

# **KİLO TAKİBİ ANDROİD TABANLI UYGULAMA**

**2020**  
**BİLGİSAYAR MÜHENDİSLİĞİ**  
**BİTİRME PROJESİ TEZİ**

**Süleyman ALTUNAKAR**  
**Fatoş DABAKOĞLU**

# **KİLO TAKİBİ ANDROİD TABANLI UYGULAMA**

**Süleyman ALTUNAKAR**

**Fatoş DABAKOĞLU**

**Karabük Üniversitesi**

**Mühendislik Fakültesi**

**Bilgisayar Mühendisliği Bölümünde**

**Bitirme Projesi Tezi**

**Olarak Hazırlanmıştır.**

**KARABÜK**

**June 2020**

Süleyman ALTUNAKAR ve Fatoş DABAKOĞLU tarafından hazırlanan “Kilo Takibi Android Tabanlı Uygulama” başlıklı bu projenin Bitirme Projesi Tezi olarak uygun olduğunu onaylarım.

Dr. Öğr. Üyesi Zafer ALBAYRAK

Bitirme Projesi Danışmanı, Bilgisayar Mühendisliği

...../...../2020

Bilgisayar Mühendisliği bölümü, bu tez ile Bitirme Projesi Tezini onaylamıştır

Dr. Öğr. Hakan KUTUCU

.....

Bölüm Başkanı

*“Bu projedeki tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu; ayrıca bu kuralların ve ilkelerin gerektirdiği şekilde, bu çalışmadan kaynaklanmayan bütün atıfları yaptığımı beyan ederim.”*

Süleyman ALTUNAKAR

Fatoş DABAKOĞLU

## **ÖZET**

**Bitime Projesi Tezi**

### **KİLO TAKİBİ ANDROİD TABANLI UYGULAMA**

**Süleyman ALTUNAKAR**

**Fatoş DABAKOĞLU**

**Karabük Üniversitesi**

**Bilgisayar Mühendisliği**

**Bilgisayar Mühendisliği Bölümü**

**Tez Danışmanı:**

**Dr. Öğr. Üyesi Zafer ALBAYRAK**

**June 2020, 21 sayfa**

Günümüzde bilgisayar ve iletişim teknolojisinin yaygın olarak kullanılmasıyla android tabanlı uygulamalar üzerinde çok büyük bir etkisi vardır. Bu bilgilerin doğrultusunda kişisel olarak kullanıcının aldığı besinlerin, tıbbi verilerinin ve kendi diyet listesinin takibi için düzenlenmiş bir mobil platform. Uygulama, içerisinde kategori sistemiyle ayrılmış su miktarı, vücut analizi, diyet listesi ve besin değerlerini gösteren özel veriler bulunmaktadır. Kullanıcı ana menü içerisinde bütün istediği bilgilere erişim ve takibini kolaylıkla yapabilmektedir. Kullanıcılar uygulama içerisinde veritabanında bulunan besinlerin değerlerini öğrenebilmektedir. Proje içerisinde bir vücut kitle endeksi hesaplama aracı ile sistemde kayıt altına alınabiliyor. Girilen bilgilere göre öneriler ve yapılması gereken eylemler hesaplanıyor. Kullanıcıya her gün düzenli bir şekilde tavsiyede bulunuyor ve bunlar bildirim yoluyla sunuluyor.

**Anahtar Sözcükler :** Android, Mobil Uygulama, Dart, Flutter, Json, Data

## **ABSTRACT**

### **Senior Project Thesis**

## **WEIGHT TRACKING ANDROID BASED APPLICATION**

**Süleyman ALTUNAKAR**

**Fatoş DABAKOĞLU**

**Karabük University**

**Faculty of Engineering**

**Department of Computer Engineering**

**Project Supervisor:**

**Dr. Prof. Dr. Zafer ALBAYRAK**

**June 2020, 21 pages**

Nowadays, applications based on computer and communication technology are common and android-based applications have a huge impact. In line with this information, it is a mobile platform that is personally arranged for the tracking of the nutrients, medical data and its own diet list. In the application, there are special data showing the amount of water separated by category system, body analysis diet and nutritional values. The user can easily access and follow all the desired information in the main menu. Users can learn the values of nutrients in the database within the application. Within the project, a body mass index can be recorded in the system with the calculation tool. Based on the information entered, suggestions and actions to be taken are calculated. It advises the user on a regular basis every day, and these are offered through notification.

**Key Words :** Android, Mobile Based Application, Dart, Flutter, Json, Data

## **TEŞEKKÜR**

Bu tez çalışmasının planlanmasında, araştırılmasında, yürütülmesinde, oluşumunda ilgi ve desteğini esirgemeyen, engin bilgi ve tecrübelerinden yararlandığım, yönlendirme ve bilgilendirmeleriyle çalışmamı bilimsel temeller ışığında şekillendiren sayın hocamız Dr. Öğr. Üyesi Zafer ALBAYRAK'a sonsuz teşekkürlerimizi sunarız.

## İÇİNDEKİLER

	<u>Sayfa</u>
KABUL .....	ii
ÖZET.....	iv
ABSTRACT .....	v
TEŞEKKÜR.....	vi
İÇİNDEKİLER .....	vii
ŞEKİLLER LİSTESİ .....	viii
BÖLÜM 1 .....	1
GİRİŞ .....	1
1.1. LİTERATÜR ÖZETİ .....	1
1.2. PROJENİN AMACI.....	2
BÖLÜM 2 .....	3
PROJEDE KULLANILAN TEKNOLOJİLER .....	3
2.1. ANDROID İŞLETİM SİSTEMİ .....	3
2.1.1.DART PROGRAMLAMA DİLİ .....	4
2.1.2..FLUTTER PROGRAMLAMA DİLİ .....	4
2.2. FIREBASE PLATFORMU .....	6
2.2.1. FIREBASE REAL TIME DATABASE .....	8
2.2.2. FIREBASE AUTHENTICATION .....	9
2.2.3. FIREBASE STORAGE .....	10
2.2.4. FIREBASE FUNCTIONS .....	10
2.3. SQLITE .....	11
2.3.1.FLUTTER-SQLITE(SQFLITE) .....	13
BÖLÜM 3 .....	14
PROJE YAZILIMI.....	14
3.1. PROJENİN ARAYÜZLERİ.....	14
3.2. PROJENİN AKIŞ ŞEMALARI .....	16
3.2.1. GENEL ÇALIŞMA İÇİN FLOWCHART .....	16



3.2.2. ÜRÜN EKLEME İÇİN FLOWCHART .....	17
3.3. VERİTABANI ŞEMALARI .....	18
3.3.1. KULLANICILAR VERİTABANI ŞEMASI.....	18
3.3.2. KATEGORİLER VERİTABANI ŞEMASI .....	18
3.3.3. ÜRÜNLER VERİTABANI ŞEMASI .....	19
3.3.4. BİLDİRİMLER VERİTABANI ŞEMASI.....	19
3.4. PROJE USER CASE DİYAGRAMI .....	20
3.5. PROJE KODLARININ AÇIKLAMASI .....	21
 BÖLÜM 4 .....	 29
SONUÇ VE DEĞERLENDİRME .....	29
KAYNAKLAR .....	30

## ŞEKİLLER LİSTESİ

Şekil 2.1.	Android Mimarisi.....	4
Şekil 2.2.	IOS, Android Gösterimi.....	5
Şekil 2.3.	Flutter System.....	6
Şekil 2.4.	Firebase Platformu.....	8
Şekil 2.5.	Realtime Database.....	9
Şekil 2.6.	Firebase Authentication.....	9
Şekil 2.7.	Firebase Storage.....	10
Şekil 2.8.	Firebase Functions.....	11
Şekil 2.9.	DB Browser SQLite.....	12
Şekil 2.10.	Flutter-SQLite.....	13



## BÖLÜM 1

### GİRİŞ

#### 1.1. LİTERATÜR ÖZETİ

Nisan-Mayıs 2017 tarihleri arasında yüz yüze veya online diyetisyen kontrolü altında eğitim verilerek diyet yapan 18-65 yaş grubunun da ki erkek (n:28) ve kadın (n:73) toplam 101 bireyin diyet uyumları ve ağırlık kayıplarını karşılaştırmak amacıyla planlanmış ve yürütülmüştür. Araştırmaya online diyet yapan bireylerin %69,8'inin kadın, %30,2'sinin erkek olduğu; yüz yüze diyet yapan bireylerin %74,1'inin kadın, %25,9'unun de erkek olduğu belirlenmiştir ( $p>0,05$ ). Online diyet yapan bireylerin %60,5'inin, yüz yüze diyet yapan bireylerin ise %65,5'inin öğün atladığı belirlenmiştir ( $p>0,05$ ). Online diyet yapan bireylerin %41,9'u normal, %37,2'sinin hafif şişman, %20,0'unun şişman olduğu saptanmıştır. Yüz yüze diyet yapan bireylerin ise %31'i normal, %34,5'i hafif şişman ve şişman olduğu saptanmıştır ( $p>0,05$ ). Online diyet grubundaki bireylerin kaybettikleri ağırlık ortalamasının 6,82 kg, yüz yüze diyet grubundaki bireylerin kaybettiği ağırlık ortalamasının ise 5,90 kg olduğu saptanmıştır ( $p>0,05$ ). Gruplar arasındaki ağırlık kaybı ve kaybedilen ağırlığı koruma bakımından farklılık bulunmamıştır ( $p>0,05$ ). Sonuç olarak bireylerin sağlıklı beslenmeyi öğrenmek amacıyla doğru bilgiye ulaşmaları adına internet aracılığı ile de diyetisyen yardımı almasının faydalı olacağı görüşüne varılmıştır. [1].

