

**T.C.
GEBZE TEKNİK ÜNİVERSİTESİ**

Bilgisayar Mühendisliği Bölümü

**KÖŞE YAZISI
UYGULAMASI PROJESİ
RAPORU**

Fatih Selim YAKAR

**Danışman
Doç. Dr. Habil KALKAN**

**Haziran, 2021
Gebze, KOCAELİ**

**T.C.
GEBZE TEKNİK ÜNİVERSİTESİ**

Bilgisayar Mühendisliği Bölümü

**KÖŞE YAZISI
UYGULAMASI PROJESİ
RAPORU**

Fatih Selim YAKAR

**Danışman
Doç. Dr. Habil KALKAN**

**Haziran, 2021
Gebze, KOCAELİ**

ÖNSÖZ

Bu projede emeği geçen ve beni her hafta yaptığı toplantılarıyla yönlendiren danışmanım Doç. Dr. Habil KALKAN hocama ve her izleme buluşmalarında beni dinleyen Prof. Dr. Erkan Zergeroğlu ve Dr. Öğr. Üyesi Alp Arslan BAYRAKÇI hocalarıma en içten teşekkürlerimi sunarım.

Ayrıca eğitimim süresince bana her konuda tam destek veren aileme ve bana bilgi katan, gelişmemde yardımcı olan tüm hocalarıma saygı ve sevgilerimi sunarım.

Haziran, 2021

Fatih Selim YAKAR

İÇİNDEKİLER

ÖNSÖZ	III
İÇİNDEKİLER	IV
ŞEKİL LİSTESİ	VII
TABLO LİSTESİ	IX
KISALTMA LİSTESİ	X
ÖZET	11
SUMMARY	13
1. GİRİŞ	15
2. YÖNTEM VE MALZEME	17
2.1. KULLANILAN PROGRAMLAR VE UYGULAMADAKİ İŞLEVLERİ	18
2.1.1. Android Studio	18
2.1.1.1. Model	18
2.1.1.2. Görünüm	18
2.1.1.3. Denetleyici	19
2.1.2. Flutter	19
2.1.3. SQLite	19
2.1.4. Flask	20
2.1.5. Heroku	20
2.1.6. Adobe XD	21
2.2. TASARIM VE ARAYÜZLER	21
2.2.1. Animasyonlu Giriş Ekranı	22
2.2.2. Ana Sayfa	23
2.2.3. Köşe Yazılarım Sayfası	24
2.2.4. Konuya Göre Ara Sayfası	26
2.2.5. Bilgiler Sayfası	29
2.2.6. Ayarlar Sayfası	29
2.3. VERİ TABANI	33

2.3.1. Yazar Tablosu (author_table)	33
2.3.2. Ayar Tablosu (settings_table)	33
2.3.3. Yazı Tablosu (news_table)	34
2.3.4. Filtrelenebilir Yazı Tablosu (news_table_filter)	34
2.3.5. Filtreleme Günü Tablosu (filter_day_table)	34
2.4. UYGULAMA İÇİNDE ÇALIŞAN İŞLEVLER	35
2.4.1. Android Studio İşlevleri	35
2.4.1.1. Activity	36
2.4.1.1.1. AuthorSettingsActivity.java	36
2.4.1.1.2. FilterableNewsActivity.java	37
2.4.1.1.3. InformationActivity.java	37
2.4.1.1.4. MainActivity.java	37
2.4.1.1.5. MainSettingsActivity.java	38
2.4.1.1.6. NewsActivity.java	38
2.4.1.1.7. NewsDetailActivity.java	38
2.4.1.1.8. SplashScreenActivity.java	39
2.4.1.1.9. UpdateSettingsActivity.java	39
2.4.1.2. Adapter	39
2.4.1.3. Database	39
2.4.1.4. Helper	40
2.4.1.5. Holder	40
2.4.1.6. Model	40
2.4.1.7. Receiver	41
2.4.1.8. Service	41
2.4.2. Flutter İşlevleri	41
2.4.2.1. AuthorSetting.dart	42
2.4.2.2. Informations.dart	42
2.4.2.3. Main.dart	42
2.4.2.4. Models.dart	43

2.4.2.5.	<i>NewsRequest.dart</i>	43
2.4.2.6.	<i>ReadNews.dart</i>	44
2.4.2.7.	<i>ReadNewsDetail.dart</i>	44
2.4.2.8.	<i>ReadNewsFilterable.dart</i>	44
2.4.2.9.	<i>Settings.dart</i>	45
2.4.2.10.	<i>SqliteHelper.dart</i>	45
2.4.2.11.	<i>UpdateSettings.dart</i>	45
2.5. API KURULUMU VE WEB'DE YAYILMASI		46
2.5.1. API Kurulumu		46
2.5.2. API'ın Web'de Yayılması		46
3. BULGULAR		49
3.1. CİHAZ TESTLERİ		49
3.1.1. Android Studio Tarafındaki Testler		49
3.1.1.1. <i>Pixel 3a (Emülatör)</i>		49
3.1.1.2. <i>Nexus 4 (Emülatör)</i>		50
3.1.1.3. <i>sdk_gphone_arm64 (Geliştirme Ortamı)</i>		51
3.1.1.4. <i>Sony Xperia X (Gerçek Cihaz)</i>		52
3.1.1.5. <i>Samsung Galaxy S8 (Gerçek Cihaz)</i>		53
3.1.1.6. <i>Huawei Mate 20 lite (Gerçek Cihaz)</i>		54
3.1.2. Flutter Tarafındaki Testler		55
3.1.2.1. <i>Apple iPhone 8 (Emülatör)</i>		55
3.1.2.2. <i>Apple iPhone 12 Pro (Emülatör)</i>		56
3.1.2.3. <i>Apple iPhone 11 (Gerçek Cihaz)</i>		57
3.2. KULLANICI TESTLERİ		58
4. TARTIŞMA VE SONUÇ		60
5. KAYNAKLAR		61
6. EKLER		63

ŞEKİL LİSTESİ

- ŞEKİL 2.1. Uygulamanın İlk Tasarım Planı
- ŞEKİL 2.2. Projenin İlk Arayüzü
- ŞEKİL 2.3. Uygulamanın Akış Şeması
- ŞEKİL 2.4. Giriş Ekranı Arka Planı
- ŞEKİL 2.5. Ana Sayfa Android Studio
- ŞEKİL 2.6. Ana Sayfa Flutter
- ŞEKİL 2.7. Köşe Yazılarım Android Studio
- ŞEKİL 2.8. Köşe Yazısı Android Studio
- ŞEKİL 2.9. Köşe Yazılarım Flutter
- ŞEKİL 2.10. Köşe Yazısı Flutter
- ŞEKİL 2.11. Köşe Yazısı Ara Android Studio
- ŞEKİL 2.12. Köşe Yazısı Android Studio
- ŞEKİL 2.13. Köşe Yazısı Ara Flutter
- ŞEKİL 2.14. Köşe Yazısı Flutter
- ŞEKİL 2.15. Sesli Dinle Android Studio
- ŞEKİL 2.16. Sesli Dinle Flutter
- ŞEKİL 2.17. Bilgiler Android Studio
- ŞEKİL 2.18. Bilgiler Flutter
- ŞEKİL 2.19. Ayarlar Android Studio
- ŞEKİL 2.20. Ayarlar Flutter
- ŞEKİL 2.21. Güncelleme Ayarları Android Studio
- ŞEKİL 2.22. Güncelleme Ayarları Flutter
- ŞEKİL 2.23. Günlük Güncelleme Saat Diyalogu AS
- ŞEKİL 2.24. Günlük Güncelleme Saat Diyalogu F
- ŞEKİL 2.25. Yazarlar Android Studio

ŞEKİL 2.26. Yazarlar Flutter

ŞEKİL 2.27. Yazar Ekleme Android Studio

ŞEKİL 2.28. Yazar Ekleme Flutter

ŞEKİL 2.29. Veri Tabanı ER diyagramı

ŞEKİL 2.30. AS Dosyalama Sistemi

ŞEKİL 2.31. Flutter Sınıfları

ŞEKİL 2.32. Örnek JSON Dönütü

ŞEKİL 2.33. Procfile İçeriği

ŞEKİL 2.34. Requirements.txt İçeriği

ŞEKİL 2.35. Runtime..txt İçeriği

ŞEKİL 2.36. Heroku'da Yayınlanan API'in Seyir Defteri (Log)

ŞEKİL 3.1. Pixel 3a Emülatöründeki Çalışma

ŞEKİL 3.2. Nexus 4 Emülatöründeki Çalışma

ŞEKİL 3.3. sdk_gphone_arm64 Emülatöründeki Çalışma

ŞEKİL 3.4. Xperia X Cihazındaki Çalışma

ŞEKİL 3.5. Samsung Galaxy S8 Cihazındaki Çalışma

ŞEKİL 3.6. Huawei Mate 20 Lite Cihazındaki Çalışma

ŞEKİL 3.7. iPhone 8 Simülatöründeki Çalışma

ŞEKİL 3.8. iPhone 12 Pro Simülatöründeki Çalışma

ŞEKİL 3.9. iPhone 11 Cihazındaki Çalışma

ŞEKİL 3.10. Kullanıcı Testi Bar Grafiği

TABLO LİSTESİ

TABLO 3.1. Kullanıcı Testi Tablosu

KISALTMA LİSTESİ

TTS	: Yazıdan Sese Çeviri (Text-to-Speech)
iOS	: Iphone İşletim Sistemi (Iphone Operating System)
API	: Uygulama Programlama Arayüzü (Application Programming Interface)
IDE	: Tümleşik Geliştirme Ortamı (Integrated Development Environment)
OOP	: Nesne Yönelimli Programlama (Object Oriented Programming)
UI	: Kullanıcı Arayüzü (User Interface)
MVC	: Model View Controller Pattern
PK	: Primary Key
ID	: Identification Number
HTTP	: Hyper-Text Transfer Protocol (Hiper-Metin Transfer Protokolü)
HTML	: Hypertext Markup Language (Hiper Metin İşaretleme Dili)

ÖZET

Mobil uygulamalar geliştirici dünyasında çok önemli rol oynamaktadır ve sürekli gelişmektedir. Bu projede bu gelişime ve yeni teknolojilere adapte olmuş olan bir mobil uygulama oluşturmak üzere kurgulandı. Bu bağlamda projedeki mobil uygulama haber uygulamaları ana başlığı içindeki köşe yazıları alt başlığı içinde bir işlev sahip olacak şekilde bir geliştirme yapıldı. Kullanıcının sağlanan yazarlar arasından seçim yapabilmesi ve istediği yazarların günlük köşe yazısını okuyup dinleyebilmesi amacıyla bir uygulama planlandı.

Bu projedeki asıl ve özgün olan amaç uygulamanın sadece gerçekten gerekli işlevleri için ağ erişimi kullanmak ve diğer ana işlevleri ise ağ erişimi olmadanda sağlayabilmektir. Bu bağlamda uygulamada günlük güncelleme ve elle güncelleme kısımları özelinde gerçekten ağ erişim gerektiği için sadece bu kısımlarda sınırlayıcılık sağlandı. Diğer işlevler olan köşe yazısı okuma, yazarların resminin tutulması, sesli dinleme, yazarlar arasında arama, yazıları belirli gün dahilinde kaydetme, yazar ekleme ve çıkarma işlevleri için ağ erişimi sınırlayıcılığı kaldırıldı. Ayrıca tüm bu işlevler hem Android mobil cihazlar hem de iOS mobil cihazlar için sağlanacak şekilde geliştirme yapıldı. Bu bahsedilen özellikleri sağlamak amacıyla geliştirme aracı olarak Flutter ve Android Studio kullanılırken, veri tabanı sistemi olarak SQLite, sesli dinleme için Google TTS ve iOS öntanımlı TTS sistemi, Arka planda güncelleme için Android tarafında AlarmManager Flutter tarafında ise isolates kullanıldı. Yazıların ayrıştırılması için Python Flask kullanarak bir API yazıldı ve heroku üzerinden web'de dağıtıımı sağlandı.

