İSTANBUL ÜNİVERSİTESİ CERRAHPAŞA MÜHENDİSLİK FAKÜLTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİTİRME PROJESİ

VERİ ANALİZİ YÖNTEMİYLE EKONOMİK SEPET UYGULAMASI

Hazırlayanlar: Ömer Can Uyar 1306170082
Umut Can Öztürk 1306170070

Danışman: Doç. Dr. DERYA YILTAŞ KAPLAN

HAZİRAN - 2023

ÖNSÖZ

Fiyat karşılaştırma servisleri, hem teorik hem de pratik açıdan geniş bir ilgi alanıdır. İlk uygulamalarından bu yana, bu servislerin hızlı bir ilerleme kaydettiği görülmektedir. Fiyat karşılaştırma servisleri, hayatın çeşitli alanlarında uygulanabilirliği olması ve kullanımının giderek yaygınlaşması nedeniyle önem kazanmıştır. Anlık veri takibi ve kişiye özel uygulamaların geliştirilmesi sayesinde, bu servislerin hayatımızın önemli bir parçası haline gelmesi beklenmektedir.[1]

Ancak, günümüzde fiyat karşılaştırma servisleriyle ilgili en büyük sorun, veri toplamadır. Kaynaktan alınan verilerin sistemlere aktarılması, aktarım sonrası doğrulanması ve verilerin karşılaştırılması süreci çeşitli sorunlara yol açabilmektedir. 7 Aralık 2022 tarihinde yayınlanan 32036 sayılı Resmi Gazete'ye göre, zincir marketlerin verilerini Ticaret Bakanlığı'nın sistemine girmesi zorunluluk hâline gelmiştir. Bu durum, belirtilen sorunlara bir ölçüde çözüm getirebilmektedir. Ticaret Bakanlığı'nın verilerini açık kaynak olarak sunması sonrasında fiyat karşılaştırma servislerinin hizmet kalitesinde artış beklenebilir.[2] Sistem gecikmelerinin azaltılması için geliştirilen yöntemler, ortamlar ve mimariler, fiyat karşılaştırma servislerindeki kalitenin belirleyici faktörleri olacaktır.

Bu tez aynı zamanda, kullanıcı-tabanlı bir fiyat karşılaştırma uygulamasının teknik yönlerini de ele almaktadır. Uygulama geliştirme sürecinde kullanılan programlama dilleri, seçilen çerçeveler (frameworks), veritabanı yönetimi, kuyruk yönetimi, kullanıcı arayüzü tasarımı ve güvenlik önlemleri gibi konular üzerinde detaylı bir şekilde durulmuştur.

Son olarak, bu tez çalışmasıyla elde edilen sonuçların, perakende ve e-ticaret alanında akademik araştırmalara ve pratik uygulamalara katkı sağlamasını umuyoruz. Uygulama, kullanıcıların alışveriş deneyimini optimize etmeyi hedeflemekte ve gelecekteki çalışmalara ilham kaynağı olabilecektir.

Bu tez çalışmasını tamamlamak için yoğun bir emek sarf edildi ve birçok zorlukla karşılaşıldı. Ancak, bu süreçte kazanılan deneyimler ve elde edilen bilgi birikimi, gelecekteki projelerde ve çalışmalarda değerli bir temel oluşturacaktır.

Sonuç olarak, bu tez çalışması, kullanıcı-tabanlı bir fiyat karşılaştırma mobil uygulamasının ayrıntılı bir incelemesini sunmaktadır. Umarız, bu çalışma perakende ve e-ticaret sektörüne yeni bir bakış açısı getirir ve kullanıcıların alışveriş deneyimini optimize etmeyi hedefleyen yenilikçi çözümler sunar.
Bu çalışmanın gerçekleştirilmesi boyunca bize her türlü destek ve yardımı sağlayan değerli hocamız Doç. Dr. Derya Yıltış Kaplan’a en içten dileklerimizle teşekkürlerimizi sunarız.

Ayrıca, tüm eğitim hayatımız boyunca bize rehberlik eden, destekleyen ve en önemlisi eğitimin önemini ve çalışma disiplinini bize kazandıran ailemize de teşekkürlerimizi sunarız.

Ömer Can Uyar 1306170082
Umut Can Öztürk 1306170070

İÇİNDEKİLER

ŞEKİL LİSTESİ

TABLO LİSTESİ

SEMBOL LİSTESİ

KISALTMA LİSTESİ

UI: Kullanıcı arayüzü
IOS: iPhone Operating System
JSX: JavaScript Syntax eXtension
SQL: Structured Query Language
MACOS: Macintosh Operating System
UNIX: Uniplexed Information and Computing System
UUID: Universally Unique Identifier
XML: Extensible Markup Language
JSON: JavaScript Object Notation
LDAP: Lightweight Directory Access Protocol
SSPI: Security Support Provider Interface
MVCC: Multi-Version Concurrency Control
SDK: Software Development Kit
API: Application Programming Interface
ITO: İstanbul Ticaret Odası
TÜİK: Türkiye İstatistik Kurumu
ORM: Object Relational Mapping

ÖZET

FİYAT KARŞILAŞTIRMA SERVİSLERİ

Bu tez çalışması, fiyat karşılaştırma servislerini mobil platforma taşıyan ve aynı zamanda bu hizmeti fiziki perakende mağazalarına da genişleten bir mobil uygulama geliştirme projesinin detaylı bir incelemesini sunmaktadır. E-ticaret sektöründeki fiyat karşılaştırma servislerinin büyüklüğü ve etkisi, Akakçe ve Cimri gibi popüler platformların varlığı ile açıkça görülmektedir. Ancak, bu servislerin çoğu çevrimiçi alışveriş ile sınırlı kalmış ve fiziki perakende mağazaları büyük ölçüde göz ardı edilmiştir. Bu çalışma, bu boşluğu doldurmayı hedeflemektedir.[3]

Fiyat karşılaştırma servisleri, Türkiye'de Akakçe'nin 2000 yılında kurulması ile popülerleşmiştir. Sonraki yıllarda Cimri gibi alternatiflerin ortaya çıkmasıyla kullanımı artmış ve hizmetler geniş bir ürün yelpazesine yayılmıştır. İlk başta web tabanlı hizmetler sunulurken, son zamanlarda bu hizmetler mobil uygulamalar aracılığıyla kullanıcılarla buluşmaya başlamıştır.

Bu tez, kullanıcının önceki alışveriş deneyimleri ve oluşturdukları sepetleri temel alarak, güncel fiyatlarla en uygun sepeti oluşturmayı sağlayan bir mobil uygulama geliştirme sürecini detaylı bir şekilde inceler. Bu yaklaşım, kullanıcıların süreç içerisinde alışveriş sürelerini azaltmayı ve onlara en uygun fiyatı sunmayı amaçlamaktadır. Yapay zeka teknolojileri kullanılarak, ürünler tek bir sonuç üzerinde kategorize edilir ve bu da doğru sonucu elde etmeyi kolaylaştırır.

Ayrıca, bu çalışma, kullanıcının alışveriş sonrası deneyimini, mağazanın hizmet kalitesini ve konumunu göz önünde bulundurarak değerlendirir. Bu veriler, kullanıcının bir sonraki alışverişinde hangi mağazayı tercih edeceğini belirlemek için bir yapay zeka algoritmasında işlenir. Bu durum, müşteri memnuniyetini ve hizmet kalitesini artırarak kullanıcılara katkı sağlar.

Sonuç olarak, bu tez çalışması, özel algoritmalara ve yapay zeka teknolojilerine dayalı bir mobil uygulama geliştirme süreci sunarak, perakende sektöründe fiyat karşılaştırma servislerini genişletmeyi hedeflemektedir. Bu hizmet, hem e-ticarette hem de fiziki perakende mağazalarında ürünlerin fiyatlarını karşılaştırarak, tüketicilere zaman ve maliyet tasarrufu sağlar.[4] Bu çalışmanın, alışveriş deneyimini daha verimli ve kullanıcı odaklı hale getirerek perakende sektörüne ve e-ticaret ekosistemine önemli bir katkı sağlaması beklenmektedir.

SUMMARY

PRICE COMPARISON SERVICES

This thesis study presents a detailed examination of a mobile application development project that brings price comparison services to the mobile platform and also expands this service to physical retail stores. The size and impact of price comparison services in the e-commerce sector are clearly evident with the presence of popular platforms such as Akakçe and Cimri. However, most of these services have been limited to online shopping, largely ignoring physical retail stores. This study aims to fill this gap.

Price comparison services gained popularity in Turkey with the establishment of Akakçe in 2000. In the following years, usage increased with the emergence of alternatives like Cimri, and the services expanded to a wide range of products. While initially offering web-based services, these services have recently started reaching users through mobile applications.