YAZIO tarafından geliştirilmiş ücretsiz Kalori Sayacı Uygulaması ile yiyecek günlüğünü yönetebilir, aktivitelerini takip edebilir ve başarıyla kilo verilebilir [2]. Diyet planı, yemek günlüğü, makro hesap makinesi, kalori sayacı ve sağlıklı yemek tarifleri gibi seçeneklere sahip Lifesum uygulaması [3].

Android için en hızlı ve görsel anlamda sade olan kalori sayacı MyFitnessPal uygulaması kalori sayaçları arasında en geniş yiyecek veritabanı (6.000.000'dan fazla yiyecek) ve egzersiz girişi seçenekleriyle uygulama marketlerinde büyük bir yer sahibidir. [4].

Uygulamamızda da olduđu gibi su, yaşamımız için vazgeçilmezdir ve yeterli miktarda su içmek sağlığımız açısından çok önemlidir. VGFit'in Water Reminder uygulaması, vücudunuzun ne kadar suya ihtiyaç duyduğunu hesaplamaya yardımcı olur, su içme sıklığınızı takip eder ve seçilen hedefe ulaşmak için size hatırlatmalarda bulunur. [5].

## 1.2. PROJENİN AMACI

Kişisel sağlık ile ilgili verilerin değerlendirilerek, tıbbi anlamda yardımda bulunulması, bu verilere kolay bir şekilde ulaşılması amaçlanmaktadır. Kilo sorunu yaşayan vatandaşların hayatlarını kolaylaştırmak ve onlara yeni bir alışkanlık kazandırmak kendilerini sevmelerine katkıda bulunmaktır.

İnsanların erteleyerek kendi sağlıklarına zarar vermelerine engel olmak ve bunu yaparken onlara beslenme alışkanlığı kazandırmak amaçlanmıştır. Bu uygulama ile alınması gereken tıbbi destek için zamandan tasarruf edildiği gibi günümüzde önemli bir problem olan obezite'nin önüne geçilmesi hedeflenmiştir.

Uygulama android platformda geliştirildiği için her yaş grubuna hitap edecek şekilde dizayn edilmiştir. Bu sayede daha fazla insana ulaşım sağlanacaktır.

**Hedef 1:** İnsanlar genel anlamda diyetisyene gitmek ya da zayıflamak için bir çaba göstermiyor. Bunu onlar için kolay bir hale getirmek.

**Hedef 2:** Bu süreç içerisinde, diyetisyenler sayesinde gerekli tıbbi verilerin elde edilmesi.

**Hedef 3:** Verilerin toplanması ve gerekli yazılım dillerinin kararlaştırılması ve kaynakların incelenmesi.

**Hedef 4:** Şablonların ve genel tasarımın yapılması ve koda dökülmesi.

**Hedef 5:** Verilerin veri tabanı ve internet üzerinden toplanıp tek parça haline getirilmesi ve uygulama içerisine yerleştirilmesi.

**Hedef 6:** Bir bütün halinde uygulamanın tamamlanması.

## BÖLÜM 2

Bu bölümde android işletim sistemi, dart ve flutter programlama dili gibi kullanılan teknolojiler ve yazılım dilleri hakkında bilgi verilmiştir.

### 2.1.Android İşletim Sistemi

Android Google öncülüğündeki bir organizasyon (Open Handset Alliance) tarafından geliştirilen Linux türevi bir işletim sistemidir. Android; Open Handset Alliance tarafından mobil cihazlar için geliştirilmiş, Linux 2.6 çekirdeğine dayanan, açık kaynak kodlu özel bir Linux dağıtımıdır. Android'in yazılı tarihi Google' ın 2005 yılında Android Inc isimli firmayı satın almasıyla başlamıştır. Google Android'i açık kaynak kod esasına dayanarak geliştirmek istediği için 2007 yılında Open Handset Alliance isimli organizasyonu oluşturmuştur. Open Handset Alliance, Google dışında daha birçok donanım ve yazılım üreticisinin dahil olduğu bir organizasyondur. Örneğin; Samsung, HTC, LG, T-Mobile, Motorola, Intel, Dell, Texas Instruments, Nvidia, Broadcom gibi şuan otuzun üzerinde firma bu organizasyona dahildir.

2007 yılında pek çok yazılım, donanım, telekomünikasyon firmasının katkıda bulunduğu Open Handset Alliance adı verilen şirketler birliği sayesinde ortaya çıkan Android'in kaynak kodları iki farklı lisans kullanır. Kullandığı Linux çekirdeği GPL, diğer dış bileşenler ise Apache Lisansı ile dağıtılmaktadır. Android 2.6 çekirdeği üzerine geliştirilmiş özel bir Linux dağıtımıdır ve process, bellek yönetimi, dosya sistemi, güvenlik gibi temel sistem servisleri diğer Linux türevi sistemlerde olduğu gibidir. Android çekirdek kodlarını açık olarak yazılımcılara sunmuştur. Android açık kaynak kod olmasına rağmen GNU lisansı altında dağıtımı yapılmamaktadır.

Bu platformda uygulamalar Android Yazılım Geliştirme Kiti (SDK) kullanarak Java veya Kotlin dilinde yazılır. Bu SDK yazılımcıya hata ayıklayıcı, yazılım kütüphaneleri ve emülatör gibi yardımcı araçlar sunar.

Android güç kullanımını en aza indirmek ve hafızayı yeterli kullanabilmek için çeşitli yöntemler kullanır. Kullanılmayan uygulamaların bekleme moduna alınması; yetersiz hafıza durumunda uzun süredir aktif olmayan uygulamaların kapatılması bu yöntemlerden bazılarıdır.[6]



Şekil 2.1. Android Mimarisi

Android işletim sistemi 4 ana katmandan oluşur:

1. Linux Çekirdek (Linux Kernel)
2. Kütüphaneler (Libraries)
3. Android Çalışma Zamanı (Android Runtime)
  - 3.1.Çekirdek Kütüphaneleri (Core Libraries)
  - 3.2.Dalvik Sanal Makinesi (DVM – Dalvik Virtual Machine)
4. Uygulama Çatısı (Application Framework)
5. Uygulamalar (Applications) [6]

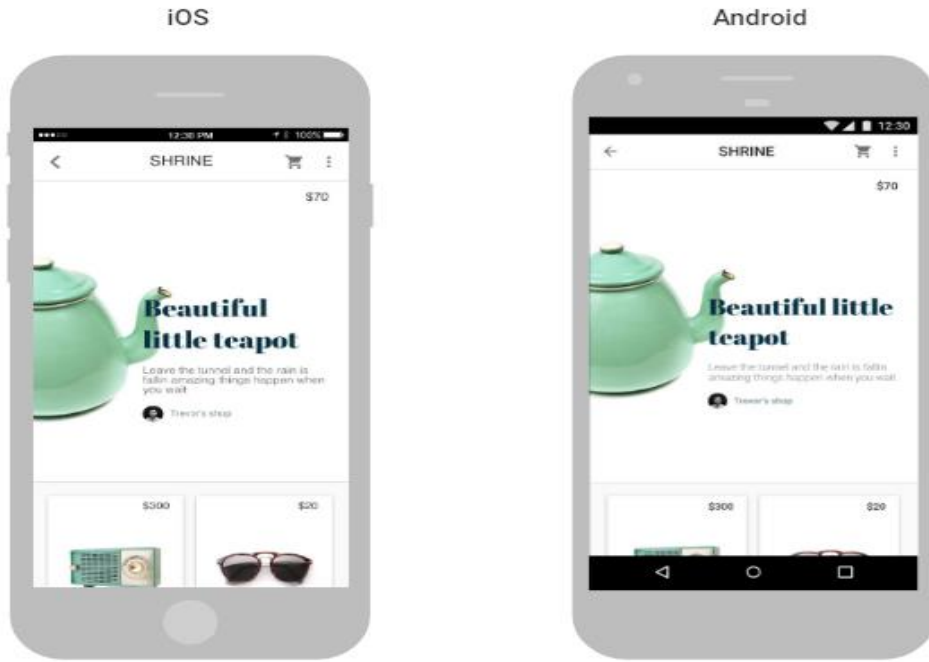
### 2.1.1. Dart Programlama Dili

Dart, ilk kez Google tarafından 2011 yılında nesne yönelimli, smalltalk tarzı ve JavaScript'in yerini almasını düşünülerek geliştirilmiş bir programlama dilidir. Ecma International tarafından standart haline getirilen açık kaynaklı dildir. Flutter SDK sayesinde Dart dilini kullanarak hem iOS hem de Android uygulamalar geliştirilebiliyor. Mobil uygulamalar haricinde web, sunucu ve IoT cihazlar içinde uygulamalar geliştirilebilir. Dart dili sınıf tabanlı, tek kalıtıma sahip nesne tabanlı, C

programlama dilinin kod dizilimine benzemektedir. JavaScript diline veya çalıştığı sistemdeki native dile çevrilebilir. Interface'lar, Abstracts, generic type ve opsiyonel tipleri destekler.