Sonuç olarak kullanıcı ve cihaz testlerine göre uygulamaların düzenlemeleri yapıldı. Kullanıcı testleri istatistik bazında grafiğe döküldü. Uygulama ise yapılan geliştirmeler sonucu şu işlevleri yerine getirir hale geldi:

Bu uygulama köşe yazarlarının efektif biçimde takip edilmesi için geliştirildi. Uygulamanın bilgiler kısmında uygulama ile alakalı kullanım bilgileri ve notlar okunabilir hale geldi. Ayarlar kısmında ise iki seçenek sunuldu. Bu seçeneklerden ilki günlük güncelleme özelliği, bu özellik açıldığında uygulama arka planda kullanıcının girdiği saatte olacak şekilde günde bir defa her gün internet olduğu taktirde takip edilen köşe yazarlarının en son yazısını günceller hale geldi. Yine günlük güncellemenin altında konu araması yapılmak için tutulan köşe yazılarının tutulma sınırı da gün sayısı bazında ayarlanabilir olarak eklendi. Ayarlar kısmındaki ikinci seçenek ise takip edilen köşe yazarlarını eklemek veya silmek için geliştirildi, bu kısımda takip edilmek istenen köşe yazarları birçok farklı yazar içinden gazete veya isim filtrelemesi yapılarak seçilebilir hale geldi. Köşe Yazılarım kısmında ise takip edilen köşe yazarlarının en güncel yazıları okunur ve dinlenilebilir hale geldi. Üstüne basıldığı taktirde tüm yazı okunulabiliyor ve eğer istenirse sadece geçerli yazı dinlenilebiliyordu. Köşe Yazısı Ara kısmında ise kullanıcının kendi belirlediği aralık boyunca takip edip güncellediği yazıların tümü yer alacak şekilde uygulandı. Ayrıca istenildiği taktirde tüm bu yazılar arasında başlığa, içeriğe veya yazara göre arama yapıp üstüne tıklayıp okuma veya dinleme yapabilir şekilde kurgulandı. Ana sayfanın sağ üstündeki butondan ise kullanıcıya elle güncelleme yapabilme seçeneği eklendi.

SUMMARY

Mobile applications play a very important role in the developer world and are constantly evolving. In this project, it was designed to create a mobile application that has adapted to this development and new technologies. In this context, the mobile application in the project has been developed in such a way that it has a function in the subheading of the columns in the main heading of news applications. An application has been planned so that the user can choose among the provided authors and read and listen to the daily columns of the authors they want.

The main and original purpose of this project is to use network access only for the really necessary functions of the application and to provide other main functions without network access. In this context, since network access is really required for daily update and manual update parts, only these parts are limited. The network access limitation has been removed for other functions such as reading columns, keeping pictures of authors, listening to audio, searching between articles, saving articles within a certain day, adding and removing authors. In addition, all these functions have been developed to be available for both Android mobile devices and iOS mobile devices. To provide these features, Flutter and Android Studio were used as development tools, SQLite was used as database system, Google TTS and iOS default TTS system for voice listening, AlarmManager on Android side and isolates on Flutter for background update. An API was written using Python Flask for parsing the articles and it was deployed on the web via Heroku.

As a result, the applications were arranged according to the user and device tests. User tests were graphed on the basis of statistics. As a result of the improvements made, the application has become to perform the following functions:

This application was developed to effectively follow columnists. In the information section of the application, usage information and notes related to the application have become readable. In the settings section, two options are presented. The first of these options is the daily update feature, and when this feature is turned on, the application updates the last article of the columnists followed once a day, if the internet is available every day, at the time the user enters in the background. Again, under the daily update, the retention limit of the columns kept for searching the subject was added as adjustable on the basis of the number of days. The second option in the settings section has been developed to add or delete the columnists that are followed. In this section, the columnists to be followed can be selected from many different authors by filtering newspapers or names. In the My Columns section, the most up-to-date articles of the columnists followed became readable and listenable. If it was pressed, the entire text could be read and only the current text could be listened if desired. In the Search Columns section, it was applied to include all the articles that the user followed and updated during the interval determined by himself. In addition, if desired, it has been designed in such a way that you can search by title, content or author among all these articles and click on them to read or listen. The option to manually update the user from the button on the top right of the home page has been added.

1. GİRİŞ

Günümüzde haber okumak veya köşe yazısı okumak gündemi takip etmek adına normalleşmiş ve günlük hayatı yaygın olan bir olaydır. Fakat zaman zaman tüm bu takip edilen yazarların teker teker farklı haber sitelerinde her gün aranması durumu kullanıcı için verimsizlik ve zaman kaybı getirir. Ayrıca araba gibi yerlerde veya internet erişimi bulunmayan yeraltı toplu ulaşım alanlarında bu yazıları okumak veya dinlemek mümkün olmayabilir. İşte bu problemlerin çözümü için tüm bu verimsizliği ve zaman kaybını önleyebilecek bir uygulama geliştirilmesi amaçlandı. Bu bağlamda yapılan projede köşe yazılarının takibini ve çeşitli işlevlerin gerçekleştirilmesi için bazı yöntemler ve implementasyonlar yer alacaktır.

Bütün bu anlatılanlar eşliğinde ele alınan problemi çözen bir uygulamayı oluşturmanın en iyi yolu öncelikle kapsamlı bir araştırma yapmaktan geçiyordu. Bu alandaki yapılanları ve yapılabilecekleri anlamak adına çeşitli platformlar üzerinde araştırmalar yapıldı. [1][2][3][4][5][6][7][8][9][10][11][12][13]

Bağlam hakkında yeterli bilgi sağlandıktan sonra yapılacak uygulamayı iskelet olarak belirlemek ve benzer işleri yapan uygulamalar üzerinde araştırmalar yapıldı. Buradan elde edilen bilgi birikimi ve çeşitli bilgiler dahilinde yeni bir tasarım yapıldı.

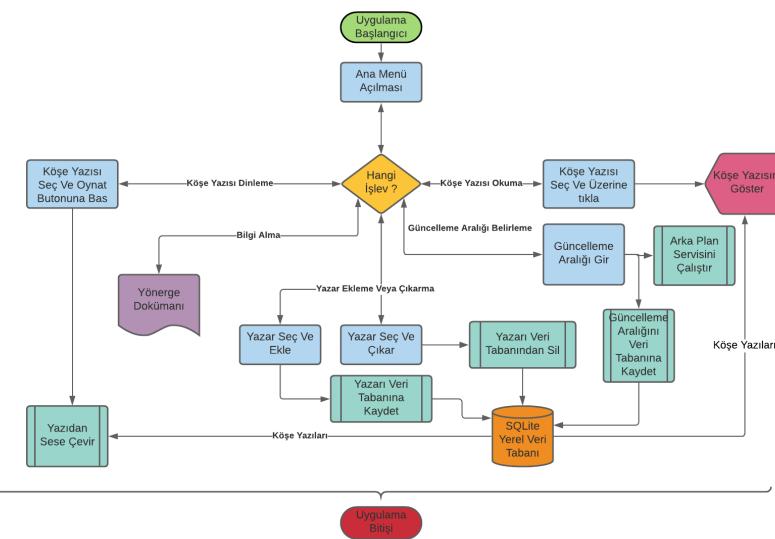
Yapılan ilk tasarımda ana amaçlar yer alıyordu. Yani aslında uygulamanın kullanıcı kitlesinin son derece geniş bir alanı kapsaması için basit, anlaşılır ve işlevsel bir kullanım senaryosu hazırlandı. Bu senaryoda ana ekranda açık ve büyük şekilde ana işlevler olacak ve ana işlevlere tıklandığında alt işlevler seçili kolayca kullanılabilirdi. Fakat daha önceki satırlarda bahsedilen tasarım anlayışı sürekli korundu. Bu bağlamda son olarak Uygulama dört ana işlevde bölündü. Bu ana işlevler; Köşe Yazılarım, Konuya Göre Ara, Ayarlar, Bilgiler olarak ayarlandı. Kullanıcıya Köşe Yazılarım kısmında istediği yazıyı

seçip okuyabilecek veya dinleyebilecek şekilde bir arayüz sağlandı. Konuya Göre Ara kısmında yine kullanıcının belirlediği gün sınırı dahilinde tutulan yazıların arasında içeriğe, başlığa, yazara, gazeteye görefiltrelenebilecek bir arama bölümü sağlandı. Böylece kullanıcı buradan filtreleme sonucu aradığı yazıyı dinleyebilir veya okuyabilir hale geldi. Ayarlar kısmında uygulamanın değişimlerini ayarlarına yer verildi. Burada günlük güncelleme özelliği ve filtreleme aralığı ile ilgili kısım bulunurken ayrıca takip edilen yazarların seçilmesi gibi kısımlarda sağlandı. Bilgiler kısmında ise kullanıcının uygulamayı nasıl kullanacağı ve uygulama ile ilgili notlar yer aldı.

Rapor tüm yukarıda bahsedilen ve kısaca bilgi verilen konular detaylıca açıklanması, yanı sırasıyla kullanılan programların listelenmesi ve uygulamada hangi amaç için kullanıldığı, tasarım ve arayüzlerin açıklanması, veri tabanının açıklanması, uygulama içinde çalışan işlevler, API kurulumu ve web' de yayılması, yapılan cihaz ve kullanıcı testleri şekilde devam edecektir.

2. YÖNTEM VE MALZEME

Bu projede öncelikle genel bir plan belirlendi. Öncelikle uygulamada kabaca kullanılacak bir tasarım belirlendi ve işlevleri için kabaca bir tasarım planı çıkarıldı. Bu tasarım planı doğrultusunda arayüz çalışmaları yapılarak projeye başlandı. Aşağıda projenin ilk tasarım planı ve arayüzü görülmektedir. Ardından bu tasarım planı ve arayüz geliştirildi.



Şekil 2.1. Uygulamanın İlk Tasarım Planı



Şekil 2.2. Projenin İlk Arayüzü

2.1. KULLANILAN PROGRAMLAR VE UYGULAMADAKİ İŞLEVLERİ

Projedeki tasarım planı yapıldıktan sonra benzer uygulamaların hangi geliştirme araçlarını ve geliştirme özellikleri kullanıldığı araştırıldı.[REFERANS EKLE]

2.1.1. Android Studio

Android Studio Google ve JetBrains'in geliştiriciliğini yaptığı IntelliJ üzerine kurulmuş bir tümleşik geliştirme ortamı yani IDE'dir. Bu ortamda genel olarak Android işletim sistemine sahip cihazlarda çalışabilen uygulamalar yapılır. Java ve Kotlin programlama dilini destekler.

Bu projedede Android Studio programı köşe yazısı okuma uygulamasını geliştirmek için geliştirme ortamı olarak kullanıldı. Java veya Kotlin arasından daha geniş kaynaklar içерdiği düşünüldüğü için Java programlama dili seçildi. Programlama prensibi olarak ise OOP yani Nesne Yönelimli Programlama prensibi kullanıldı. Programlama modeli olarak ise Android Studio IDE'sinin varsayılan model anlayışı olan MVC modeli kullanıldı. Bu modelde ise genel anlayışlar şu şekilde yönetildi.

2.1.1.1. Model

Bu bileşen, uygulama verilerini depolar. Arayüz hakkında bilgisi yok. Model, etki alanı mantığını işlemekten ve veri tabanı ve ağ katmanlarıyla iletişimden sorumludur. Bu kısım uygulamada Java dosyaları tarafından sağlandı

2.1.1.2. Görünüm

Ekranda görünen bileşenleri tutan UI(Kullanıcı Arayüzü) katmanıdır. Ayrıca Modelde depolanan verilerin görselleştirilmesini sağlar ve kullanıcıya etkileşim sunar. Bu kısım uygulamada .xml dosyaları ile sağlandı.

2.1.1.3. *Denetleyici*

Bu bileşen, Görünüm ve Model arasındaki ilişkiyi kurar. Temel uygulama mantığını içerir ve kullanıcının davranışından haberdar olur ve Modeli ihtiyaca göre günceller. Bu kısım ise Java dosyaları içindeki adaptörlerle sağlandı.

2.1.2. Flutter

Flutter, Google tarafından oluşturulan açık kaynaklı bir UI yazılım geliştirme kitidir. Bu geliştirme kitinin en önemli özelliği birçok platformda çıktı verebilmesidir. Android, iOS, Google Fuchsia, Web, Windows, macOS ve Linux gibi platformlara çıktı verebilir. Dart programlama diliyle yazılmıştır ve Android Studiodaki varsayılan olarak gelen MVC anlayışının aksine bir tek bir uygulama ile hem görünüm hem model hem denetleyici görevi üstlenilir.