This thesis thoroughly examines the process of developing a mobile application that enables users to create the most suitable shopping basket with current prices based on their previous shopping experiences and created baskets. This approach aims to reduce users' shopping time during the process and provide them with the best price. By utilizing artificial intelligence technologies, products are categorized into a single result, making it easier to obtain accurate outcomes.

Additionally, this study evaluates the user's post-shopping experience, considering the store's service quality and location. These data are processed in an artificial intelligence algorithm to determine which store the user will prefer for their next shopping. This enhances customer satisfaction and service quality, providing value to users.

In conclusion, this thesis aims to expand price comparison services in the retail sector by presenting a mobile application development process based on specialized algorithms and artificial intelligence technologies. This service saves consumers time and costs by comparing product prices both in e-commerce and physical retail stores. It is expected that this study will make a significant contribution to the retail industry and e-commerce ecosystem by making the shopping experience more efficient and user-focused.

1. GİRİŞ

Projemiz, temel tüketim malzemelerinin satıldığı birçok mağaza olmasına rağmen, bu ürünleri satan mağazaların fiyatlarının karşılaştırılabileceği bir uygulamanın eksikliğinden doğmuştur. E-ticaret ürünlerinin fiyatları internet ortamında veya bazı mobil uygulamalarda karşılaştırılabilir olsa da bu çözüm fiziksel perakende mağazalarının çoğunu kapsamamaktadır. Aynı zamanda kullanıcılar, tek tek ürün fiyatlarını araştırmak yerine farklı ihtiyaçlar için farklı mağazalara gitmek yerine toplamda en uygun sepet tutarını hesaplayan bir uygulama bulunmamaktadır. Bu şekilde kullanıcılar, harcadıkları yol ve ulaşım maliyetlerini ve zamanlarını minimum düzeye indirebilirler.

"Perakende mağazaları indirim yaparken araştırma yapan kullanıcıyı hedefliyor." * Uygulama sayesinde kullanıcılar, indirimli ürünleri tek tek araştırmak yerine ihtiyaçları doğrultusunda bir sepet oluşturarak hızlı bir şekilde en uygun maliyetle alışveriş yapabilirler. Ayrıca, yaptıkları alışverişleri analiz ederek bir sonraki siparişlerinde istediklerine daha yakın bir alışveriş deneyimi yaşarlar.

2. GENEL KISIMLAR

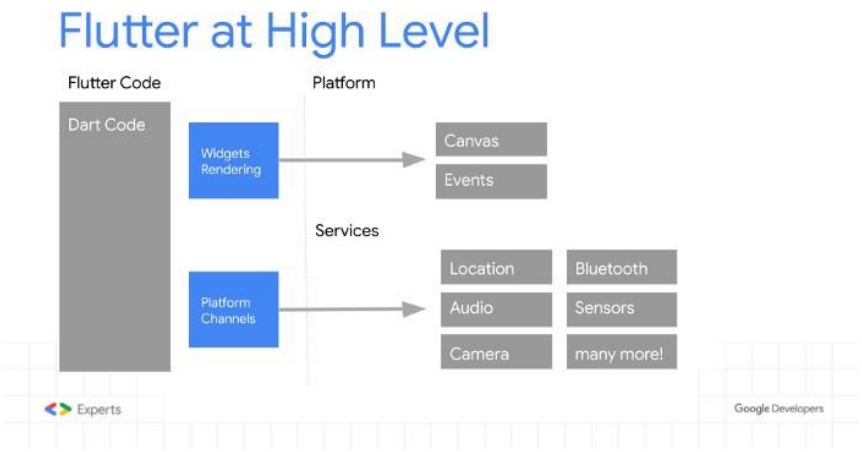
2.1. MOBİL UYGULAMADA KULLANILACAK TEKNOLOJİLER

2.1.1. Flutter

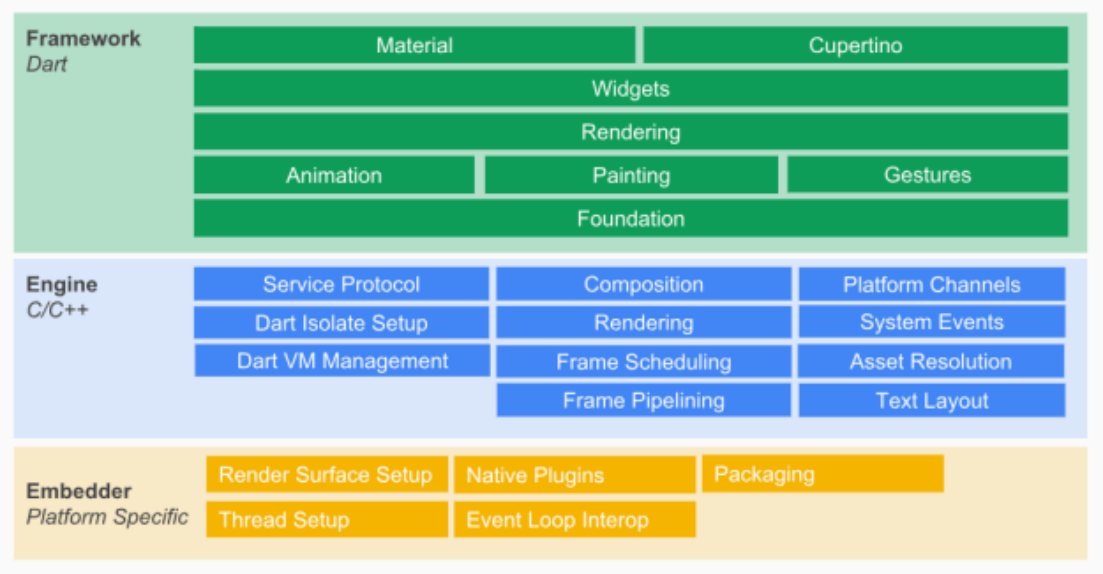
Flutter, Google tarafından geliştirilen bir açık kaynaklı mobil UI Framework'üdür. Mayıs 2017'de piyasaya sürülmüş olan Flutter, mobil, web ve masaüstü uygulamalarının geliştirilmesi için kullanılabilir. Bu Framework, tek bir kod tabanı üzerinde hem iOS hem de Android platformları için uygulama geliştirmeyi sağlar[5].

Flutter, Dart programlama dilini kullanır. Dart, nesne yönelimli bir dildir ve hot-reload gibi hızlı geliştirme özellikleri sunar. Flutter, Dart kodlarını derleyerek, donanım seviyesindeki native koda dönüştürür ve kendi grafik motoru olan Skia ile platforma özgü arayüzler render eder. Bu, Flutter'ı diğer Hybrid Framework'lerden ayıran önemli bir özelliktir.

Flutter mimari olarak Şekil 1 ve Şekil 2 de belirtildiği gibi çalışır.



Şekil 1. Flutter mimarisi



Şekil 2. Flutter pipeline çalışması

Flutter, declarative (bildirimsel) programlama paradigmalarını benimser. React Native'de olduğu gibi, geliştiriciye bildirimsel yapılara (StatefulWidget, StatelessWidget, InheritedWidget, BuildContext vb.) doğrudan erişim sağlar. Flutter, programlama dili ile arayüz geliştirmesini (ui-as-code) benzer şekilde JSX syntax'ine benzeyen yapılarla gerçekleştirir.

Flutter'da, UI geliştirmesi temel olarak "Widget" ismi verilen yapılarla yapılır. Stateless Widget ve Stateful Widget olmak üzere iki tür Widget sınıfı vardır. Stateless Widget'lar, sabit ve değişmeyen ekran tasarımlarında veya parçacıklarında kullanılırken, Stateful Widget'lar kullanıcının etkileşimde bulunduğu ve olaylara bağlı olarak değişen yapılarda kullanılır. Flutter, widgetlar aracılığıyla esnek ve etkileşimli kullanıcı arayüzleri oluşturmayı sağlar.
Flutter'ın avantajlarından biri, hızlı geliştirme süreci ve hot-reload özelliği sayesinde anında değişiklikleri görebilme imkanı sunmasıdır. Ayrıca, zengin widget koleksiyonu, performans odaklı yapısı ve platforma özgü arayüzlerin birleşimi, kullanıcı dostu ve çekici uygulamaların geliştirilmesini sağlar.
Flutter, geliştiricilere geniş bir topluluk ve destek sunar. Sürekli olarak güncellenen ve geliştirilen bir Framework olması, yeni özelliklerin eklenmesi ve sorunların giderilmesi için aktif bir ekosisteme sahip olduğu anlamına gelir. Bu da Flutter'ın popülerliğini artıran bir faktördür.

