..) olmasına rağmen XML dilinde veri tipi her şey olabilir hatta veri tipi olmayabilir. Json formatının tanımlanması, üretilmesi ve işlenmesi XML'e oranla daha kolaydır [78].

GeoJson formatı (RFC 7946) Internet Engineering Task Force (IETF) tarafından yayımlanmış ve Json yapısında coğrafi veri tanımlanabilmesini sağlamaktadır. Coğrafi veriyi temsil etmesi için Json formatı içine birçok coğrafi veri tipi tanımlamıştır. Bu veri tipleri "Point", "MultiPoint", "LineString", "MultiLineString", "Polygon", "MultiPolygon" ve "GeometryCollection" olmak üzere yedi tanedir ve coğrafi koordinat referans sistemi World Geodetic System 1984 (WGS84) olarak seçilmiştir[79]. GeoJson veri tipleri ve örnekleri Çizelge 3.5'te verilmiştir. Her bir GeoJson öğesi verinin tipinin belirlendiği "type" ve koordinatlarının bulunduğu "coordinates" etiketlerini içermelidir.

Çizelge 3.5. GeoJson Coğrafi Veri Tipleri ve Örnekleri.

GeoJson Coğrafi Veri Tipi	**Örnek**
Point	{ "type": "point", "coordinates": [10.0,11.2] }
LineString	{ "type": "Linestring", "coordinates": [[10.0,11.2], [10.5, 11.9]] }
Polygon	{ "type": "Polygon", "coordinates": [[[10.0, 11.2], [10.5, 11.9], [10.8, 12.0], [10.0, 11.2]]] }
MultiPoint	{ "type": "MultiPoint", "coordinates": [[10.0, 11.2], [10.5, 11.9]] }
MultiLineString	{ "type": "MultiLinestring", "coordinates": [[[10.0, 11.2], [10.5, 11.9]], [[11.0, 12.2], [11.5, 12.9], [12.0, 13.0]] }

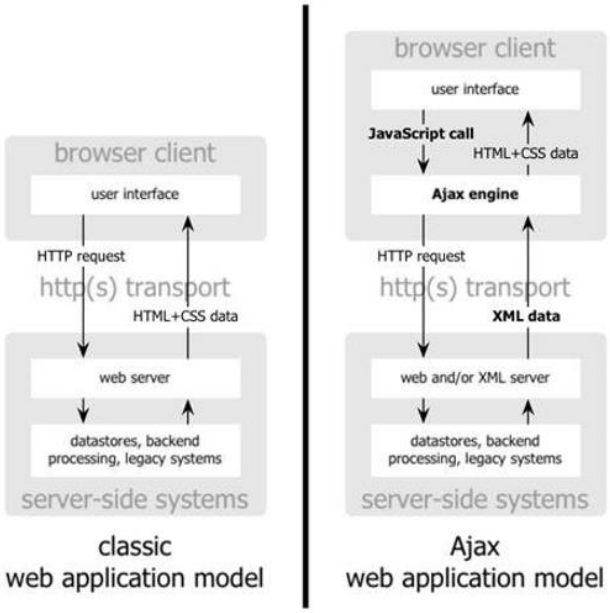
MultiPolygon	{ "type": "MultiPolygon", "coordinates": [[[[10.0, 11.2], [10.5, 11.9], [10.8, 12.0], [10.0, 11.2]]], [[[9.0, 11.2], [10.5, 11.9], [10.3, 13.0], [9.0, 11.2]]]] }
GeometryCollection	{ "type": "GeometryCollection", "geometries": [{ "type": "Linestring", "coordinates": [[10.0, 11.2], [10.5, 11.9]] }, { "type": "Point", "coordinates": [10.0, 20.0] }] }

3.1.3.3. Ajax

Ajax (Asynchronous JavaScript and XML) istemci tarafında sunucu ile iletişim kurmak için geliştirilmiş bir web teknolojisidir. Sunucu istemci arasında asenkron bir şekilde veri alışverişi yapabilmektedir. Asenkron iletişim yapılması sebebiyle web sayfaları tekrar yüklenmeden (reload) güncellenebilmektedir. İsminden de anlaşılacağı üzere Ajax JavaScript tabanlı bir teknolojidir ve XML formatında veri transferi yapabilmektedir. Stackoverflow topluluğunun 2020 anketine göre JavaScript %67,7 ile dünya genelinde en çok kullanılan web teknolojisidir[80]. 1990'lı yılların başında World Wide Web (WWW) hizmete sunulmasıyla web teknolojileri gelişmeye başlamıştır. Başlarda günümüzde hala kullanılmakta olan HTML diliyle başlayan web teknoloji serüveni 1995 yılından sonra Netscape Navigator internet tarayıcısının çıkmasıyla birlikte script dillerini de içine alarak devam etmektedir. Web sayfalarının statik yapısını dinamik hale getirmek için çalışan Netscape Communications Corporation ve Sun Microsystems 1995 yılı sonunda Javascript'in ilk versiyonunu yayımlamışlardır[81]. Javascript istemci tarafında, sunucuyla tarayıcı üstünden iletişime geçerek web sayfalarının içeriklerini güncelleyebilmektedir. İstemci tarafında işlem yapması sebebiyle çeşitli güvenlik açıklarına sebep verse de bu konu üzerine iyileştirme çalışmaları devam etmektedir. JavaScript (ECMA-262) ECMAScript standartları baz alınarak geliştirilmektedir. Bu standartta yazılım dilbilgisi kuralları, standart veri tipleri, fonksiyon ve sınıf tanımlama gibi işlemlerin standartları oluşturulmuştur[82].

Web sayfaları içine yazılan Ajax çağrıları statik HTML sayfalarını sunucuyla asenkron olarak haberleştirerek istek verilerinin sayfayı tekrar yüklemeye gerek bırakmadan güncelleyebilmekte, bu esnada kullanıcı web sayfasının başka bir bölgesinde farklı bir işlemi aynı anda yapabilmektedir. Şekil 3.12. de görüldüğü üzere klasik web

uygulamalarında sunucuyla haberleşmek için bir tetikleyici (buton ya da link) yardımıyla istek doğrudan sunucuya gönderilmektedir. Sunucu ise cevap olarak HTML sayfasının tamamını geri döndürmektedir. Bu sebeple klasik web sayfası isteklerinde sayfanın güncellenebilmesi için tekrar yüklenmesi gerekmektedir[83].



Şekil 3.11. Klasik ve Ajax Web Teknolojilerinin karşılaştırılması [84]

3.1.4. İstemci Tarafı Uygulama Geliştirme Çerçeveleri

İstemci-Sunucu web mobil teknolojilerinde istemci tarafı uygulamalar, kullanıcının doğrudan etkileştiği kısım olması sebebiyle kullanım kolaylığı, performans, güvenlik, kaynak yönetimi gibi konular önemli hale gelmektedir. İstemci tarafı için geliştirilen mobil uygulamalar üç başlık altında toplanabilmektedir. Bunlar mobil cihazlar için özel üretilen ve uygulama şeklinde geliştirilen doğal (native) uygulamalar, web tarayıcıları için geliştirilmiş fakat mobil tarayıcılarda da çalışabilen esnek (responsive) uygulamalar ve her iki yöntemi aynı anda karşılayabilen hibrit uygulamalardır.

3.1.4.1 Doğal (Native) Uygulamalar

Günümüzde mobil cihazlarda yaygın olarak kullanılan işletim sistemi Android ve iOS'tur. Mobil cihazlar için geliştirilen uygulamalar genellikle aplikasyon olarak tanımlanmaktadır. Android ve iOS işletim sistemli cihazlar için geliştirilen aplikasyonlar teknolojilerinin farklı olması sebebiyle ayrı ayrı geliştirilmektedir. Mobil cihazın teknolojik yeteneklerini (sensörler, kamera, dokunmatik ekran vb.) kullanabilmek için doğal (native) uygulamalar geliştirmek gerekmektedir.

Android işletim sistemi için geliştirilecek aplikasyonlar genellikle Java dilinde geliştirilebilmekle birlikte Kotlin ve C++ gibi diller de kullanılabilmektedir. Aplikasyon geliştirmek için Android SDK'ya ihtiyaç duyulmaktadır. Android SDK içinde barındırdığı araçlar sayesinde Java, Kotlin veya C++ dillerinde geliştirilmiş olan mobil uygulamayı Android işletim sisteminin çalıştırabileceği APK (Android Package Kit) haline paketleyebilmektedir. Android işletim sistemi üstünde çalışan her aplikasyona bir Linux user-id vermektedir. Bu sayede her aplikasyonun, cihazın teknolojik yeteneklerine erişim izinleri ayrı ayrı tanımlanabilmektedir. Her aplikasyon kendi sanal makinesi içinde çalıştırılmakta bu sebeple diğer tüm aplikasyonlardan izole olabilmektedir. Bu yetenekler cihaz güvenliği için önem arz etmektedir. Android aplikasyonları dört ana bileşene sahiptir. Bunlardan ilki kullanıcıyla etkileşim kuran Activies bileşenidir. Uygulama için tasarlanan her ara yüz bir Activies elemanıdır. Uzun süreli ve sürekli çalışan uygulamaların diğer uygulamaları kesmemesi ve beraber çalışabilmeleri için arka planda çalışabilen Services bileşeni bulunmaktadır. Bir diğer bileşen ise Broadcast Receivers'tır. Bu bileşen tanımlandığı uygulamanın çalışmasına gerek kalmadan tamamlayabilmektedir. Örneğin bir aplikasyon üzerinden ileri saatte çalması için bir alarm tanımlandığında Broadcast Receiver bu alarmı aplikasyonun çalışır olmasına gerek duymadan zamanı geldiğinde çalıştırabilir. Son bileşen ise Content Providers'tır. Bu bileşen ise uygulama verilerini depolamak için kullanılmaktadır. SQLite veri tabanı kullanarak kullanıcı verilerini depolayabilmekte ve istenildiğinde ulaşılmasını sağlayabilmektedir[85].

Apple firması tarafından yayımlanan Xcode geliştirici araçları kullanılarak swift dilinde aplikasyon geliştirilebilmektedir. Otomatik (atık kaynak temizleme) garbage collection sayesinde cihaz hafızasının kullanımını en düşük seviyede tutmaya çalışmaktadır. Basit yazım kurallarına sahip olması sebebiyle yazılan kod kolay anlaşılabilmektedir. Tüm değişkenler kullanılmadan önce ilk değerleri verilerek ve diziler

ve sayısal değerler için taşma kontrolü yapılarak güvenli kod geliştirme yapılması sağlanmaktadır. Güvenlikle ilgili bir diğer özelliği ise hiçbir sınıfın "null" değer alamamasıdır. Bunu çalışma zamanında değil derleme zamanında kontrol ederek aplikasyonun çalışırken hata vermesine engel olmaktadır. Açık kaynak kodlu olması sebebiyle Apple ve uygulama geliştirme toplulukları tarafından sürekli olarak güncellenmektedir[86].

Günümüzde birçok cep telefonu, tablet hatta saat gibi giyilebilir teknolojilerde konum bilgisi alma yeteneği bulunmaktadır. Konum bilgisi aplikasyonlar tarafından, trafik, yol bulma, spor aktiviteleri, coğrafi bilgi sistemi uygulamaları hatta bu tezin konusu olan taşınmaz değerleme uygulamalarında kullanılabilmektedir. Android işletim sistemleri içinde bu yeteneği kullanmak için Location sınıfı bulunmaktadır. Bu sınıf sayesinde cihazın bulunduğu enlem ve boylam, iki nokta arasında mesafe, mevcut konumun yataydaki doğruluk hassasiyeti, rakım ve hız bilgisi gibi değerler alınabilmekte ve geliştirilen uygulamada kullanılabilmektedir[87]. iOS işletim sistemine sahip cihazlarda ise konum bilgisini kullanmak için Core Location servisi geliştirilmiştir. Bu servis sayesinde mobil cihaz çevrede bulunan ve bağlanabildiği tüm GPS, Wi-Fi, Bluetooth gibi cihazlardan mevcut konumunu alabilmektedir. Hassasiyet ayarı belirtilerek mobil cihazın konumunu takip edebilmekte, tanımlı ve belirli bir bölgeye girildiğini anlayıp uyarı verebilmekte, yakında bulunan mekanları tarayabilmektedir[88].

3.1.4.2 Esnek (Responsive) Uygulamalar

Esnek uygulamalarda kullanılan cihazın ekran büyüklüğü ve çözünürlüğü tespit edilerek otomatik olarak yeniden boyutlandırma (resize), saklama (hide), genişletme (enlarge) işlemleri uygulanmaktadır. Bu yetenekler için temelde HTML ve CSS kullanılmaktadır. CSS (Cascading Style Sheets) web dokümanlarına stil eklemek için kullanılan bir teknolojidir. HTML sayfalarının içine yazılabileceği gibi farklı bir dosyada da saklanıp HTML sayfaları içinden çağrılabilmektedir. Örnek bir css kodu aşağıdaki şekilde tanımlanmaktadır. Html sayfası içine eklenen bu tanım sayesinde, sayfanın ana kısmının arka plan rengi ve yazı rengi değiştirilmiştir [89].

```
<style type="text/css">
    body {
```

color: purple;

background-color: #d8da3d

}

</style>

Esnek bir web uygulaması yapmak için ***<meta name="viewport" content="width=device-width, initial-scale=1.0">*** etiketi siteye yerleştirilmelidir. Bu etiket sayesinde sayfanın görüntülenebilir alanı cihazın genişliğine eşitlenmiş olmaktadır. Daha sonra sayfa içinde kullanılacak diğer bileşenlerin genişliklerine yüzdelik değerler verilerek cihazın boyutuna göre otomatik olarak ayarlanmış olmaktadır. Örnek olarak **<img src="img.jpg" style="width:50%;">** etiketi sayfanın içine bir resim eklemek için kullanılmıştır. Meta etiketiyle cihazın ekran genişliği daha önceden tanımlandığı için resim width:50% sitili ile cihaz ekranın yarısına sığacak şekilde otomatik ayarlanmaktadır. Bu sayede bilgisayar, tablet veya cep telefonu için ayrı ayrı uygulama geliştirmek yerine web tarayıcısı üzerinden aynı uygulama sitillerine göre kendini otomatik ayarlayabilmektedir[90].

3.1.4.3 Hibrit Uygulamalar

Hibrit uygulama geliştirme mobil, masaüstü ve web uygulamalarını tek çatı altında toplamak için tasarlanmıştır. Tek kod tabanı kullanılarak hem masaüstü hem mobil hem de web için aynı anda uygulama geliştirmeyi sağlamaktadır. En yaygın kullanılan hibrit uygulama geliştirme çerçevesi Google tarafından geliştirilen Flutter'dır.

Flutter yazılım geliştirme sürecini değiştirerek tek seferde mobil, web ve masaüstü uygulamaları geliştirme, test etme ve derleme imkânı sunmaktadır. Google tarafından açık kaynak kodlu olarak geliştirilen Flutter sayesinde hızlı ve esnek uygulamalar yaratılabilmektedir. Windows, macOs, Linux ve Chrome OS işletim sistemlerinde kullanılabilmektedir. Android Studio ve VS Code gibi yazılım geliştirme araçları üzerinden Flutter projeleri oluşturulabilmektedir. Flutter temel olarak widget'lar üzerinden çalışmaktadır. Tanımlanmış widget'lar Android, iOS ve web için ortak kullanıma müsait şekilde geliştirilmiştir. Bu sayede geliştirilen uygulama Android, iOS ve web için ayrı ayrı derlenebilmekte ve her biri için çalıştırılabilir bir uygulama sunmaktadır. Flutter teknolojisinin merkezinde Flutter motoru (engine) bulunmaktadır.

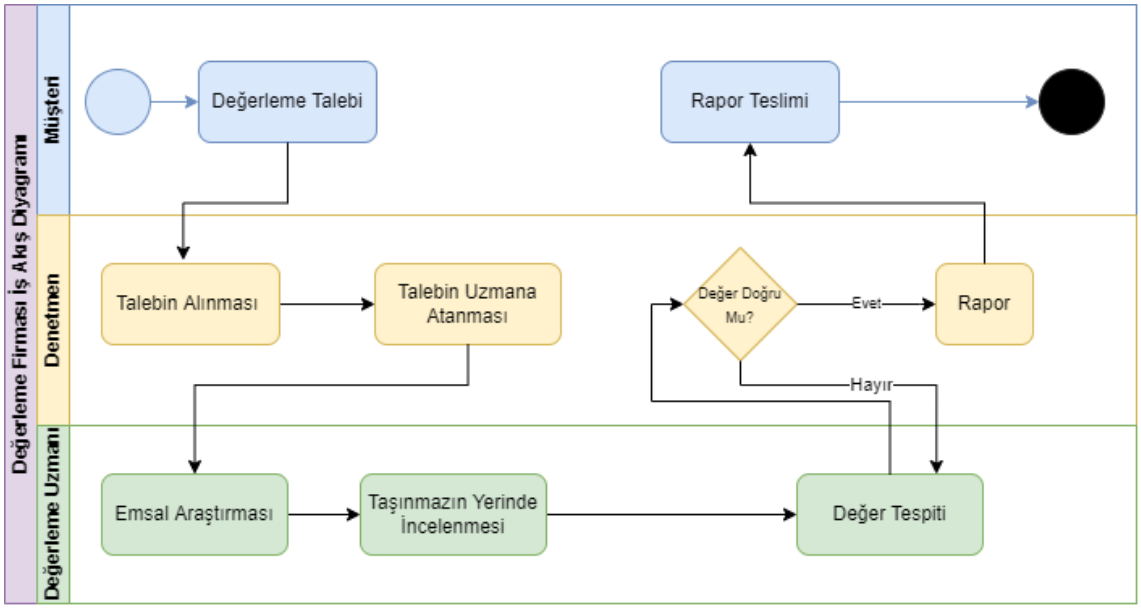
Çoğunluğu C++ dilinde yazılmış bu motor metin işlemleri, dosya işlemleri, ekran tasarımları gibi merkezi işlemlerden sorumludur. Flutter, Dart yazılım geliştirme dilini kullanmaktadır. Flutter motoru geliştiricilerin Dart dilinde kodladığı uygulamayı daha düşük seviye bir dil olan C++ diline çevirerek işlemlerini yapmaktadır. Geliştiriciler Flutter motoru ile Flutter çerçevesi (framework) üzerinden haberleşmektedir[91].

4. YÖNTEM

Bu bölümde geliştirilecek uygulamanın kavramsal tasarımı anlatılmaktadır. Sistemin temel yetenekleri ve kullanım senaryolarının yanı sıra başarı kriterleri listelenmektedir. Daha sonra uygulama için gerekli olan veriler ve kullanılacak emsal değeri üretim yöntemleri açıklanmaktadır. Veri tabanı, sunucu yazılımı ve istemci yazılımı tasarımları açıklandıktan sonra test ve doğrulama senaryoları hakkında bilgi verilmektedir.