Bu projede de Flutter kiti, köşe yazısı okuma uygulamasını çoklu platformda geliştirmek için kullanıldı. Dart programlama diliyle kullanıldı. Programlama prensibi olarak ise OOP yani Nesne Yönelimli Programlama prensibi kullanıldı.

2.1.3. SQLite

SQLite, bağımsız, yüksek güvenilirliğe sahip, gömülü, tam özellikli, herkese açık bir SQL veritabanı motorudur. Herhangi bir amaç için, ticari veya özel kullanım için ücretsizdir. Sıradan disk dosyaları SQL gibi ayrı bir sunucuya sahip olmadığı için SQLite tarafından kolayca okunup yazılabilir. SQLite veritabanı dosya formatı çok platformludur, böylece herkes tarafından bir veritabanını 32-bit ve 64-bit sistemler arasında kullanabilir. Tüm bu özellikleri nedeniyle Uygulama Dosya Formatı olarak çokça kullanılan bir seçimdir

Bu projede veri tabanı motoru olarak ,uygulamanın ana amacı internet olmadan da birçok işlevi desteklemesi gerektiği için, yerel bir alanda veri tabanının tutulması gerekiyordu. Bu bağlamda bazı araştırmalar yapıldı. Bu araştırmalar sonucu çok kullanıldığı için, diğer programlar tarafından desteği ve kaynak tarafından da zengin olduğu dolayısıyla SQLite seçildi. Hem Android Studio tarafından hem de Flutter tarafından yardımcı sınıflarla SQLite ile veri tabanına erişim sağlandı.

2.1.4. Flask

Flask, Python ile yazılmış bir mikro web framework’üdür. Belirli araçlar veya kitaplıklar gerektirmeden mikro iskelet olarak adlandırılır. Veri tabanı soyutlama katmanı, form doğrulama veya önceden var olan üçüncü taraf kütüphanelerinin ortak işlevler sağladığı diğer bileşenlere sahip değildir. Ancak Flask, kendine uygulama özellikleri ekleyebilen uzantıları destekler. Nesne-ilişkisel eşleyiciler, form doğrulama, karşıya yükleme işleme, çeşitli açık kimlik doğrulama teknolojileri ve çeşitli ortak çerçeve ile ilgili araçlar için uzantılar mevcuttur.

Bu projede Flask yazıların gazete üzerindeki web adreslerindeki html dönütlerini ayırtmak üzere bir API yazmak için kullanılmıştır. Böylece sorgu olarak atılan web adresinin dönütünü ayrıstırılmış bir json dosyası şeklinde alıp ardından uygulamadaki gerekli kısımda gösterilmesi sağlandı.

2.1.5. Heroku

Heroku birçok programlama dili tarafından geliştirilen uygulama ya da programın yayılmasını sağlayan bir web bulut platformudur. Heroku tarafından yayılan bir uygulama kendine özel bir web adresine sahip olur ve bu adresden kullanılabilir.

Bu projede Heroku Flask ile yazılan API kodunun yerel bir API olarak değil de web üzerinde çalışması için kullanıldı. Böylece yazılan API ve dolayısıyla uygulama sadece ön gösterimi için değil gerçek hayat kullanımı için de kullanılabilir hale gelmiş oldu.

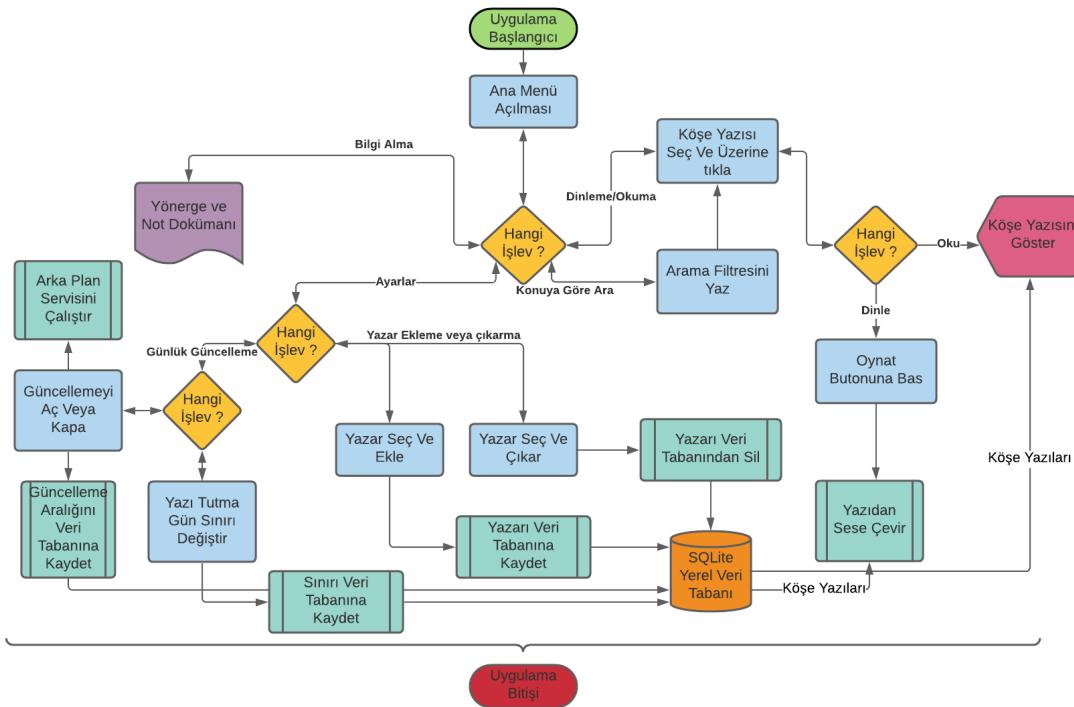
2.1.6. Adobe XD

Adobe XD (Adobe Experince Design) internet siteleri ya da uygulamalar için ux/ui arayüz tasarımları yapmak için kullanılan bir Adobe programıdır. Bu uygulama ile bir mobil uygulama veya internet uygulaması yaparken daha basit biçimde ön çalışması görülebilir.

Bu projede ise Adobe XD uygulamadaki ikonların, sayfaların ve sayfa arka planlarının tasarlanması için kullanıldı. Böylece daha göze hoş gelebilecek ve uğraşılmış dizaynlar yapıldı.

2.2. TASARIM VE ARAYÜZLER

Raporun yöntem kısmında da belirtildiği üzere öncelikle bir başlangıç tasarımları ve başlangıç arayüzü yapıldı. Fakat bu tasarım ve arayüz haftalar eşliğinde geribildirimler dahilinde daha da geliştirildi. Bu kısımda ise uygulamanın geldiği son hal açıklanacak ve anlatılacaktır. Aşağıda uygulamanın son halindeki işlevlerini gösteren bir akış şeması bulunmaktadır.



Şekil 1.3. Uygulamanın Akış Şeması

2.2.1. Animasyonlu Giriş Ekranı

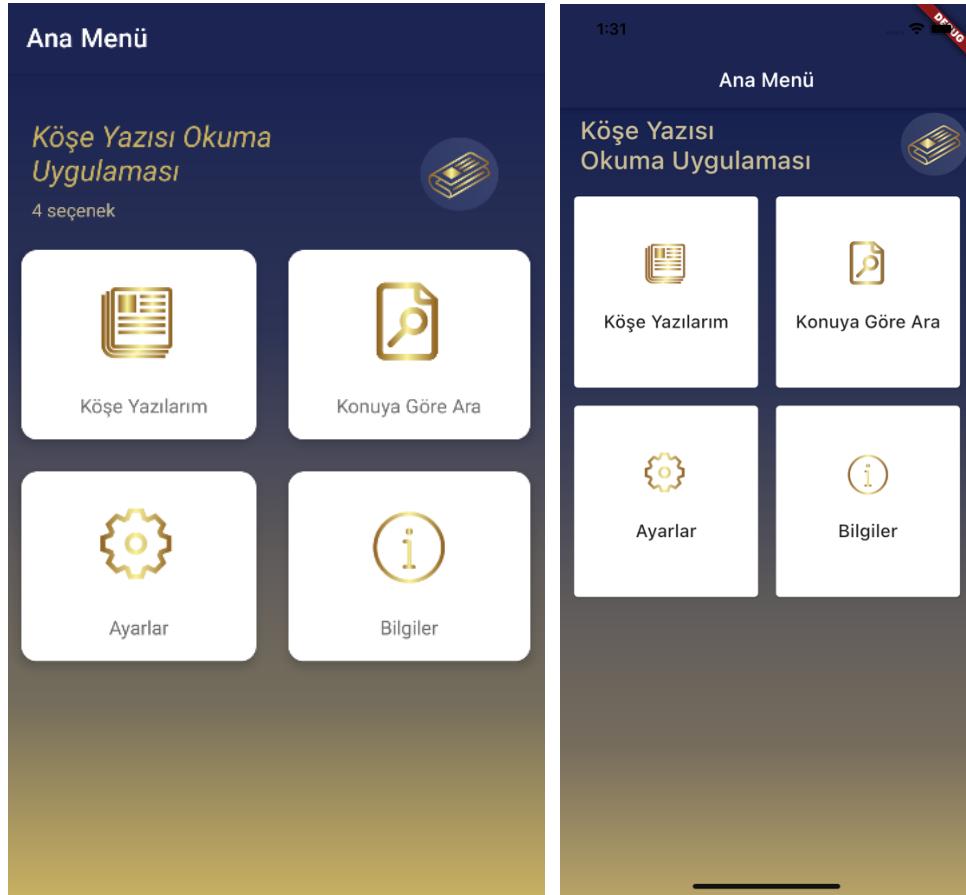
Uygulamanın Android Studio kısmında giriş yapılırken soldan sağa kayacak şekilde bir animasyonlu giriş ekranı hazırlandı. Bu ekran Adobe XD kullanılarak yapıldı. Bu ekranda içerik olarak uygulamanın simgesi olan bükülmüş gazete, uygulamanın ismi olan “kYOU” ve uygulamanın isminin açılımı olan “Köşe Yazısı Okuma Uygulaması” yazıları yer aldı. Böylece kullanıcının daha güzel ve ilgi çekici bir giriş yapılması sağlandı.



Şekil 2.4. Giriş Ekranı Arkaplani

2.2.2. Ana Sayfa

Uygulamanın Ana sayfa kısmında mümkün olduğunca basitliği ve anlaşılabilirliği korumak amaçlandı. Ayrıca kullanıcının ana işlevlerini kolayca yerine getirebilmesi sağlandı. Bu bağlamda Ana ekranın sol üst köşesine uygulamanın ismi sağ üst köşesine uygulamanın simgesini içeren bir elle güncelleme butonu ve alttaki dört karta ise üzerlerinde ilgili işlevin ismi ve ikonu bulunan işlevlere yönlendirecek basılabilir alanlar koyuldu. Bu kısımdaki arka plan da Adobe XD ile yapıldı.