2.2. SİSTEM VE DATABASE ALANINDA KULLANILACAK TEKNOLOJİLER

2.2.1 PostgreSQL

PostgreSQL, açık kaynaklı bir veritabanı yönetim sistemidir ve hem ilişkisel (SQL) hem de ilişkisel olmayan (JSON) veri sorgulamalarını destekler. Geniş bir özellik setine sahip olan

PostgreSQL, kurumsal sınıf ve gelişmiş bir nesne-ilişkisel veritabanıdır. Aşağıda PostgreSQL'in bazı önemli özelliklerini bulabilirsiniz:

Geniş İşletim Sistemi Uyumluluğu: PostgreSQL, Windows, macOS, Linux, UNIX vb. gibi birçok işletim sistemiyle uyumlu olarak çalışabilir. Bu, farklı platformlarda kullanım esnekliği sağlar ve çeşitli uygulama gereksinimlerini karşılar.

Dil Desteği: PostgreSQL, C#, C/C+, Java, Python, JavaScript (Node.js), Ruby gibi popüler programlama dilleriyle entegrasyonu destekler. Bu, PostgreSQL veritabanına erişirken ve veritabanı işlemleri yaparken tercih ettiğiniz dili kullanmanıza olanak sağlar.

Geniş Veri Tipleri Uyumluluğu: PostgreSQL, çeşitli veri tiplerini destekler. Bunlar arasında temel veri tipleri (String, Numeric, Integer, Boolean), yapılandırılmış veri tipleri (Dizi, Date/Time, UUID, Range), geometri veri tipleri (Poligon, Line, Point, Circle) ve belge tabanlı veri tipleri (XML, JSON/JSONB) bulunur. Bu çeşitlilik, verilerin farklı formatlarda ve yapıda depolanabilmesini ve işlenebilmesini sağlar.

Veri Bütünlüğü Desteği: PostgreSQL, veri bütünlüğünü sağlamak için çeşitli mekanizmalar sunar. Yabancı anahtarlar, birincil anahtarlar, dışlama kısıtları, tavsiye kilitleri, açık kilitler, NOT NULL ve UNIQUE gibi özellikler veri bütünlüğünü korumaya yardımcı olur. Bu, veritabanında tutarlı ve doğru veri sağlar.

Güvenlik: PostgreSQL, sağlam bir erişim kontrol sistemi sunar ve kimlik doğrulama yöntemleriyle güvenliği artırır. LDAP, SSPI, GSSAPI, sertifikalar gibi farklı kimlik doğrulama yöntemlerini destekler. Ayrıca, sütun ve satır düzeyinde güvenlik seçenekleri sunarak verilerin güvenliğini sağlar.

PostgreSQL, birçok ek özellik sunar. Bunlar arasında belirli bir noktadan sonra kurtarma, eşzamanlı çoğaltma, çok sürümlü eşzamanlılık kontrolü (Multi-Version Concurrency Control - MVCC), iç içe işlemler, tablo bölümleme gibi özellikler bulunur. Bu özellikler, performansı artırır, veritabanı yönetimini kolaylaştırır ve daha etkili veri işleme imkanı sunar.

PostgreSQL, geniş bir topluluğa ve aktif bir geliştirici ekosistemine sahiptir. Sürekli olarak güncellenir ve yeni özellikler eklenir. Bu da kullanıcılar için destek ve geliştirme açısından önemli bir avantaj sağlar. PostgreSQL'in sağladığı özellikler, güvenilirlik, performans ve esneklik gibi faktörlerle birlikte, veritabanı ihtiyaçlarınızı karşılamak için güçlü bir seçenek olmasını sağlar.

2.2.2 .Net Core

.NET Core, Microsoft tarafından geliştirilen, açık kaynaklı, modüler ve platformlar arası bir yazılım geliştirme framework'üdür. .NET Core, çeşitli platformlarda çalışabilme özelliğiyle öne

çıkar. Windows, macOS ve Linux gibi farklı işletim sistemlerinde uygulama geliştirmenizi sağlar.

.NET Core, C# gibi programlama dilleriyle kullanılabilir ve geniş bir sınıf kitaplığına sahiptir. Hem masaüstü uygulamaları hem de web uygulamaları için kullanılabilir.

.NET Core'un bazı özellikleri şunlardır:

Çoklu Platform Desteği: .NET Core, Windows, macOS ve Linux gibi farklı işletim sistemlerinde çalışabilir. Bu, uygulamanızı hedefleyeceğiniz platforma bağımlı olmadan kod yazabilmenizi sağlar.

Modüler ve Hafif: .NET Core, modüler bir yapıya sahiptir ve ihtiyaç duyduğunuz bileşenleri projenize ekleyebilirsiniz. Bu da uygulamanızın gereksinimlerine uygun bir şekilde ölçeklendirilebilir ve optimize edilebilir olmasını sağlar.

Yüksek Performans: .NET Core, hızlı ve verimli bir şekilde çalışabilmesi için optimize edilmiştir. Just-in-Time (JIT) derlemesi ve Ahead-of-Time (AOT) derlemesi gibi derleme teknikleri kullanarak uygulamalarınızın performansını artırır.

Web Geliştirme: ASP.NET Core, .NET Core'un bir parçası olarak sunulan bir web uygulama geliştirme Framework'üdür. Modern ve hızlı web uygulamaları oluşturmanızı sağlar. Ayrıca, WebSocket, gRPC gibi iletişim protokollerini destekler.

Verimli Kaynak Kullanımı: .NET Core, az bellek tüketimi ve hızlı başlatma süreleriyle tanınır. Bu, uygulamalarınızın daha verimli çalışmasını ve kullanıcı deneyiminin iyileştirilmesini sağlar.

.NET Core, güçlü bir topluluğa sahiptir ve sürekli olarak geliştirilmektedir. Microsoft tarafından desteklenir ve yeni özelliklerin eklenmesi ve hataların düzeltilmesi için sürekli olarak güncellenir. Bu, geliştiricilere güvenilir bir platform sunar ve projelerin hızlı bir şekilde ilerlemesini sağlar.

2.2.3 Google Haritalar

Google Haritalar SDK'sı, mobil uygulama tasarımlarında kullanılan bir araçtır. Bu SDK, kullanıcılara haritalar ve navigasyon özellikleri sunar. Kullanıcılar, haritaları yakınlaştırabilir, konumlarını belirleyebilir, yol tarifleri alabilir ve işletme bilgilerini görüntüleyebilir. Ayrıca, trafik bilgilerini kontrol edebilir ve konumlarını gerçek zamanlı olarak paylaşabilirler. Google Haritalar SDK'sı, kullanıcıların interaktif ve kullanışlı bir harita deneyimi yaşamalarını sağlar.

2.2.4 Hangfire

Hangfire, .NET tabanlı bir arka plan iş yöneticisidir. Bu araç, .NET uygulamalarında tekrarlanan, zamanlanmış veya gecikmeli görevleri yönetmek ve çalıştırmak için kullanılır. Hangfire, basit ve kolay kullanımlı bir API sunar ve birçok farklı senaryoda kullanılabilir.

Hangfire'in bazı özellikleri şunlardır:

Arka Plan İş Yönetimi: Hangfire, arka planda çalışan işleri yönetmek ve planlamak için kullanılır. Örneğin, belirli bir zaman aralığında bir işi tekrarlayan bir görevi ayarlayabilir veya gecikmeli olarak çalışması gereken bir işi planlayabilirsiniz.

Esnek Zamanlama: Hangfire, işleri farklı şekillerde zamanlayabilme esnekliği sağlar. Belirli bir tarih ve saat, belirli bir zaman aralığı veya belirli bir periyodik tekrarlama (örneğin, her saat başı) gibi çeşitli zamanlama seçeneklerini destekler.

Veritabanı Bağımsızlık: Hangfire, çeşitli veritabanı sağlayıcılarıyla çalışabilme esnekliği sunar. Hangfire, PostgreSQL, SQL Server, MySQL, Redis gibi yaygın kullanılan veritabanlarıyla entegre olabilir.

İş İlerleme İzleme: Hangfire, çalışan işlerin ilerlemesini izlemek ve takip etmek için kullanılabilir. Bu sayede işlerin tamamlanma durumunu ve hataları görebilir ve gerektiğinde müdahale edebilirsiniz.

Ölçeklenebilirlik: Hangfire, büyük miktarda işi işleyebilme yeteneğiyle ölçeklenebilir bir yapı sunar. Birden fazla işçi süreci kullanarak yüksek performanslı işlemleri destekler ve büyük iş yüklerini etkili bir şekilde yönetebilir.

Hangfire, .NET projelerinde arka plan işlerini yönetmek ve otomatikleştirmek için kullanılan güçlü bir araçtır. Basit API'ı ve esnek zamanlama seçenekleri ile geliştiricilere işlerini düzenleme ve izleme kolaylığı sağlar.

2.3. YAPAY ZEKA VE MAKİNE ÖĞRENMESİNDE KULLANILACAK TEKNOLOJİ

2.3.1. Karar Ağaçları

Karar ağaçları, veri madenciliği ve makine öğrenimi alanlarında kullanılan bir öğrenme algoritmasıdır. Bu algoritma, veri setlerinden öğrenme yaparak karar verme sürecini modelleyen bir ağaç yapısı oluşturur. Karar ağacı, bir dizi karar noktası ve karar kurallarından oluşur.