4.1. Kavramsal Tasarım

Değerleme firmalarında iş akışı Şekil 4.1.'de gösterildiği gibi gelen değerleme talebinin bir uzmana atanması şeklinde başlamaktadır. Daha sonra değerleme uzmanı, taşınmazı yerinde inceleyerek değere etki edecek özellikleri tespit etmektedir. Taşınmazın bulunduğu konumdaki komşular ve emlakçılar ile mülakatlar yaparak ve emlak alım satımının yapıldığı web sitelerini inceleyerek emsal araştırması yapmaktadır. Taşınmaz değerleme uzmanı nihai değere ulaştıktan sonra bunu bir denetmenin onayına sunmaktadır. Denetmen onayından sonra değerleme raporu son halini almaktadır.



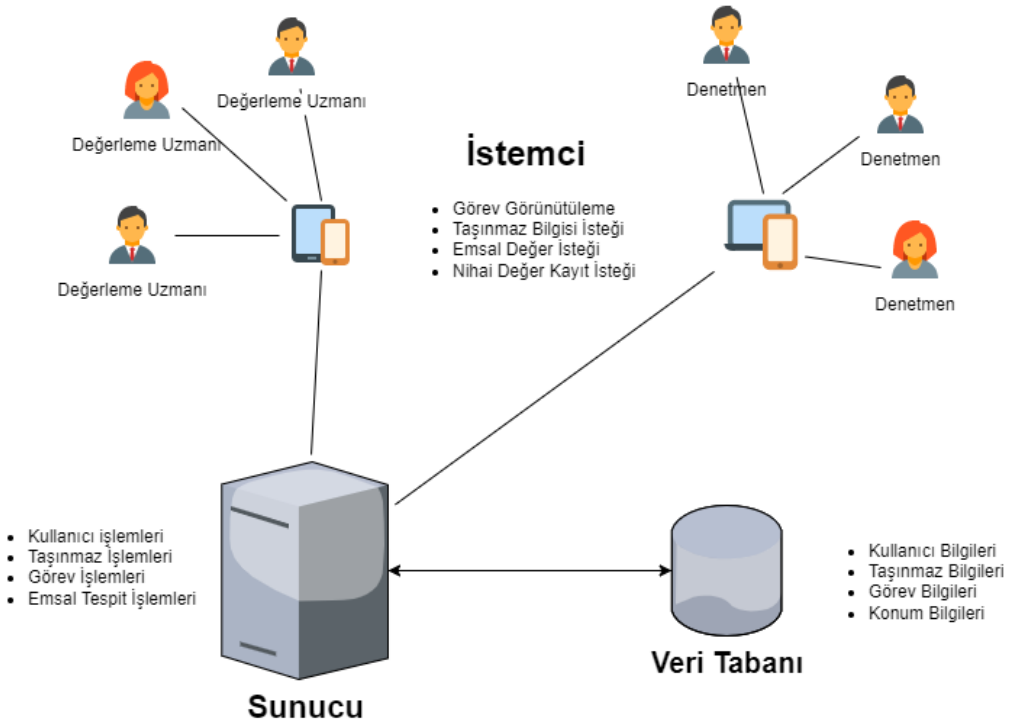
Şekil 4.1. Değerleme Firması İş Akış Diyagramı

Tez kapsamında geliştirilecek uygulamaya “Taşınmaz Emsal Tespit Sistemi” isminin baş harflerinin kısaltılması olan TETS ismi verilmiştir. TETS uygulamasının temel amacı, değerleme uzmanlarına değerleyecekleri taşınmaz için etrafındaki benzer özellikteki taşınmazları kullanarak emsal değer tahmini sunmak ve uzmanın durumsal farkındalığını arttırarak karar desteğine yardımcı olmaktır. Bu sayede değerleme uzmanının emsal tespit süresinin kısaltılması ve olası emlakçı ve emlak alım satım sitelerindeki gerçeği yansıtmayan bilgilerin değere etkisinin azaltılması düşünülmektedir. Uygulama içinde uzman ve denetmen rolünde kullanıcılar eklenebilmelidir. Denetmen kullanıcısı değerlenecek taşınmazla ilgili temel bilgileri ve konum bilgisini kaydedecektir. Kaydettiği bu taşınmazın değerleme görevini bir uzmana ve kontrol etmesi için bir denetmene atayabilecektir. Görev listesinden kendisine atanmış görevlerini gören uzman kullanıcısı taşınmazın değerine esas bilgilerini güncelledikten sonra, uygulama o taşınmaz için emsal değeri üretip uzmana sunacaktır. Uzman bu tahmin edilmiş emsal değeri yardımıyla oluşturduğu nihai değeri sisteme kaydedip denetmen oluruna sunacaktır. Denetmen kullanıcısı ise uzman tarafından kontrole gönderilen değer ile birlikte sistemin tahmin ettiği emsal değerini görerek karşılaştırabilecektir. Kontrolü tamamlanmış değer kesin veri olarak denetmen tarafından sonraki otomatik emsal tahminlerinde kullanılmak üzere kaydedilecektir. Bu sayede sistemin her kaydedilen kesinleşmiş değer ile beslenerek tahmin doğruluğunu artıracağı düşünülmektedir.

TETS uygulaması web tabanlı ve esnek yapıda geliştirilerek hem bilgisayar internet tarayıcılarında hem de mobil cihazlarda kullanılması sağlanacaktır. Uygulamanın tasarımı için istemci-sunucu mimarisi kullanılacaktır. Şekil 4.2.’de görüldüğü gibi uygulama verileri konumsal veriyi de tutabilecek ilişkisel bir veri tabanı yönetim sisteminde tutulacaktır. Sunucu tarafı veri tabanı işlemleri, istenilen konum için emsal değerinin üretilmesi ve istemci uygulaması tarafından gelecek isteklerin karşılanmasından sorumlu olacaktır. İstemci tarafında denetmenler değeri tespit edilmek istenen taşınmaz bilgilerini sisteme girebilecek ve değerleme uzmanına görev atayabilecektir. Uzmanlar ise kendilerine atanan görevleri listeleyebilecek, taşınmaz için değere esas bilgileri girebilecek, emsal değerini görüntüleyebilecek ve nihai taşınmaz değerini kaydedebilecektir.

Bölüm 3’te Mobil Web-CBS Teknolojileri hakkında bilgi verilmiştir. Bu teknolojilerden PostgreSQL TETS uygulamasının veri tabanı yönetim sistemi olarak

seçilmiştir. Bu seçimde PostgreSQL veri tabanı yönetim sisteminin açık kaynak kodlu olmasının yanı sıra PostGIS eklentisi sayesinde mekânsal veriyi kolayca işleyebilmesidir. Bununla beraber PostGIS eklentisinin shape dosyalarını okuyup tablolara kaydetme ve içinde coğrafi veri bulunan tabloları shape formatında dışarı aktarma özellikleri uygulama geliştirilirken yardımcı olacağı düşünülmüştür. Yazılım teknolojisi ise Django olarak belirlenmiştir. Python dili makine öğrenmesi ve yapay zekâ uygulamaları için geliştirilmiş geniş bir kütüphane setine sahip olması sebebiyle seçilmiştir. Ayrıca Django çerçevesinin GeoDjango eklentisi, harita ile etkileşimi kolaylaştırmaktadır. Uygulama içinde kullanılan haritalar açık kaynak kodlu olması sebebiyle OpenStreetMap olarak belirlenmiş ve harita üstünde işlemler gerçekleştirebilmek için Javascript Leaflet kütüphanesi kullanılmıştır.



Şekil 4.2. TETS Uygulaması İstemci- Sunucu Mimarisi

4.2. Veri Tanımları

TETS uygulaması kullanıcı etkileşimini mümkün kılan bir uygulama olarak geliştirileceği için geniş bir veri tipi yelpazesine sahip olacaktır. İlişkisel veri tabanlarında

veriler, tiplerine göre saklanmaktadır. Bu sayede verilerin geçerlilik kontrolleri kolaylaşmakta ve indekslemeler sayesinde veriye erişim hızlanabilmektedir. TETS uygulaması kullanıcı, görev, taşınmaz gibi modüllerden oluşacaktır. Bu modülleri veri tabanında modelleyebilmek için birçok veri tipine ihtiyaç duyulmaktadır. Veri tabanı yönetim sistemlerinde veriler yaygın olarak sayısal (integer, double, float, numeric), metin (string, text), tarih-saat (date, datetime, timestamp), mantıksal (bool, boolean) tiplerinde saklanmaktadır. Bunun yanında konum verisini saklamak için coğrafi (point, polygon) veri tipleri bulunmaktadır.

TETS uygulamasında kullanılacak modeller veri tabanı yönetim sistemi üzerinde tablolar halinde tutulacaktır. Uygulamanın kullanacağı veriler kullanıcı, yetki, taşınmaz ve görev tablolarında saklanacaktır.

Kullanıcı tablosunda kullanıcı adı, parola, ad, soyad, mail adresi alanları metin, aktiflik durumu mantıksal, indekslemek için kullanılacak id sütunu sayısal ve kullanıcının oluşturulma zamanı ise tarih-zaman tipinde tanımlanacaktır. Yetki tablosunda kullanıcı id'si sayısal, yetki tanımı metin tipinde olacaktır. Taşınmaz tablosu taşınmaz ile ilgili bilgileri saklayacaktır. Bu bilgilerden bulunduğu kat, yüzölçümü, sokak genişliği, önemli noktalara uzaklığı (üniversite, park, şehir merkezi vb.) gibi değerler sayısal veri tipinde saklanacaktır. Ada, parsel, mahalle, il, ilçe gibi bilgiler metin, taşınmazın konumu ise coğrafi veri tipinde saklanacaktır. Görev tablosu ise değerlemesi yapılacak taşınmazın id'sine referans verecek şekilde görevin atandığı uzman ve denetmen bilgilerini saklayacaktır.

4.3. Emsal Değer Üretim Yöntemi

TETS uygulamasının temel amacı sahadaki değerleme uzmanının değerlenecek taşınmaz için emsal değerini hızlıca hesaplamasıdır. TETS yazılımı farklı emsal değerleme yöntemlerinin kolay entegre edilebileceği nesne tabanlı bir yaklaşımla tasarlanmalıdır. Bu kapsamda yeni emsal değerleme yöntemlerinin kolayca entegre edilebilir olması için tasarım yapılmalıdır.

Bu çalışma kapsamında değerleme yöntemi eklentisi olarak öncelikle En küçük Kareler (Ordinary Least Square -OLS) metodunun bölgesel olarak kullanılması planlanmıştır. Ek olarak Ters Uzaklık Ağırlıklandırma yöntemi (Inverse Distance

Weighted -IDW) ve Uzamsal Gecikmeli Regresyon yöntemi (Spatially Lagged Regression - SLR) tasarım ve kavram doğrulama amacıyla geliştirilecektir.

4.3.1. En Küçük Kareler Yöntemi (OLS)

TETS uygulamasının çözmek istediği problem doğruluğu bilinen emsal değerlerini kullanarak yeni bir taşınmazın değerini tahmin etmektir. Veri setinde bulunan taşınmaz değerleri, taşınmaz yüzölçümü, bulunduğu kat, parka uzaklığı, üniversiteye uzaklığı, asansör durumu, sokak genişliği, yapı kullanım izin belgesi durumu, otopark durumu, bağımsız salon durumu ve şehir merkezine olan uzaklık bilgilerini de barındırmaktadır. Buna göre bir taşınmazın değeri, değere etki eden özniteliklerin doğrusal bir fonksiyonu olarak ifade edilebilir.

$$P_i = \alpha + \sum_k x_{ik}\beta_k + \varepsilon_i$$

Modelde P taşınmazın değerini, α sabit katsayıyı, x taşınmazın değerine etki eden özellikleri, β değere etki eden özelliklerinin katsayılarını, ε ise hatayı temsil etmektedir. Bu denklem sistemini oluşturan katsayılar kestirilebildiğinde sisteme yeni eklenecek taşınmazlar için tahmin üretmek amacıyla kullanılabilir. Katsayılar halihazırda sistemde bulunan emsal değerleri ve öznitelikler kullanılarak en küçük kareler yöntemi kestirilebilir. Bu çalışmada emsal değerinin tahmini için bölgesel modelleme yaklaşımı benimsenmiştir. Bu yaklaşımda temel varsayım her bölgede emsal değerine etki eden faktörlerin ve katsayılarının farklı olabileceğidir. Bu kapsamda değerlendirilecek bir taşınmazın en yakın N komşusunun her verisi kullanılarak yukarıda verilen emsal tahmin denklemi bir denklem sistemi olarak aşağıdaki gibi yazılabilir.

$$\boldsymbol{p} = \boldsymbol{X\beta} + \boldsymbol{e}$$

Burada $\boldsymbol{p}$ komşuluktaki taşınmazların normalize edilmiş emsal metrekare birim değerlerini içeren vektördür, $\boldsymbol{X}$, her satırında komşuların özniteliklerini barındıran tasarım matrisi, $\boldsymbol{\beta}$, ise kestirilmesi gereken özniteliklerin katsayılarını ifade etmektedir. En küçük kareler yöntemi ile katsayılar bölgesel olarak aşağıdaki gibi kestirilebilir:

$$\widehat{\boldsymbol{\beta}} = (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T p}$$

Elde edilen katsayılar ile tahmin edilmesi beklenen taşınmaza ait öznitelikler kullanılarak emsal değeri tahmin edilebilir. Yazılım sürekli olarak seçilen bir taşınmaza ait en yakın

komşular ile yukarıdaki açıklanan yöntemler yeni katsayılar öğrenecek şekilde tasarlanacaktır. Bu kapsamda sisteme yeni eklenen veriler ile doğruluğun giderek arttırılması ve kendi kendine öğrenen bir yazılım çerçevesi oluşturulması hedeflenmiştir.

4.3.2. Ters Uzaklık Ağırlıklandırma (IDW)

IDW yöntemi dayanak noktalar ile kestirim yapılacak nokta arasındaki mesafenin tersinin ağırlık olarak belirlendiği ve bu ağırlıklara dayalı olarak komşuluktaki emsal değerlerinin ağırlıklı ortalamasının emsal değer tahmini olarak kullanıldığı bir yöntemdir. Bu yöntem de tahmin edilmek istenen konum ile değeri bilinen noktalar arasında mesafe arttıkça ilgili değerin etkisinin azalacağı varsayımı yapılmıştır [92].

$$P_k = \frac{\sum_i^n \left(\frac{P_i}{d_i^k}\right)}{\sum_i^n \left(\frac{1}{d_i^k}\right)}$$

Yukarıdaki formülde P_k tahmin edilmek istenen değeri temsil etmektedir, d_i^k ise emsali tahmin edilecek taşınmazın, değeri bilinen komşuluktaki diğer taşınmazlara uzaklıklarını temsil etmektedir. IDW yönteminde değeri bilinen taşınmazların değerleri, emsali hesaplanmak istenen taşınmazlara uzaklıklarıyla ters orantılı olacak şekilde ağırlıklandırılarak toplanmakta ve hesaplanan bu ağırlıkların toplamına bölünerek normalize edilmektedir. Bu sayede emsal hesabı yapılacak taşınmazın değerinin yakın olan taşınmazların değerinden aykırı olmaması sağlanmaktadır.

4.3.3. Uzamsal Gecikmeli Regresyon (SLR)

OLS yönteminde şehir merkezi veya alışveriş merkezlerine yakınlık değişkenleri dışında konuma bağlı bir ifade bulunmamaktadır. IDW yöntemi ise taşınmazın diğer özelliklerini dikkate almadan sadece mesafeye bağlı bir ağırlıklı ortalama hesaplamaktadır. Literatürde uzamsal gecikmeli regresyon (SLR - Spatially Lagged Regression [93]) olarak geçen yöntemde ise hem bir taşınmazın yakın komşularının ağırlıklı ortalaması ile konuma bağlı değişim hem de taşınmazın diğer öznitelikleri dikkate alınarak bir regresyon modeli oluşturulmaktadır. SLR yöntemi genel anlamda,

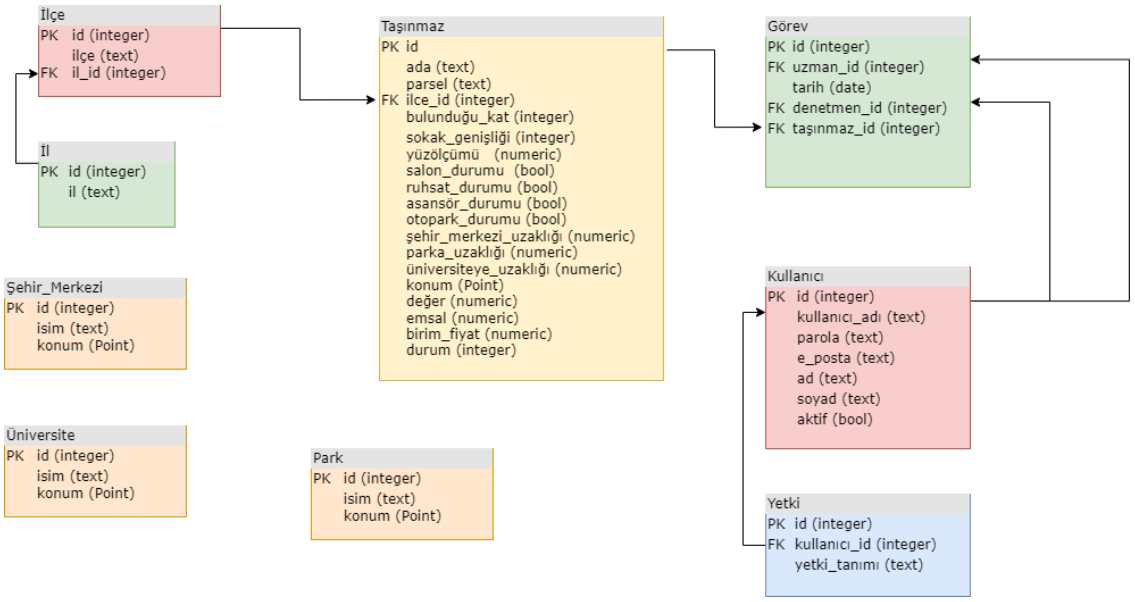
değerlenecek taşınmaza en yakın taşınmazların emsal değerlerinin uzamsal bir ağırlıklandırma ile bulunan ortalamasını uzamsal gecikmeli değişken olarak ifade etmektedir. Bu değişkene ilişkin katsayı yine OLS yöntemi ile kestirilebilmektedir. Genel denklem aşağıdaki gibi yazılabilir.

$$P_i = \alpha + \sum_k x_{ik}\beta_k + \beta_l \sum_j w_{ij}P_j + \epsilon_i$$

Bu ifadede $\sum_j w_{ij}P_j$ uzamsal gecikmeli değişkeni (spatially lagged variable), w_{ij} genelde uzaklığa bağlı olarak ifade edilen ağırlıkları (bu çalışmada IDW yöntemi ile aynı alınmıştır), β_l ise uzamsal gecikmeli değişkenin katsayısını ifade etmektedir.