Şekil 2.5. Ana Sayfa Android Studio

Şekil 2.6. Ana Sayfa Flutter

2.2.3. Köşe Yazılarım Sayfası

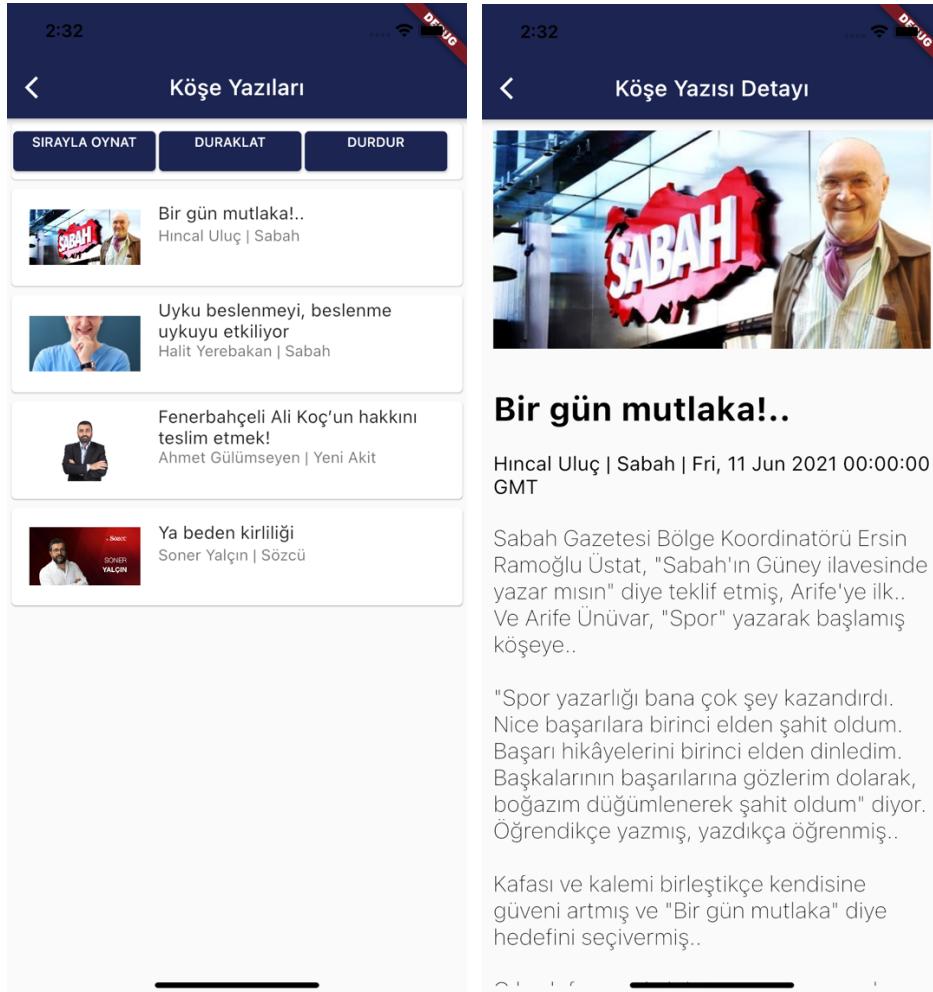
Uygulamanın köşe yazılarım sayfasında kullanıcının takip ettiği yazarların eğer internet varsa en güncel köşe yazıları listelenecek şekilde gösterimi sağlandı. Eğer internet yoksa en son güncellenen zamana ait olan köşe yazıları gösterildi. Her bir köşe yazısı için bir görsel, köşe yazısı başlığı ve altında ise yazarı ve gazetesi yazılacak şekilde liste elemanları yapıldı. En üstte ise yazıların tümünü dinlemek için butonlara yer verildi. Kullanıcı herhangi bir yazının üzerine tıkladığında ise en üstte görsel altında başlık, yayın tarihi, yazar ismi ve gazetesi yer alacak şekilde ayarlamalar yapıldı. Bu kısmın altında

yazılan köşe yazısı, linki ve en altta ise özel olarak tıklanmış olan yazının dinlemesinin yapılabileceği butonlara yer verildi.



Şekil 2.7. Köşe Yazılarım Android Studio

Şekil 2.8. Köşe Yazısı Android Studio



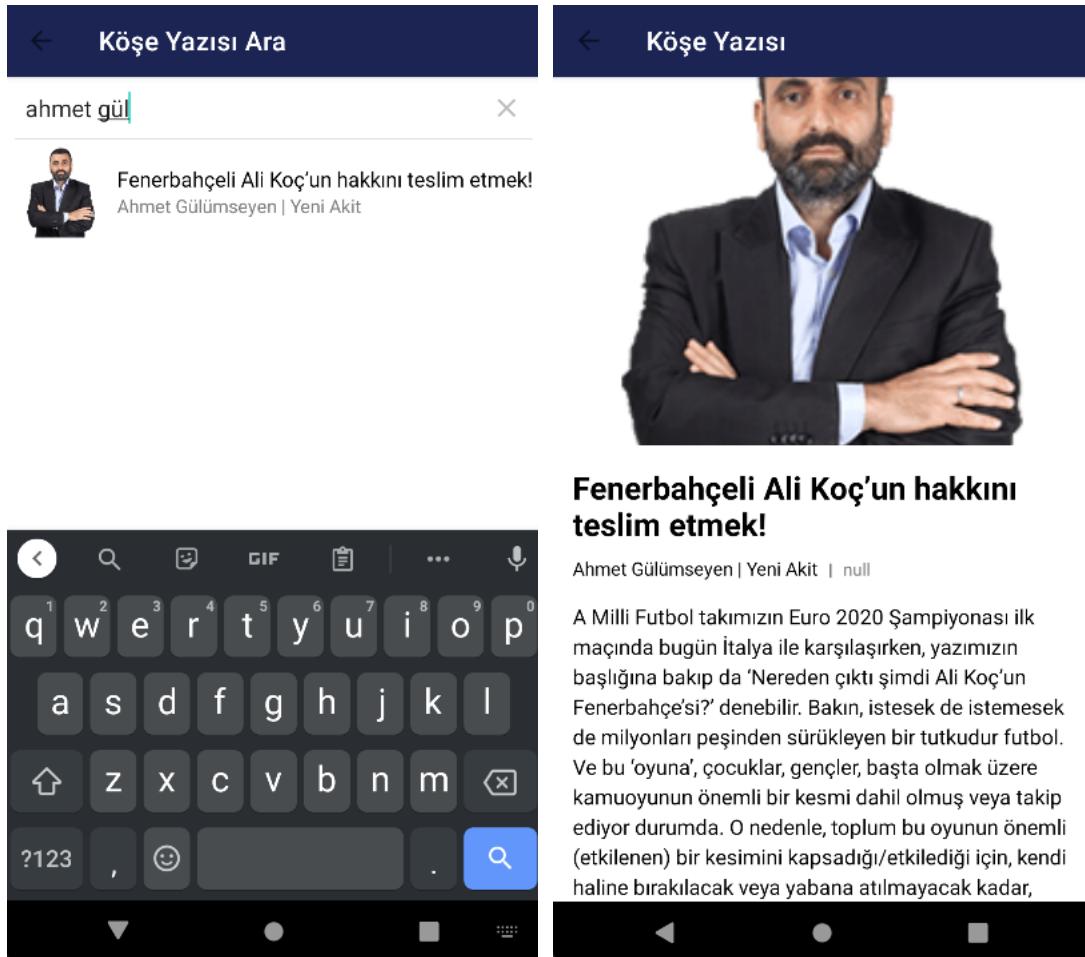
Şekil 2.9. Köşe Yazılarım Flutter

Şekil 2.10. Köşe Yazısı Flutter

2.2.4. Konuya Göre Ara Sayfası

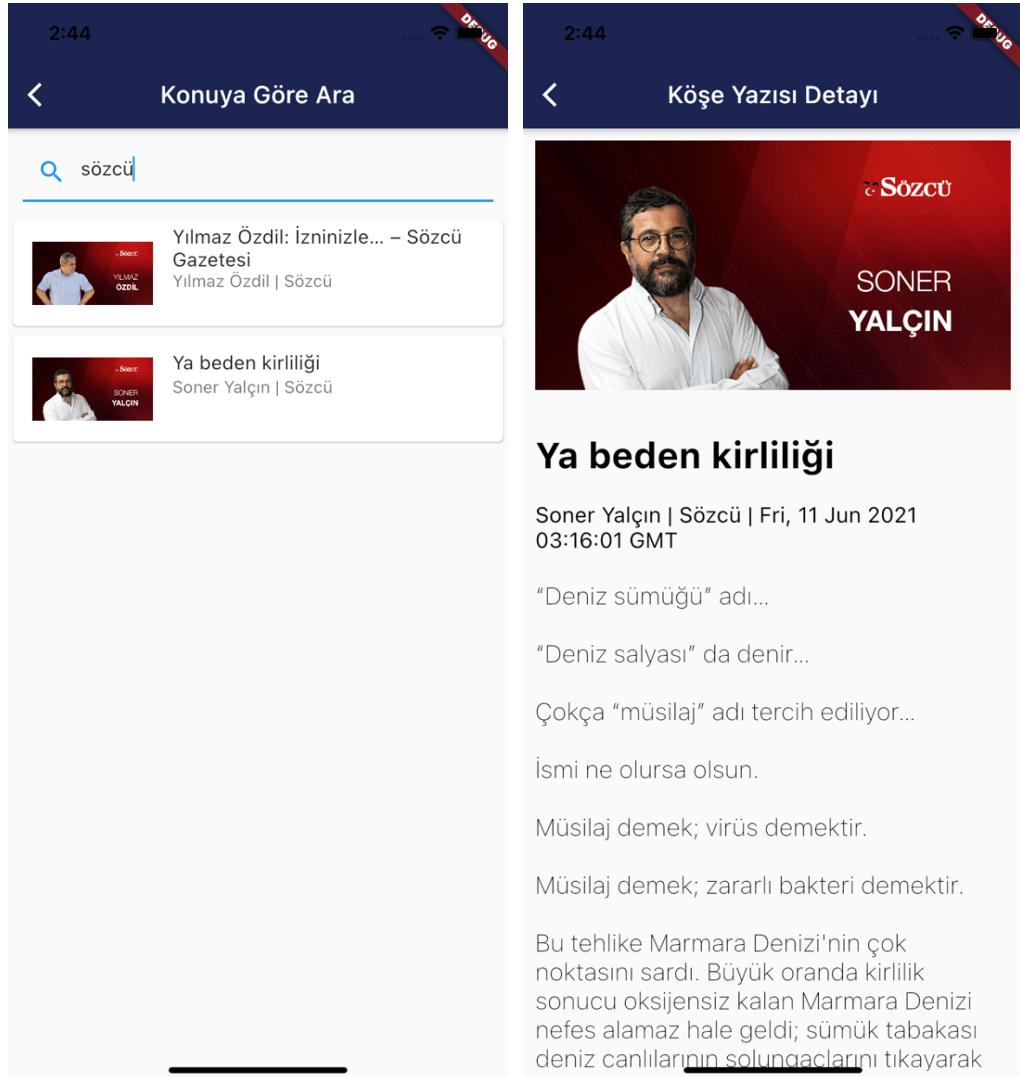
Uygulamanın konuya göre ara sayfasında kullanıcının belirdiği aralıktaki (eğer belirlemediyse 7 gün) daha önceden güncellenmiş olan yazılar kaydedilecek şekilde ayarlandı. En üste yazılabilir bir arama kısmı eklendi. Bu kısımda köşe yazıları; başlığı, yazarı, içeriği, gazetesi dahilinde arama yapılabilir hale getirildi. Her bir köşe yazısı için bir görsel, köşe yazısı başlığı ve altında ise yazarı ve gazetesi yazılacak şekilde liste elemanları yapıldı. Kullanıcı herhangi bir yazının üzerine tıkladığında ise en üstte görsel

altında başlık, yayın tarihi, yazar ismi ve gazetesi yer alacak şekilde ayarlamalar yapıldı. Bu kısmın altında yazılan köşe yazısı, linki ve en alta ise özel olarak tıklanmış olan yazının dinlemesinin yapılabileceği butonlara yer verildi.