Karar ağacı, veri setindeki her bir özelliğe (değişken) göre karar noktaları oluşturur. Her karar noktası, bir karar kuralını temsil eder ve bu kurala göre veri setindeki bir örneğin sınıflandırılması veya bir sonraki karar noktasına yönlendirilmesi sağlanır. Karar noktaları, veri setinin yapısına bağlı olarak belirlenir ve verilerin sınıflandırılması veya regresyon tahminlerinin yapılması için kullanılır.

Karar ağaçlarının oluşturulması, öğrenme sürecini içerir. Bu süreçte, veri setindeki örneklerin özellikleri analiz edilir ve en uygun karar noktaları ve karar kuralları belirlenir. Karar ağacı, veri setindeki örneklerin sınıflandırılması veya tahmin edilmesi için kullanıldığında, ağaç yapısı üzerinde ilerler ve karar kurallarını uygular. Böylece, yeni veriler için doğru kararlar verilebilir.

Karar ağaçları, veri setlerinin içerdiği desenleri ve ilişkileri anlamak ve kullanmak için kullanılır. Bu yöntem, veri setinin yapısını ve özelliklerini anlamak için görselleştirme ve analiz araçları kullanır. Karar ağaçları, sınıflandırma, regresyon, kümeleme ve özellik seçimi gibi birçok veri analizi görevinde kullanılabilir.

Karar ağaçları, basit bir anlaşılabilirlik ve yorumlanabilirlik sunar, çünkü oluşturulan ağaç yapısı insan tarafından okunabilir ve anlaşılabilir bir yapıya sahiptir. Bu özellikleri, karar ağaçlarını makine öğrenimi projelerinde yaygın bir şekilde kullanılan ve tercih edilen bir yöntem haline getirir.

2.4. İNCELENEN VE İLHAM ALINAN BAŞLIKLAR

2.4.1. Geçmişten Günümüze Türkiye'de Enflasyon

Enflasyon, bir ekonomide genel fiyat seviyelerinin sürekli ve dikkate değer bir artışını ifade eder ve ekonomik istikrar, reel gelir, alım gücü ve genel refah üzerinde önemli etkileri vardır. Bu makalede, Türkiye'nin enflasyon tarihine genel bir bakış sunulacak ve özellikle enflasyonun hangi dönemlerde öne çıktığına dikkat çekilecektir.

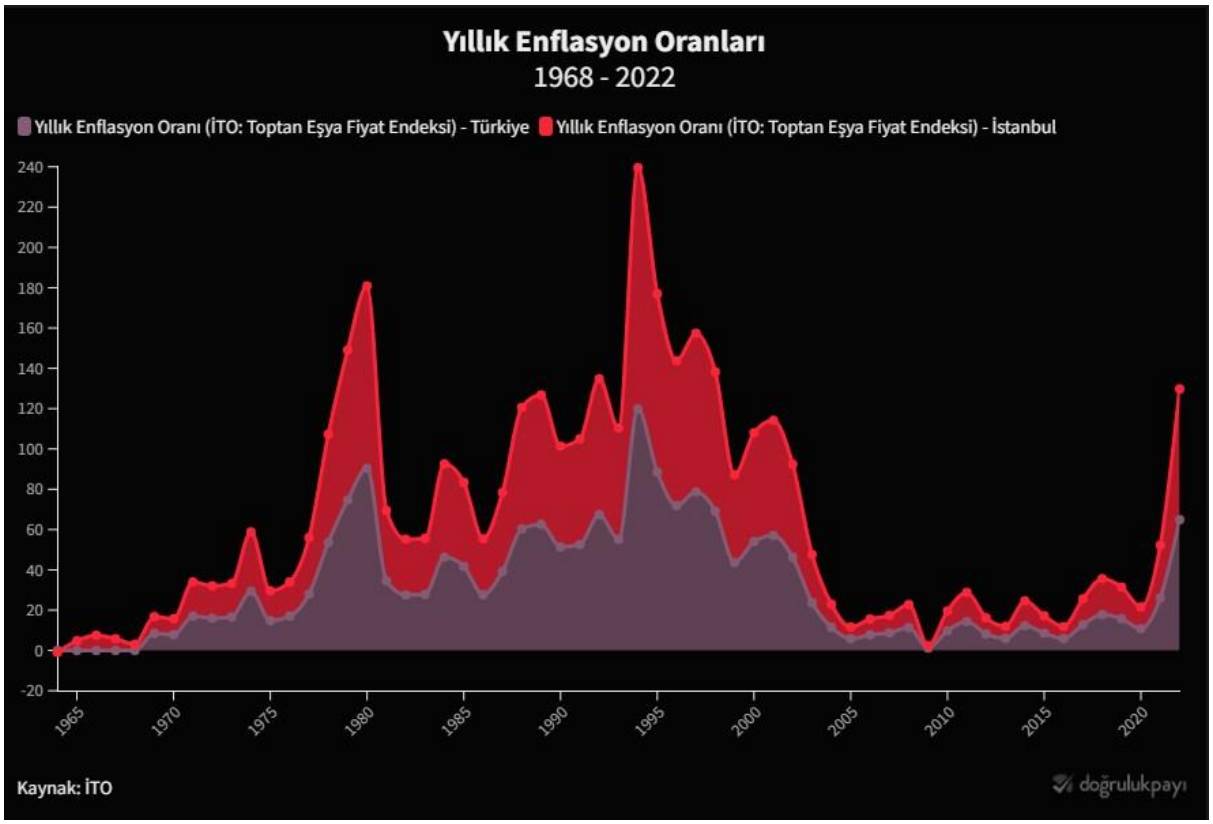
Türkiye'de Enflasyonun Tarihsel Evrimi

Türkiye'de enflasyonun tarihçesi, genellikle birkaç belirgin döneme ayrılabilir: düşük enflasyon dönemleri (1923-1970), yüksek enflasyon dönemleri (1970-2000), ve daha istikrarlı bir dönem (2000'ler sonrası).

1923'ten 1970'lerin başına kadar olan dönemde, Türkiye genellikle düşük enflasyon oranlarına sahipti. Bu dönemde, Türkiye ekonomisi tarıma dayalıydı ve ekonominin sanayileşmeye geçiş süreci henüz tamamlanmamıştı.[6]

1970'lerin başından itibaren, Türkiye'de enflasyon önemli ölçüde artmaya başladı. Bu dönemde, Türkiye ekonomisi önemli yapısal değişiklikler geçirdi, petrol krizleri, politik istikrarsızlıklar ve borçlanma oranlarının artması gibi faktörler enflasyonu tetikledi.[8]

2000'ler sonrası dönemde, Türkiye'nin enflasyonla mücadelesi daha başarılı hale geldi. Türkiye Merkez Bankası'nın bağımsızlık kazanması ve enflasyon hedeflemesi gibi politikaların uygulanması bu süreçte önemli rol oynadı.[9] Ancak, bu dönemde dahi enflasyonun kontrol altında tutulması zor oldu ve zaman zaman çift haneli rakamlara ulaştı.



Şekil 3. 1965-2021 tarihleri arası Türkiye'de Enflasyon

Türkiye'deki enflasyonun tarihsel incelemesi, yüksek enflasyonun Türkiye ekonomisi için neredeyse bir norm haline geldiğini göstermektedir. 1970'lerden bu yana, Türkiye defalarca yüksek enflasyon dönemleriyle karşı karşıya kalmış ve bu durum, ekonominin yapısal unsurları, dış şoklar ve politik istikrarsızlıklar dahil olmak üzere bir dizi faktörden kaynaklanmıştır.

2000'lerden sonra Merkez Bankası'nın daha bağımsız bir yapıya kavuşması ve enflasyon hedeflemesi politikalarının uygulanması gibi stratejiler, enflasyonun kontrolüne yardımcı olmuştur. Ancak, hala dikkat çekici enflasyon dalgalanmaları görülmektedir ve bu durum, Türkiye'nin enflasyonla sürekli bir mücadele içinde olduğunun kanıtıdır. Bu nedenle, Türkiye'de yüksek enflasyonun bir alışkanlık haline gelmiş olduğunu söylemek ne yazık ki doğru olacaktır. Ancak bu, ekonomik politikaların ve uygulamaların etkili bir şekilde yönetilmesiyle değiştirilebilecek bir durumdur.[9]

Şekil 4. 1996-2022 tarihleri arası Türkiye'de Enflasyon

2.4.2. Türkiye'de Mobil Alışveriş ve Web Alışveriş

Günümüzde teknolojiye olan ilginin artması ve dijitalleşmenin hız kazanması, Türkiye'de alışveriş alışkanlıklarında belirgin bir değişikliğe yol açmıştır. Bu kısımda, Türkiye'de mobil alışveriş ve web alışverişin yüzdeleri hakkında genel bir bakış sunacağız.