4.4. Veri Tabanı Tasarımı

Uygulama veri tabanı Şekil 4.3.'te görüldüğü gibi kullanıcı, taşınmaz ve görev süreçleri ile ilgili verileri saklamak için tasarlanmıştır. Taşınmaz tablosunda taşınmazla ilgili tanım bilgileri (ada, parsel, il, ilçe), taşınmaz değerini etkileyen özellik bilgileri (bulunduğu kat, yüzölçümü, sokak genişliği, ruhsat, otopark ve asansör durumu), taşınmazın konum bilgisi, şehir merkezi, üniversite ve park alanına olan uzaklıkları bulunacaktır. Bunlarla beraber taşınmaz için hesaplanan emsal değeri, taşınmazın değeri ve birim fiyatlar taşınmaz tablosunda saklanacaktır. Uygulama içinde bir taşınmaz, yeni eklenmiş, uzmana değerleme için gönderilmiş, denetmene onay için gönderilmiş veya değeri tespit edilmiş durumlarında bulunabilecektir. Bu bilgiyi tutmak için taşınmaz tablosunun durum sütunu kullanılacaktır. Görev tablosunda ise değerlenecek taşınmaz ve görevin atandığı uzman, denetmen bilgileri saklanacaktır. Kullanıcı ve yetki tabloları kullanıcılar için oturum açma ve yetkilendirme bilgilerini saklayacaktır. TETS uygulamasında denetmen ve uzman rolleri bulunacaktır. Yeni eklenecek taşınmazların değerine etki eden şehir merkezi, üniversite ve park alanına uzaklık bilgisinin hesaplanabilmesi için bu yerlerin konum bilgisini saklayan tablolar tasarlanmıştır.



Şekil 4.3. Veri Tabanı Tasarımı

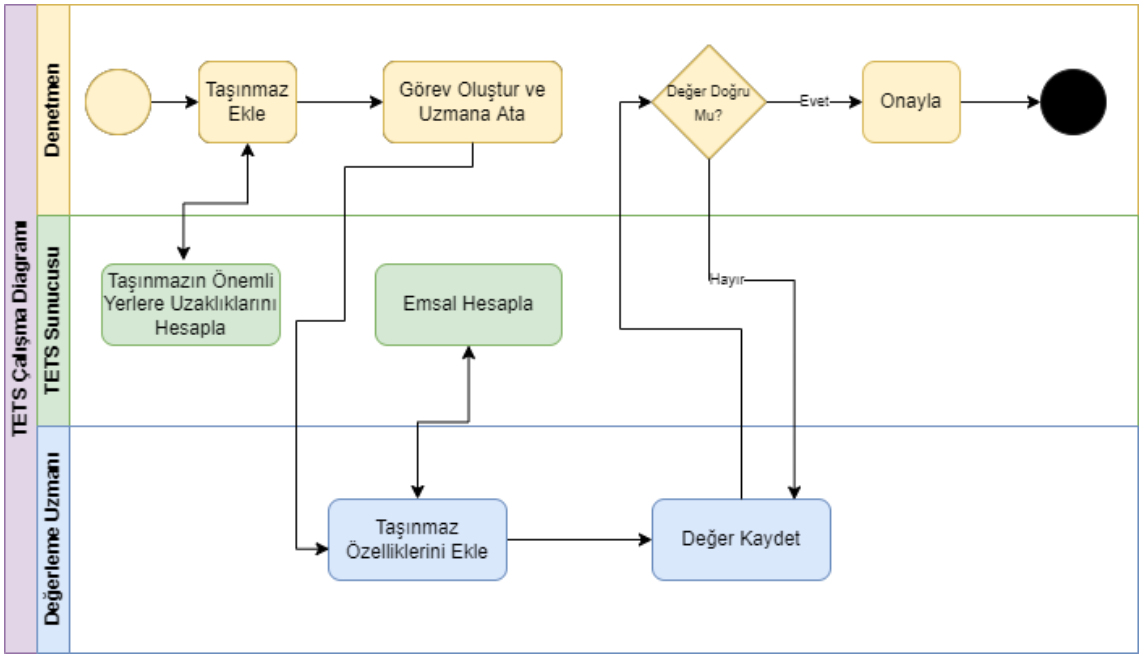
4.5. Sunucu Yazılımı Tasarımı

TETS uygulaması sunucu istemci mimarisi ile tasarlanacaktır. Sunucu uygulaması Django çerçevesi kullanılarak Python dilinde geliştirilecektir. Yazılım geliştirme ortamı olarak Visual Studio Code kullanılacaktır. Visual Studio Code Microsoft tarafından Windows, Linux ve MacOS için geliştirilen bir kaynak kodu düzenleyicisidir. https://code.visualstudio.com/ web sitesinden indirilip kurulum yapılabilmektedir. Daha sonra ücretsiz ve açık kaynaklı, Python ve R programlama dillerinin kullanımında paket yönetimini kolaylaştırmayı amaçlayan Anaconda paket yönetim sistemi kurulmuştur. Conda paket yönetim sistemi ile Django çerçevesinin çalışabilmesi için sanal bir ortam oluşturulmuştur. Bu sanal ortam içine TETS uygulaması için gerekli olan pysal, numpy, spreg gibi kütüphaneler eklenmiş ve yazılım geliştirme ortamı hazırlanmıştır.

Kullanıcı ekleme çıkarma yetkilendirme işlemleri sunucu tarafında gerçekleştirilecektir. Uygulama üzerinden eklenmek istenen kullanıcı bilgileri sunucu katmanına iletilip modeller aracılığı ile veri tabanına kaydedilecektir. Kullanıcının denetmen ya da uzman rolüne sahip olması da bu katman aracılığı ile belirlenecektir.

Taşınmaz bilgileri veri tabanına konumuyla birlikte eklenecektir. Harita üzerinden seçilen konum bilgisi taşınmazın diğer tanım bilgileri (ada, parsel, il, ilçe, mahalle vb.) ile birlikte kaydedilecektir. Bu veriler otomatik emsal tespit yönteminin

uygulanması için kullanılacaktır. Bununla birlikte kaydedilecek taşınmaz konumunun değere etki edecek okul, park, üniversite gibi önemli noktalara uzaklıkları otomatik olarak hesaplanarak ilgili alanlara yazılacaktır. Denetmen tarafından görev oluşturularak daha önce tanım bilgisi ve konumu girilmiş bir taşınmaz için değer talebi uzmana gönderilebilecektir. Uzman ise kendine tanımlanmış değer tespit görevlerini listeleyebilecek ve değere esas taşınmaz bilgilerini uygulama üzerinden girerek otomatik emsal tahmini isteyebilecektir. Daha sonra nihai değeri oluşturarak uygulama üzerinden sisteme kaydedip denetmenin onayına sunacaktır. Denetmen tahmin edilen emsal bilgisi yardımıyla denetmenin belirlediği değeri kontrol ederek onaylayacaktır. Onay sonrası bu taşınmaz değeri bir sonraki emsal tahmininde kullanılarak kendi kendini besleyen bir sistem oluşturulacaktır. Uygulamanın çalışma adımlar Şekil 4.4. teki akış diyagramında görülmektedir.



Şekil 4.4. TETS Uygulaması çalışma adımları diyagramı

4.6. İstemci Yazılımı Tasarımı

Uygulamanın kullanıcı ile olan etkileşimleri istemci yazılımı aracılığı ile gerçekleştirilecektir. Uygulama üzerinde denetmen rolündeki kullanıcıların yeni bir taşınmazı sisteme eklemek için kullanacağı form istemci tarafında geliştirilecektir. Bu form HTML sayfasından oluşmaktadır. Kullanıcının doldurduğu form bilgileri AJAX ve

Javascript teknolojileri ile sunucu tarafına gönderilecektir. Buna benzer şekilde denetmenin görev eklemesi, uzmanın taşınmazın değere esas verilerini girmesini sağlayacak formlarda istemci tarafında HTML, AJAX ve Javascript kullanılarak gerçekleştirilecektir. Sadece form değil görev listesinin kullanıcı ekranında gösterecek tabloların oluşması da aynı teknolojiyle istemci tarafında geliştirilecektir.

TETS uygulaması mobil veya bilgisayar gibi her ekran boyutu ve çözünürlüğünde çalışabilmesi için esnek olarak geliştirilecektir. Bunu gerçekleştirebilmek için css ve Javascript kütüphanelerinden faydalanılacaktır. Günümüzde yaygın olarak kullanılan bootstrap kütüphanesi temel alınarak kullanıcıya kullanım kolaylığı sağlayacak şekilde geliştirilecektir. Bunun yanında javascript kütüphanesinin türevi olan "jquery" istemci tarafı uygulama geliştirmede kullanılacaktır.

Uygulamanın istemci tarafında ara yüzlerinin daha kullanışlı ve görselinin güzel olması için baz versiyonu ücretsiz olarak sunulan "Django Pixel Template" kullanılacaktır. Aynı şekilde uygulama içinde listeleri daha düzenli ve kullanışlı gösterebilmek için https://datatables.net/ tarafından sunulan tablo mimarisi, uygulama için oluşturulmuş sanal Python ortamına "pip install django-datatables-view" komutuyla yüklenmiştir.

4.7. Test ve Doğrulama Senaryoları

TETS Uygulaması, taşınmaz değerlemesinde kullanılan emsal tespit yöntemini kolaylaştırmak için tasarlanmıştır. İlk olarak uygulamanın temelini oluşturan yeni taşınmaz ekleme, görev oluşturma, emsal tahmini ve denetmen kontrolü modülleri test edilecektir. Uygulama kayıtlarının ve listelemelerinin doğru gelip gelmediği kontrol edilecektir.

Bir sonraki aşamada uygulamanın emsal tahmin yöntemlerinin hataları test edilecektir. Test için "Leave-One-Out Cross Validation" yöntemi kullanılacaktır [94]. Bu yönteme göre veri setindeki tüm taşınmazlar için ayrı ayrı emsal tahmini üretilecektir. Emsal tahmini için kullanılacak veri seti kendisi hariç diğer taşınmaz verilerinden elde edilecektir. Daha sonra tahmin edilen emsal değerlerinden karekök ortalama tahmin hatası (KOTH) aşağıdaki formül ile elde edilecektir.

$$KOTH = \sqrt{\frac{\sum_1^n (D_i - T_i)^2}{n}}$$

Denklemde D_i taşınmazın gerçek birim fiyatını, T_i emsal tahminini, n ise test için kullanılacak veri sayısını ifade etmektedir. Daha sonra bu tahmin hataları ışığında veri seti incelenerek hata kaynakları tespit edilmeye çalışılacaktır.

5. UYGULAMA

Bu bölümde yöntem kısmında anlatılan süreçlerin yazılım ortamına nasıl aktarıldığı anlatılacaktır. Bu kapsamda veri tabanı tasarımı, uygulama geliştirme süreci ve ortamları, var olan verilerin sisteme entegrasyonu ve kullanılan emsal tespit yöntemlerinin karşılaştırmalarından bahsedilecektir.

5.1. Çalışma Ortamı Hazırlığı

Bir Django uygulaması django-admin aracı ile kolaylıkla oluşturulabilir. Öncelikle çalışma ortamına Anaconda Python dağıtımı ve django kütüphaneleri yüklenmiştir. Daha sonra TETS uygulaması ve ilintili dizinler "django-admin startproject tets" komutu kullanılarak oluşturulmuştur. Django çerçevesi ile proje yaratıldığında uygulamanın temel bileşenlerinden olan uygulama ayarlarının bulunduğu "settings.py", modellerin tanımlandığı "models.py", sayfa yönlendirmelerinin bulunduğu "urls.py" dosyaları otomatik olarak oluşmaktadır. Kullanıcı ara yüzlerini geliştirmek için kullanılacak HTML dosyaları "templates" dizini altında saklanmaktadır.

Projeyi çalıştırmak için öncelikle Python sanal ortamının etkin hale getirilmesi gerekmektedir. Uygulama Visual Studio Code geliştirme aracı ile geliştirilmiştir. Bir terminal ekranı aracılığı ile aşağıdaki komut çalıştırıldığında daha önce yaratılan ve gerekli kütüphaneleri yüklenen "django" sanal ortamı etkin hale gelmektedir.

```
C:\git_proje\hustol2020\tets\tets>C:\ProgramData\Anaconda3\Scripts\activate django

(django) C:\git_proje\hustol2020\tets\tets>
```

Django projesi yaratıldığında yönetim paneli otomatik olarak oluşmaktadır. Bu panele erişim için kullanılacak kullanıcı adı ve parolası aşağıdaki gibi "createsuperuser" komutu ile oluşturulmuştur.

```
(django) C:\git_proje\hustol2020\tets\tets>python manage.py createsuperuser
```

Projeyi çalıştırmak için ise "python manage.py runserver" komutu kullanılmaktadır.

```
System check identified no issues (0 silenced).
January 01, 2022 - 14:38:19
Django version 3.2.9, using settings 'emsal.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Bu sayede uygulamamız yerel makinada http://127.0.0.1:8000/ adresinde çalışmaya başlayacaktır. Yönetim paneline ulaşmak için yerel makinada http://127.0.0.1:8000/admin adresi kullanılmalıdır. Daha önce yarattığımız kullanıcı adı ve parola ile giriş yapıldığında Şekil 5.1. de görüldüğü gibi bir kullanıcı paneli açılmakta, yeni kullanıcı ve grupları yaratmak için gerekli modüller otomatik olarak yaratılmaktadır.

Şekil 5.1. TETS Uygulaması yönetim paneli

TETS uygulaması için yönetim paneli sadece kullanıcı tanımlamaları ve rol tanımlamaları için kullanılmıştır. Taşınmaz ve görev modülleri için ekleme işlemleri site görsellerine uyumlu olması için formlar şeklinde yeniden tasarlanmıştır. Bu bölümde verilen ekran görüntülerinde mobil uygulama kullanımı öncelikli olarak verilmiştir. Ancak geliştirilen yazılım Esnek Web uygulaması kurallarına uygun olarak geliştirilmiştir ve masaüstü platformları da desteklemektedir.

5.1. Veri ve Veri Tabanı Tasarımı

Veri tabanı yönetim sistemi olarak PostgreSQL kullanılacaktır. Veri tabanı ortamını hazırlamak için öncelikle https://www.postgresql.org/ web sitesinden Windows işletim sistemi uyumlu veri tabanı sunucusu indirilerek kurulmuştur. Daha sonra tablo ve verilerle daha kolay çalışmak için PostgreSQL'in grafik ara yüzü olan pgAdmin uygulaması indirilip yüklenmiştir. Bu sayede yerel bilgisayarda çalışan bir PostgreSQL veri tabanı sunucusu hazırlanmıştır.

TETS uygulaması içeriği bakımından konumsal veriyi sıklıkla kullanacaktır. Konumsal verinin saklanması ve üzerinde işlem yapılabilmesi için veri tabanı sunucusunun konumsal veriyi desteklemesi gerekmektedir. PostgreSQL veri tabanı yönetim sisteminin PostGIS eklentisi geometrik ve coğrafi veri tipini desteklemektedir. PostgreSQL kurulumu ile birlikte gelen "Application Stack Builder" uygulaması ile daha önce kurduğumuz PostgreSQL sunucusu üstüne PostGIS eklentisi yüklenmiştir. Yüklenen sunucu üzerinde "emsal" isminde bir veri tabanı yaratılarak uygulamanın veri tabanı ortamı hazır hale getirilmiştir.

Uygulama veri tabanı üç temel model üzerine oturtularak taşınmaz değerleme sürecinin yönetilmesi sağlanmıştır. Üç temel model kullanıcı, taşınmaz ve görev modelleridir. Bu modellerden kullanıcı modeli Django çerçevesi tarafından otomatik olarak yaratılmıştır. Uygulama denetmen ve uzman olmak üzere 2 rol üzerinden çalışacaktır. Denetmen rolü Django çerçevesinin "superuser", uzman ise "user" rolü olarak kabul edilmiştir. Bu sebeple uygulama için kullanıcı tanımları Django yönetim panelinden gerçekleştirilmiştir.

Yöntem bölümünde anlatılan veri tabanı tasarımını uygulamak için öncelikle projenin veri tabanı bağlantısı yapılmıştır. Django çerçevesi ile proje oluşturulduğunda settings.py dosyası otomatik oluşmakta, veri tabanı bağlantısı "DATABASE" bölümü içinde bulunmaktadır. Aşağıda görüldüğü gibi daha önce yerel bilgisayara kurduğumuz PostgreSQL sunucusunun bağlantı ayarları yapılmıştır.

```
DATABASES = {
  'default': {
    'ENGINE': 'django.contrib.gis.db.backends.postgis',
    'NAME': 'emsal',
    'USER': 'postgres',
    'PASSWORD': '******',
    'HOST': 'localhost',
    'PORT': '5432',
```

```
    }
}
```

Veri tabanı bağlantısı yapıldıktan sonra veri tabanı tablolarını oluşturmak için "models.py" dosyası içine modeller eklenmiştir. Taşınmaz görev tabloları için oluşturulmuş model sınıfları aşağıda görülmektedir.

```
class TasinmazModel(models.Model):
    sira= models.IntegerField(default=0,null=True)
    ilce = models.ForeignKey(IlceModel,on_delete=models.DO_NOTHING,default=1,)
    mahalle =models.CharField(max_length=100,default="",null=True)
    zemin_id= models.IntegerField(default=0,null=True)
    ada = models.CharField(default=0,max_length=10)
    parsel =models.CharField(default=0,max_length=10)
    enlem=models.FloatField(default=0,null=True)
    boylam=models.FloatField(default=0,null=True)
    konum = gmodels.PointField(default=None,db_index=True,null=True,srid=4326)
    brut_alan=models.FloatField(default=0,null=True)
    bulundugu_kat = models.IntegerField(default=0,null=True)
    rekreasyon_uzakligi=models.IntegerField(default=0,null=True)
    universite_uzakligi=models.IntegerField(default=0,null=True)
    sokak_genisligi=models.IntegerField(default=0,null=True)
    sehir_merkezi_uzakligi=models.IntegerField(default=0,null=True)
    bina_yasi=models.IntegerField(default=0,null=True)
    class EvetHayir(models.IntegerChoices):
        Evet = 1
        Hayır = 0

    class DurumSecim(models.IntegerChoices):
        Yeni = 0
        Uzmanda = 1
        Uzman  Onayladı = 2
        Denetmen_Onayladı = 3
    durum=models.IntegerField(choices=DurumSecim.choices,default=0,null=True)
    yapi_kullanim_izin_belgesi_var_mi_int =
models.IntegerField(choices=EvetHayir.choices,default=0,null=True)
    asansor_var_mi_int = models.IntegerField(choices=EvetHayir.choices,default=0,null=True)
    otopark_durumu_int = models.IntegerField(choices=EvetHayir.choices,default=0,null=True)
    salon_bagimsiz_mi_int = models.IntegerField(choices=EvetHayir.choices,default=0,null=True)
    degerleme_tarihi=models.DateField(null=True)
    deger=models.FloatField(default=0,null=True)
    birim_fiyat=models.FloatField(default=0,null=True)
    emsal=models.FloatField(default=0,null=True)
    normalize_deger=models.FloatField(default=0,null=True)
    def __str__(self):
      return str(self.ada)+'-'+str(self.parsel)
```

```
class GorevModel(models.Model):
    uzman =
models.ForeignKey(settings.AUTH_USER_MODEL,on_delete=models.CASCADE,default=1)
    denetmen =
models.ForeignKey(settings.AUTH_USER_MODEL,on_delete=models.CASCADE,default=1,relat
ed_name='denetmen_user')
    tasinmaz=models.ForeignKey(TasinmazModel,db_column='tasinmaz_id',
on_delete=DO_NOTHING,null=True, default=1)
    aciklama =models.CharField(max_length=100,null=True)
    last_modified = models.DateTimeField(auto_now_add = True)
    musteri_ad_soyad =models.CharField(max_length=100,default="",null=True)
    def __str__(self):
        return str(self.emlak.mahalle)+'-'+str(self.emlak.ada)+'-'+str(self.emlak.parsel)+'-
'+self.musteri_ad_soyad
```

Django çerçevesi içinde veri tabanı işlemeleri için göç ettirme (migration) modülü bulunmaktadır. Yukarıdaki taşınmaz ve görev sınıfları oluşturulduktan sonra aşağıdaki komutlarla veri tabanına tablo olarak kaydedilmiştir.