Şekil 2.11. Köşe Yazısı Ara Android Studio

Şekil 2.12. Köşe Yazısı Android Studio

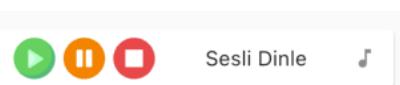


Şekil 2.13. Köşe Yazısı Ara Flutter

Şekil 2.14. Köşe Yazısı Flutter



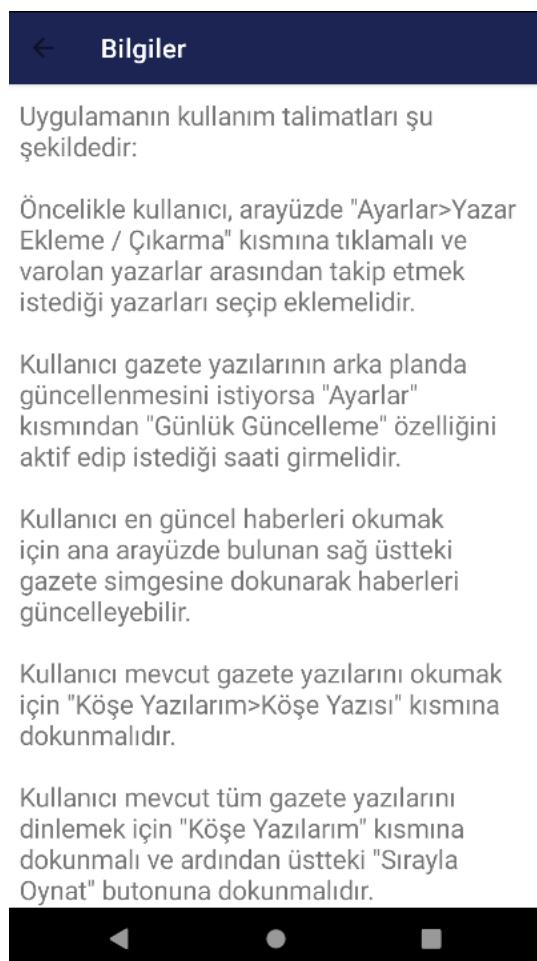
Şekil 2.15. Sesli Dinle Android Studio



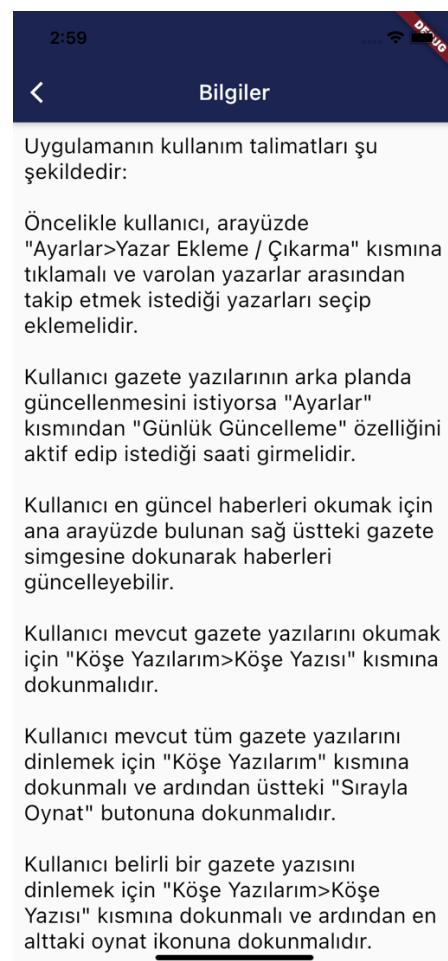
Şekil 2.16. Sesli Dinle Flutter

2.2.5. Bilgiler Sayfası

Uygulamanın bilgiler sayfasında uygulamayı tam verimlilikte kullanmak için gereken bilgilere yer verildi. Ayrıca uygulamanın çalışmasını etkileyebilecek bazı notlar eklendi.



Şekil 2.17. Bilgiler Android Studio

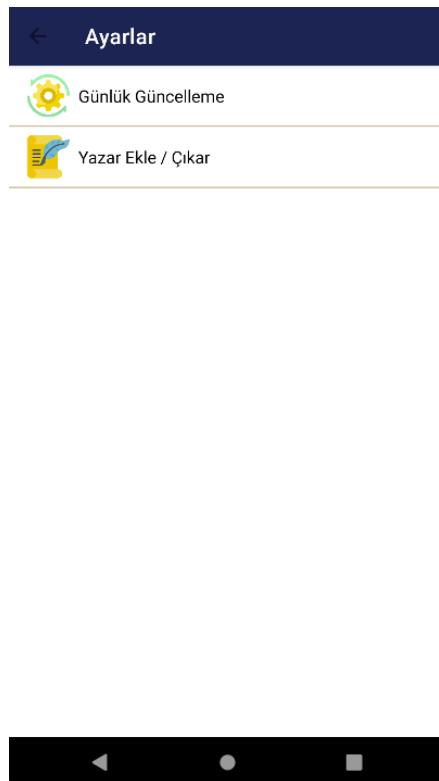


Şekil 2.18. Bilgiler Flutter

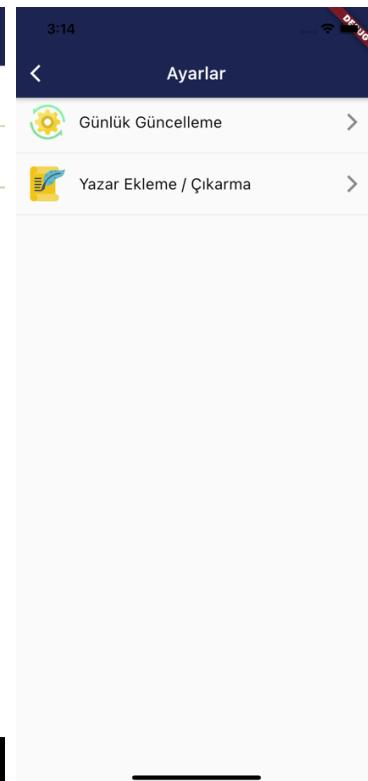
2.2.6. Ayarlar Sayfası

Uygulamanın ayarlar sayfasında liste halinde iki ana başlık halinde Günlük Güncelleme ve Yazar Ekle / Çıkar ayarlarına yer verildi. Ayarlar solda uygulamaya bağlı ikonlar,

sağlarında ayarın açıklayıcı ismi olacak şekilde alt alta konumlandırıldı. Ayrıca bu bahsedilen ayarlar altın rengi ayırıcı çizgiler ile ayrıldı. Günlük Güncelleme ayarı içinde günlük güncelleme özelliğinin açılması veya kapanmasını sağlayacak bir switch'e yer verildi. Bu switch açılırken bir saat belirleme diyalogu çıkacak şekilde ayarlamalar yapıldı. Böylece kullanıcı her gün köşe yazılarının güncelleneceği saati seçebilir hale geldi. Bahsedilen switch'in altında ise yazı tutma sınırının ayarlanabileceği bir buton ve seçiciye yer verildi. Bu kısımdan ise kullanıcı arama yapabileceği köşe yazılarının kaç gün tutulacağını seçebilir hale geldi. Ek olarak yazı tutma sınırının seçiminin yapıldığı seçici bu sayfanın açılışı sırasında hangi ayardaysa onu gösterecek şekilde ayarlandı. Bundan önceki sayfadaki Günlük Güncelleme ayarının altında ise Yazar Ekle / Çıkar kısmı konumlandırıldı. Bu ayarın içinde ise mevcut takip edilen yazarların gösterildiği, silinebildiği bir alan yapıldı. Bu alana ek olarak üst kısmda bir çoklu kutucuk eşliğinde takip edilen yazar eklemenin yapılabileceği bir kısım eklendi. Bu kısmda ayrıca yazarın adı ve gazetesine göre arama yapılır hale geldi.



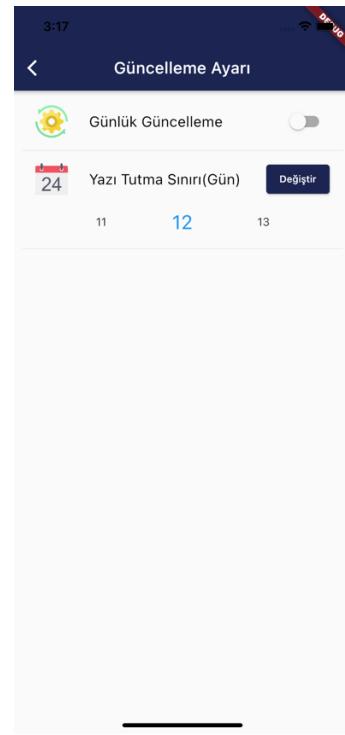
Şekil 2.19. Ayarlar Android Studio



Şekil 2.20. Ayarlar Flutter



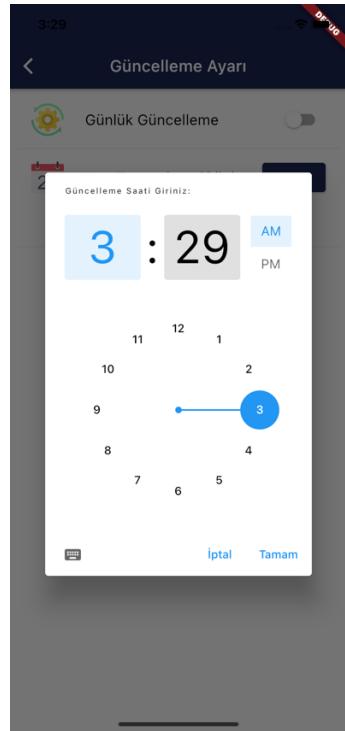
Şekil 2.21. Güncelleme Ayarları Android Studio



Şekil 2.22. Güncelleme Ayarları Flutter



Şekil 2.23. Günlük Güncelleme Saat Diyalogu AS



Şekil 2.24. Günlük Güncelleme Saat Diyalogu F



2.3. VERİ TABANI

Daha önce de bahsedildiği üzere veri tabanı yönetimi için SQLite veri tabanı motoru kullanıldı. Bu sistem uygulamadaki yerelliği sağlamak ve çok platformdaki desteği sağlamak adına çok önemli bir rol oynadı. Bu bağlamda veri tabanında hem Flutter' da hem Android Studio' da olacak şekilde verileri yerel olarak tutmak için beş farklı tablo oluşturuldu. Bu tablolar aşağıdaki şekildeydi.

2.3.1. Yazar Tablosu (author_table)

Yazar tablosunun içinde PK olarak bir ID ve author_name elemanları kullanıldı. Bu elemanlardan ilki yani ID çeşitli aramalarda ve işlemlerde benzersizlik kazanması için kullanıldı. İkinci eleman yani author_name ise kullanıcının çeşitli yazarlar arasından eklediği yazarın ismi için kullanıldı. Bu tablodaki ana amaç kullanıcının takip etmek istediği yazarların tutulmasıydı. Ayrıca bu tabloda kopya yazar isimlerinin önlenmesi sağlandı.

2.3.2. Ayar Tablosu (settings_table)

Ayar tablosunda kullanıcının haberlerinin kaç dakika sonra güncelleneceği tutuldu. Yine PK olarak ID kullanıldı ve bir diğer eleman olarak updateInterval kullanıldı. Hem uygulamanın gelişme aşamasında bu durumun değişebilir olabilmesi nedeniyle ayrı bir tablo ile sağlandı. Böylece pratikte tabloya ilk olarak kullanıcı tarafından seçilen güncelleme saatı ile şimdiki saatin arasındaki farkın dakika olarak karşılığı kaydedildi ve ilk güncelleme buna göre yapıldı. Ardından her gün aynı saatte güncelleneceği üzere 1440 dakika (1 gün) olarak ayarlandı. Ayrıca güncelleme zamanının 0 olduğu durumda da servis eğer çalışıyorsa kapanacak çalışmıyorsa durumunu koruyacak şekilde kurgulandı.

2.3.3. Yazı Tablosu (news_table)

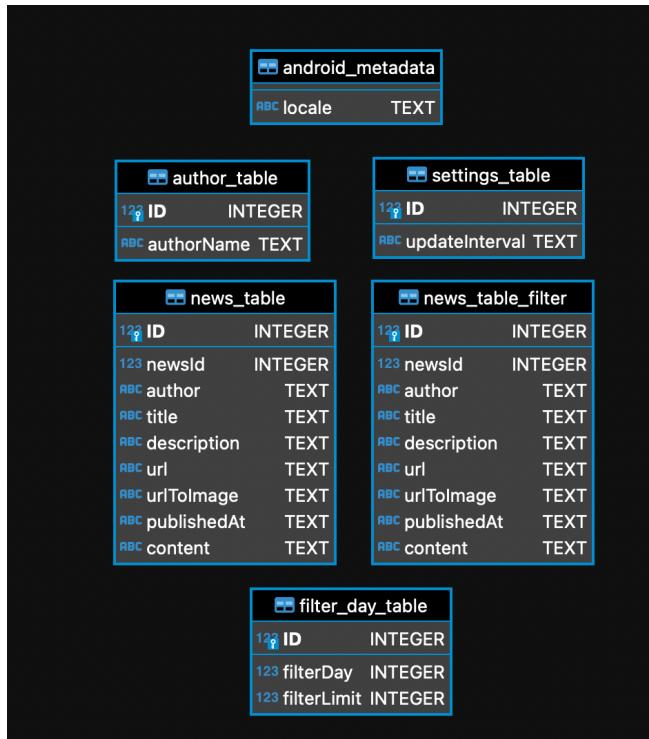
Bu tabloda uygulamanın ana işlevi olan haberlerin tutulması için kullanıldı. Burada aynı haber kopyalanamayacak şekilde bir düzen kuruldu. PK olarak yine ID kullanılırken diğer elemanlar olarak yazının bölümleri kullanıldı. Bahsedilen bu bölümler API' dan gelecek olan ayırtırılmış json dosyasına göre ayarlandı. Bunlar yazarın ismi, yazının başlığı, yazının açıklaması, yazının url linki, yazının görsel linki, yazının paylaşım zamanı ve yazının içeriğiydi.