### 2.1.2. Flutter Programlama Dili

Flutter, Google tarafından geliştirilen açık kaynak kodlu bir mobil uygulama geliştirme SDK'sıdır. Amaç, geliştiricilerin farklı platformlarda doğal hissettikleri yüksek performanslı uygulamalar sunmalarını sağlamaktır örneğin; Windows, Linux ya da Mac ortamlarında rahatlıkla çalıştırılabilir. Android ve iOS için uygulama geliştirmek ve Google Fuchsia işletim sistemi için uygulama geliştirmek için kullanılır.



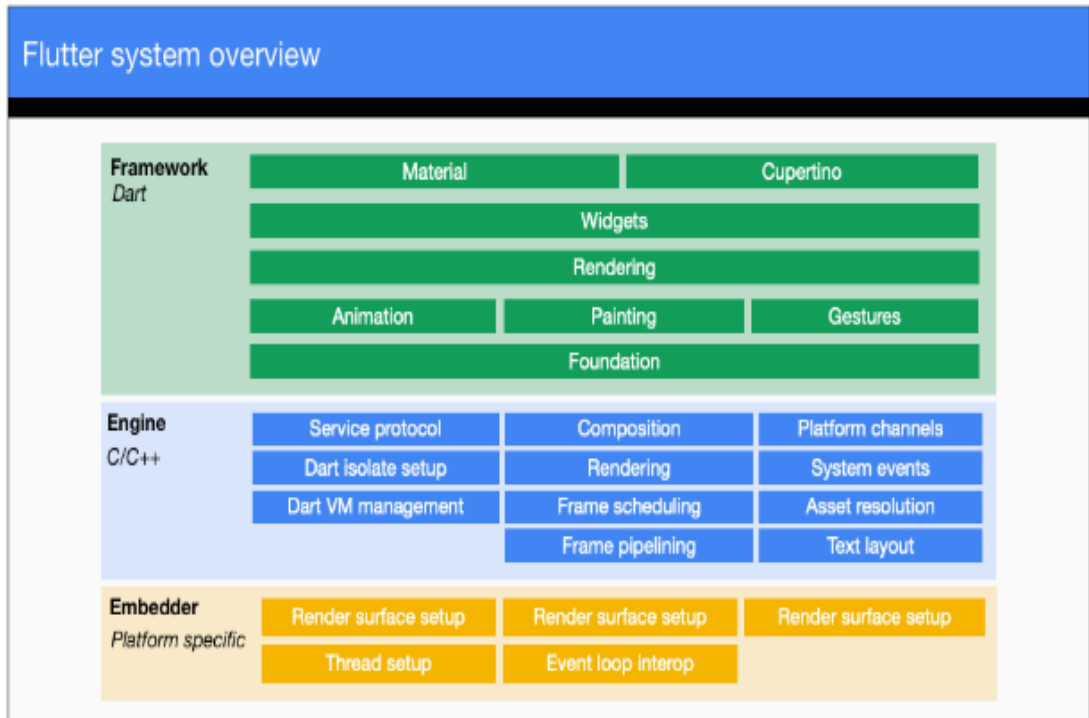
Şekil 2.2. IOS, ANDROID Gösterimi

### Flutter Avantajları

Tek bir kod tabanından iOS ve Android için geliştirilen modern, etkileyici bir dil ve bildirimsel bir yaklaşımla tek işletim sisteminde bile daha az kodla daha fazlasını yapmamıza olanak tanır. Kolayca prototip oluşturabilir ve uygulama çalışırken kodları değiştirerek yeniden yükleme yapılabilir. Flutter'ın kendi çerçevesi kullanılarak



oluşturulmuş zengin bir materyal tasarım ve OEM widget setlerinin sınırlamaları olmadan özel, marka odaklı tasarımlar gerçekleştirilebilir. Flutter, modern tepki tarzı bir çerçeve, bir 2B oluşturma motoru, hazır widget'lar ve geliştirme araçları içerir ve bileşenler uygulamaları tasarlamamıza, oluşturmamıza, test etmemize ve hataları ayıklamamıza yardımcı olmak için birlikte çalışır. Her şey birkaç temel ilke etrafında düzenlenmiştir. Flutter her katman bir önceki katman üzerine inşa edilecek şekilde bir dizi katman halinde düzenlenmiştir. [7]



Şekil 2.3. Flutter System

## 2.2. Firebase Platformu

Firebase, geliştiricilere yüksek kaliteli uygulamalar geliştirmelerine, kullanıcı tabanlarını büyütmelerine ve daha fazla kar elde etmelerine yardımcı olacak çok sayıda araç ve hizmet sunan bir mobil ve web uygulama geliştirme platformudur.

### Kısa bir tarihçe

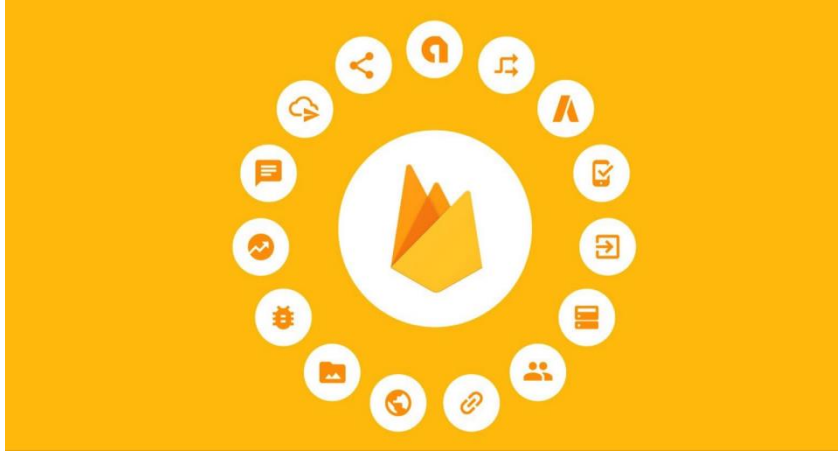
2011 yılında Firebase Firebase'den önce, Envolv adlı bir başlangıçtı. Envolv olarak, geliştiricilere çevrimiçi sohbet işlevselliğinin web sitelerine entegrasyonunu sağlayan bir API sağladı. Fakat insanlar sadece sohbet mesajlarından daha fazlası olan

uygulama verilerini iletmek için Envolv kullandılar. Geliştiriciler, kullanıcıları arasında gerçek zamanlı olarak oyun durumu gibi uygulama verilerini senkronize etmek için Envolv kullanıyorlardı. Bu Envolv kurucularının ( [James](#) Tamplin ve [Andrew Lee'nin](#)) sohbet sistemini ve gerçek zamanlı mimariyi ayırmalarına yol açtı . 2012 yılının Nisan ayında, Firebase, *gerçek zamanlı işlevsellikle* Backend as a Service sağlayan ayrı bir şirket olarak kuruldu . 2014 yılında Google tarafından satın alındıktan sonra, Firebase hızla bugünün çok fonksiyonlu mobil ve web platformu haline geldi.

## Firestore Hizmetleri

Firestore hizmetleri 3 gruba ayrılabilir:

1. Daha iyi uygulamalar oluşturun
  - Cloud Firestore (BETA)
  - ML Kit (BETA)
  - Cloud Functions
  - Authentication
  - Hosting
  - Cloud Storage
  - Realtime Database
2. Uygulama kalitesini iyileştirin
  - Crashlytics
  - Performance Monitoring
  - Test Lab
3. İşletmenizi büyütün
  - In-App Messaging
  - Google Analytics
  - Predictions (BETA)
  - A/B Testing (BETA)
  - Cloud Messaging
  - Remote Config
  - Dynamic Links
  - App Indexing



Şekil 2.4. Firebase Platformu

### 2.2.1. Firebase Realtime Database

Firebase Gerçek Zamanlı Veritabanı, verilerinizi gerçek zamanlı olarak kullanıcılarınız arasında depolamanıza ve senkronize etmenize olanak tanıyan bulut tarafından barındırılan bir NoSQL veritabanıdır.

Gerçek Zamanlı Veritabanı, geliştiricilerin gerçek zamanlı olarak yönetebileceği büyük bir JSON nesnesidir.

Gerçek zamanlı senkronizasyon, kullanıcılarınızın verilerinin herhangi bir cihazdan erişilmesini kolaylaştırır. Web veya mobil ve kullanıcılarınızın birbirleriyle işbirliği yapmasına yardımcı olur. Gerçek Zamanlı Veri Tabanı, mobil ve web SDK ile birlikte gelir ve böylece sunuculara ihtiyaç duymadan uygulamalar oluşturabilirsiniz. Ayrıca [Firebase](#) Cloud Function kullanarak veritabanınızın tetiklediği olaylara yanıt veren arka uç kodunu da çalıştırabilirsiniz.

Kullanıcılarınız çevrimdışı duruma geçtiğinde, Gerçek Zamanlı Veritabanı SDK'ları, değişiklikleri sunmak ve depolamak için cihazdaki yerel önbelleği kullanır. Cihaz çevrimiçi olduğunda, yerel veriler otomatik olarak senkronize edilir. Gerçek Zamanlı Veritabanı, geliştiriciler için basit ve sezgisel kimlik doğrulaması sağlamak için Firebase Authentication ile bütünleşir.