```
(django) C:\git_proje\hustol2020\tets\tets>python manage.py makemigrations
```

```
(django) C:\git_proje\hustol2020\tets\tets>python manage.py migrate
```

“makemigrations” komutu “models.py” dosyasındaki değişikleri algılayarak, bu değişiklikleri veri tabanı sorgusu haline getirip bir dosyaya kaydetmektedir. “migrate” komutu ile de daha önce uygulanmamış “migration” dosyasındaki veri tabanı sorguları çalıştırılarak veri tabanı güncellenmektedir. Taşınmaz ve Görev modelleri oluşturulduktan sonra diğer gerekli olan (şehir merkezi, üniversite, park vb.) model sınıfları oluşturularak veri tabanına tablo olarak kaydedilmiştir.

TETS uygulaması için başlangıçta kullanılacak veri seti Ankara ili Mamak ilçesinde bulunan 1799 tane taşınmazdan oluşmaktadır. Veri içinde bulunduğu kat, yüzölçümü, otopark durumu, bağımsız salon durumu, asansör durumu, yapı kullanım izin belgesi durumu gibi birçok değere esas veri bulunmaktadır. Bunun yanında üniversite, şehir merkezi, park alanı, AVM, okul gibi yerlere uzaklık bilgisi de bulunmaktadır. Aynı

veri seti için yapılan bir çalışmada değere etki eden faktörler incelenmiş ve korelasyonları bulunmuştur[94]. TETS uygulamasında bahsedilen bu çalışmadaki değere etki eden taşınmaz yüzölçümü, bulunduğu kat, parka uzaklığı, üniversiteye uzaklığı, asansör durumu, sokak genişliği, yapı kullanım izin belgesi durumu, otopark durumu, bağımsız salon durumu ve şehir merkezine olan uzaklık bilgileri dikkate alınmıştır.

Uygulama üzerinden taşınmaz modeli oluşturulduktan sonra veriler PostgreSQL veri tabanına atılmıştır. Veri seti içinde konum bilgisi "point" tipinde bulunmaktadır. Bu sebeple veri seti önce bir "shape" dosyasına dönüştürülmüştür. Oluşturulan shape dosyası PostgreSQL veri tabanı için geliştirilmiş olan "PostGIS Shapefile Import/Export Manager" aracı ile taşınmaz tablosu içine aktarılmıştır.

Veri setinde ki taşınmaz değerleri 2011,2012,2013 yıllarında tespit edilmiş değerlerdir. Ankara ili için Merkez Bankası konut birim fiyat ortalamaları incelenmiştir[96]. 2010 yılından itibaren Ankara ili için aylık konut ortalama birim değerleri Merkez Bankası tarafından yayımlanmaktadır. 2021 yılı Eylül ayı için Ankara için yayımlanan ortalama birim fiyat 3631,76 TL olarak belirlenmiştir. Çizelge 5.1.'de test verisinin oluşturulduğu aylardaki taşınmazların ortalama birim fiyatları ve bu fiyatları 2021 yılı Eylül ayına güncellemek için bulunan katsayılar gösterilmektedir.

Çizelge 5.1. Merkez Bankası 2011, 2012, 2013 Konut Birim Fiyatları

Yıl - Ay	Birim Fiyat	Katsayı
2011-1	957.5	3.79
2011-2	969.2	3.75
2011-3	978.2	3.71
2011-4	986.4	3.68
2011-5	1002	3.62
2011-6	1009.5	3.60
2011-7	1014.5	3.58
2011-8	1019	3.56
2011-9	1032	3.52
2011-10	1035.7	3.51
2011-11	1039.6	3.49
2011-12	1039.7	3.49
2012-1	1045.4	3.47
2012-2	1058.2	3.43
2012-3	1074.3	3.38
2012-4	1091.9	3.33

2012-5	1097.4	3.31
2012-6	1101.1	3.30
2012-7	1106	3.28
2012-8	1109.8	3.27
2012-9	1118	3.25
2012-10	1122.1	3.24
2012-11	1130.1	3.21
2012-12	1137.2	3.19
2013-1	1143.1	3.18
2013-2	1155.1	3.14
2013-3	1172.1	3.10
2013-4	1189.7	3.05
2013-5	1201.3	3.02
2013-6	1213.1	2.99
2013-7	1221.1	2.97
2013-8	1226.4	2.96
2013-9	1234	2.94
2013-10	1242.1	2.92
2013-11	1258.9	2.88
2013-12	1263.2	2.88

Bu veriler alınarak PostgreSQL veri tabanına kaydedilmiştir. Veri setinde bulunan değerlerin değerleme tarihleri göz önüne alınarak tüm değerler 2021 yılı eylül ayındaki olması gereken değerlere dönüştürülmüştür. Ayrıca daha kolay ve anlaşılır işlem yapmak için otopark durumu, asansör durumu gibi mantıksal alanlar Var=1, Yok=0 olmak üzere sayısal hale getirilmiştir.

5.2. Uygulama Geliştirme

Veri tabanı modellemesi tamamlandıktan sonra uygulama geliştirilmeye başlanmıştır. TETS uygulaması geliştirilirken, değerleme uzmanının sahada mobil bir cihaz yardımıyla emsal tahmini üreterek değerleme sürecinin süresinin kısaltması hedeflenmiştir. Bu sebeple tüm uygulama ekran görüntüleri, uygulamanın mobil bir cihazda nasıl göründüğünü gösterecek şekilde oluşturulmuştur. İlk olarak Şekil 5.2. de görülen kullanıcı giriş ekranı hazırlanmıştır. Giriş ekranında kullanıcı adı ve parolası alınarak sisteme giriş Django çerçevesinin oturum yönetimi kullanılarak yapılmıştır.

Şekil 5.2. TETS Uygulaması giriş ekranı

Kullanıcı girişi yapıldıktan sonra uygulama ana sayfaya yönlenmektedir. Uygulama içindeki tüm yönlendirmeler "urls.py" dosyasında tutulmaktadır. Adres satırından gelen tüm istekler bu dosyadaki karşılıklarına göre yönlendirilmektedir. Aşağıda TETS uygulamasında "urls.py" dosyasının bir bölümü görülmektedir. Örnek olarak adres çubuğundan gelen "gorev" isteği (http://127.0.0.1:8000/gorev) views modülündeki "gorev" sınıfına yönlenmektedir.

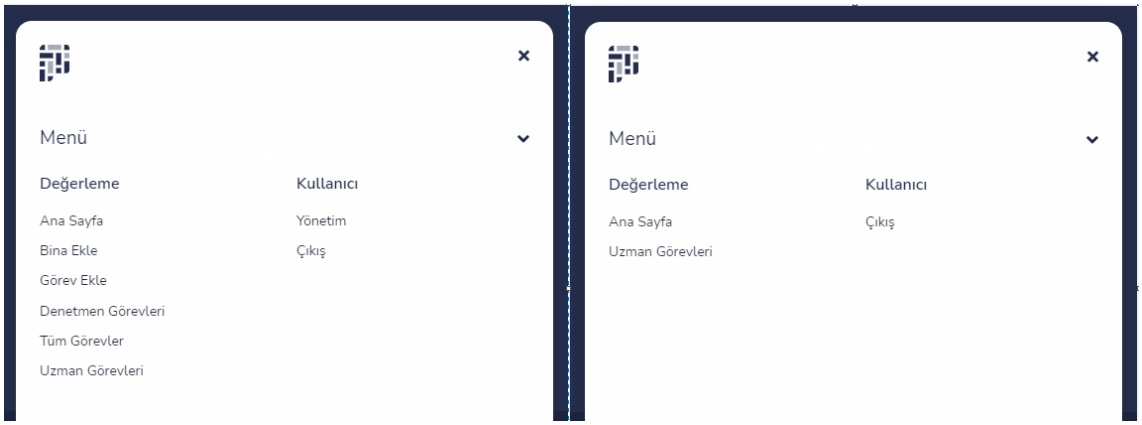
```
urlpatterns = [
    path('', views.home, name='home'),
    path('gorev/', views.gorev, name='gorev'),
    path('bina/', views.bina, name='bina'),
    path('uzman_bina/<int:pk>/', views.uzman_bina, name='uzman_bina'),
    path('denetmen_bina/<int:pk>/', views.denetmen_bina, name='denetmen_bina'),
    path('gorevlist/', views.gorevlist, name='gorevlist'),
    path('denetmen_gorev/', views.denetmen_gorev, name='denetmen_gorev'),
    path('uzman_gorev/', views.uzman_gorev, name='uzman_gorev'),
    path('login/',
        LoginView.as_view
        (
            template_name='login.html',
            authentication_form=forms.BootstrapAuthenticationForm,
            extra_context=
            {
                'title': 'Giriş',
                'year' : datetime.now().year,
```

```
            }
        ),
        name='login'),
    path('logout/', views.logoutUser, name='logout'),
    path('admin/', admin.site.urls),
    path('api/',include('tets_app.urls')),

    url(r'^data/$', GorevListJson.as_view(), name="gorev_list_json"),

    url(r'^data_denetmen_gorev/$', DenetmenGorevListJson.as_view(),
name="denetmen_gorev_list_json"),
    url(r'^data_uzman_gorev/$', UzmanGorevListJson.as_view(), name="uzman_gorev_list_json"),
     url('uzmana_geri_gonder/$', views.uzmana_geri_gonder, name='uzmana_geri_gonder'),
     url('emsal_getir/$', views.emsal_getir, name='emsal_getir'),
     url('deger_analiz/', views.deger_analiz, name='deger_analiz'),
]
```

Kullanıcı girişi yapıldıktan sonra uygulama ana sayfaya yönlenmekte ve uygulama menüsü oluşmaktadır. Uygulama menüsü kullanıcının yetkisine göre şekillendirilmiştir. Denetmen rolünde bir kullanıcı giriş yaptığında denetmen rolünün ulaşması gereken sayfalar menüye eklenmekte, uzman kullanıcısı giriş yaptığında ise uzmanın ulaşması gereken sayfalar menüde listelenmektedir. Menü her sayfada görüntüleneceği için "templates" dizini içinde "layout.html" dosyası içine yazılmıştır. Bu sayede diğer oluşturulacak tüm sayfalarda bu dosyanın içe aktarılmasıyla menü görünür hale getirilmiştir. Şekil 5.7.' de denetmen rolü ve uzman rolündeki kullanıcıların uygulama menüleri görülmektedir. "layout.html" içinde denetmen rolünün belirlenip menünün buna uygun hazırlanması için kullanıcının "superuser" olup olmadığı kontrol edilmiştir. Örnek olarak denetmen kullanıcısının bina ekleme, görev ekleme, kendi görevlerini ve tüm görevlerini görme yetkisi vardır. Bu yetkiler uzman kullanıcısında bulunmadığı için aşağıdaki kod parçasında da görüleceği üzere istekte bulunan kullanıcının "superuser" olup olmadığı kontrol edilmiştir.

```
<div class="col-6 col-lg-4">
    <h6 class="d-block mb-3 text-primary">Değerleme</h6>

        <ul class="list-style-none mb-4">
            <li class="mb-2 megamenu-item">
              <a class="megamenu-link" href="/">Ana Sayfa</a> </li>
              {% if request.user.is_superuser %}
             <li class="mb-2 megamenu-item">
                <a class="megamenu-link" href="/bina">Bina Ekle</a></li>
              <li class="mb-2 megamenu-item">
```

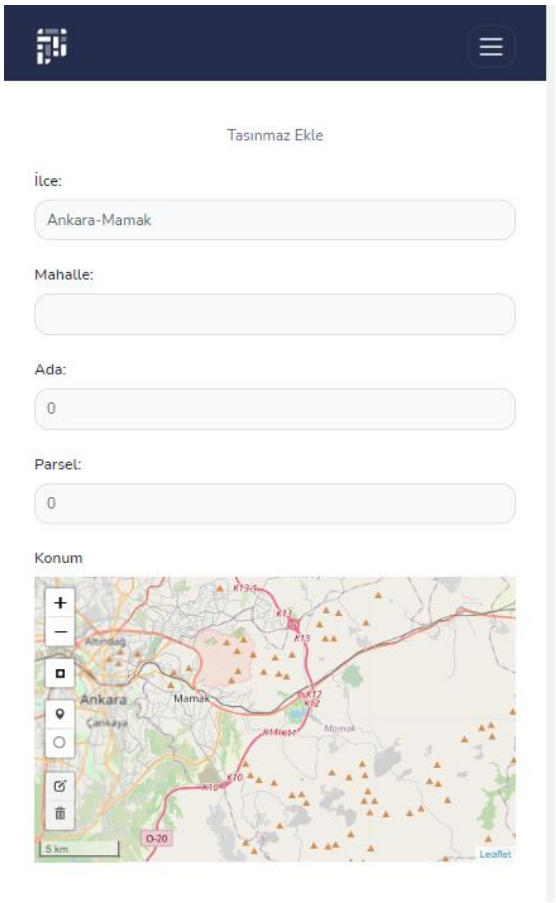
```
                <a class="megamenu-link" href="/gorev">Görev Ekle</a></li>
              <li class="mb-2 megamenu-item">
                <a class="megamenu-link" href="/denetmen_gorev">Denetmen Görevleri</a>
                </li>
              <li class="mb-2 megamenu-item">
                <a class="megamenu-link" href="/gorevlist">Tüm Görevler</a></li>
                {% endif %}
              <li class="mb-2 megamenu-item">
                <a class="megamenu-link" href="/uzman_gorev">Uzman Görevleri</a>
              </li>
        </ul>
<div>
```



Şekil 5.3. Denetmen ve Uzman menüleri

Menüler oluşturulduktan sonra Şekil 4.4. teki çalışma adımlarının geliştirilmesi yapılmıştır. Uygulamanın ilk adımı denetmen kullanıcısının değerlenmesi istenen taşınmazı sisteme kaydetmesidir. TETS uygulaması içinde kullanılacak tüm formlar "forms.py" isminde bir dosyaya kaydedilmiştir. Django çerçevesinde formlar daha önce oluşturulmuş model sınıflarından oluşturulabilmektedir. Taşınmaz modeli daha önce oluşturulduğu için yeni taşınmaz ekleme formu aşağıdaki BinaForm sınıfı ile oluşturulmuştur. Modelimizin TasinmazModel olduğu belirtildikten sonra form içinde bulunmasını istediğimiz alanlar "fields" özelliği içine yazılmıştır. Django çerçevesi bu alanları tipine göre otomatik olarak ayarlayabilmektedir. Denetmen tarafından yeni taşınmaz eklenirken taşınmazın konum bilgisi de alınması gerekmektedir. Bu sebeple haritanın açılması için "LeafLetWidget" kullanılmıştır. Oluşturulan formun ekran görüntüsü ve form sınıfı Şekil 5.4.'de görülmektedir.

```
class BinaForm(forms.ModelForm):
    class Meta:
        model = TasinmazModel
        fields = ('ilce','mahalle','ada','parsel','konum',)
        widgets = {'konum': LeafletWidget()}
    def __init__(self, *args, **kwargs):
        super(BinaForm, self).__init__(*args, **kwargs)
        for visible in self.visible_fields():
            visible.field.widget.attrs['class'] = 'form-control'
```



Şekil 5.4. Yeni taşınmaz ekleme

Form içindeki alanlar doldurulup form kaydedilmek için gönderildiğinde ilgili veriler sunucu tarafında ayrıştırılıp taşınmaz veri tabanına kaydedilmektedir. Veri tabanına kayıt işi views modülünde tanımladığımız bina metodu içinde gerçekleşmektedir. Aşağıda görüleceği üzere "forms.save()" kod parçası ile taşınmaz bilgileri taşınmaz veri tabanına kaydedilmektedir. Daha sonra emsal değerinin hesaplanmasında kullanılacak şehir merkezi, park ve üniversiteye uzaklığı hesaplanarak kayıt güncellenmiştir. Yeni kaydedilmiş taşınmazın varsayılan durum değeri "0" olarak

kaydedilmektedir. Bu sayede görev ekleme adımında hangi taşınmazlar için görev oluşturulmadığı durum alanın “0” olanlarının seçilmesiyle bulunması sağlanmıştır.

```
@login_required
def bina(request):
   if request.user.is_superuser:
      submitted = False
      if request.method == 'POST':
         form = BinaForm(request.POST)
         if form.is_valid():
           bina=form.save()

           df_wms=gpd.geopandas.GeoDataFrame.from_records(SehirMerkeziModel.objects.all()
         .annotate(mesafe=Distance('konum', bina.konum, spheroid=True))
         .order_by('mesafe').values())
           sm_uzaklik=df_wms['mesafe'][0].m

           df_wmu=gpd.geopandas.GeoDataFrame.from_records(UniversiteModel.objects.all()
         .annotate(mesafe=Distance('konum', bina.konum, spheroid=True))
         .order_by('mesafe').values())
           um_uzaklik=df_wmu['mesafe'][0].m

           df_wmr=gpd.geopandas.GeoDataFrame.from_records(RekreasyonModel.objects.all()
         .annotate(mesafe=Distance('konum', bina.konum, spheroid=True))
         .order_by('mesafe').values())
           rm_uzaklik=df_wmr['mesafe'][0].m

           TasinmazModel.objects.filter(id=bina.id).update(
                  rekreasyon_uzakligi=rm_uzaklik,universite_uzakligi=um_uzaklik,
                  sehir_merkezi_uzakligi=sm_uzaklik)

           return HttpResponseRedirect('/bina/?submitted=True')
      else:
         form = BinaForm()
         if 'submitted' in request.GET:
            submitted = True
      return render(request,
         'bina.html',
         {'form': form, 'submitted': submitted}
         )
   else :
     return HttpResponseRedirect('/')
```

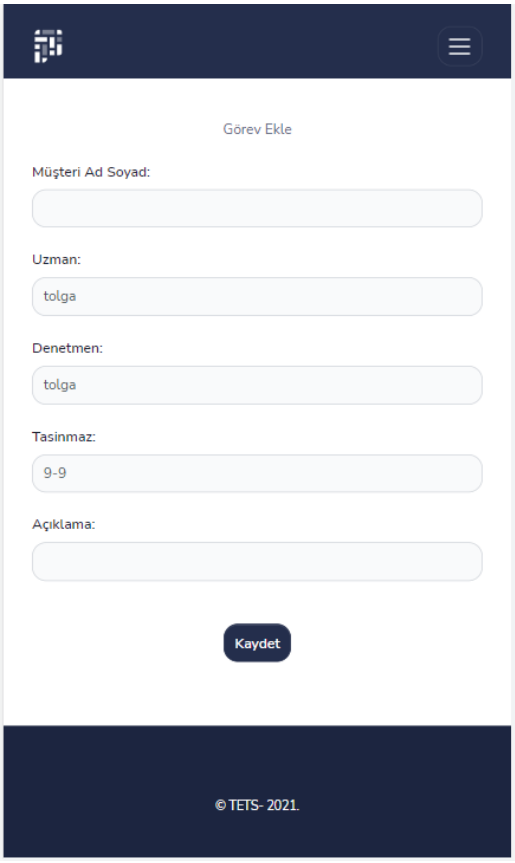
Denetmen kullanıcısı tarafından değerlenmesi istenen yeni taşınmaz kaydedildikten sonra bu değerlemenin görevlendirilmesi adımı geliştirilmiştir. Yeni taşınmaz eklenmesinde olduğu gibi görev modelinden bir form üretilmiştir. Şekil 5.5’te

görülen bu form ile daha önce görev atanmamış taşınmazlar bir listeden seçtirilmektedir. Müşteri bilgisi alındıktan sonra değerleme görevinin atanacağı uzman ve kontrol için görevlendirilecek denetmen seçilerek form kaydedilmektedir. Görev formunda seçilebilecek taşınmazlar aşağıdaki kod parçasında görüleceği üzere durumu "0" olanlar olacak şekilde seçilmiştir.