2.3.4. Filtrelenebilir Yazı Tablosu (news_table_filter)

Bu tabloda uygulamanın ana işlevlerinden olan filtrelenebilir haberlerin tutulması için kullanıldı. Yazı tablosundan farklı bir tablo olmasının nedeni burada tutulacak yazıların günlük değil kullanıcının seçeceği belli bir gün aralığında saklanabilir olması gerekiydi. Burada aynı haber kopyalanamayacak şekilde bir düzen kuruldu. PK olarak yine ID kullanılırken diğer elemanlar olarak yazının bölümleri kullanıldı. Bahsedilen bu bölümler API' dan gelecek olan ayırtırılmış json dosyasına göre ayarlandı. Bunlar yazarın ismi, yazının başlığı, yazının açıklaması, yazının URL linki, yazının görsel linki, yazının paylaşım zamanı ve yazının içeriğiydi.

2.3.5. Filtreleme Günü Tablosu (filter_day_table)

Bu tabloda filrelenebilir haberler tutulma sınırı ve başlangıçtan itibaren kaçinci gün tutulduğu bilgileri sağlandı. PK olarak yine ID kullanılırken diğer elemanlar olarak filterDay mevcut yazıların kaç gündür saklandığı ve filterLimit yani yazıların limit olarak kaç gün saklanacağı yer aldı.



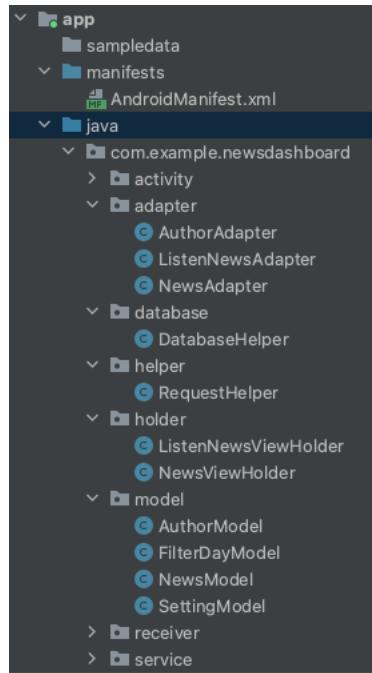
Şekil 2.29. Veri Tabanı ER diyagramı

2.4. UYGULAMA İÇİNDE ÇALIŞAN İŞLEVLER

Bu bölümde uygulamadaki modüller ve modüllerin işlevleri ele alındı. Bu işlevler Flutter ve Android Studio'da farklılık gösterebileceğinden dolayı iki ayrı başlıkta incelandı.

2.4.1. Android Studio İşlevleri

Android Studio kısmında işlevler OOP düzeninde yazıldığı için sınıflara bölünerek oluşturuldu. Bu sınıflarda kendi yaptığı işlemlere göre dosyalandı. Bu dosyalar sonucu activity, adapter, database, helper, holder, model, receiver, service gibi dosyalar oluşturuldu ve Java sınıfları bu dosyaların altına amacına uygun olacak şekilde yerleştirildi. Bu dosyalama aşağıda gösterildi.



Şekil 2.30. AS Dosyalama Sistemi

2.4.1.1. Activity

Dosyalamanın Activity kısmında Android Studio tarafından sağlanan MVC modeline uygun şekilde .xml dosyalarına bağlı olan işlevleri yazılacak şekilde bir programlama yapıldı. Bu activity'nin içeriği ise aşağıda açıklanmıştır.

2.4.1.1.1. AuthorSettingsActivity.java

Bu activity'de yazarların ekleme ve çıkarma kısımlarının işlevleri uygulandı. Bu işlevlerin görsel tanımı ise activity_author_settings.xml kısmında yapıldı. RelativeLayout içinde Toolbar, Searchable Spinner, TextView, RecyclerView gibi alt elemanlar kullanıldı. Buradaki ekleme veya çıkarma durumunda ise veri tabanında değişiklik yapılarak işlem sağlandı. Liste gösterimi ise veri tabanından yazar verisi alınarak sağlandı. RecyclerView içinde adaptörler ile işlevler sağlandı. Satırlardaki görsel

tanım için ise author_row.xml içinde Linear Layout içinde TextView, ImageView ve ImageButton kullanıldı.

2.4.1.1.2. FilterableNewsActivity.java

Bu activity’de yazarların yazılarının görüntülenmesi işlevi uygulandı. Buna ek olarak arama kısmı sağlandı. Bu işlevlerin görsel tanımı ise activity_list_filterable_news.xml kısmında yapıldı. RelativeLayout içinde Toolbar, SearchView ve RecyclerView gibi alt elemanlar kullanıldı. Bu kısımdaki yazı gösterimleri veri tabanından filtrelenebilir yazı verisi alınarak sağlandı. RecyclerView içinde adaptörler ile işlevler sağlandı. Satırlardaki görsel tanım için ise news_row.xml içinde LinearLayout, TextView ve ImageView kullanıldı.

2.4.1.1.3. InformationActivity.java

Bu activity’de kullanıcının uygulama hakkında alabileceği bilgiler ve notlar sağlandı. Bu işlevin görsel tanımı ise activity_information.xml kısmında yapıldı. RelativeLayout içinde Toolbar, ScrollView ve TextView kullanıldı.

2.4.1.1.4. MainActivity.java

Bu activity’de uygulamanın ana sayfası yapılandırıldı. Dört adet ana işlev dört adet basılabilir alana yerleştirildi. Böylece bu basılabilir alanlara dokunarak istenilen işleve geçiş imkanı sağlandı. Sağ üstte ise elle güncelleme için gazete ikonu yerleştirildi. Bu işlevin görsel tanımı ise activity_main.xml kısmında yapıldı. ScrollView içinde Toolbar, TextView, ImageButton, GridLayout, CardView ve ImageView kullanıldı.

2.4.1.1.5. MainSettingsActivity.java

Bu activity’de uygulamanın ana ayarlar sekmesi yapıldı. Üstte Güncellemeye Ayarları altta Yazar Ekleme / Çıkarma olacak şekilde ikiye bölündü. Bu işlevin görsel tanımı ise activity_main.xml kısmında yapıldı. RelativeLayout içinde ImageView, TextView ve View kullanıldı,

2.4.1.1.6. NewsActivity.java

Bu activity’de yazarların yazılarının görüntülenmesi işlevi uygulandı. En üstte yazıların dinlenilebilmesi için oynatma ve durdurma işlevlerine ve butonlarına yer verildi. Dinleme işlevi TextToSpeech tipinin yerel veri tabanında bulunan yazıları okuması ile sağlandı. Bu işlevlerin görsel tanımı ise activity_list_news.xml kısmında yapıldı. RelativeLayout içinde Button, Toolbar ve RecyclerView gibi alt elemanlar kullanıldı. Bu kısımdaki yazı gösterimleri veri tabanından yazı verisi alınarak sağlandı. RecyclerView içinde adaptörler ile işlevler sağlandı. Satırlardaki görsel tanım için ise news_row.xml içinde LinearLayout, TextView ve ImageView kullanıldı.

2.4.1.1.7. NewsDetailActivity.java

Bu activity’de üzerine tıklanan yazının detaylıca görüntülenmesi işlevi uygulandı. En üstte cached bir görsel altta yazının başlığı, yazarın ismi, gazetesi, yayınlanma tarihi ve yazının içeriğine yer verildi. En altta yazının dinlenilebilmesi için oynatma ve durdurma işlevlerine ve butonlarına yer verildi. Dinleme işlevi TextToSpeech tipinin yerel veri tabanında bulunan yazıları okuması ile sağlandı. Bu işlevlerin görsel tanımı ise activity_news_detail.xml kısmında yapıldı. RelativeLayout içinde ImageView, ImageButton, Toolbar ve RecyclerView gibi alt elemanlar kullanıldı. Bu kısımdaki yazı gösterimi veri tabanından yazı verisi aktarılarak sağlandı.

2.4.1.1.8. SplashScreenActivity.java

Bu activity’de uygulamanın başlangıcında kayarak gelen ekran ve animasyonu uygulandı. Görsel kısmı activity_splash_screen.xml de yapıldı. ConstraintLayout içinde ImageView bileşenleri kullanıldı. Animasyon ise Animation ve Handler tipleri ile uygulandı.

2.4.1.1.9. UpdateSettingsActivity.java

Bu activity’de uygulamanın güncelleme ayarlarına yer verildi iki ana ayar gösterildi. Bu ayarlar yazıları tutma sınırı ve günlük güncellemenin açılıp kapanmasını sağlayan ayarlardı. Günlük güncellemenin açılımında ise saatin seçilebileceği bir ekran geliyordu. Bu kısmın görsel tarafı activity_update_settings.xml de yapıldı. RelativeLayout içinde Toolbar, ImageView, View, TextView, NumberPicker, ImageButton, SwitchCompat kullanıldı.

2.4.1.2. Adapter

Adaptör bölümünde AuthorAdapter.java, NewsAdapter.java yazıldı. Burada AuthorAdapter kısmında RecyclerView içindeki listelemenin hem arayüzde hem databasede silinmesinin, güncellenmesinin ve eklenmesin kontrolcü kısımları yapıldı. NewsAdapter kısmında ise sadece parametre olarak gelen NewsModel ArrayList’inin liste halinde gösterimi sağlandı.

2.4.1.3. Database

Bu bölümde daha önceki kısımlarda belirtilen SQLite veri tabanını manipüle etmek için SQLiteOpenHelper ana sınıfından türemiş DatabaseHelper sınıfı yazıldı. Bu sınıf içinde veri tabanını oluşturma, veri tabanına ekleme yapma, veri tabanından çıkarma yapma,

veri tabanını güncelleme, veri tabanından veri alma gibi işlevleri sağlayan metotlar yazıldı.

2.4.1.4. *Helper*

Bu bölümde uygulamanın verilerinin çekilmesi için bazı işlemler yapıldı. Bu işlemleri yapmak amacıyla RequestHelper.java sınıfı yazıldı. Bu sınıfta basitçe servis veya elle güncelleme sırasında çalışacak olan HTTP isteklerinin uygulaması yapıldı. Bu uygulama sonucu parametre olarak verilen yazar ismine karşılık gelen en güncel köşe yazısı çekilebilir hale geldi. Bunu yapmak için öncelikle her yazarın gazetesindeki ana URL linkleri kaydedildi. Ardından bir istek atılacağıda çeşitli koşullarla hangi yazarın hangi URL'e karşılık geldiği bulundu ve bulunan URL'e istek atılarak sayfanın HTML düz yazısı alındı. Bu düz yazında bazı metotlarla arama yaparak yazarın en güncel yazısının URL'i bulundu. Bulunan URL'deki en güncel yazının içeriğini ayrıstırılmış halde alabilmek için ise Parser API adında Python Flask ile bir API yazıldı. Bu API'a yazının URL'i gönderildiğinde ana başlıklara ayrıstırılmış şekilde JSON dosyası olarak döndürüldü. Bu döndürülen JSON dosyasındaki gerekli alanlar ayrıstırılıp kullanılarak veri tabanına en güncel yazılar kaydedildi.

2.4.1.5. *Holder*

Holder kısmında ise NewsViewHolder.java isminde bir sınıf yazıldı. Bu sınıf yazıların gösterimindeki RecyclerView'in kullanılması için görsel, başlık, yazı ve zaman sağladı.

2.4.1.6. *Model*

Model kısmında veri tabanına kaydedilen ve veri tabanından getirilen verinin düzgün şekilde eklenmesi ve ayırtılmasını sağlamak için model sınıfları yazıldı. Bu sayede alınan veya eklenecek veri düzgün şekilde yani bir tip halinde saklanabilir oldu. Bu

kısımında yazılan sınıflar şunlardı; AuthorModel.java, FilterDayModel.java, NewsModel.java, SettingModel.java.