Şekil 2.5. Realtime Database

### 2.2.2. Firebase Authentication

Firebase Authentication, son kullanıcılar için oturum açma ve onboard oluşturma deneyimini geliştirirken güvenli kimlik doğrulama sistemleri oluşturmayı kolaylaştırmayı amaçlar.

E-posta ve şifre hesaplarını, telefon yetkisini ve Google, Twitter, Facebook ve GitHub girişlerini ve daha fazlasını destekleyen, uçtan uca bir kimlik çözümü sağlar. FirebaseUI, kullanıcılara imzalamak için UI akışlarını işleyen özelleştirilebilir, açık kaynaklı bir drop-in yetkisiz çözüm sunar.

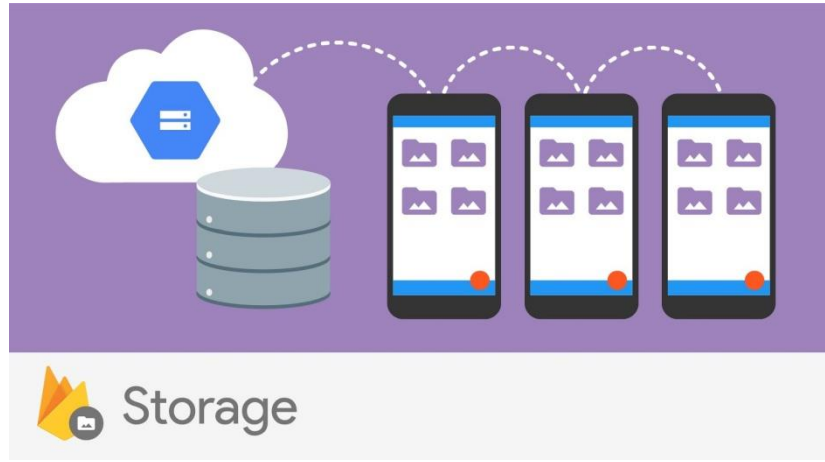
FirebaseUI Auth bileşeni, uygulamanız için oturum açma ve kaydolma dönüşümünü en üst düzeye çıkarabilen mobil cihazlarda ve web sitelerinde kimlik doğrulaması için en iyi uygulamaları kullanır.



Şekil 2.6. Firebase Authentication

### 2.2.3. Firebase Storage

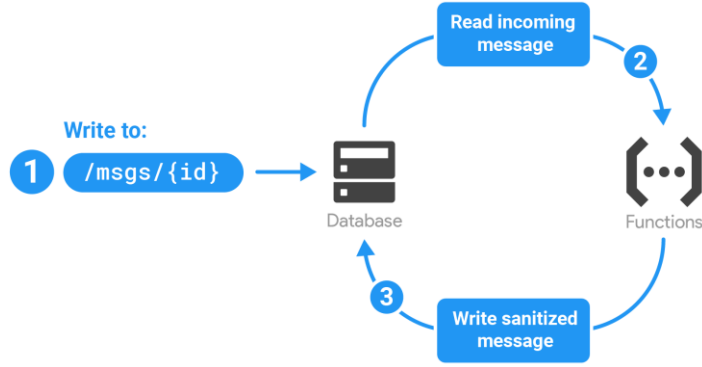
Cloud Storage, fotoğraf ve videolar gibi kullanıcı tarafından oluşturulan içerikleri hızlı ve kolay bir şekilde depolamaya ve sunmaya yardımcı olmak için tasarlanmıştır. Uygulama, kullanıcıların zamanını ve bant genişliğini azaltarak mobil bağlantıyı kaybedip yeniden bağlandıklarında aktarımlarını otomatik olarak duraklatır ve devam ettirir. Cloud Storage için Firebase SDK'sı, basit ve sezgisel erişim kontrolü sağlamak için Firebase Authentication ile bütünleşir.



Şekil 2.7. Firebase Storage

### 2.2.2 Firebase Functions

Bulut İşlevleri, güvenli, yönetilen bir Node.js ortamında yürütülen tek amaçlı JavaScript işlevleridir. Sadece izlenen belirli bir olay yayınlandığında çalıştırılırlar. Bundan sonra, Cloud Functions, uygulama kaynaklarının uygulamanın kullanım düzenleriyle eşleşmesi için otomatik olarak ölçeklendirir. Çoğu durumda, uygulama mantığı, istemci tarafında kurcalamasını önlemek için sunucu üzerinde en iyi şekilde kontrol edilir. Bulut İşlevleri istemciden tamamen yalıtılmıştır. [8]

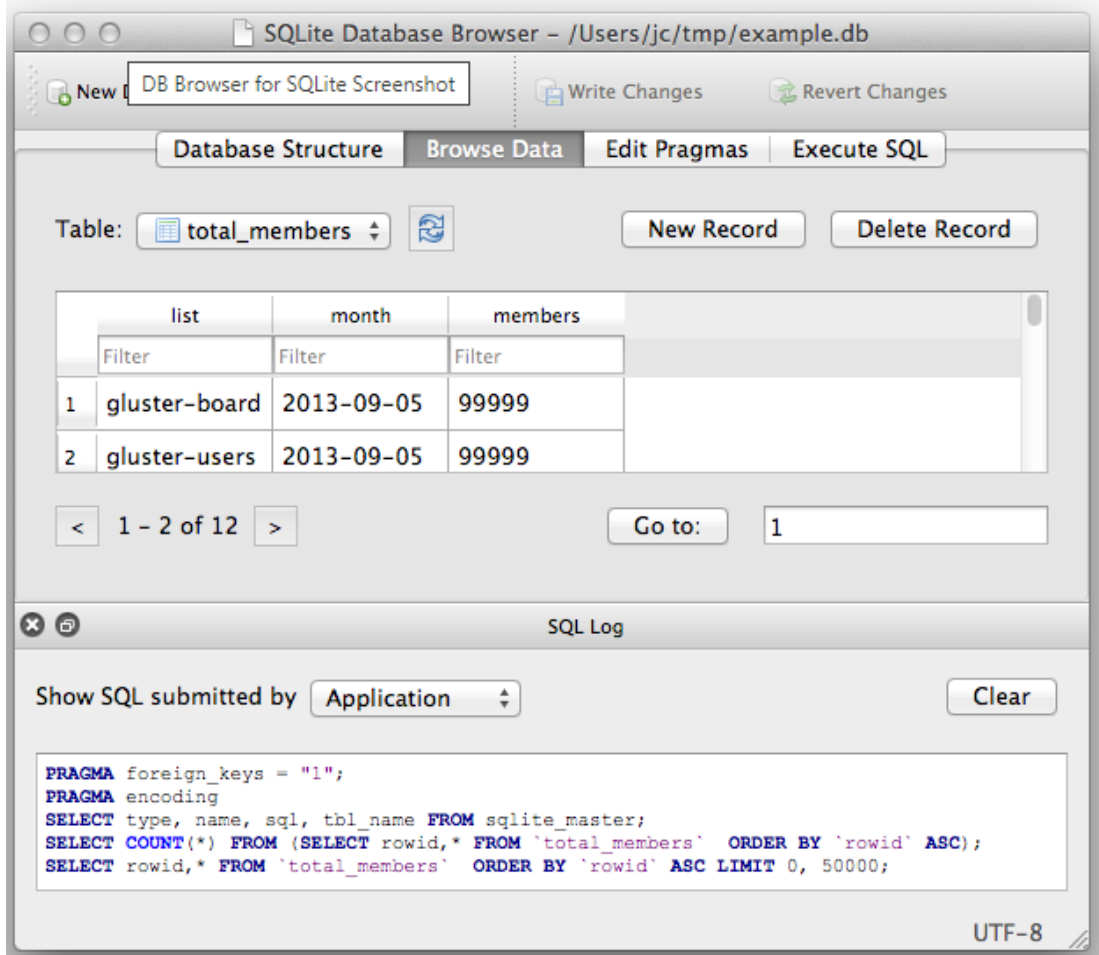


Şekil 2.8. Firebase Functions

### 2.3.SQLite

SQLite; küçük, hızlı, bağımsız, sıfır konfigürasyonlu, işlemsel bir SQL veritabanı motorunu uygulayan işlem içi bir kütüphanedir. Dünyada en yaygın olarak kullanılan ücretsiz veritabanıdır. Diğer SQL veritabanlarının çoğundan farklı olarak, SQLite'nin ayrı bir sunucu işlemi yoktur. SQLite doğrudan sıradan disk dosyalarını okur ve yazar. Birden çok tablo, izin ve görünüm içeren eksiksiz bir SQL veritabanı tek bir dosyada bulunur. Veritabanı dosya biçimi platformlar arasındır. 32 bit ve 64 bit sistemler arasında veya big-endian ve little-endian mimarileri arasında veritabanını kopyalamaya izin vermektedir. Bu özelliği sayesinde popüler bir seçenek haline getirir. SQLite veritabanı dosyaları, ABD Konge Kütüphanesi tarafından önerilen bir depolama biçimidir. SQLite kompakt bir kütüphanedir, tüm özellikler etkinleştirildiğinde hedef platforma ve derleyici optimizasyon ayarlarına bağlı olarak kütüphane boyutu 600 KB'tan az olabilir. Bellek kullanımı ile hız arasında bir denge vardır. SQLite genellikle ne kadar çok bellek vererseniz o kadar hızlı çalışabilmektedir. Bununla birlikte düşük bellekli ortamlarda da performans genellikle oldukça iyidir. Kullanım şekline bağlı olarak, SQLite doğrudan dosya sistemi G/Ç'den daha hızlı olabilir. SQLite her sürümden önce dikkatli bir şekilde test edildiği için çok güvenilir bir üne sahiptir. SQLite kaynak kodunun çoğu yalnızca test ve doğrulamaya ayrılmıştır. SQLite bellek ayırma hatalarına ve disk G/Ç hatalarına yanıt verir. İşlemler system çökmeleri veya elektrik kesintileri nedeniyle kesilse bile ACID'dir. (ACID (atomicity, consistency, isolation, durability) geçerliliği garanti etmeyi amaçlayan