```
self.fields['tasinmaz'].queryset = TasinmazModel.objects.filter(durum=0)
```

Görev eklendikten sonra ise o taşınmazın durumu "1" olarak güncellenerek uzmana atanmış olduğu bilgisi kaydedilmiştir.

```
TasinmazModel.objects.filter(id=form.data['tasinmaz']).update(durum=1)
```



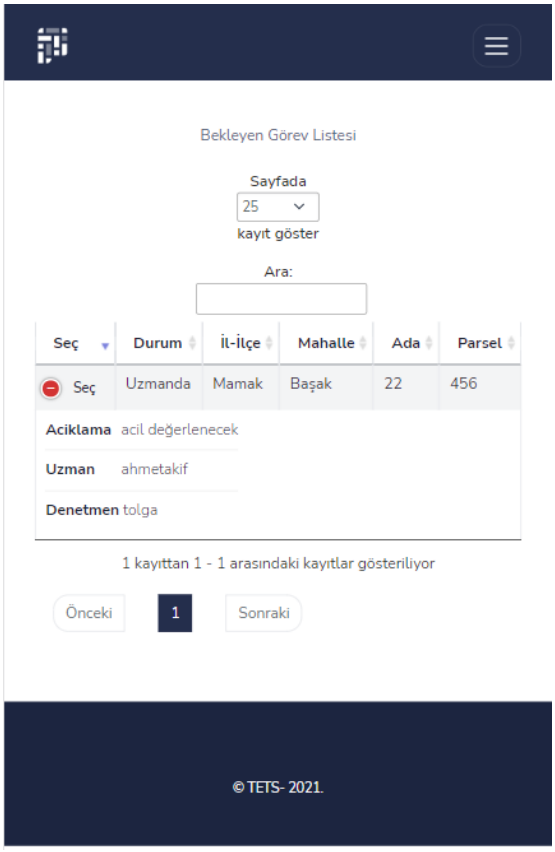
Şekil 5.5. Görev kaydetme

Görev ekleme işlemi tamamlandıktan sonra uzman kullanıcısı kendine atanmış görevleri listeleyebilecektir. Görev listesini göstermek için "datatables.net"in sunduğu

tablo yapısı kullanılmıştır. “views” modülü içine durumu “1” ve atanmış uzmanın kendisinin olduğu taşınmaz kayıtlarını döndüren bir metot yazılmıştır.

```
return GorevModel.objects.filter(tasinmaz__durum=1,uzman=self.request.user)
```

Bu sayede her uzman kendine atanmış değerleri görebilmektedir. Aşağıdaki şekilde görüldüğü gibi kendine atanmış görevleri gören uzman “seç” butonu ile o taşınmaz için işlem yapmaya başlayabilecektir.



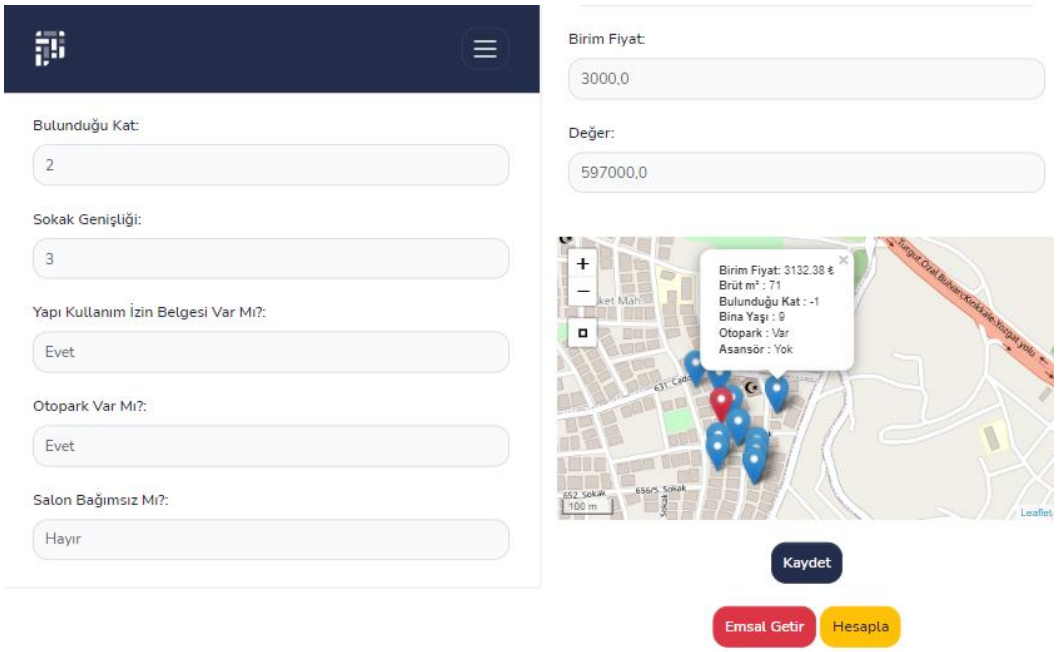
Şekil 5.6. Uzman görev listesi

Uzman kullanıcısı kendine atanan görevlerden birini seçtikten sonra o taşınmazın temel bilgilerini ve konumunu görüntüleyebilmektedir. Gerekli diğer bilgileri (bulunduğu kat, sokak genişliği, yüzölçümü vb.) doldurduktan sonra emsal getir butonu ile girdiği değerlerin ürettiği emsal tahmini emsal hanesine yazılmaktadır. Şekil 5.7’de uzman kullanıcısının bir taşınmazı değerlemesi için tasarlanmış ekran görülmektedir. Uzman

kullanıcısı gerekli bilgileri girdikten sonra emsal getir metodu çağrılarak uygulama tarafından hesaplanan emsal tahmini metodu çağrılmaktadır.

```
$.get('/emsal_getir/', {tasinmaz_id: tasinmaz_id,salon:salon,
        otopark:otopark,yapi_izin:yapi_izin,
        bulundugu_kat:bulundugu_kat,sokak_genisligi:sokak_genisligi,
        brut_alan:brut_alan,bina_yasi:bina_yasi}, function(data,response){

        if(response=="success")
        {
          $("input[name='emsal']").val(data.emsal);

          deger_hesapla();

        }

      });
```

Emsal tahmini uygulama içinde farklı yöntemlerle geliştirilmiş ve bu bölge için karşılaştırılmıştır. Emsal tahmini için kullanılan yöntemlerin geliştirilmesi bir sonraki bölümde anlatılacaktır. Uzman kullanıcısı tahmin edilen emsal değerini dikkate alarak belirlediği birim fiyatı gerekli alana girerek hesapla butonuna basmakta ve taşınmazın değeri belirlenmektedir. Bir harita üzerinde kırmızı işaretle değerlenmesi istenen taşınmaz ve mavi renkte ona en yakın 15 taşınmaz gösterilmektedir. Yakın taşınmazların üstüne tıklanarak daha önce değerlemesi yapılmış taşınmazların özellikleri görünerek uzmanın değer kararına yardımcı olması sağlanmıştır. Daha sonra bu değeri kaydederek denetmenin onayına sunmaktadır. Kaydet butonuna basıldığında taşınmazın durumu “2” yapılarak denetmene havale edildiği belirlenmiş olmaktadır.



Şekil 5.7. Uzman taşınmaz değerleme formu

Bir sonraki adımda denetmen kendi onayına sunulmuş taşınamaz değerlerini listelemektedir. Uzman görevleri listelemesinde kullanılan yöntem burada da kullanılmış sadece taşınmaz filtrelemesi, durumu "2" olanlar olarak uygulanmıştır.

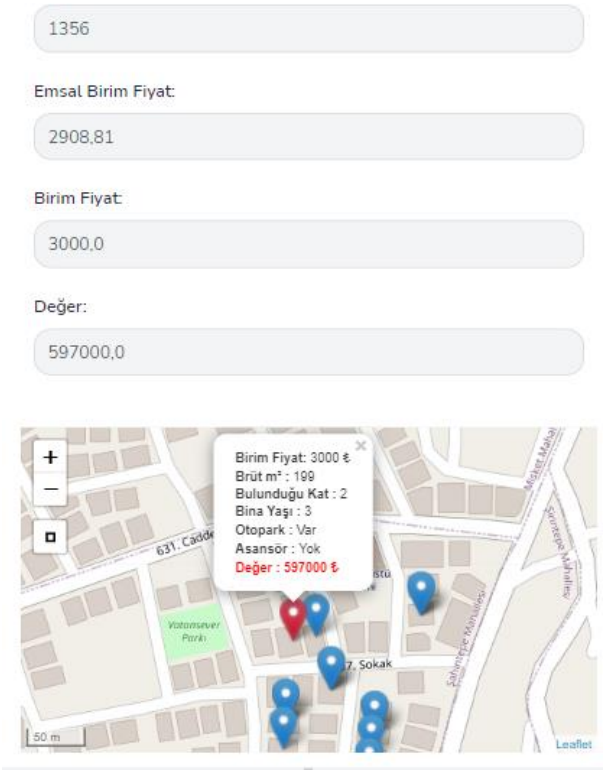
```
return GorevModel.objects.filter(tasinmaz__durum=2,denetmen=self.request.user)
```

Denetmen listeden kontrol etmek istediği görevi seçerek Şekil 5.8.'te görüldüğü gibi onay ekranına getirmektedir. Onay ekranında görevli uzmanın girdiği bilgiler ile bir harita yardımıyla taşınmazın yeri ve etrafındaki daha önce değerlenmiş taşınmazlar görüntülenmektedir. Bu sayede denetmenin uzmanın belirlediği değeri kontrol etmesinin kolaylaşacağı düşünülmüştür. Denetmen bu form ile görevi uzmana tekrar değerlemesi için iade edebilmekte veya onaylayabilmektedir. Form onay şeklinde gönderildiğinde taşınmazın durumu "3" yapılarak değeri onaylanmış olarak kaydedilmektedir.

```
TasinmazModel.objects.filter(id=pk).update(durum=3)
```

Uzmana iade ederken ise durum alanını tekrar "1" olarak güncellemektedir.

```
TasinmazModel.objects.filter(id=tasinmaz_id).update(durum=1)
```



Şekil 5.8. Denetmen onay formu

Denetmenin onayından sonra uygulama çalışma adımları tamamlanmış olmaktadır. Onaylanmış değerler bir sonraki emsal tespitinde kullanılır hale getirilerek uygulamanın kendi kendini beslemesi sağlanmıştır.

5.3. Emsal Tespiti Yöntemleri

TETS uygulamasının temel amacı taşınmaz değerlemesi yapacak uzmana emsal tahmini yaparak zamandan kazandırmaktır. Ayrıca değer kontrolü yapan denetmene kontrol esnasında kolaylık sağlaması düşünülmektedir. Bu kapsamda emsal tahmini yapmak için farklı yöntemlerin desteklenebildiği genişletilebilir bir mimari benimsenmiştir. Bu kapsamda emsal değerleme yöntemlerini soyutlayan nesne tabanlı tasarım uygulanmış ve bu tasarım iki farklı yöntem ile uygulanmıştır. Bu yöntemler test verisi için denenmiş ve test bölgesi için sonuçları tartışılmıştır.

Uygulama içinde emsal tahmin yöntemi için sınıf hiyerarşisi oluşturulmuştur. Bunun için önce EmsalDegerleme sınıfı geliştirilmiştir.

```
class EmsalDegerleme():
    def __init__(self, emlak):
        self.emlak = emlak

    def emsal(self):
        pass

    def get_df(self):
        tt=self.emlak
        df_wm=gpd.geopandas.GeoDataFrame.from_records(TasinmazModel.objects.filter(durum=3).
exclude(id= tt.id)
        .annotate(mesafe=Distance('konum', self.emlak.konum,
spheroid=True)).order_by('mesafe')[:200].values())
        df_wm['geometry']=df_wm.apply(lambda
row:shapely.geometry.Point(row['konum'].x,row['konum'].y),axis=1)
        df_wm.set_geometry("geometry")
        df_wm = df_wm.set_crs('epsg:4326')
        df_wm = df_wm.to_crs(epsg=3857)

        return df_wm
```

EmsalDegerleme sınıfı içinde emsal tahmin yöntemlerini uygulamak için gerekecek veri setini oluşturan bir metot tanımlanmıştır. “get_df” metodu değerlenmesi istenen taşınmaza en yakın taşınmazları (200 adet) bir Geodataframe içine atmaktadır. Bunu yaparken doğruluğu kesinleşmiş yani durumu “3” olan taşınmazları seçmektedir.

5.3.1 En Küçük Kareler Yöntemi (OLS)

OLS çözümü için EmsalDegerleme sınıfından türetilmiş OLSEmsal sınıfı yaratılmıştır ve emsal metodu ezilmiştir. Aşağıda görüldüğü gibi ilk olarak türetildiği EmsalDegerleme sınıfının “get_df” metodunu çağırarak modeli besleyecek veri seti oluşturulmuştur.

```
class OLSEmsal(EmsalDegerleme):
    def __init__(self, emlak):
        super().__init__(emlak)

    def emsal(self):
        df_wm = self.get_df()
        y = df_wm['normalize_deger'].values

        columns = ['brut_alan',  'bulundugu_kat', 'universite_uzakligi',
            'asansor_var_mi_int','sokak_genisligi',
```

```
            'yapi_kullanim_izin_belgesi_var_mi_int',
            'otopark_durumu_int','rekreasyon_uzakligi',
            'salon_bagimsiz_mi_int','sehir_merkezi_uzakligi','bina_yasi' ]

        yxs = df_wm.loc[:,columns]
        m1 = spreg.OLS(y, yxs.values, name_x=yxs.columns.tolist(), name_y='deger')
        b_vals = model_to_dict(self.emlak)
        b_vals['CONSTANT']=1
        vals = [b_vals[c] for c in m1.name_x]
        emsal = np.matmul(m1.betas.T, vals)

        return emsal
```

OLS model çözümü için "spreg" kütüphanesi kullanılmıştır. Modele girdi parametresi olarak verilecek öznitelik bilgileri (yxs değerleri) ve veri setinin sonuçlarının bulunduğu "normalize_değer" sütunu (y değeri) belirtilmiştir. Daha sonra "spreg" kütüphanesiyle model çözülerek katsayılar (β) ve sabit değer (α) bulunmuştur. Yeni gelen taşınmaz bilgileri (vals) bu katsayılarla çarpılarak emsal değeri tespit edilmiştir.

5.3.2 Ters Uzaklık Ağırlıklandırma Yöntemi (IDW)

Emsal tahmin edilmesi için kullanılan bir diğer yöntem ise IDW (Inverse Distance Weighted) yöntemidir. OLS yönteminde olduğu gibi IDWEmsal sınıfı da EmsalDegerleme sınıfından türetilmiştir.

```
class IDWEmsal(EmsalDegerleme):
    def __init__(self, emlak):
        super().__init__(emlak)

    def emsal(self):
        df_wm = self.get_df()
        y = df_wm['normalize_deger'].values

        d = np.array([d.m for d in df_wm['mesafe'].values])
        d=np.where(d>0,d,1)
        emsal = np.dot(y,1/d)
        emsal = emsal / (np.sum(1/d))
        return emsal
```

IDW modeli için gereken veri seti çekildikten sonra değerlemesi istenen taşınmaza uzaklıkları bir diziye atılmıştır. Veri setindeki mesafelerin tersi emsal değeri

ile çarpılarak ağırlıklandırılmıştır. Veri setinde bulunan bazı taşınmazlar aynı binada bulundukları için konumları aynı olabilmekte dolayısı ile aralarındaki mesafe "0" çıkabilmektedir. Bu sebeple mesafenin "0" olduğu durumlarda ağırlık "1" ile değiştirilmiştir. Böylelikle mesafenin olmadığı aynı binadaki taşınmazların ağırlıkları artırılmıştır. Son olarak ağırlıklandırılmış değer toplamları ağırlıklar toplamına bölünerek emsal değeri normalize edilmiştir.

5.3.3 Uzamsal Gecikmeli Regresyon (SLR)

SLP çözümü için EmsalDegerleme sınıfından türetilmiş SLPEmsal sınıfı yaratılmıştır. OLS yöntemine benzer olarak aşağıda görüldüğü gibi ilk olarak türetildiği EmsalDegerleme sınıfının "get_df" metodunu çağırarak modeli besleyecek veri seti oluşturulmuştur.