Mobil Alışverişin Türkiye'deki Yükselişi

Son dönemde dünya genelinde yeni tip koronavirüs (Kovid-19) salgını nedeniyle yaşam tarzlarımız ve alışkanlıklarımız ciddi bir dönüşüm geçirdi. Bu değişikliklerden biri de alışveriş alışkanlıkları oldu. Fiziksel mağazalar yerine çevrim içi alışverişin tercih edilmesiyle birlikte, özellikle mobil alışveriş uygulamaları önemli ölçüde popülerlik kazandı.

Anadolu Ajansı'nın Rekabet Kurumu verilerine dayanarak yaptığı derlemeye göre, 2020 yılında internetten alışveriş yapan tüketicilerin %84'ü mobil cihazları tercih etti. Mobil cihazlar üzerinden gerçekleştirilen alışverişlerde ise tüketicilerin %79,6'sı mobil uygulamayı kullandı. 2019 yılında mobil uygulamalar üzerinden yapılan alışverişlerin oranı %56,4 iken, bu oran 2020 yılında %82,3'e çıktı (Anadolu Ajansı, 2021).

Bu veriler, salgın sürecinde tüketicilerin çevrim içi alışverişe yöneliminde büyük bir artış olduğunu ve bu eğilimin özellikle mobil uygulamalar lehine olduğunu gösteriyor. Tüketicilerin çevrim içi alışveriş yaparken en hızlı işlem yapabildikleri ve kullanım kolaylığı sunan araçlara yönelmeleri, mobil uygulamaların popülaritesinin artmasında önemli bir faktör oldu.

Bu eğilim, genellikle genç tüketiciler arasında daha belirgin oldu. 18-24 yaş arası tüketiciler mobil cihaz ve uygulama kullanımında en yüksek seviyeye ulaşırken, 45 ve üzeri yaş grubunda bu kullanım daha düşük seviyelere indi.

Dolayısıyla, bu veriler mobil alışverişin Türkiye'de hızla yükseldiğini ve bu trendin önümüzdeki yıllarda da devam etmesinin beklendiğini göstermektedir. Biz de bu araştırmaları göz önünde bulundurarak işe mobil platformda uygulama geliştirerek başlamak istedik.

2.4.3. Market Verilerinin Ticaret Bakanlığının Ortak Veri Havuzunda Toplanması Kararı

Son olarak, yakın zamanda Ticaret Bakanlığı tarafından yapılan bir yönetmelik değişikliği, çevrim içi perakende sektörünün gelecekteki gelişimine önemli bir etkide bulunacak gibi görünüyor. Bakanlık, "Veri Paylaşımı" başlıklı yeni bir madde ekleyerek, 200'den fazla şubesi olan perakende zincir mağazaların satışa sundukları ürünler ve şubelerine ilişkin verileri belirlenen bir sistem üzerinden paylaşmalarını zorunlu kıldı [2].

Bu düzenleme, sektördeki rekabet koşullarının iyileştirilmesi, kamuoyunun bilgilendirilmesi ve tüketicilerin fiyat karşılaştırması yapabilmesi amacıyla getirildi. Ayrıca, bu tür veri paylaşımının, çevrim içi perakende uygulamalarının kullanımını daha da yaygınlaştırabileceğini düşünmekteyiz. Çünkü bu sayede tüketicilerin bilgiye erişimi artacak, daha bilinçli alışveriş kararları verme imkânı sağlanacak ve nihayetinde çevrim içi alışverişe olan güvenleri daha da artacaktır.

Şekil 5. Ticaret Bakanlığı tarafından yayınlanan afiş

Bu değişiklik, bizim çalışmalarımızda değerlendirdiğimiz çevrim içi alışveriş uygulamalarının önemini ve etkisini daha da güçlendirecektir. Dolayısıyla, bu yeni düzenlemenin, bizim

tarzımızdaki uygulamaların önünü açacağını ve çevrim içi alışverişin Türkiye'deki gelişimini daha da hızlandıracağını düşünmekteyiz.[10]

3. KULLANILAN ARAÇ VE YÖNTEM

3.1. Uygulama Geliştirme Süreci

3.1.1. Kullanılan Programlama Dilleri ve Platformlar

Flutter: Flutter, çoklu platform desteği sağlayan ve hızlı kullanıcı arayüzü geliştirme imkanı sunan bir Framework olarak tercih edildi. Aynı kod tabanıyla hem iOS hem de Android platformlarında çalışabilme avantajı sağladı. Bu sayede uygulamanın birden fazla platformda hızlı ve tutarlı bir şekilde yayınlanması mümkün oldu.

Google Haritalar SDK: Google Haritalar, kullanıcıların konum tabanlı hizmetlere erişimini sağlayarak uygulamanın kullanıcı deneyimini zenginleştirdi. Kullanıcılar haritaları görüntüleyebilir, yerleri işaretleyebilir, yol tarifleri alabilir ve daha fazlasını yapabilir. Bu, kullanıcıların uygulama içinde daha etkileşimli ve kullanışlı bir deneyim yaşamasını sağladı.

Firebase Authentication: Firebase Authentication, güvenli kimlik doğrulama işlemlerini sağlayarak kullanıcı girişi ve hesap yönetimini kolaylaştırdı. Kullanıcılar uygulamaya kaydolabilir, giriş yapabilir ve sosyal medya hesaplarıyla entegre bir şekilde oturum açabilir. Bu, kullanıcı verilerinin güvenliğini sağlamak ve gelişmiş kullanıcı yönetimi sağlamak için tercih edildi.

.NET 6.0 ve Entity Framework ORM: .NET 6.0 ve Entity Framework ORM, veri tabanı işlemlerini kolaylaştırarak veri yönetimini daha etkili bir şekilde gerçekleştirmeyi sağladı. .NET 6.0'nın sunduğu gelişmiş özellikler ve performans artışı, uygulamanın daha verimli çalışmasına katkı sağladı. Entity Framework ORM ise veritabanı işlemlerini nesne yönelimli bir şekilde gerçekleştirerek kod okunabilirliğini ve bakım kolaylığını artırdı.

PostgreSQL: PostgreSQL, açık kaynaklı ve performanslı bir veritabanı yönetim sistemi olarak tercih edildi. İlişkisel ve ilişkisel olmayan veri tiplerini desteklemesi, güçlü veri bütünlüğü sağlaması ve geniş işletim sistemi uyumluluğu gibi özellikleriyle projeye uygun bir veritabanı çözümü sunması nedeniyle tercih edildi.

Hangfire: Hangfire, zamanlanmış görevleri yönetmek ve arka planda çalışan işleri kontrol etmek için kullanılan bir araçtır. Proje içinde belirlenen ortalama fiyatlara göre saat başı fiyatların değiştirilmesi gibi zamanlı görevlerin sorunsuz bir şekilde yönetilmesini sağladı. Bu, otomatik ve güvenilir bir şekilde fiyat değişikliklerinin yapılmasını sağladı.

Bu şekilde, kullanılan teknolojiler projenin farklı yönlerini destekleyerek daha güçlü bir uygulama geliştirmeyi sağladı. Her teknoloji, belirli avantajlar sunarak uygulamanın kullanıcı deneyimini zenginleştirmeye ve geliştirme sürecini kolaylaştırmaya yardımcı oldu.

4. SİSTEMİN GERÇEKLENMESİ YA DA BULGULAR

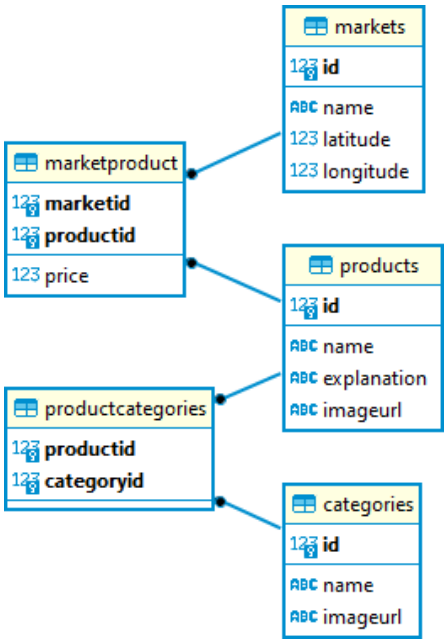
4.1 BACKEND PROJESİNİN VE VERİTABANININ KURGULANMASI VE GERÇEKLEŞTİRİLMESİ

Uygulama veritabanı olarak karar verilen PostgreSQL de ilgili tablolar, sonrasında ilişkileri kullanarak Nesne Tabanlı Mapping (ORM) uygulayabilmek adına Foreign Key ve Primary Key kısıtları kullanarak oluşturuldu. Şekil 6'da görüldüğü üzere temelde kullanılacak 3 ana tablo bulunuyor. Bu tablolar Products, Markets, ve Categories tablolarıdır. Bu tabloları ilişkilendirmek adına fiyatları da tutacağımız MarketProducts tablosu ve ProductCategories tabloları da ilişkili olarak oluşturuldu. Database uygulamasının tamamlanmasının ardından EntityFramework kütüphanesi kullanılarak bir tersine mühendislik işlemi olan DB-First yaklaşımıyla ORM projeye dahil edildi. Tablo sınıfları oluşturulmuş oldu.
Restful servislerin oluşturulması için gerekli Request, Response sınıfları oluşturuldu. Bu sınıflar ilgili isteklerin karşılanmasında ve ardından Mapping işleminin yapılmasında kullanıldı. Bu noktada önemli olan Controller sınıfları;