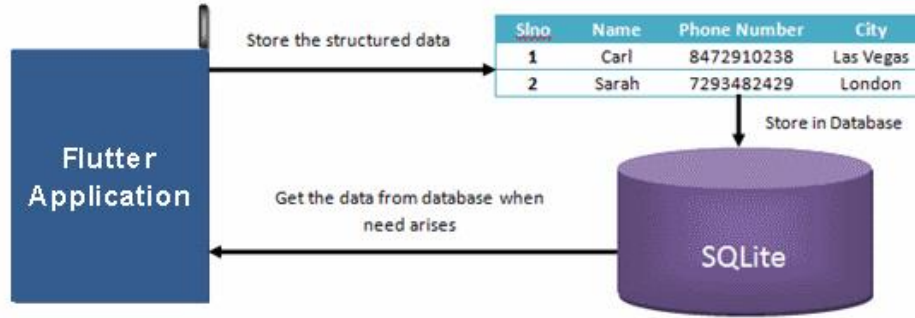
veritabanı işlemlerinin bir dizi özelliğidir.) Tüm bunlar, sistem arızalarını simüle eden özel test donanımları kullanılarak doğrulanır. Tabii ki, tüm bu testlerde bile hatalar olabilir. SQLite hatalar hakkında açık ve dürüstür, hata listeleri dakika dakika kod değişikliği kronolojileri ile imkan sağlar. SQLite kod tabanı, SQLite üzerinde tam zamanlı çalışabilen uluslararası bir geliştiriciler ekibi tarafından desteklenmektedir. Geliştiriciler, SQLite'nın yeteneklerini genişletmeye ve yayımlanan arabirim özellikleri, SQL sözdizimi ve veritabanı dosya biçimi ile geriye dönük uyumluluğu korurken güvenilirliğini ve performansını arttırmaya devam ediyor. SQLite projesi 2000 yılında başlatılmış geleceği tahmin etmek zor ama geliştiricilerin amacı SQLite'yi 2050 yılına kadar desteklemektir. Tasarım kararları bu amaç doğrultusunda oluşturuluyor. [9]



Şekil 2.9. DB Browser SQLite

### 2.3.1 Flutter-SQLite (SQFLite)

Flutter uygulamaları, pub'da bulunan sqflite eklentisi aracılığıyla SQLite veritabanlarından yararlanabilir. Sqflite paketi, SQLite veritabanıyla etkileşim kurmak için sınıflar ve işlevler sağlar. Paket veritabanını diskte depolamak için konumu tanımlayan işlevler sağlar.



Şekil 2.10. Flutter-SQLite




## BÖLÜM 3

### PROJE YAZILIMI

Bu bölümde geliştirilen yazılım için kodların açıklanması, akış diyagramları, veritabanı şemaları, arayüz tasarımları açıklanmıştır.


#### 3.1. Projenin Arayüzleri

### Diyet Buddy



Devam Et

Yeni Kişi

 Google ile Giriş

<

Yeni Kişi

Başla

ADINIZ

Adınız

YAŞINIZ

Yaşınız

KILO

Kilo

BOY

Boy

ŞİFRE

Şifre

Devam Et

05:15 57%

### Vücut Endeksi

Erkek

Kadın

Boy  
**180** cm

Kilo  
**80**

Yaş  
**20**

Hesapla

Home Vücut Endeksi

05:14 57%

### Sonucunuz

NORMAL

**24.7**

Mükemmel bir kilonuz var. Hadi bunu koruyalım.