```
class SLREmsal(EmsalDegerleme):
   def __init__(self, emlak):
      super().__init__(emlak)


   def emsal(self):
      df_wm = self.get_df()
      y = df_wm['normalize_deger'].values

      columns = ['brut_alan',  'bulundugu_kat', 'universite_uzakligi',
            'asansor_var_mi_int','sokak_genisligi',
            'yapi_kullanim_izin_belgesi_var_mi_int',
            'otopark_durumu_int','rekreasyon_uzakligi',
            'salon_bagimsiz_mi_int','sehir_merkezi_uzakligi','bina_yasi','idw' ]

      yxs = df_wm.loc[:,columns]

      m1 = spreg.OLS(y, yxs.values, name_x=yxs.columns.tolist(), name_y='deger')

      b_vals = model_to_dict(self.emlak)
      b_vals['CONSTANT']=1
      vals = [b_vals[c] for c in m1.name_x]
      emsal = np.matmul(m1.betas.T, vals)

      return emsal
```

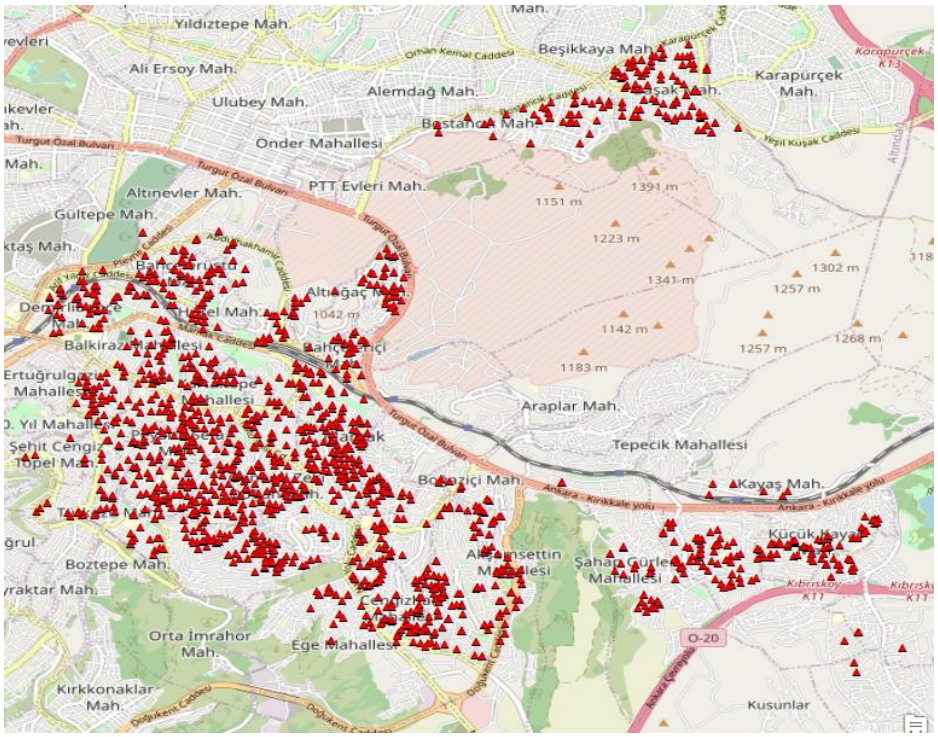
SLR model çözümü için "spreg" kütüphanesi kullanılmıştır. Modele girdi parametresi olarak verilecek öznitelik bilgileri (yxs değerleri) ve veri setinin sonuçlarının

bulunduğu “normalize_değer” sütunu (y değeri) belirtilmiştir. Daha sonra “spreg” kütüphanesiyle model çözülerek katsayılar (β) ve sabit değer (α) bulunmuştur. Yeni gelen taşınmaz bilgileri (vals) bu katsayılarla çarpılarak emsal değeri tespit edilmiştir. OLS yönteminden farkı öznitelik bilgilerine daha önceden hesaplanmış IDW yöntemi değerleri de eklenmiştir.

5.3.4 Testler, Sonuçlar ve Tartışmalar

Uygulama bina ekleme, görev atama, emsal tahmini ile birlikte değer tespiti ve denetmen kontrolü modüllerinden oluşmaktadır. Uygulama geliştirmesi tamamlandıktan sonra 5 denetmen, 5 uzman yetkisine sahip kullanıcı oluşturulmuştur. 20 tane yeni taşınmaz uygulama üzerinden eklenerek rastgele uzman ve denetmenlere görev olarak atanmıştır. Daha sonra uzman kullanıcıları uygulama üzerinden kendilerine atanan görevleri listeleyip, değer hesabı için gerekli olan alanlar doldurulmuştur. Uygulama emsal tahmini üretmiş ve bu emsal tahmini yardımıyla nihai birim fiyat oluşturularak denetmen onayına gönderilmiştir. Denetmen kullanıcısı bazı taşınmazları yeniden değerlenmesi için uzmana geri göndermiş bazılarını ise onaylayarak uygulamanın hatasız bir şekilde çalıştığı test edilmiştir. Uygulama ekran görüntüleri EK 1 ‘de görülmektedir.

TETS uygulamasında emsal tahmin testleri IDW, OLS ve SLR yöntemleri kullanılarak yapılmıştır. 1799 tane test verisine her bir yöntem için TETS uygulaması üzerinden emsal tahmini yaptırılmıştır ve veri tabanına kaydedilmiştir. 1799 taşınmazın ortalama birim fiyatı 3043,74 TL. olarak hesaplanmıştır. Aşağıdaki şekilde taşınmazların harita üzerindeki dağılımları görülmektedir.



Şekil 5.9. Test Verisinin Dağılımı

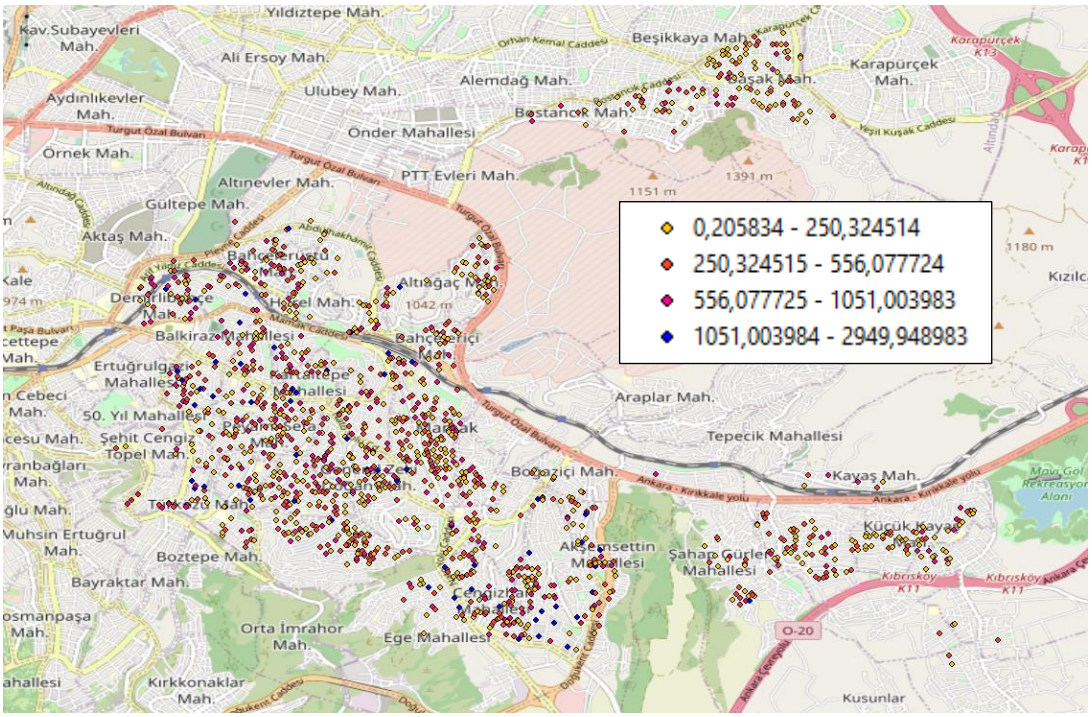
Emsal tahmini için IDW, OLS ve SLR metotları test edilmiştir. Yöntem kısmında anlatılan "Leave One Out Cross Validation" yöntemiyle 1799 taşınmaz için tek tek emsal tahmini üretilmiş ve kaydedilmiştir. İlk testte emsal tahmini yaptırılacak taşınmaza en yakın 200 adet taşınmaz alınarak IDW, OLS ve SLR yöntemleri kullanılmıştır. Bu kriterlere göre çıkan karekök ortalama tahmin hatası IDW yöntemine göre 502,85 TL, OLS yöntemine göre 450,38 TL, SLR yöntemine göre ise 437,70 TL olduğu görülmüştür. SLR yönteminin OLS ve IDW yöntemine göre test bölgesinde daha iyi sonuçlar verdiği tespit edilmiştir.

Sekil 5.9'de görüldüğü gibi test bölgesinde taşınmazların birbirine yakın oldukları görülmektedir. Bu sebeple modelleri oluşturmak için kullanılan 200 en yakın taşınmaz 100'e düşürülerek ortalama karekök tahmin hataları tekrar hesaplanmıştır. IDW için ortalama karekök tahmin hatası 503,16 TL olarak görülmüştür. OLS için ise 451,09 TL, SLR için ise 445,51 olduğu görülmüştür. Modele dahil edilecek taşınmaz sayısının düşürülmesinin hata oranı çok fazla etkilemediği anlaşılmıştır.

IDW yönteminin değere etki eden hiçbir özelliği barındırmaması ve sadece konumuna göre yakın taşınmazlar üzerinden hesap yapması sebebiyle OLS ve SLR yönteminden daha kötü sonuç üretmesine yol açmıştır. OLS ve SLR yöntemi uygulamak

için seçilen yakın taşınmazların otopark, asansör veya yapı ruhsatı özelliklerinin aynı olması ayırt edici özelliğini kaybetmesine sebep olmaktadır. Bu durum OLS ve SLR sonuçlarının hata oranını artırmaktadır.

Bir sonraki aşamada veri Postgis ShapeFile Exporter uygulaması ile shape dosyasına dönüştürülmüştür. Bu shape dosyası ArcMap uygulaması ile açılıp gerçek değerle emsal tahmininin mutlak değer farkı Şekil 5.10'da harita üzerinde renklendirilerek gösterilmiştir.



Şekil 5.10. Hata Dağılımı Haritası

ArcMap uygulaması üzerinden hata oranı yüksek taşınmazlar incelenmiştir. Emsal tahmini için kullanılan yakın taşınmazlar ile arasında bu farkın oluşmasına sebep olabilecek özellikler tespit edilmeye çalışılmıştır. Yapılan incelemelerde 1000 TL ve üzerinde fark çıkan örneklere bakılmış fakat bu değer farkına sebep olabilecek bir öznitelik bilgisi bulunamamıştır.

TETS Uygulamasının 1799 tane test verisi üzerinden ürettiği emsal tahminleri incelemeye devam edilmiş ve gerçek değer ile uygulanan emsal tahmin yöntemlerine göre yapılan tahminler arasındaki mutlak değer farkları gruplandırılmıştır. Veri setinin gerçek birim fiyat ortalaması 3043,74 TL. olarak bulunmuştur. Hata miktarlarını gruplamak için

ortalama değerin yaklaşık %10, %20, %30 ve %50'si olan 300, 600, 900 ve 1500 TL. değerleri sınır olarak kullanılmıştır.

Çizelge 5.2. Test verisi için hata miktarları dağılımı.

Yöntem	**0-300 TL**	**300-600 TL**	**600-900 TL**	**900-1500 TL**	**1500-3000 TL**	**Toplam**
IDW	930	525	206	118	20	1799
OLS	1000	521	205	80	13	1799
SLR	995	535	184	70	15	1799

Çizelge 5.2. incelendiğinde 0-300 TL hata yapılan taşınmaz sayısı IDW, OLS ve SLR yöntemlerine göre sırasıyla 930, 1000 ve 995, 300-600 arasının ise 525, 521 ve 535 olduğu görülmektedir. Bu sonuçlara göre en iyi sonucu veren SLR yönteminin veri setinin %55,3'ünün emsal tahmin hatası 300 TL altında kalmaktadır. Bu orana 300-600 TL aralığında hata yapılan taşınmazlar da eklendiğinde veri setinin %85'ini oluşturmaktadır. Bu istatistikler ışığında veri setinin %85'inin hatasının, ortalama birim fiyatın %20'sinde kaldığı tespit edilmiştir.

9 Ocak 2020 tarihinde resmî gazetede yayımlanan "Bankaların Değerleme Hizmeti Almaları ve Bankalara Değerleme Hizmeti Verecek Kuruluşların Yetkilendirilmesi ve Faaliyetleri Hakkında Yönetmelikte Değişiklik Yapılmasına Dair Yönetmelik" başlıklı yönetmeliğin 20. maddesinde değeri 5 milyon TL'nin altında olan taşınmazların değerine itiraz halinde bir başka değerleme kuruluşuna değerleme yaptırılabileceği, ilk değerleme ile arasında %20 'den az fark olması halinde aritmetik ortalaması alınarak değer tespit edileceği belirtilmiştir [97]. Yönetmelik değerlendirildiğinde ve değerleme kuruluşları ile yapılan görüşmeler sonucunda iki farklı uzman tarafından aynı taşınmaz için yapılan değer tespitinin farklılık oranının %20 olmasının beklenti dahilinde olduğu değerlendirilmiştir. Bu sebeple TETS uygulamasının, test verisinin yaklaşık %..'inde ortalama birim fiyata göre %20 hata yapmasının kabul edilebilir seviyede olduğu düşünülmektedir.

Uygulama içinde değer tespiti için kullanılan taşınmaz özelliklerinin taşınmazın bulunduğu bölgeye göre farklı ağırlıklarda değere etki ettiği görülmüştür. Örnek olarak bazı bölgelerde otopark durumunun çok önemli olması sonucu değere daha çok etki ettiği

görülürken bazı bölgelerde otopark durumunun değer için önemli olmadığı düşünülmektedir. Mobil bir uygulama olan TETS uygulaması sahada uzmana yardımcı olması için tasarlandığından, kullanım kolaylığı sağlaması için taşınmaz özellik girişi 10 tane ile sınırlandırılmıştır. Konu ile ilgili gelecek çalışmalarda taşınmazın bulunduğu bölgeye göre otomatik özellik seçimi üzerinde çalışılabileceği düşünülmektedir.

Taşınmaz değerlemesine etki eden bir diğer konu ise insan faktördür. Tapu ve Kadastro Genel Müdürlüğü Taşınmaz Değerleme Başkanlığı, kurumun yaptığı toplu değerleme pilot çalışmalarında insan faktörünün değere etki ettiğinden bahsetmiştir (kişisel iletişim,15.12.2021). Aynı zamanda taşınmazın özelliklerinden bağımsız olarak komşuluk ilişkileri, yakın taşınmazlarda ikamet edenlerin gelir düzeylerinin ve mesleklerinin değeri etkilediği vurgulanmıştır. Bu değerlendirmeler uygulama testlerinde çıkan sayıca az da olsa yüksek farklılıkların belli oranda insan faktörüyle alakalı olabileceğini açıklamaktadır. Bununla birlikte insanların güvenliğe verdiği öneminin artmasıyla birlikte site içinde bulunan ve depreme dayanıklı olduğu bilinen taşınmazlara ilgi artmaktadır. Taşınmaz bilgileri içine deprem ve yapı dayanıklılığı ile ilgili öznitelik bilgilerinin eklenmesinin emsal tahminin daha doğru yapılabileceği fikrini oluşturmuştur.

TETS uygulaması test edilirken Tapu ve Kadastro Genel Müdürlüğünün 2013 yılında yaptığı pilot çalışmanın verileri kullanılmıştır. Bu verilerde taşınmaz değerleri 2011-2013 yılları arasında belirlenmiş değerlerdir. Değerler Merkez Bankasının Ankara ili için yayımladığı aylık birim fiyat değişimlerinin oranlanması ile 2021 yılı eylül ayına taşınmıştır. Yayımlanan birim fiyatların Ankara il geneli ortalaması olduğu düşünüldüğünde test bölgesi olan MAMAK ilçesini ne kadar temsil ettiği tartışmalıdır. Bu kapsamda konu ile ilgili gelecekte yapılabilecek çalışmalar arasına bölgesel olarak birim fiyat normalizasyonunun eklenmesi gerektiği düşünülmektedir. Özellikle sistemin kendi kendini beslediği düşünülecek olursa, verinin kendi içinde yıllık veya aylık değişim oranını bölgesel olarak otomatik belirlenebileceği bir yöntemin geliştirilebileceği düşünülmektedir. Ayrıca konuma göre ağırlıklandırmanın yanı sıra zamana bağlı ağırlıklandırmanın da sonuçlara katkı sağlayabileceği değerlendirilmiştir.