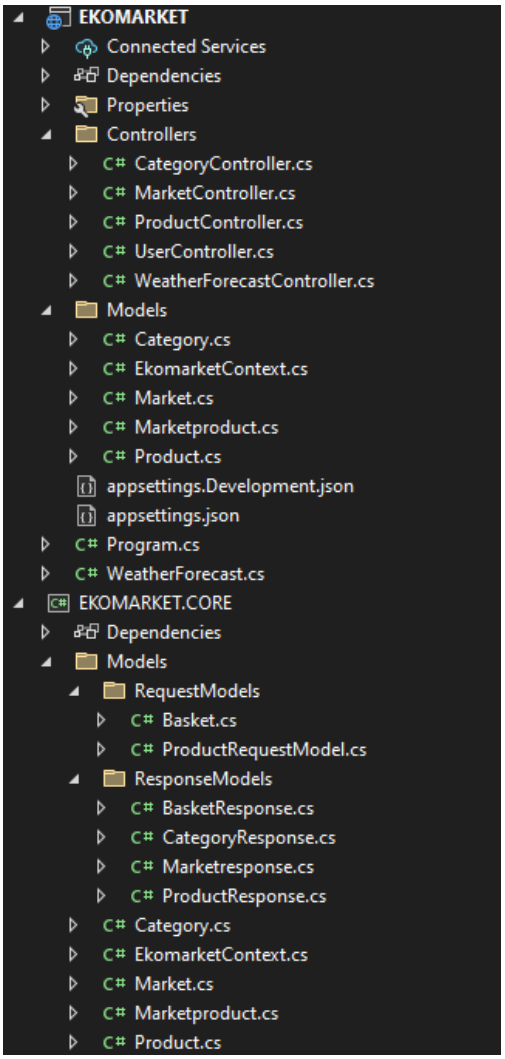
- Bütün ürünleri getiren GetAllProducts,
- Ürünleri belirtilen kategoriler için Getiren GetProductsByCategoryId,
- Ürünlerin marketlerdeki fiyatlarını kümülatif olarak hesaplayıp ortalama sepet tutarını ve en düşük fiyatı hesaplayan CalculateBasketPrice

Olarak sıralanabilir.
Uygulamada mimari yaklaşım olarak monolitik bir yaklaşım izlendi. Bunun sebebi geliştirme sürecini hızlandırıp maliyeti düşürmek olarak söylenebilir. Uygulamanın klasörleme yapısı Şekil 7 de belirtilmiştir.

İlgili Restful API'ların oluşturulmasından sonra marketlerdeki ürün fiyatlarını düzenli olarak güncellemek adına periyodik olarak çalışacak Hangfire uygulamasına başlandı. Burada belirlenen işlerin zamanlanması ve takibi için bir veritabanına tablo oluşturulması kararlaştırıldı. İlgili tabloların konfigürasyonları sağlandıktan sonra Hangfire projesinde kullanılacak market fiyatlarını güncelleyen uygulama yazıldı. İlk etapta gerçek market verisi sağlanamadığı için 3 saatte bir market verilerinin güncellenmesinde karar kılındı.



Şekil 6. Veri tabanı şeması



Şekil 7. API projesi yapısı

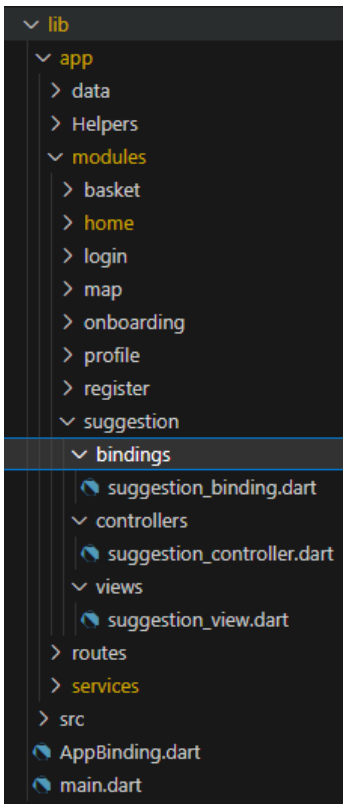
4.2 MOBİL UYGULAMANIN GERÇEKLEŞTİRİLMESİ

4.2.1 MOBİL UYGULAMANIN MİMARİ KURGUSUNUN YAPILMASI

Uygulamanın mimari kurgusunda MVC mimari modelinden faydalanıldı. Getx State Management yöntemi kullanıldığı için GetxController ve GetxView sınıflarından faydalanılarak modüler bir uygulama mimarisi oluşturuldu Uygulama mimarisinin yapısına Şekil 8'den bakabilirsiniz.

Bu yaklaşımda Her View bir Controller sınıfına bağlı. Uygulamada yapılan mantık işlemleri bu Controller'lar vasıtasıyla çalıştırılırken, sade yazılmış View sınıflarında arayüz ve tasarımsal

işlemler mevcut. Aynı zamanda her modülün bir Binding sınıfı bulunmakta. Bu sınıf bir View ve Controller'ı birbirine bağlayarak bağımlılık dahil edilmesi işlemini sağlıyor.



Şekil 8. Mobil uygulama klasörlemesi

4.2.2 MOBİL UYGULAMANIN SAYFALARI VE FONKSİYONLARI

4.2.2.1 LOGİN SAYFASI VE FİREBASE KULLANIMI

Uygulamada bir Google servisi olan Firebase'den faydalanıldı. Firebase kurulumu mobil uygulamaya önce firebase_auth ve firebase_core kütüphanelerinin yüklenmesinin ardından Android Native dosyalara ilgili konfigurasyonların eklenmesiyle yapıldı.

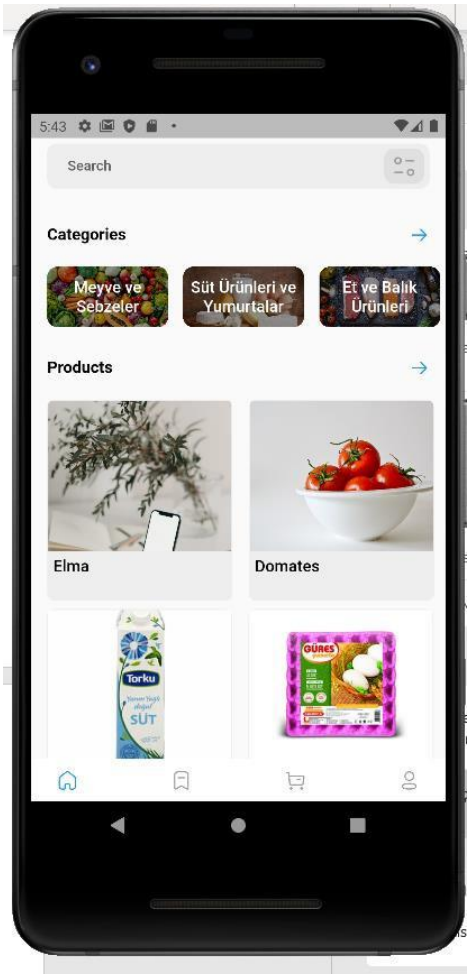
Flutter main fonksiyonu içerisinde eklenen firebaseInitialize fonksiyonu ile de uygulama içinde sürekli şekilde çalışacak duruma getirilmiş oldu. Bu noktada bir farklılık enjeksiyonu yöntemiyle uygulamaya eklenen AuthService sınıfının içerisine Firebase üzerinden kaydolma, giriş yapma ve oturumu kapatma işlemlerini sağlayan fonksiyonlar yazıldı. Resim 9' da bu sayfanın örnek bir görüntüsü mevcut.

Şekil 9. Giriş Sayfası

4.2.2.2 ANA SAYFA VE API PROJESİYLE İLETİŞİM

Ana sayfanın tasarımında dinamik olarak sayfanın alt tarafından değiştirilebilen bir BottomNavigationBar Widget'i mevcut aynı zamanda ana sayfanın mantık işlemlerinin yürütüldüğü HomeController sınıfı içerisinde ilgili Repository'lerin örneklerini de barındırıyor.