2.4.1.7. *Receiver*

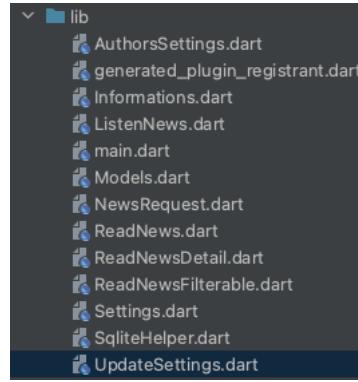
Bu kısımda AlarmReceiver.java adında bir sınıf yazıldı. Bu sınıfın işlevi bir alarm tetiklemesini kurmaktı. Bu alarm tetikleyici BroadcastReceiver'dan türetildi. Belirli bir zaman aralığı boyunca süre veriliyor ve daha sonra çalışacak işlevi tekliyordu. Basitçe belirtmek gerekirse arka planda çalışabilen bir zamanlayıcıydı. Bu zamanlayıcının belirli saatlerde çalışabilmesi için ilk çalışma saatinin bulunabilmesi amacıyla kullanıcının girdiği anki saat ile istediği saat arasındaki zaman farkı bulunup dakikaya çevrildi. Böylece ilk tetikleme o saatte yapılacak şekilde ayarlandı. İlk tetikleme gerçekleştirildikten sonra ise bir sonraki periyot 1440 dakika yani 1 gün olacak şekilde güncellendi ve böylece kullanıcının istediği saatte her gün güncelleme sağlandı.

2.4.1.8. *Service*

Bu kısımda MakeAlarmTask adında Service ana sınıfından türetilen bir sınıf yazıldı. Bu basitçe tetikleme olduğu zaman çalışacak bir işlev barındırdı. Her tetiklemede RequestHelper'ın güncelleme işlevini çalıştıracak şekilde programlandı.

2.4.2. Flutter İşlevleri

Flutter tarafından Android Studio tarafına nazaran daha basit bir geliştirme anlayışı kullanıldı. Her işlev/sayfa bir sınıf olacak şekilde Dart dilinde sınıflar yazıldı. Görsel kısım ve işlev kısmı beraber olduğu için genel yapısı bakımından da daha anlaşılabilir bir uygulama oldu.



Şekil 2.31. Flutter Sınıfları

2.4.2.1. *AuthorSetting.dart*

Bu sınıfıta yazarların ekleme ve çıkarma kısımlarının işlevleri ve görsel kısmı uygulandı. SingleChildScrollView içinde SearchableDropdown, ListTile, IconButton, Center, Text gibi alt elemanlar kullanıldı. Buradaki ekleme veya çıkarma durumunda ise veri tabanında değişiklik yapılarak işlem sağlandı. Liste gösterimi ise veri tabanından yazar verisi alınarak sağlandı.

2.4.2.2. *Informations.dart*

Bu sınıfıta kullanıcının uygulama hakkında alabileceği bilgiler ve notlar sağlandı. SingleChildScrollView içinde Padding ve Text alt elemanları kullanıldı.

2.4.2.3. *Main.dart*

Bu sınıfıta uygulamanın ana sayfası yapılandırıldı. Dört adet ana işlev dört adet basılabilir alana yerleştirildi. Böylece bu basılabilir alanlara dokunarak istenilen işlev geçiş imkanı sağlandı. Sağ üstte ise elle güncelleme için gazete ikonu yerleştirildi Column ve Row veri tiplerinin içinde Text, TextStyle, Align, GestureDetector, Container,

`CircularProgressIndicator`, `FittedBox`, `ListTile`, `Center` gibi veri tipleri kullanılarak yapıldı.

2.4.2.4. *Models.dart*

Model kısmında veri tabanına kaydedilen ve veri tabanından getirilen verinin düzgün şekilde eklenmesi ve ayırtırılmasını sağlamak için model sınıfları yazıldı. Bu sayede alınan veya eklenecek veri düzgün şekilde yani bir tip halinde saklanabilir oldu. Bu kısımda yazılan sınıflar şunlardı; `AuthorModel`, `FilterDayModel`, `NewsModel`, `SettingModel`.

2.4.2.5. *NewsRequest.dart*

Bu bölümde uygulamanın verilerinin çekilmesi için bazı işlemler yapıldı. Bu işlemleri yapmak amacıyla bahsi geçen sınıfı yazıldı. Bu sınıfta basitçe servis veya elle güncelleme sırasında çalışacak olan HTTP isteklerinin uygulaması yapıldı. Bu uygulama sonucu parametre olarak verilen yazar ismine karşılık gelen en güncel köşe yazısı çekilebilir hale geldi. Bunu yapmak için öncelikle her yazarın gazetesindeki ana URL linkleri kaydedildi. Ardından bir istek atılacağında çeşitli koşullarla hangi yazarın hangi URL'e karşılık geldiği bulundu ve bulunan URL'e istek atılarak sayfanın HTML düz yazısı alındı. Bu düz yazda bazı metodlarla arama yaparak yazarın en güncel yazısının URL'i bulundu. Bulunan URL'deki en güncel yazının içeriğini ayırtırılmış halde alabilmek için ise Parser API adında Python Flask ile bir API yazıldı. Bu API' a yazının URL'i gönderildiğinde ana başlıklara ayırtırılmış şekilde JSON dosyası olarak döndürüldü. Bu döndürülen JSON dosyasındaki gerekli alanlar ayırtırılıp kullanılarak veri tabanına en güncel yazılar kaydedildi.

2.4.2.6. *ReadNews.dart*

Bu sınıfıta yazarların yazılarının görüntülenmesi ve listelenmesi işlevi uygulandı. En üstte yazıların dinlenilebilmesi için oynatma, duraklatma, durdurma işlevlerine ve butonlarına yer verildi. Dinleme işlevi FlutterTts tipinin yerel veri tabanında bulunan yazıları okuması ile sağlandı. SingleChildScrollView içinde Card, ListTile, CachedNetworkImage, SizedBox, CircularProgressIndicator, Text, Button, TextStyle, Container, ElevatedButton tipleri kullanıldı. Bu kısımdaki yazı gösterimleri veri tabanından yazı verisi alınarak sağlandı.

2.4.2.7. *ReadNewsDetail.dart*

Bu sınıfıta üzerine tıklanan yazının detaylıca görüntülenmesi işlevi uygulandı. En üstte cached bir görsel alta yazının başlığı, yazarın ismi, gazetesi, yayınlanma tarihi ve yazının içeriğine yer verildi. En altta yazının dinlenilebilmesi için oynatma, duraklatma ve durdurma işlevlerine/butonlarına yer verildi. Dinleme işlevi FlutterTts tipinin yerel veri tabanında bulunan yazıları okuması ile sağlandı. Container içinde SingleChildScrollView, Column, Align, CachedNetworkImage, Text, TextStyle, SizedBox gibi tipler kullanıldı. Bu kısımdaki yazı gösterimi veri tabanından yazı verisi aktarılarak sağlandı.

2.4.2.8. *ReadNewsFilterable.dart*

Bu sınıfıta yazarların yazılarının görüntülenmesi ve listelenmesi işlevi uygulandı. En üstte yazıların arasında arama yapılabilesi için yazılabilir filtreleme kısmına yer verildi. SingleChildScrollView içinde TextEditingController, Padding Card, ListTile, CachedNetworkImage, SizedBox, CircularProgressIndicator, Text, Button, TextStyle, Container, ElevatedButton tipleri kullanıldı. Bu kısımdaki yazı gösterimleri veri tabanından yazı verisi alınarak sağlandı.

2.4.2.9. *Settings.dart*

Bu sınıfıta uygulamanın ana ayarlar sekmesi yapıldı. Üstte Güncelleme Ayarları altta Yazar Ekleme / Çıkarma olacak şekilde ikiye bölündü. SingleChildScrollView içinde Column, ListTile, Divider, Icon kullanılarak yapıldı.

2.4.2.10. *SqliteHelper.dart*

Bu bölümde daha önceki kısımlarda belirtilen SQLite veri tabanını manipüle etmek için SqliteHelper sınıfı yazıldı. Bu sınıf içinde veri tabanını oluşturma, veri tabanına ekleme yapma, veri tabanından çıkışma yapma, veri tabanını güncelleme, veri tabanından veri alma gibi işlevleri sağlayan metodlar yazıldı.

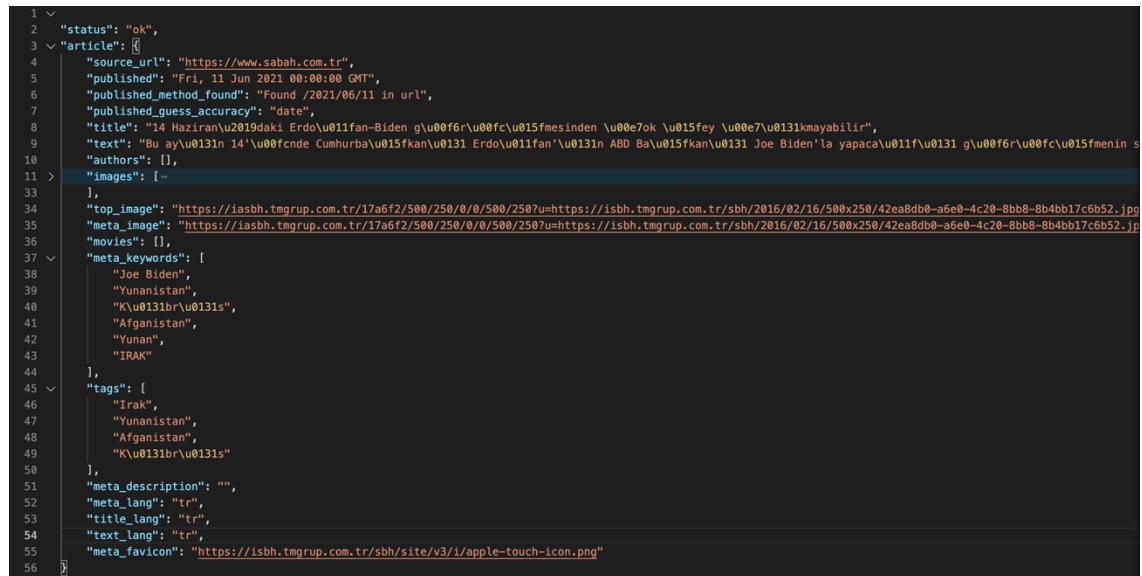
2.4.2.11. *UpdateSettings.dart*

Bu activity'de uygulamanın güncelleme ayarlarına yer verildi iki ana ayar gösterildi. Bu ayarlar yazıları tutma sınırı ve günlük güncellemenin açılıp kapanmasını sağlayan ayarlardı. Günlük güncellemenin açılımında ise saatin seçilebileceği bir ekran geliyordu. Ayrıca arka plan servislerinin çalışması da burada yapıldı. Isolates kullanarak arka planda çalışma yeteneği kazandırıldı. Future.delay ve Timer işlevleriyle de tekrarlı güncelleme işlemi sağlandı. Bu zamanlayıcının belirli saatlerde çalışabilmesi için ilk çalışma saatinin bulunabilmesi amacıyla kullanıcının girdiği anki saat ile istediği saat arasındaki zaman farkı bulunup dakikaya çevrildi. Böylece ilk tetikleme o saatte yapılacak ertelendi. İlk tetikleme gerçekleştirildikten sonra ise bir sonraki periyot 1440 dakika yani 1 gün olacak şekilde güncellendi ve böylece kullanıcının istediği saatte her gün güncelleme sağlandı.

2.5. API KURULUMU VE WEB'DE YAYILMASI

2.5.1. API Kurulumu

Daha önceki kısımlarda bahsedilen yazarın en güncel yazısını parametre olarak alan ve buna göre ayırtırıp json dosyası olarak geri döndüren API Python Flask kullanılarak yazıldı. Python kodunda Flask'e ek olarak newspaper, date_guesser ve langdetect eklentileri kullanıldı. Bu şekilde parametre olarak alınan URL newspaper eklentisi ile elde edilip ardından ayırtırıldı. Ayırtırılan ve elde edilen bilgiler bir sözlük (dictionary) değişkenine kaydedildi. Son olarak ise json dosyası olarak geri döndürüldü.