Taşınmaz değerlemesi ile ilgili yapılan literatür taramaları incelendiğinde genel olarak toplu değerleme ile ilgili çalışmaların olduğu görülmektedir. Tekil değerleme için uygulama haline getirilmiş bir çalışma bulunamamıştır. Uygulamanın tekil değerleme konusunda, bulunduğu konuma ve özelliklerine bağlı olarak emsal tahmini yapması güçlü özelliklerinden biri olarak değerlendirilebilir. Değerleme firmalarındaki süreç akışının

uygulama üzerinden yönetilmesi ve sistemin kendi kendini besleyerek hem kurumsal hafıza oluşturması hem de belirlenmiş tüm değerleri bir sonraki emsal tahmininde kullanması açısından önemlidir. Uygulamanın kullanıcı dostu bir ara yüze sahip olması hem bilgisayar hem de mobil ortamda kolayca kullanılmasını sağlamaktadır. Ayrıca yaratılan emsal belirleme sınıf hiyerarşisi, farklı emsal tahmin yöntemlerinin sisteme entegre edilmesini kolaylaştırmaktadır. Sistemin daha verimli kullanılması ve emsal tahminindeki doğruluğun artırılması için taşınmaz piyasasındaki fiyat değişimlerinin fazla olduğu bölgelerde daha önceden belirlenmiş birim fiyatların normalizasyonu konusu üstünde çalışılması gerekmektedir. Bununla birlikte değer belirlenirken kullanılan taşınmaz özelliklerinin bölgesel olarak farklılık gösterebileceğinin dikkate alınarak otomatik özellik seçme yeteneği üzerinde çalışılması gerektiği düşünülmektedir.

6. SONUÇ

Günümüzde taşınmaz sektörünün büyüklüğü düşünüldüğünde taşınmaz değerlemesinin önemi anlaşılmaktadır. Dünyada ve ülkemizde taşınmaz değerlemesi değerleme uzmanları tarafından yapılmaktadır. Bilgi teknolojilerinin gelişmesi ile birlikte taşınmaz değerlemesinin de mobil veya web uygulamaları yardımıyla karar desteği vermesi kaçınılmaz olmaktadır. Bu çalışmada değerleme firmaları için taşınmaz değerleme uzmanlarının sahada geçireceği süreyi azaltacak ve denetmenler için değer kontrolünü kolaylaştıracak web tabanlı bir mobil uygulama ile birlikte bölgesel emsal tahmin yaklaşımı geliştirilmiştir.

Uygulama taşınmaz tanım bilgilerinin sisteme girilmesini, taşınmazın değerlemesi ile ilgili görevin atanması, değerleme uzmanının değere esas bilgileri girerek uygulamadan emsal tahminini istemesi ve değeri belirlemesi, denetmenin onay mekanizması adımlarından oluşmaktadır.

TETS uygulaması değerleme süreci için karar desteği sağlamaktadır. Emsal tahmini üreterek uzman için değer kararına ve emsal tahmini ile uzmanın belirlediği değeri karşılaştırma imkânı vererek denetmenin onay kararına yardımcı olması beklenmektedir. Uygulamanın karar desteği için en önemli özelliği emsal tahmini üretmesidir. Bu tahmin için Ankara ili Mamak ilçesi test bölgesi olarak seçilmiştir. OLS, IDW ve SLR yöntemleri test verisi için uygulanmıştır. 1799 kayıttan oluşan test verisindeki değerler 2021 yılındaki olması gereken değerlere dönüştürülerek testler gerçekleştirilmiştir. Test sonuçlarına göre uygulama taşınmaz değerleme süreç adımlarını başarı ile gerçekleştirmiştir. Emsal tahmin yöntemleri için konum bazlı IDW yöntemi, öznitelik bazlı OLS yöntemi ve hem konum hem öznitelik bazlı SLR yöntemi kullanılmıştır. SLR yönteminin test bölgesi için daha iyi sonuç verdiği görülmüştür.

Uygulama, farklı emsal tahmin yöntemlerinin entegre edilmesine imkân verecek şekilde tasarlanmıştır. Uygulama, test amaçlı kullanılan IDW, OLS ve SLR yöntemlerinin yanı sıra toplu değerleme için kullanılan random forest, yapay sinir ağları, kriging gibi yöntemler de tekil değerleme için kolayca entegre edilebilecek şekilde tasarlanmıştır.

7. KAYNAKLAR

[1] T. Yomralıoğlu, R. Nişancı, M. Çete, E. Candaş, Dünya'da ve Türkiye'de Taşınmaz Değerlemesi, (2011) 1–18.

[2] A. Bayraktar, Türkiye'de Taşınmaz Değerlemesinde Bankaların Gözettiği Değerleme Faktörlerinin Karşılaştırılması Ve Ortak Bir Rapor Formatı Önerisi, 2019.

[3] Türkiye'de 9 Ayda 2 Milyon 118 Bin gayrimenkul Satıldı | Tapu ve Kadastro Genel Müdürlüğü (TKGM), (n.d.). https://www.tkgm.gov.tr/turkiye%27de-9-ayda-2-milyon-118-bin-gayrimenkul-satildi (accessed November 10, 2021).

[4] GYODER, Türkiye Gayrimenkul Sektörü 2021 2.Çeyrek Raporu -, (2021).

[5] Başbakanlık Mevzuatı Geliştirme ve Yayın Genel Müdürlüğü, (n.d.). https://www.resmigazete.gov.tr/eskiler/2017/02/20170201-8.htm (accessed November 11, 2021).

[6] UDSK, Uluslararası Değerleme Standartları, (2017).

[7] E. Hromada, Real Estate Valuation Using Data Mining Software, Procedia Engineering. 164 (2016) 284–291. https://doi.org/10.1016/j.proeng.2016.11.621.

[8] International Association of Assessing Officers (IAAO), Guidance on International Mass Appraisal and Related Tax Policy, Journal of Property Tax Assessment & Administration. 11 (2014) 5–33.

[9] A. Yılmaz, Çok Ölçütlü Karar Destek Sistemleri İle Taşınmaz Değerleme Ve Oran Çalışması, 2010.

[10] Ü. Yıldız, Dünyada Küme Değerleme Uygulamaları Ve Türkiye Modeli Önerisi, 2012.

[11] T. Pişkin, Coğrafi Bilgi Sistemleri Yardımıyla Taşınmaz Değerlemesi: Kınalı-Tekirdağ-Çanakkale-Savaştepe Otoyolu Malkara-Çanakkale Kesimi (Çanakkale 1915 Köprüsü Dahil) Örneği, 2021.

[12] B. Park, J. Kwon Bae, Using Machine Learning Algorithms For Housing Price Prediction: The Case Of Fairfax County, Virginia Housing Data,

Expert Systems with Applications. 42 (2015) 2928–2934. https://doi.org/10.1016/j.eswa.2014.11.040.

[13] V. Chiarazzo, L. Caggiani, M. Marinelli, M. Ottomanelli, A Neural Network Based Model For Real Estate Price Estimation Considering Environmental Quality Of Property Location, Transportation Research Procedia. 3 (2014) 810–817. https://doi.org/10.1016/j.trpro.2014.10.067.

[14] M.E. Tabar, Yapay Sinir Ağları ve Bulanık Mantıkla Gayrimenkul Değerleme Modelinin Oluşturulması: Samsun Örneği, 2020.

[15] R. Cellmer, The Possibilities and Limitations of Geostatistical Methods in Real Estate Market Analyses, Real Estate Management and Valuation. 22 (2014) 54–62. https://doi.org/10.2478/remav-2014-0027.

[16] S. Basu, T.G. Thibodeau, Analysis of Spatial Autocorrelation in House Prices, Journal of Real Estate Finance and Economics. 17 (1998) 61–85.

[17] B. Calka, Estimating Residential Property Values On The Basis Of Clustering And Geostatistics, Geosciences (Switzerland). 9 (2019) 1–14. https://doi.org/10.3390/geosciences9030143.

[18] H. Crosby, T. Damoulas, A. Caton, P. Davis, J. Porto de Albuquerque, S.A. Jarvis, Road Distance And Travel Time For An İmproved House Price Kriging Predictor, Geo-Spatial Information Science. 21 (2018) 185–194. https://doi.org/10.1080/10095020.2018.1503775.

[19] M.C. Morillo Balsera, S. Martínez-Cuevas, I. Molina Sánchez, C. García-Aranda, M.E. Martinez Izquierdo, Artificial Neural Networks And Geostatistical Models For Housing Valuations in Urban Residential Areas, Geografisk Tidsskrift - Danish Journal of Geography . 118 (2018) 184–193. https://doi.org/10.1080/00167223.2018.1498364.

[20] W.R. Tobler, A Computer Movie Simulating Urban Growth in the Detroit Region, 1970.

[21] J. Chica-Olmo, Prediction Of Housing Location Price By A Multivariate Spatial Method: Cokriging, Journal of Real Estate Research. 29 (2007) 91–114. https://doi.org/10.1080/10835547.2007.12091188.

[22] J.M. Montero-Lorenzo, B. Larraz-Iribas, Space-Time Approach To Commercial Property Prices Valuation, Applied Economics. 44 (2012) 3705–3715. https://doi.org/10.1080/00036846.2011.581212.

[23] F. Bünyan Ünel, Ş. Yalpır, Türkiye'de Taşınmazların Değerini Etkileyen Kriterlere Yaklaşım, Geomatik. (2019). https://doi.org/10.29128/geomatik.499681.

[24] TÜİK Kurumsal, (n.d.). https://data.tuik.gov.tr/Bulten/Index?p=Konut-Satis-Istatistikleri-Eylul-2021-37477 (accessed November 13, 2021).

[25] Ü. Yıldız, Gayrimenkul Bilimlerinde Kitlesel Değerleme Uygulamaları Ve Türkiye İçin Model Önerisi, 2014.

[26] Başkanlık Birimleri ve Görevleri | Tapu ve Kadastro Genel Müdürlüğü (TKGM), (n.d.). https://www.tkgm.gov.tr/tasinmazd-db/baskanlik-birimleri-ve-gorevleri (accessed November 13, 2021).

[27] C. Ulvi, Taşınmaz Değerlemede Yapay Zekâ Tekniklerinin Kullanılabilirliği Ve Yöntemlerin Karşılaştırılması, 2018.

[28] M.A. Bayrak, Avrupa İnsan Hakları Mahkemesi Ve Yargı Kararları Bağlamında Türkiye'de Taşınmaz Değerleme Sistemindeki Temel Sorunların Tespiti, 2021.

[29] Z. Bulut, Real Estate Appraisal Methods And Their Application In Ankara, 2011. http://www.ainfo.inia.uy/digital/bitstream/item/7130/1/LUZARDO-BUIATRIA-2017.pdf.

[30] A.D. Dongare, R.R. Kharde, A.D. Kachare, Introduction to Artificial Neural Network (ANN) Methods, International Journal of Engineering and Innovative Technology (IJEIT). 2 (2012) 189–194. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1082.1323&rep=rep1&type=pdf.

[31] M.S. Wanker, Research Paper on Basic of Artificial Neural Network, International Journal on Recent and Innovation Trends in Computing and Communication. 2 (2014) 96–100.

[32] T. Mete, Kesikli Bir Biyoreaktörde Yapay Sinir Ağlarının Kullanımı, 2008.

[33] M.E. TABAR, A.C. BAŞARA, Y. ŞİŞMAN, Çoklu Regresyon ve Yapay Sinir Ağları ile Tokat İlinde Konut Değerleme Çalışması, Türkiye Arazi Yönetimi Dergisi. 3 (2021) 1–7. https://doi.org/10.51765/tayod.832227.

[34] B. Savaş, Makine Öğrenme Algoritmalarının Konut Değer Tahmininde Kullanımı: Ankara Gölbaşı Uygulaması, 2019.

[35] L.A. Zadeh, Fuzzy Sets, (1965). https://doi.org/10.1061/9780784413616.194.

[36] G. Biau, E. Scornet, A random forest guided tour, Test. 25 (2016) 197–227. https://doi.org/10.1007/s11749-016-0481-7.

[37] G. Matheron, Principles of geostatistics, Economic Geology. 58 (1963) 1246–1266. https://doi.org/10.2113/gsecongeo.58.8.1246.