Tekrar Hesapla

Home

05:18 57%

### Yemek Listesi

Yiyecek İsmi Giriniz

Kalori Miktarı Giriniz

Ekle Temizle

MAGNUM DONDURMA  
700 Kalori

KARPUZ  
120 Kalori

TAVUK  
250 Kalori

ELMA  
68 Kalori

Home Diyet Listesi

05:23 56%

### Nasıl Yakarız

Çikolata

Bitter

420 Kalori

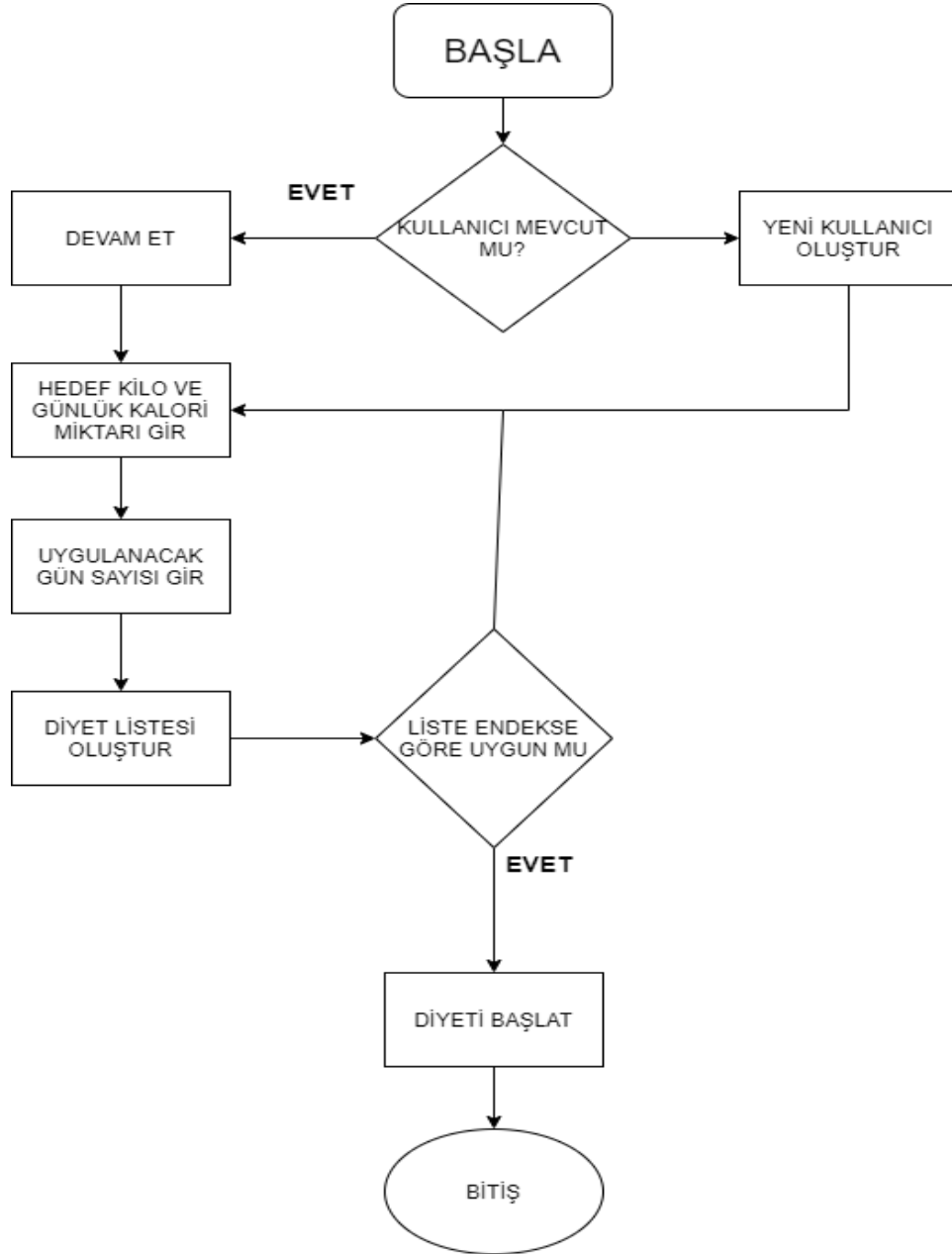
47 Dakika 45 Dakika 41 Dakika 52 Dakika

Tavsiyeler

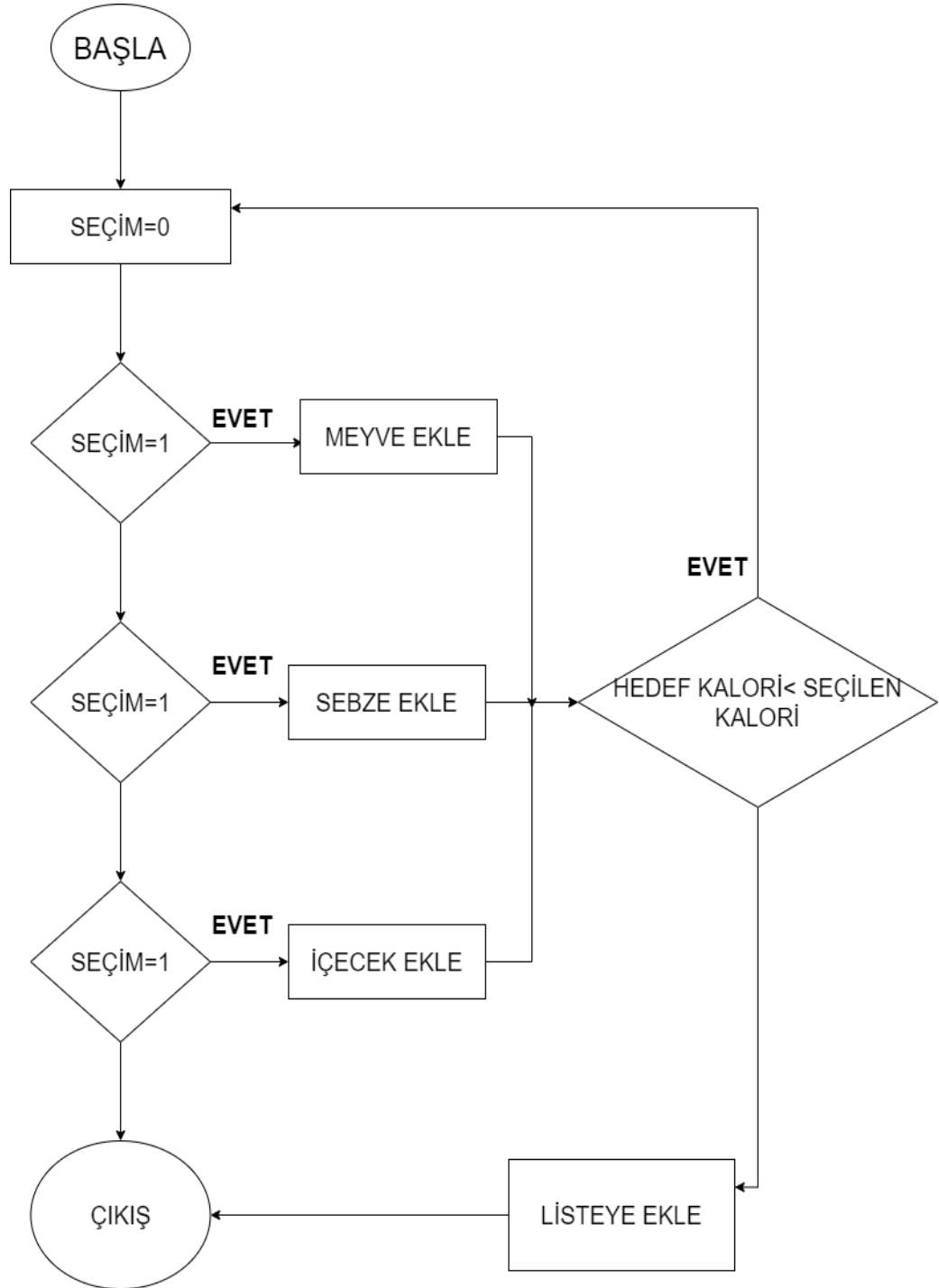
Home

### 3.2. Projenin Akış Şemaları

#### 3.2.1 Genel Çalışma İçin Flowchart

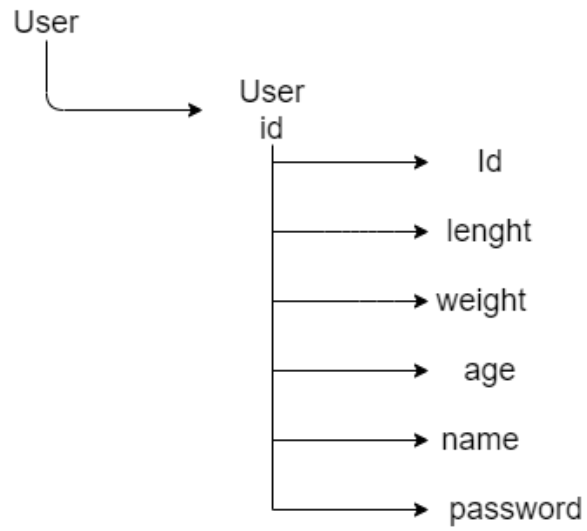


### 3.2.2 Ürün Ekleme İçin Flowchart

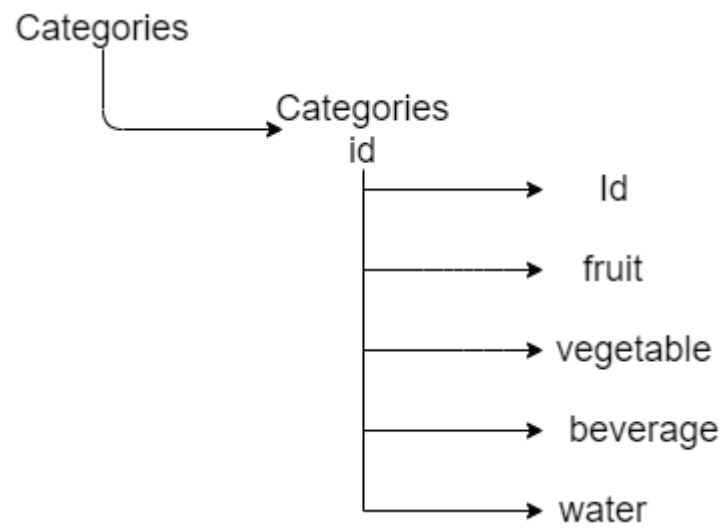


### 3.3 Veritabanı Şemaları

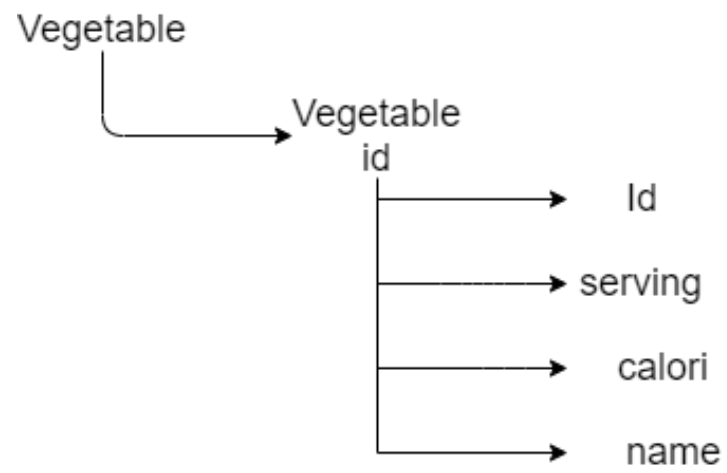
#### 3.3.1 Kullanıcılar Veritabanı Şeması



#### 3.3.2 Kategoriler Veritabanı Şeması

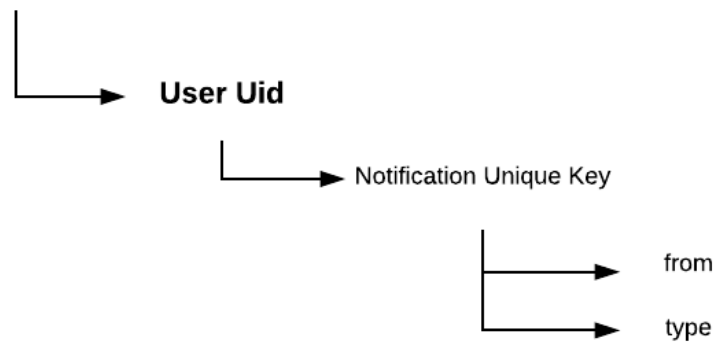


### 3.3.3 Ürünler Veritabanı Şeması

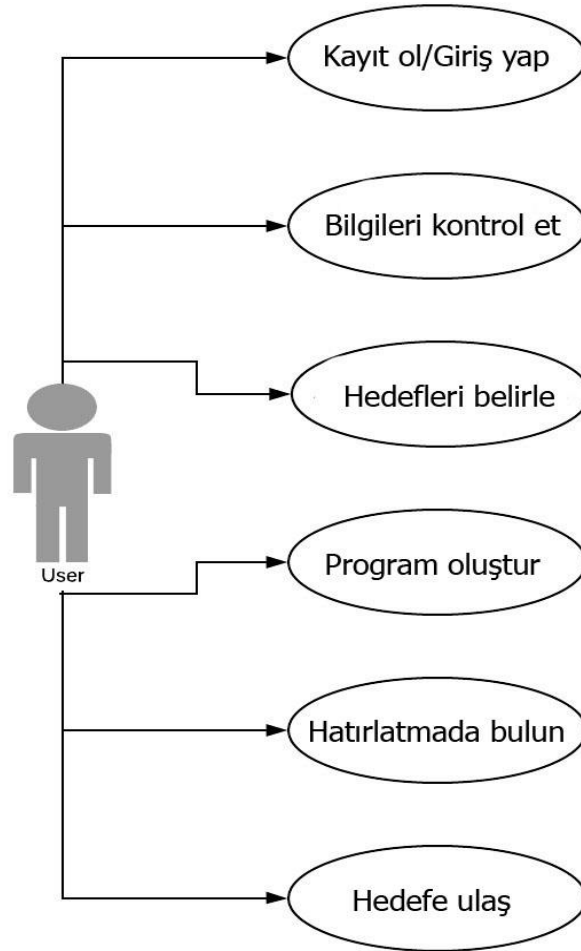


### 3.3.4 Bildirimler Veritabanı Şeması

#### Notifications



### 3.4 Proje User Case Diyagramı



### 3.5 Proje Kodlarının Açıklanması

```
void _emailveSifreileUserOlustur() async {
  String mail = "suleymanaltunakar@gmail.com";
  String sifre = "123456";
  var authResult = await _auth
    .createUserWithEmailAndPassword(
      email: mail,
      password: sifre,
    )
    .catchError((e) => debugPrint("Hata : " + e.toString()));
  var firebaseUser = authResult.user;
  if (firebaseUser != null) {
    firebaseUser
      .sendEmailVerification()
      .then((data) {
        _auth.signOut();
      })
      .catchError((e) => debugPrint("Mail gönderirken hata $e"));

    setState(() {
      mesaj =
        "Uid ${firebaseUser.uid} \nmail : ${firebaseUser.email} \nmailOnayı : ${firebaseUser.isEmailVerified}\n Email gönderildi lütfen onaylayın";
    });
    debugPrint(
      "Uid ${firebaseUser.uid} mail : ${firebaseUser.email} mailOnayı : ${firebaseUser.isEmailVerified} ";
    );
  } else {
    setState(() {
      mesaj = "bu mail zaten kullanımda";
    });
  }
}
```