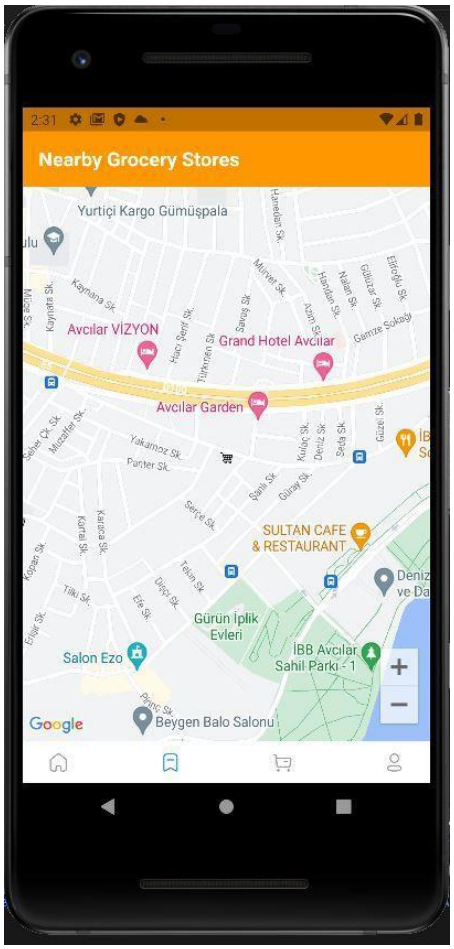
Uygulamada servislerle iletişim için Repositoy Pattern mimari yaklaşımı izlendi. Servis sağlayıcı Provider sınıfları, içerisinde ilgili Repository'lerin verilerini Map'leyip ilgili sayfalar içerisinde kullanılabilmesini sağlıyor. Bu Repository sınıfları ise Generic olarak farklı veri tipleri için Restful isteklerde bulunabilen ApiService sınıfından faydalanıyor. Bu ApiService sınıfı uygulamanın başında farklılıkların içeri alınması yöntemi sayesinde uygulamanın ayakta olduğu süre boyunca çalışmaya devam edip Restful istekleri gerçekleyebiliyor. Şekil 10'da belirtilen örnek üzerinde açılan sayfada istenilen ürünlerin aranarak filtrelenebileceği bir arama çubuğu, kategorilerin listesi ve sade bir tasarımla başlıca ürünler gösteriliyor. Bu noktada herhangi bir kategoriye tıklanarak sadece o kategorideki ürünlerin gösterilmesi de sağlanabiliyor. Aynı zamanda ürünlerin üzerlerine tıklayarak ürünlerle ilgili açıklamaları görebileceğimiz bir ekrana girip ürünleri sepetimize ekleyebiliyoruz.



Şekil 10.Ana Sayfa, Kategoriler

4.2.2.3 HARİTA SAYFASI GOOGLE MAPS SERVİSLERİ

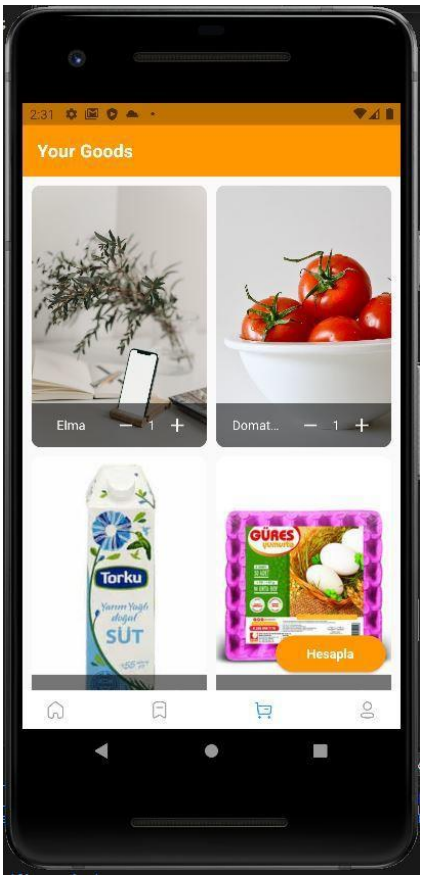
Harita sayfasında kullanıcılar çevrelerindeki yakın marketleri görebiliyorlar. Bu kısımda harita entegrasyonu için google_maps_flutter kütüphanesinden faydalanıldı. Google harita sağlayıcısının üzerine işaretçiler, veritabanına kayıtlı olan market koordinatlarından yola çıkılarak harita üzerine çiziliyor.
Google haritasının uygulamaya entegrasyonu için Google'ın kullanıcıya eşsiz olarak sağladı bir API anahtarı gerekiyor. Bu anahtar Google bulut servisleri üzerinden edinilip, uygulamanın çalışması gereken platform için ilgili dosyaya ekleniyor uygulamamız Android için konfigüre edildiğinden AndroidManifest.xml dosyasının içerisinde bu Key belirtilerek kütüphane kullanılmaya başlayabiliyor. Kullanıcı bu sayfada isterse yakınında gitmek istediği marketlerin üzerlerindeki işaretçiye tıklayarak uygulamanın kendisini Google Haritalar uygulamasına yönlendirmesini de sağlayabiliyor.



Şekil 11.Harita Sayfası

4.2.2.4 SEPET SAYFASI

Sepet sayfasında kullanıcı sepetine eklediği ürünleri ve adetlerini görebiliyor. Adetlerini değiştirip sepetten çıkartabiliyor. Almak istenilen ürünlere karar verilmesi durumunda en uygun sepet tutarının hesaplandığı öneri sayfasına da bu sayfa üzerinden yönlendirme yapılıyor. Kullanıcının sepetindeki ürünleri, BasketController isimli, uygulamaya farklılık enjektesi yöntemiyle dahil edilmiş bir Controller yönetiyor. Ürün silme, ürün ekleme veya adet güncellemeyle ilgili fonksiyonlar bu Controller sınıfı içerisinde bulunuyor. Hesapla tuşuna basılmasının ardından uygulama API'a erişiyor, en uygun tutarlı marketi, bu marketin konumunu, ortalama sepet tutarını alıp, öneri sayfasında kullanıcıya gösteriyor.



Şekil 12.Sepet Sayfası

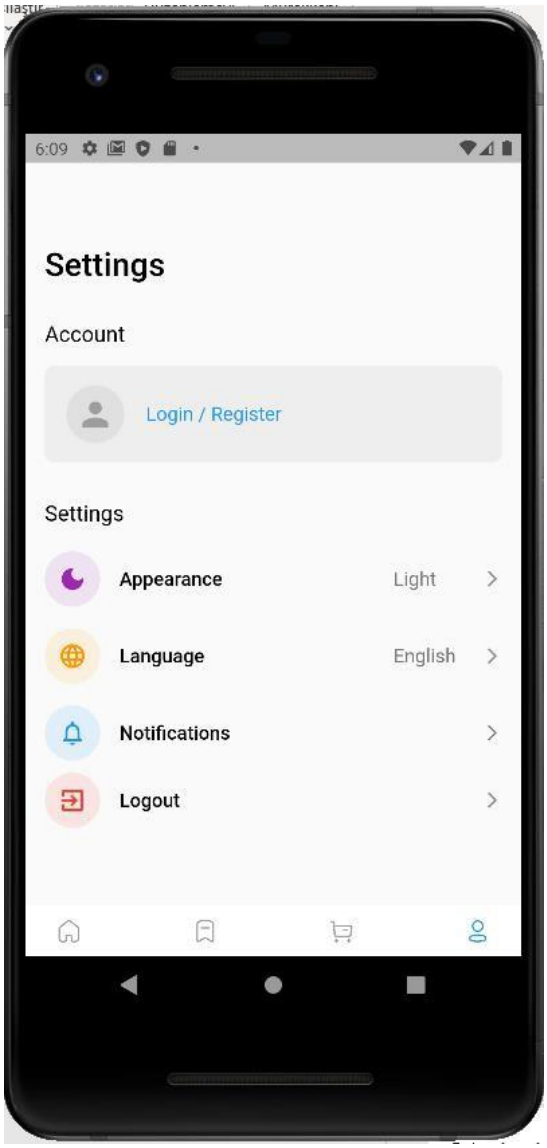
4.2.2.5 ÖNERİ SAYFASI

Bu sayfada kullanıcı maliyet açısından en ekonomik marketi görüntülüyor ve almak istediği ürünlerin belirtilen marketteki fiyatını öğreniyor. Aynı zamanda bu sepet için hesaplanan ortalama tutar konusunda da bilgi sahibi olup ne kadar kazançlı olduğunu analiz edebiliyor. Sayfanın sonunda iki buton bulunuyor bu butonlardan biri kullanıcıyı otomatik olarak Google haritalar uygulamasına yönlendiriyor. Diğer buton ise kullanıcıyı bu marketten belirtilen ürünleri aldıktan sonra alışverişten duyduğu memnuniyeti ölçekleyebileceği bir değerlendirme sayfasına yönlendiriyor. Bu sayda sayesinde kullanıcının gittiği mesafe ve tercihleri göz önünde bulundurularak basit bir karar ağacı algoritması sayesinde sonraki alışverişlerinde kullanıcıya daha uygun olabilecek market seçenekleri öneriliyor.