[38] A. Stein, L.C.A. Corsten, Universal Kriging and Cokriging as a Regression Procedure, Biometrics. 47 (1991) 575. https://doi.org/10.2307/2532147.

[39] M. SUCU, KARAR DESTEK SİSTEMLERİ: BİR LİTERATÜR DEĞERLENDİRMESİ, Pamukkale University Journal of Social Sciences Institute. (2020). https://doi.org/10.30794/pausbed.649150.

[40] H. Zhang, Architecture of Network and Client-Server model, (2013) 1–3. http://arxiv.org/abs/1307.6665.

[41] H.S. Oluwatosin, Client-Server Model, IOSR Journal of Computer Engineering. 16 (2014) 57–71. https://doi.org/10.9790/0661-16195771.

[42] N. ULUĞTEKİN, A.Ö. DOĞRU, Coğrafi Bilgi Sistemleri Ve Harita: Kartografya, Ege Üniversitesi CBS Sempozyumu. (2005) 209–2015.

[43] Becoming A Data-Driven CEO | Domo, (n.d.). https://www.domo.com/solution/data-never-sleeps-6 (accessed November 16, 2021).

[44] S. Spaccapietra, C. Parent, C. Vangenot, GIS databases: From multiscale to multirepresentation, Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science). 1864 (2000) 57–70. https://doi.org/10.1007/3-540-44914-0_4.

[45] MySQL, (n.d.). https://www.mysql.com/ (accessed November 15, 2021).

[46] Oracle, MySQL Workbench. Design. Database. Administration. Development, (2013) 1–25.

[47] PostgreSQL: The world’s most advanced open source database, (n.d.). https://www.postgresql.org/ (accessed November 15, 2021).

[48] PostGIS — Spatial and Geographic Objects for PostgreSQL, (n.d.). https://postgis.net/ (accessed November 15, 2021).

[49] A. Kütükçü, MEKANSAL VERİTABANLARINDA HIZLI SORGULAMA, 2009.

[50] L. Zhang, J. Yi, Management methods of spatial data based on PostGIS, 2010 2nd Pacific-Asia Conference on Circuits, Communications and System, PACCS 2010. 1 (2010) 410–413. https://doi.org/10.1109/PACCS.2010.5626962.

[51] K. Gorman, A. Hirt, D. Noderer, J. Rowland-Jones, A. Sirpal, D. Ryan, B. Woody, Introducing Microsoft SQL Server 2019, 2019. https://www.winpro.com.sg/wp-content/uploads/2020/10/Introducing-Microsoft-SQL-Server-2019.pdf.

[52] SQL Server 2019 | Microsoft, (n.d.). https://www.microsoft.com/tr-tr/sql-server/sql-server-2019 (accessed November 16, 2021).

[53] OpenStreetMap Wiki, (n.d.). https://wiki.openstreetmap.org/wiki/Main_Page (accessed November 16, 2021).

[54] Leaflet - a JavaScript library for interactive maps, (n.d.). https://leafletjs.com/ (accessed November 17, 2021).

[55] P. Crickard, Leaflet . js Essentials, 2014.

[56] E. Dodsworth, A. Nicholson, Academic uses of Google Earth and Google Maps in a library setting, Information Technology and Libraries. 31 (2012) 102–117. https://doi.org/10.6017/ital.v31i2.1848.

[57] Google Maps Platform | Google Developers, (n.d.). https://developers.google.com/maps (accessed November 17, 2021).

[58] Maps API — Yandex Technologies, (n.d.). https://yandex.com/dev/maps/ (accessed November 17, 2021).

[59] GitHub - yandex/mapsapi-area: util.calculateArea: plugin for calculating geodesic features area., (n.d.). https://github.com/yandex/mapsapi-area?from=jsapi (accessed November 17, 2021).

[60] GitHub - yandex/mapsapi-heatmap: Heatmap: Yandex.Maps API plugin for data visualization, (n.d.). https://github.com/yandex/mapsapi-heatmap?from=jsapi (accessed November 17, 2021).

[61] Bing Maps Documentation - Bing Maps | Microsoft Docs, (n.d.). https://docs.microsoft.com/en-us/bingmaps/ (accessed November 17, 2021).

[62] GitHub - microsoft/MapsSDK-Native: This repository contains samples, documentation and releases history for the Bing Maps SDK for Android and iOS., (n.d.). https://github.com/Microsoft/MapsSDK-native (accessed November 17, 2021).

[63] GitHub - microsoft/MapsSDK-Unity: This repository contains samples, documentation, and supporting scripts for Maps SDK, a Microsoft Garage project., (n.d.). https://github.com/Microsoft/MapsSDK-Unity (accessed November 17, 2021).

[64] What is ASP.NET Core? | .NET, (n.d.). https://dotnet.microsoft.com/learn/aspnet/what-is-aspnet-core (accessed November 18, 2021).

[65] A. Leff, J.T. Rayfield, Web-application development using the Model/View/Controller design pattern, Proceedings - 5th IEEE International Enterprise Distributed Object Computing Conference. 2001-Janua (2001) 118–127. https://doi.org/10.1109/EDOC.2001.950428.

[66] M.R.J. Qureshi, F. Sabir, A comparison of model view controller and model view presenter, 25 (2014) 7–9. http://arxiv.org/abs/1408.5786.

[67] G.E. Krasner, S.T. Pope, A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 System, Journal Of Object Oriented Programming. 1 (1988) 26–49.

http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.47.366%5Cnhttp://portal.acm.org/citation.cfm?id=50757.50759%5Cnhttp://dl.acm.org/citation.cfm?id=50757.50759%5Cnhttp://www.itu.dk/courses/VOP/E2005/VOP2005E/8_mvc_krasner_and_pope.pdf.

[68] Snyk, JVM Ecosystem Report 2021, (2021) 1–44. https://snyk.io/blog/jvm-ecosystem-report-2020/.

[69] Spring | Home, (n.d.). https://spring.io/ (accessed November 18, 2021).

[70] PHP: Documentation, (n.d.). https://www.php.net/docs.php (accessed November 19, 2021).

[71] Laravel - The PHP Framework For Web Artisans, (n.d.). https://laravel.com/ (accessed November 19, 2021).

[72] Welcome to CodeIgniter, (n.d.). https://codeigniter.com/ (accessed November 19, 2021).

[73] The web framework for perfectionists with deadlines | Django, (n.d.). https://www.djangoproject.com/ (accessed November 19, 2021).

[74] XML Essentials - W3C, (n.d.). https://www.w3.org/standards/xml/core (accessed November 19, 2021).

[75] OGC, OGC® Geography Markup Language (GML) — Extended schemas and encoding rules, OpenGIS Recommendation Paper. (2010) 595. http://www.w3.org/TR/ws-arch/.

[76] JSON, (n.d.). https://www.json.org/json-tr.html (accessed November 20, 2021).

[77] ECMA International, The JSON Data Interchange Syntax, Standard ECMA-404. 2nd Editio (2017) 8. http://www.ecma-international.org/publications/standards/Ecma-404.htm.

[78] JSON Data (Standard), (n.d.). https://docs.oracle.com/en/database/oracle/oracle-database/21/adjsn/json-data.html#GUID-B2D82ED4-B007-4019-8B53-9D0CDA81C4FA (accessed November 20, 2021).

[79] rfc7946, (n.d.). https://datatracker.ietf.org/doc/html/rfc7946 (accessed November 20, 2021).

[80] Stack Overflow Developer Survey 2020, (n.d.). https://insights.stackoverflow.com/survey/2020/#most-popular-technologies (accessed November 20, 2021).

[81] A. Wirfs-Brock, B. Eich, JavaScript: The first 20 years, Proceedings of the ACM on Programming Languages. 4 (2020). https://doi.org/10.1145/3386327.

[82] ECMA International, ECMA-262, (2021).

[83] L.D. Paulson, Building rich Web applications with Ajax, Computer. 38 (2005) 14–17. https://doi.org/10.1109/MC.2005.330.

[84] C. Györödi, R. Györödi, G. Pecherle, T. Lorand, R. Alin, Web 2.0 technologies with jQuery and Ajax, Computer Technology and Computer Programming: New Research and Strategies. (2016) 99–110. https://doi.org/10.1201/b13124-7.

[85] Application Fundamentals | Android Developers, (n.d.). https://developer.android.com/guide/components/fundamentals (accessed November 21, 2021).

[86] Swift - Apple Developer, (n.d.). https://developer.apple.com/swift/ (accessed November 21, 2021).

[87] Location | Android Developers, (n.d.). https://developer.android.com/reference/android/location/Location (accessed November 21, 2021).

[88] Core Location | Apple Developer Documentation, (n.d.). https://developer.apple.com/documentation/corelocation (accessed November 21, 2021).

[89] Starting with HTML + CSS, (n.d.). https://www.w3.org/Style/Examples/011/firstcss (accessed November 21, 2021).

[90] HTML Responsive Web Design, (n.d.). https://www.w3schools.com/html/html_responsive.asp (accessed November 21, 2021).

[91] Flutter - Build apps for any screen, (n.d.). https://flutter.dev/?gclid=Cj0KCQiA-eeMBhCpARIsAAZfxZANrfj8j27J7EyL3GKz2B80OVF07Ldov5WRQtRMl9_NR5HNs5WhZs4aAhyUEALw_wcB&gclsrc=aw.ds (accessed November 21, 2021).

[92] S. Göğsu, K.Ö. Hastaoğlu, Ters Mesafe Ağırlıklı Enterpolasyon Yönteminde Güç Fonksiyonu Etkisinin İncelenmesi, (2019) 25–27.

[93] Spatial Regression · Geographic Data Science with PySAL and the pydata stack, (n.d.). http://darribas.org/gds_scipy16/ipynb_md/08_spatial_regression.html (accessed February 16, 2022).

[94] G. James, D. Witten, T. Hastie, R. Tibshirani, An Introduction to Statistical Learning with Applications in R, 2013.

[95] S. Yilmazer, S. Kocaman, A mass appraisal assessment study using machine learning based on multiple regression and random forest, Land Use Policy. 99 (2020). https://doi.org/10.1016/j.landusepol.2020.104889.

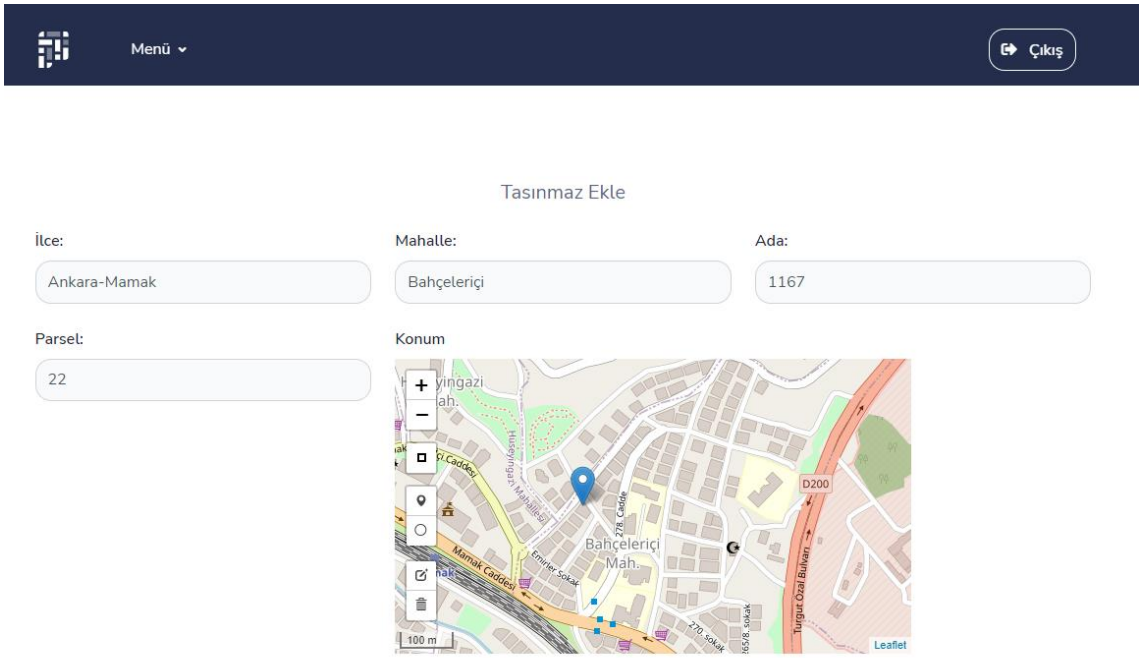
[96] EVDS | Tüm Seriler, (n.d.). https://evds2.tcmb.gov.tr/index.php?/evds/serieMarket/collapse_26/5949/DataGroup/turkish/bie_hkfe/ (accessed December 28, 2021).

[97] BANKALARIN DEĞERLEME HİZMETİ ALMALARI VE BANKALARA DEĞERLEME HİZMETİ VERECEK KURULUŞLARIN YETKİLENDİRİLMESİ VE FAALİYETLERİ HAKKINDA YÖNETMELİKTE DEĞİŞİKLİK YAPILMASINA DAİR YÖNETMELİK, (n.d.). https://www.resmigazete.gov.tr/eskiler/2020/01/20200109-3.htm.

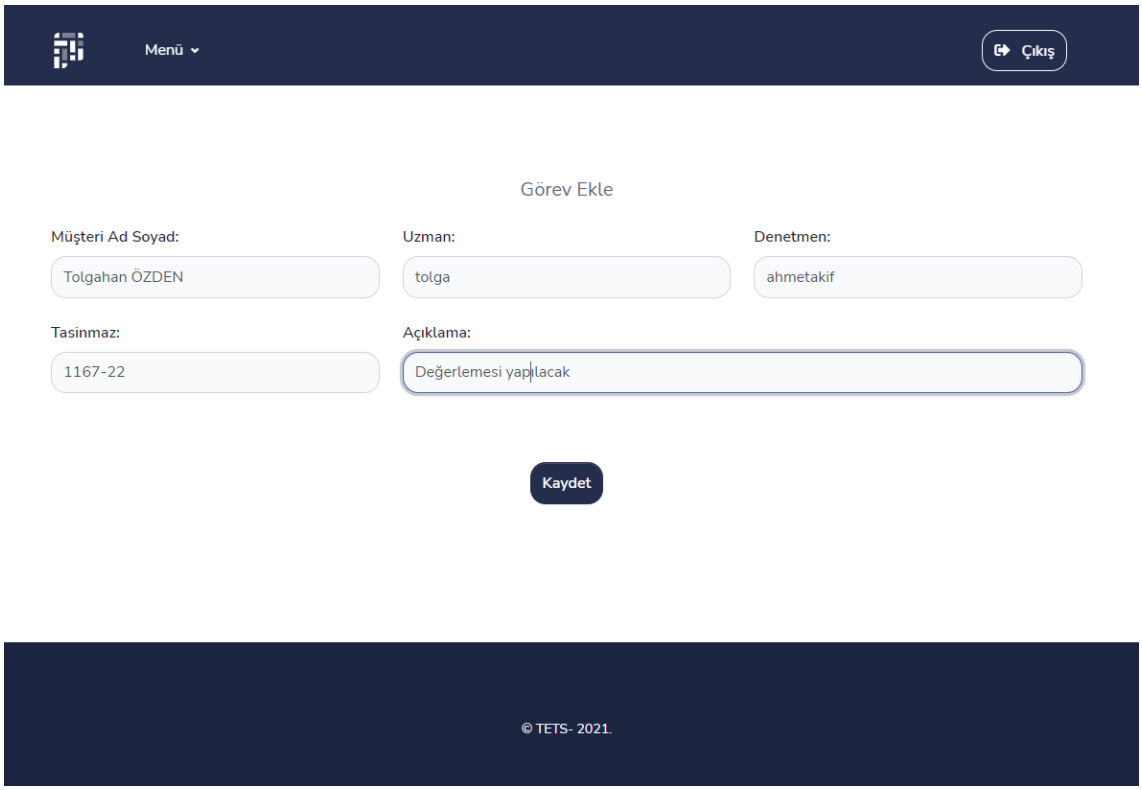
EKLER

EK 1 –TETS Uygulaması Ekran Görüntüleri

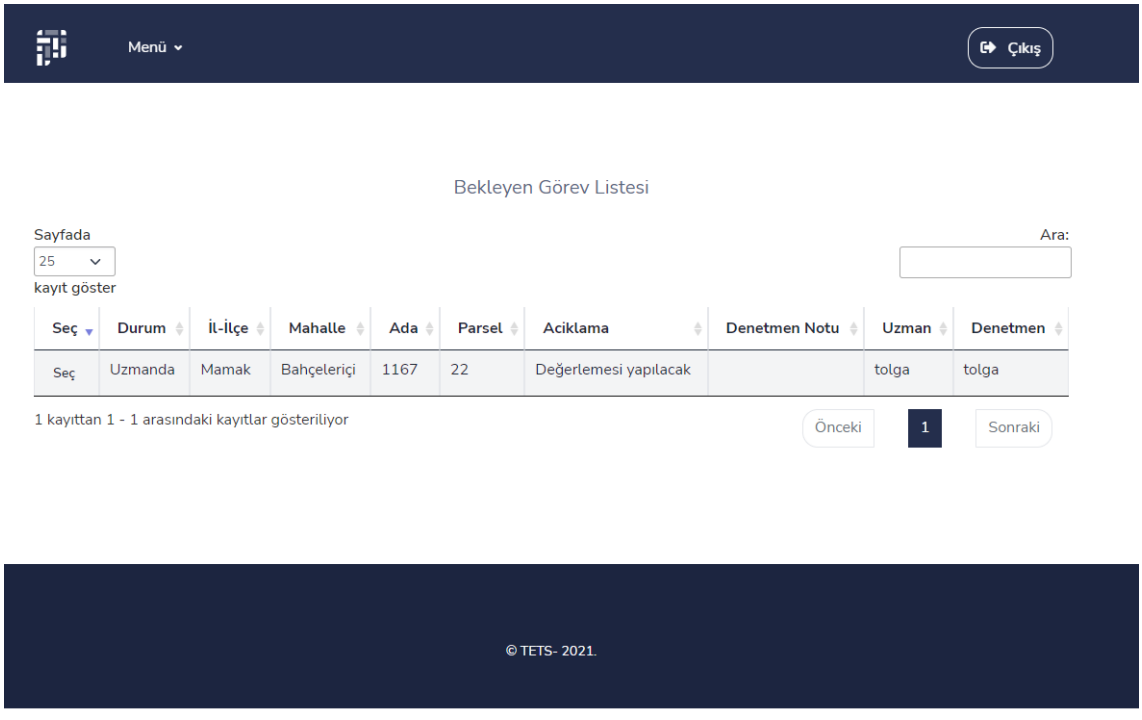
Şekil Ek1.1 Uygulama Giriş Ekranı



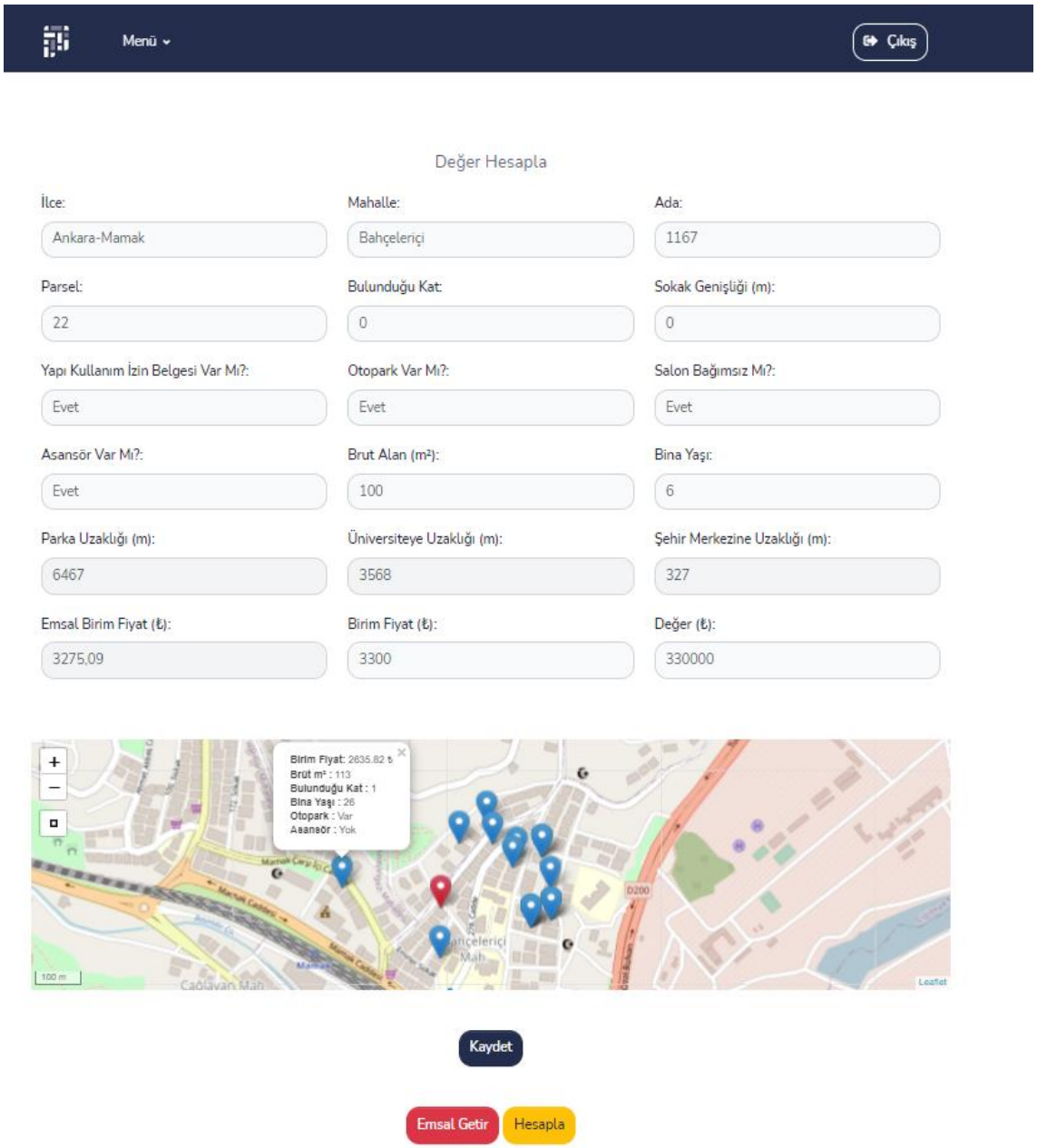
Şekil Ek1.2 Yeni Taşınmaz Ekleme Sayfası



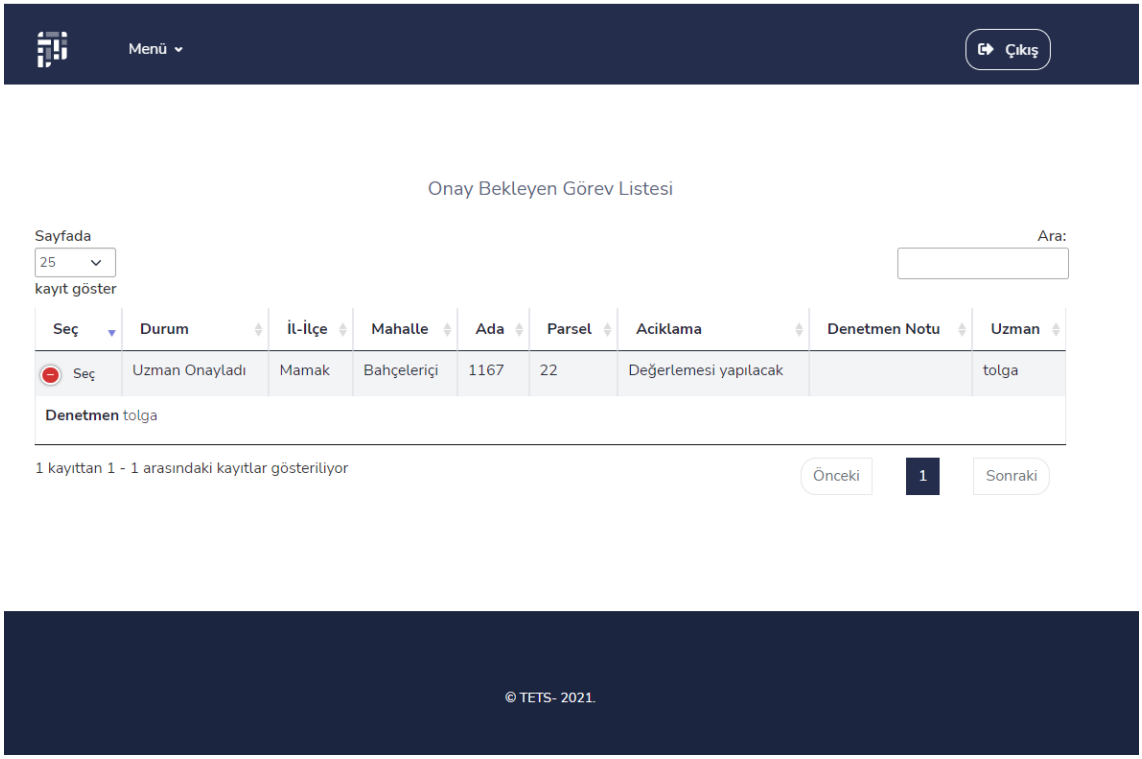
Şekil Ek1.3 Görev Ekleme Sayfası



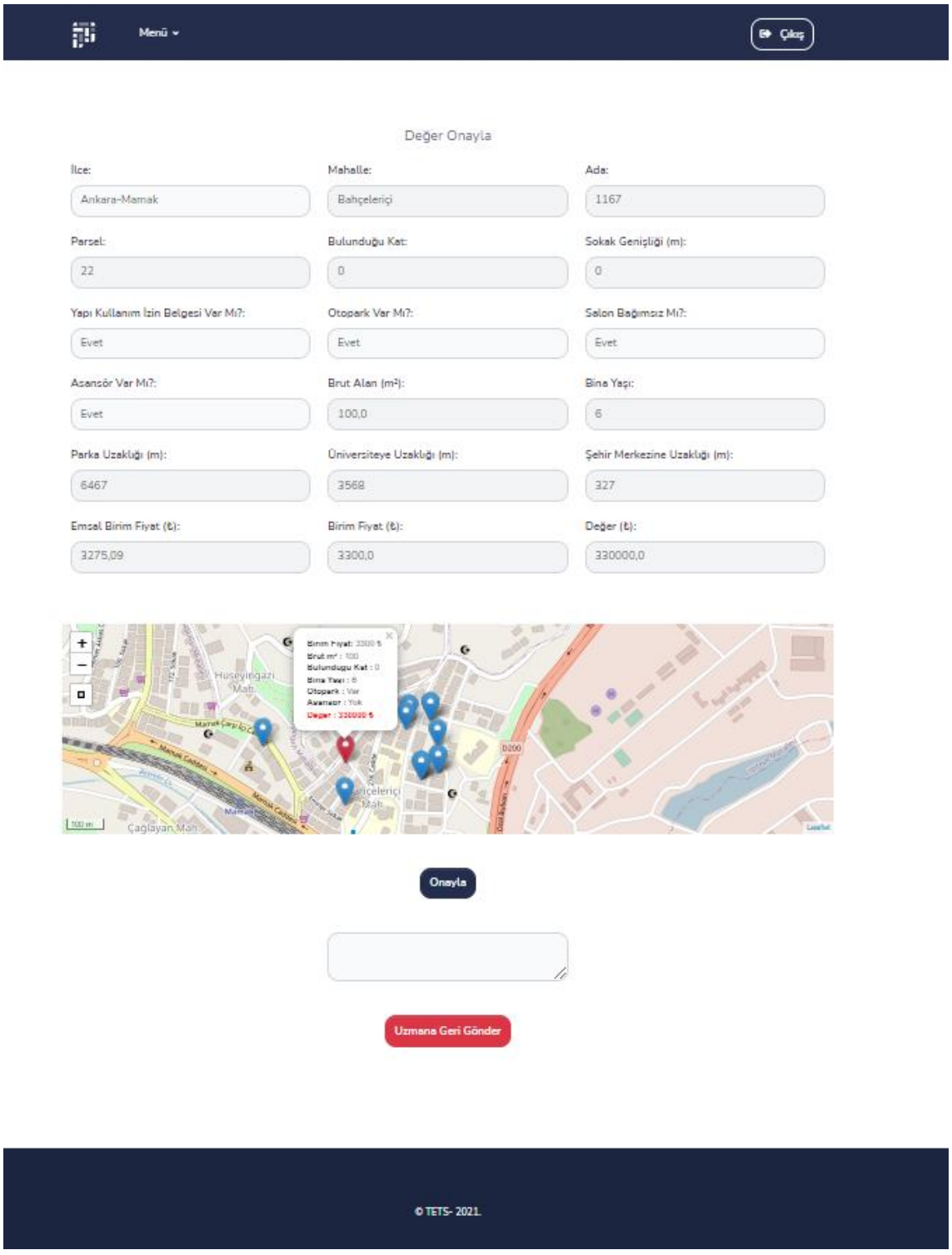
Şekil Ek1.4 Uzman Görev Listesi



Şekil Ek1.5 Uzman Değer Hesaplama Sayfası



Şekil Ek1.6 Onay Bekleyen Görev Listesi



Şekil Ek1.7 Denetmen Onay Sayfası