- Başlangıç sayfasında 3 adet buton bulunmaktadır.
- Birinci buton kayıtlı kullanıcının giriş yapması, ikinci buton yeni kullanıcının eklenmesi ve üçüncü buton Google ile giriş yapabilmesi için oluşturuldu.
- Yeni bir kullanıcı oluşturmak için 'emailveSifreileUserOlustur' metodunu kullanıyoruz.
- Kullanıcının girmiş olduğu bilgileri kontrol edilip veritabanına kaydediyoruz.



```

void _googleGirisi() {
    _googleAuth.signIn().then((sonuc){
        sonuc.authentication.then((googleKeys){
            AuthCredential credential=GoogleAuthProvider.getCredential(idToken: googleKeys.idToken, accessToken: googleKeys.accessToken);
            _auth.signInWithCredential(credential).then((userAuthResult){
                var user= userAuthResult.user;
                setState(() {
                    mesaj += "\nGmail ile giriş yapıldı\n User id:${user.uid}\nMail : ${user.email}";
                });
            }).catchError((hata){
                setState(() {
                    mesaj += "\nFirebase ve google kullanıcı hatası $hata";
                });
            });
        }).catchError((hata){
            setState(() {
                mesaj += "\nGoogle authentication hatası $hata";
            });
        });
    }).catchError((hata){
        setState(() {
            mesaj += "\nGoogle Auth signin hatası $hata";
        });
    });
}
}

```

- Google giriş butonu ile Firebase sayesinde kullanıcının bilgileri kontrol edilip uygulamaya girişi sağlandı.

```

class LoginIslemleri extends StatefulWidget {
  @override
  _LoginIslemleriState createState() => _LoginIslemleriState();
}

class _LoginIslemleriState extends State<LoginIslemleri> {
  final FirebaseAuth _auth = FirebaseAuth.instance;
  final GoogleSignIn _googleAuth = GoogleSignIn(scopes: ['email']);

  String mesaj = "";

  @override
  void initState() {
    // TODO: implement initState
    super.initState();
    _auth.onAuthStateChanged.listen((user){
      setState(() {
        if(user != null){
          mesaj += "\nListener tetiklendi kullanıcı oturum açtı";
        }else {
          mesaj += "\nListener tetiklendi kullanıcı oturumu kapattı";
        }
      });
    });
  }
}

```

- LoginIslemleri sınıfı ile kayıtlı kullanıcının uygulamaya girişi sağlandı.

```

200 // Expanded
201 BottomButton(
202   buttonTitle: 'Hesapla',
203   onTap: () {
204     CalculatorBMI calc =
205       CalculatorBMI(height: height, weight: weight);
206     Navigator.of(context).push(
207       MaterialPageRoute(
208         builder: (_) => ResultsPage(
209           bmiResult: calc.getBMI(),
210           resultText: calc.getResult(),
211           interpretation: calc.getInterpretation(),
212         ), // ResultsPage, MaterialPageRoute
213       );
214     },
215   ), // BottomButton
216 ], // <Widget>[]
217 ), // Column
218 ); // Scaffold
219 }
220 }
221

```

- Dart math kütüphanesi kullanılarak kayıtlı kullanıcının girmiş olduğu boy, kilo ve cinsiyet değerlerine göre sonuç hesaplanıp ekranda gösterilmesi işlemi yapıldı.

```

class ResultsPage extends StatelessWidget {
  final String bmiResult;
  final String resultText;
  final String interpretation;

  const ResultsPage(
    {@required this.bmiResult,
    @required this.resultText,
    @required this.interpretation});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Column(
        mainAxisAlignment: MainAxisAlignment.spaceEvenly,
        crossAxisAlignment: CrossAxisAlignment.stretch,
        children: <Widget>[
          Expanded(
            child: Container(
              padding: EdgeInsets.all(15.0),
              alignment: Alignment.bottomLeft,
              child: Text(
                'Sonucunuz',
                style: kTitleStyle,
              ), // Text
            ), // Container
          ), // Expanded
        ],
      ),
    );
  }
}

```

- ResultPage classında kayıtlı kullanıcının girmiş olduğu bilgilere göre sonucun gösterilmesi işlemi yapıldı.

```

        var foodDB = openDatabase(dbPath, version: 1, onCreate: _createDB );
        return foodDB;
    }

    Future<void> _createDB(Database db, int version) async{
        await db.execute("CREATE TABLE $_foodTablo ($_columnID INTEGER PRIMARY KEY AUTOINCREMENT, $_columnIsim TEXT, $_columnKalori INTEGER )");
    }

    Future<int> foodEkle(Food food) async{
        var db = await _getDatabase();
        var sonuc = await db.insert(_foodTablo, food.dbyeYazmakIcinMapeDonustur(), nullColumnHack: "$_columnID");
        return sonuc;
    }

    Future<List<Map<String, dynamic>>> tumYemekler() async{
        var db = await _getDatabase();
        var sonuc = await db.query(_foodTablo, orderBy: "$_columnID DESC");
        return sonuc;
    }

    Future<int> foodGuncelle(Food food) async{
        var db = await _getDatabase();
        var sonuc = await db.update(_foodTablo, food.dbyeYazmakIcinMapeDonustur(), where: "$_columnID = ?", whereArgs: [food.id]);
        return sonuc;
    }

    Future<int> foodSil(int id) async{
        var db = await _getDatabase();
        var sonuc = await db.delete(_foodTablo, where: "$_columnID = ?", whereArgs: [id]);
        return sonuc;
    }

    Future<int> listeyiTemizle() async{
        var db = await _getDatabase();
        var sonuc = await db.delete(_foodTablo);
        return sonuc;
    }
}

```

```

factory DatabaseHelper() {
    if (_databaseHelper == null) {
        _databaseHelper = DatabaseHelper._internal();
        return _databaseHelper;
    } else {
        return _databaseHelper;
    }
}

DatabaseHelper._internal();

Future<Database> _getDatabase() async {
    if(_database == null){
        _database = await _initializeDatabase();
        return _database;
    } else {
        return _database;
    }
}

_initializeDatabase() async {
    Directory klasor = await getApplicationDocumentsDirectory();
    String dbPath = join(klasor.path, "food.db");

    var foodDB = openDatabase(dbPath, version: 1, onCreate: _createDB );
    return foodDB;
}

Future<void> _createDB(Database db, int version) async{
    await db.execute("CREATE TABLE $_foodTablo ($_columnID INTEGER PRIMARY KEY AUTOINCREMENT, $_columnIsim TEXT, $_columnKalori INTEGER )");
}

Future<int> foodEkle(Food food) async{
    var db = await _getDatabase();
    var sonuc = await db.insert(_foodTablo, food.dbyeYazmakIcinMapeDonustur(), nullColumnHack: "$_columnID");
    return sonuc;
}

Future<List<Map<String, dynamic>>> tumYemekler() async{

```

- Sqflite kullanılarak database oluşturuldu. Oluşturulan yiyeceklerin veritabanına eklenmesi, silinmesi, güncellenmesi işlemlerin yapılması sağlandı.

```

void _veriEkle() {
    Map<String, dynamic> uyeEkle = Map();
    uyeEkle['ad'] = "süleyman";

    _firestore
        .collection("users")
        .document("süleyman_altunakar")
        .setData(uyeEkle, merge: true)
        .then((v) => debugPrint("Süleyman eklendi"));

    _firestore
        .collection("users")
        .document("hasan_yilmaz")
        .setData({'ad': 'Hasan', 'cinsiyet': 'erkek'}).whenComplete(
            () => debugPrint("hasan eklendi"));

    _firestore.document("/users/ayse").setData({'ad': 'ayse'});

    _firestore.collection("users").add({'ad': 'can', 'yas': 35});

    String yeniKullaniciID =
        _firestore.collection("users").document().documentID;
    debugPrint("yeni doc id: $yeniKullaniciID");
    _firestore
        .document("users/$yeniKullaniciID")
        .setData({'yas': 30, 'userID': '$yeniKullaniciID'});

    _firestore.document("users/süleyman_altunakar").updateData({
        'boy': 187,
        'yas': 25,
        'kilo': 105,
        'günSayisi': 21
    }).then((v) {
        debugPrint("süleyman güncellendi");
    });
}