Şekil 13. Öneri sayfası

4.2.2.6 KULLANICI AYARLARI VE PROFİL SAYFASI

Bu sayfada ise uygulamanın teması ve dili değiştirilebiliyor. Gece kullanımlarında yorucu bir görüntü yaratmaması açısından bir karanlık mod seçeneği de bulunuyor. Kullanıcı aynı zamanda Firebase servislerinden yararlanarak oturumunu da bu sayfada kapatabiliyor.



Şekil 14. User sayfası

5. TARTIŞMA VE SONUÇ

Ülkemizde enflasyon sebebiyle ürün fiyatlarında zaman zaman ani değişimler olabilmektedir. Bundan kaynaklıdır ki özellikle e-ticaret sektöründe fiyatların takibinin yapılabileceği uygulamalar günümüzde çok sık kullanılmaktadır. Fakat bu fiyat dalgalanmaları temel tüketim malzemelerini aldığımız marketlerde de etkiliyken ne yazık ki perakende alışveriş için fiyat karşılaştırma yapılabilecek bir uygulama ortaya çıkmamıştır. Bu çok daha uyguna yapabileceğimiz alışverişleri enflasyon gerçeğinden etkilenerek yapmamıza sebep olmuştur.
Bu sorundan yola çıkarak geliştirdiğimiz uygulamamız günümüz fiyat karşılaştırma siteleri kadar çok fonksiyon barındırmamakla beraber, kullanımı da o kadar kullanıcı dostu olmuştur. Bu sayede çok basit düzeyde akıllı telefon kullanabilen bir kullanıcı bile aslında kolayca istediği sonuca ulaşabilecektir. Ayrıca fiyat karşılaştırma servisleri ciddi kullanıcı sayılarına sahip olduğundan reklam gelirleri önemli ölçüde yüksektir. Bu da aslında rekabetin ve kazanç ölçeğinin büyük olduğu bir sektörde, henüz yapılmamış olan bu uygulamayı basit düzeyde de olsa piyasaya sürerek ekonomiye katkı sağlayabileceğimiz gerçeğini ortaya koymuştur. Sonuç olarak gerçek veri bulmanın gerçekten zor olduğundan bahsedilebilir. Çeşitli perakende marketlerle korumaya çalıştığımız iletişimler sonucunda bu marketlerden hiçbiri geçmiş ürün fiyat bilgisini paylaşmayı kabul etmemiştir. Ne yazık ki küçük ölçekteki ürün yelpazesi ve ciddi miktarlarda olabilecek veri seti büyüklüğü düşünüldüğünde uygulamanın tasarımsal ve mimari açıdan gerçek bir stres testine hazır olmadığı söylenebilir. Aynı zamanda yine veri boyutlarının, perakende markaların sayılarının ne kadar fazla olduğu düşünüldüğünde donanım ve sunucu maliyetleri sebebiyle zorlayıcı bir girişim olabileceği de söylenebilir. Fakat bunların yanı sıra aktif olarak kullanılan bir örneği olmaması sebebiyle gelişime açık, aynı zamanda kullandığımız teknolojilerin de güncellenmekte olan ve büyük toplulukları bulunan öğeler olduğu göz önünde bulundurularak gelişime oldukça açık olduğunu da söyleyebiliriz.

KAYNAKLAR

1. "E-Ticaret Firmaları İçin Fiyat Karşılaştırma Siteleri Neden Önemlidir?" [online], [Ziyaret Tarihi: 14 Haziran 2023], URL: https://www.epazarsoft.com/fiyat-karsilastirma-sitelerinin-e-ticaretteki-yeri-nedir/
2. "7 Aralık 2022 Tarihli Resmi Gazete" [online], [Ziyaret Tarihi: 14 Haziran 2023], URL: https://www.resmigazete.gov.tr/eskiler/2022/12/20221207.pdf
3. "Akakçe hakkında" [online], [Ziyaret Tarihi: 14 Haziran 2023], URL: https://www.akakce.com/info/hakkimizda.asp
4. "Kasım 2022 Genelindeki Popüler Fiyat Karşılaştırma Siteleri" [online], [Ziyaret Tarihi: 14 Haziran 2023], URL: https://www.similarweb.com/tr/top-websites/category/e-commerce-and-shopping/pricecomparison/
5. "Flutter Dökümantasyonu" [online], [Ziyaret Tarihi: 14 Haziran 2023], URL: https://docs.flutter.dev/
6. PAMUK, Ş., 1987, The Ottoman Empire and European Capitalism, 1820-1913: Trade, Investment, and Production, Cambridge University Press.
7. BORATAV, K., 1982, 1929-1980 Türkiye İktisat Tarihi, Gerçek Yayınları.
8. AYSAN, A. F., HACIHASANOĞLU, Y. S. & YILDIRIM, D., 2006, The Historical Evolution of the Independence of Central Bank of the Republic of Turkey, International Journal of Applied Econometrics and Quantitative Studies, 3 (1).
9. ÇAKIR, Merve Özlem, 2021, İnternetten alışverişte tüketiciler 'mobil uygulamaları' tercih etti, [online], [Ziyaret Tarihi: 14 Haziran 2023], URL: https://www.aa.com.tr/tr/ekonomi/internetten-alisveristetuketiciler-mobil-uygulamalari-tercih-etti/2241089.
10. PALABIYIK, Deniz Çiçek, 2023, Perakende Ticarette Uygulanacak İlke ve Kurallar Hakkında Yönetmelik'te Değişiklik Yapılmasına Dair Yönetmelik, [online], [Ziyaret Tarihi: 14 Haziran 2023], URL: https://www.aa.com.tr/tr/ekonomi/zincir-marketler-urun-vesube-verilerini-ticaret-bakanliginin-belirledigi-sisteme-aktaracak/2757315.

EKLER

EK-A

Talimat Alma-Verme Çizelgesi

Takım Liderinin Adı Soyadı: Ömer Can Uyar				
	Puanlama			
	Üye 1	**Üye 2**	**Üye 3**	**Üye 4**
i. Talimatların Açık ve Anlaşılır Olması	5			
ii. Doğru Talimatların Verilmiş Olması / Talimatların Gerekliliği	5			
iii. Talimatların Uygulanması / Talimatların Gerçekleşmesi	5			
iv. Yeterli Sayıda Talimat Verilmesi	5			
v. Talimatların Doğru Bir Üslupla Verilmiş Olması	5			

Not: *Talimat Alma Verme Çizelgesi, gruptaki her bir öğrenci takım lideri olacak şekilde çoğaltılacaktır.*

Talimat Alma-Verme Çizelgesi

Takım Liderinin Adı Soyadı: Umut Can Öztürk				
	Puanlama			
	Üye 1	**Üye 2**	**Üye 3**	**Üye 4**
i. Talimatların Açık ve Anlaşılır Olması	5			
ii. Doğru Talimatların Verilmiş Olması / Talimatların Gerekliliği	5			
iii. Talimatların Uygulanması / Talimatların Gerçekleşmesi	5			
iv. Yeterli Sayıda Talimat Verilmesi	5			
v. Talimatların Doğru Bir Üslupla Verilmiş Olması	5			

Not: *Talimat Alma Verme Çizelgesi, gruptaki her bir öğrenci takım lideri olacak şekilde çoğaltılacaktır.*

ÖZGEÇMİŞ

Öğrenci: Umut Can Öztürk

1998 Fethiye doğumlu, İlköğrenimini Ordu Hamdullah Suphi Tanrıöver İlkokulu'nda tamamladı. Liseyi Ordu Fen Lisesi'nde bitirdi. 2017 yılından beri İstanbul Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği bölümüne devam etmektedir

Öğrenci: Ömer Can Uyar

1998 Bitlis-Tatvan doğumlu, İlköğrenimini Sivas Kızılırmak İlkokulu'nda tamamladı. Liseyi Bafra Fen Lisesi bitirdi. 2017 yılından beri İstanbul Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği bölümüne devam etmektedir. Öğrenim süresinde Şubat 2022 yılında Merit Risk Management and Consultant Corporation'da stajyer olarak çalışmaya başladı. Evatro Bilişim A.Ş.'de 2 ay staj yaptı ve Globit Global Information Technology GmbH'de tam zamanlı çalışmaya başladı. Günümüzde de bu şirkette çalışmaya devam etmektedir.