```

- Kayıtlı kullanıcının veri ekleme işlemleri yapıldı.
- Kullanıcının girmiş olduğu veriler ile daha önce girdiği veriler karşılaştırılarak güncel verilerin ayarlanması işlemi sağlandı.

```

void _veriSil() {
  //Döküman silme
  _firestore.document("users/ayse").delete().then((aa) {
    debugPrint("ayse silindi");
  }).catchError((e) => debugPrint("Silerken hata çıktı" + e.toString()));

  _firestore
    .document("users/hasan_yilmaz")
    .updateData({'cinsiyet': FieldValue.delete()}).then((aa) {
      debugPrint("cinsiyet silindi");
    }).catchError((e) => debugPrint("Silerken hata çıktı" + e.toString()));
}

Future _veriOku() async {
  //tek bir dökümanın okunması
  DocumentSnapshot documentSnapshot =
  await _firestore.document("users/süleyman_altunakar").get();
  debugPrint("Döküman id:" + documentSnapshot.documentID);
  debugPrint("Döküman var mı:" + documentSnapshot.exists.toString());
  debugPrint("Döküman string: " + documentSnapshot.toString());
  debugPrint("bekleyen yazma var mı:" +
    documentSnapshot.metadata.hasPendingWrites.toString());
  debugPrint("cacheden mi geldi:" +
    documentSnapshot.metadata.isFromCache.toString());
  debugPrint("cacheden mi geldi:" + documentSnapshot.data.toString());
  debugPrint("cacheden mi geldi:" + documentSnapshot.data['ad']);
  debugPrint("cacheden mi geldi:" + documentSnapshot.data['boy'].toString());
  documentSnapshot.data.forEach((key, deger) {
    debugPrint("key : $key deger :deger");
  });
}

```

- Kayıtlı kullanıcının veri okuma ve silme işlemleri kontrolleri yapıldı.
- Kullanıcının girmiş olduğu verilerin yanlış veya hatalı girmesi sonucu verinin silme işlemi sağlandı
- Kullanıcının kendi verilerini görebilmesi işlemi yapıldı.



```

class Food {
  int _id;
  String _isim;
  int _kalori;

  int get id => _id;

  set id(int value) {
    _id = value;
  }

  String get isim => _isim;

  int get kalori => _kalori;

  set kalori(int value) {
    _kalori = value;
  }

  set isim(String value) {
    _isim = value;
  }

  Food(this._isim, this._kalori);

  Food.withID(this._id, this._isim, this._kalori);

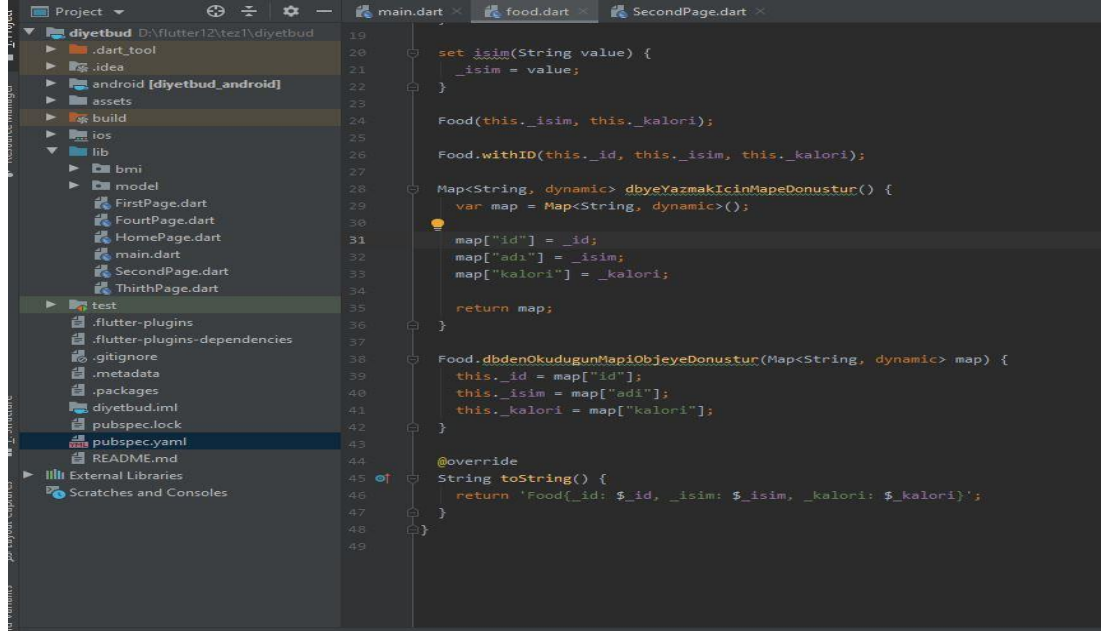
  Map<String, dynamic> dbyeYazmakIcinMapeDonustur() {
    var map = Map<String, dynamic>();

    map["id"] = _id;
    map["isim"] = _isim;
    map["kalori"] = _kalori;

    return map;
  }

  Food.dbdenOkudugunMapiObjeyeDonustur(Map<String, dynamic> map) {
    this._id = map["id"];
    this._isim = map["isim"];
    this._kalori = map["kalori"];
  }
}

```



The screenshot shows an IDE with a project named 'diyetbud' and a file named 'food.dart' open. The project structure on the left includes folders for 'diyetbud', '.dart\_tool', '.idea', 'android [diyetbud\_android]', 'assets', 'build', 'ios', 'lib', 'bmi', 'model', and 'test'. The 'lib' folder contains files like 'FirstPage.dart', 'FourtPage.dart', 'HomePage.dart', 'main.dart', 'SecondPage.dart', and 'ThirthPage.dart'. The 'test' folder contains 'flutter-plugins', 'flutter-plugins-dependencies', '.gitignore', '.metadata', '.packages', 'diyetbud.iml', 'pubspec.lock', 'pubspec.yaml', and 'README.md'. The 'food.dart' file contains the following code:

```

19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49

```

```

set isim(String value) {
  _isim = value;
}

Food(this._isim, this._kalori);

Food.withID(this._id, this._isim, this._kalori);

Map<String, dynamic> dbyeYazmakIcinMapeDonustur() {
  var map = Map<String, dynamic>();

  map["id"] = _id;
  map["adi"] = _isim;
  map["kalori"] = _kalori;

  return map;
}

Food.dbdenOkudugunMapiObjeyeDonustur(Map<String, dynamic> map) {
  this._id = map["id"];
  this._isim = map["adi"];
  this._kalori = map["kalori"];
}

@override
String toString() {
  return 'Food{_id: $_id, _isim: $_isim, _kalori: $_kalori}';
}

```

- Food class ile uygulamaya eklenen yiyeceklerin adı ve kalori miktarı kullanıcıdan alınmak üzere gerekli methodlar oluşturuluyor..
- Yiyeceklerin hesaplanan Dedeğerleri veritabanına eklenir.

## **BÖLÜM 4**

### **SONUÇ VE DEĞERLENDİRME**

Bu program android tabanlı mobil uygulamalar kapsamında oluşturulmuş bir diyet programı uygulamasıdır.

Uygulama androidde ve iOS da çalışacak şekilde çapraz platform destekli dart dili tabanlı flutter framework kullanılarak yazılmıştır. Veritabanı olarak Firebase veritabanı (bulut tabanlı bir veritabanı) ve SQL Lite kullanıldı. Firebase veritabanında kullanıcı kimlik doğrulama, gerçek zamanlı veritabanı fonksiyon özelliği ve depolama (storage) özellikleri kullanıldı. SQL Lite veritabanı ise ürün kayıtları tutulması için kullanılacak. Program test edildiğinde kayıt olma, giriş yapma, liste görüntüleme, matematiksel hesaplama, veri gönderme, veri alma gibi işlemler test sonucunda başarıyla sonuçlandırılmıştır.

Uygulamanın hızlı, kısa bir sürede yapılmasından kaynaklanan tasarım ve kod düzeninin profesyonel olmaması projenin eksik yönleridir. Kişilerin kişisel bilgilerinin güvenliği için daha çok güvenlik tedbirleri alınabilir. Uygulama daha çok geliştirilebilir. Anlık olarak kullanıcılar arası toplanan veriler istatistiksel olarak belirli bir aralıkta popülasyonun veri kaynağı olabilir.



## KAYNAKLAR

- [1] Serkan Bentli 2018 yılı 518737 nolu tezinden alıntıdır.
- [2] <https://play.google.com/store/apps/details?id=com.yazio.android>
- [3] <https://play.google.com/store/apps/details?id=com.sillens.shapeupclub>
- [4] <https://play.google.com/store/apps/details?id=com.myfitnesspal.android>
- [5] <https://play.google.com/store/apps/details?id=com.vgfit.waterbalance>
- [6] <https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-201/android-cihazlar-ve-android-isletim-sistemi-uzerine-genel-bilgiler>
- [7] <https://flutter.dev/docs/resources/technical-overview>
- [8] <https://www.npmjs.com/package/firebase>
- [9] <https://www.sqlite.org/about.html>