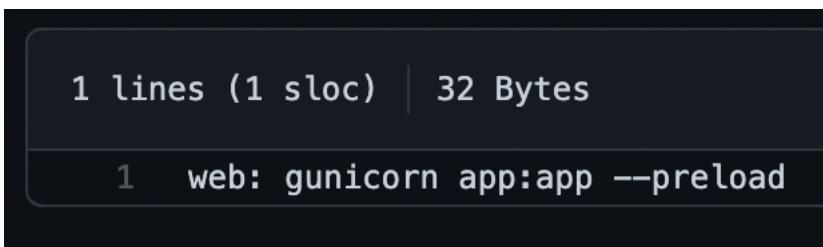
```
1 <pre>
2   "status": "ok",
3   "article": [
4     {
5       "source_url": "https://www.sabah.com.tr",
6       "published": "Fri, 11 Jun 2021 00:00:00 GMT",
7       "published_method_found": "Found /2021/06/11 in url",
8       "published_guess_accuracy": "date",
9       "title": "14 Haziran\u02019'daki Erdo\u011fan-Biden g\u00f6rüşmesinden \u000e7ok \u0015fey \u000e7 \u0131kmayabilir",
10      "text": "Bu ay\u0131n 14' \u000fcnde Cumhurba\u0131n Erdo\u011fan'\u0131n ABD Ba\u0131fkan\u0131 Joe Biden'la yapaca\u011f\u0131 g\u00f6rüşmenin s",
11      "authors": [],
12      "images": [
13        {
14          "url": "https://iasbh.tmggrup.com.tr/17a6f2/500/250/0/0/500/250?u=https://isbh.tmggrup.com.tr/sbh/2016/02/16/500x250/42ea8db0-a6e0-4c20-8bb8-8b4bb17c6b52.jpg",
15          "width": 500,
16          "height": 250
17        }
18      ],
19      "top_image": "https://iasbh.tmggrup.com.tr/17a6f2/500/250/0/0/500/250?u=https://isbh.tmggrup.com.tr/sbh/2016/02/16/500x250/42ea8db0-a6e0-4c20-8bb8-8b4bb17c6b52.jpg",
20      "meta_image": "https://iasbh.tmggrup.com.tr/17a6f2/500/250/0/0/500/250?u=https://isbh.tmggrup.com.tr/sbh/2016/02/16/500x250/42ea8db0-a6e0-4c20-8bb8-8b4bb17c6b52.jpg",
21      "movies": [],
22      "meta_keywords": [
23        "Joe Biden",
24        "Yunanistan",
25        "Ku\u0131lbr\u0131",
26        "Afganistan",
27        "Yunan",
28        "IRAK"
29      ],
30      "tags": [
31        "Irak",
32        "Yunanistan",
33        "Afganistan",
34        "Ku\u0131lbr\u0131"
35      ],
36      "meta_description": "",
37      "meta_lang": "tr",
38      "title_lang": "tr",
39      "text_lang": "tr",
40      "meta_favicon": "https://isbh.tmggrup.com.tr/sbh/site/v3/i/apple-touch-icon.png"
41    }
42  ]
43}</pre>
```

Şekil 2.32. Örnek JSON Dönüütü

2.5.2. API'ın Web'de Yayılması

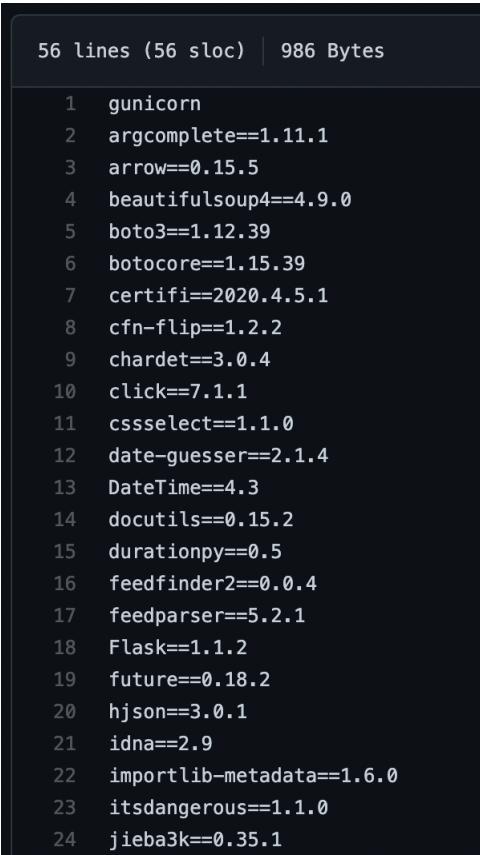
API'ın web' de yayılması (deploy) için Heroku adında bir bulut platformu kullanıldı. Bu platform sayesinde uygulamanın ismini içeren bir web URL'i sağlandı böylece bu URL'e GET tipinde bir istek atıldığında işlenmiş veri geri döndü. Heroku platformundaki

konfigürasyonlar ise Python kodunun yanına Procfile, requirements.txt ve runtime.txt eklenmesi ile sağlandı. Procfile, API kapsayıcıları tarafından Heroku platformunda hangi komutların çalıştırıldığını bildiren bir mekanizmaydı. Requirements.txt ise API'ın çalışması için gerekli olan eklentileri belirten bir dosyaydı. Runtime.txt ise hangi Python sürümü ile çalışacağını belirten bir dosyaydı. Tüm bu dosyalar Github'a yüklenikten sonra Heroku ile bağdaştırıp yayma (deploy) ile API aktif hale getirildi.



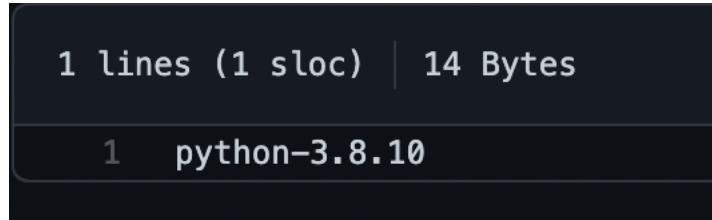
```
1 lines (1 sloc) | 32 Bytes
1 web: gunicorn app:app --preload
```

Şekil 2.33. Procfile İçeriği



```
56 lines (56 sloc) | 986 Bytes
1 unicorn
2 argcomplete==1.11.1
3 arrow==0.15.5
4 beautifulsoup4==4.9.0
5 boto3==1.12.39
6 botocore==1.15.39
7 certifi==2020.4.5.1
8 cfn-flip==1.2.2
9 chardet==3.0.4
10 click==7.1.1
11 cssselect==1.1.0
12 date-guesser==2.1.4
13 DateTime==4.3
14 docutils==0.15.2
15 durationpy==0.5
16 feedfinder2==0.0.4
17 feedparser==5.2.1
18 Flask==1.1.2
19 future==0.18.2
20 hjson==3.0.1
21 idna==2.9
22 importlib-metadata==1.6.0
23 itsdangerous==1.1.0
24 jieba3k==0.35.1
```

Şekil 2.34. Requirements.txt İçeriği



Sekil 2.35. Runtime..txt İçeriği

Application Logs ALL PROCESSES ◊

```
04/4349/* Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.77 Safari/537.36
2021-06-13T15:15:57.293454+00:00 heroku[router]: at=info method=GET path="/favicon.ico" host=extractnews.herokuapp.com request_id=d4dc7a9d-c930-4a70-8730-7bde68f5cdcd fwd="159.146.42.119" dyno=web.1 connect=0ms service=2ms status=404 bytes=378 protocol=https
2021-06-13T15:16:25.534391+00:00 app[web.1]: 10.38.245.138 - [13/Jun/2021:15:16:25 +0000] "GET /v0/article?url=https://www.sozcu.com.tr/2021/yazarlar/yilmaz-ozdil/sedat-pekerin-anlattiklarini-tee-60-yil-once-anlatan-kimdi-6474349/" HTTP/1.1" 200 6716 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.77 Safari/537.36"
2021-06-13T15:16:25.534768+00:00 heroku[router]: at=info method=GET path="/v0/article?url=https://www.sozcu.com.tr/2021/yazarlar/yilmaz-ozdil/sedat-pekerin-anlattiklarini-tee-60-yil-once-anlatan-kimdi-6474349/" host=extractnews.herokuapp.com request_id=13f9a1c6-50ec-4ede-b34e-b37129d0ee44 fwd="159.146.42.119" dyno=web.1 connect=0ms service=456ms status=200 bytes=6863 protocol=https
2021-06-13T15:16:30.948140+00:00 app[web.1]: 10.38.245.138 - [13/Jun/2021:15:16:30 +0000] "GET /v0/article?url=https://www.sabah.com.tr/yazarlar/barlas/2021/06/11/14-hazirandaki-erdogan-biden-gorusmesinden-cok-sey-cikmayabilir" HTTP/1.1" 200 4683 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.77 Safari/537.36"
2021-06-13T15:16:30.948278+00:00 heroku[router]: at=info method=GET path="/v0/article?url=https://www.sabah.com.tr/yazarlar/barlas/2021/06/11/14-hazirandaki-erdogan-biden-gorusmesinden-cok-sey-cikmayabilir" host=extractnews.herokuapp.com request_id=046f5487-a4bf-4dae-8d2f-c74bbecdcf3f fwd="159.146.42.119" dyno=web.1 connect=0ms service=1206ms status=200 bytes=4830 protocol=https
```

Autoscroll with output Save

Sekil 2.36. Heroku'da Yayınlanan API'nin Seyir Defteri (Log)

3. BULGULAR

Bu kısımda uygulamanın cihaz ve kullanıcı testlerine yer verildi. Cihaz testlerinde öncelikle Android Studio ve MacOS tarafından sağlanan Android emülatörü ve iPhone simülatörlerinde denendi. Kullanıcı testinde ise uygulama 10 tane kullanıcı tarafından test edildi ve belirli kriterlere göre puanlandı. Varsa geri bildirimleri alındı.

3.1. CİHAZ TESTLERİ

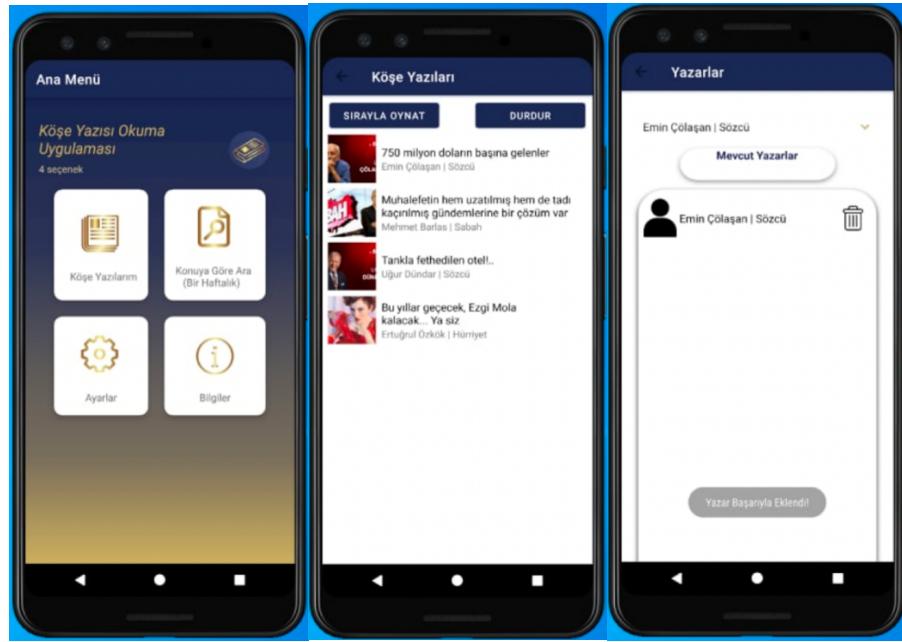
3.1.1. Android Studio Tarafındaki Testler

Android Studioda Java diliyle yazılan ve sadece Android'de çalışan uygulama hem gerçek hem sanal olacak şekilde çeşitli cihazlarda test edildi. Bu cihazlar aşağıda listelendi ve özellikleri aktarıldı.

3.1.1.1. Pixel 3a (Emülatör)

- Qualcomm Snapdragon 670.
- 5.6 inç, 2220 x 1080 (Full HD+) ekran.
- 4 GB RAM, 64 GB dahili depolama.
- Android Oreo 8.0 (API level 26) x86 / Android Nougat 7.0 (API level 24) x86

Bu emülatörde tüm özellikler ve arayüz olması gerekiği şekilde çalıştı. Fakat yazarlar kısmında arayüzde Mevcut Yazarlar kısmını kapsayan kısmda tam yerlesim sağlanamadığı için o bölüm kaldırıldı. Sadece alttaki bölüm olacak şekilde düzenlendi. Aşağıya örnek ekran görüntüleri eklendi.

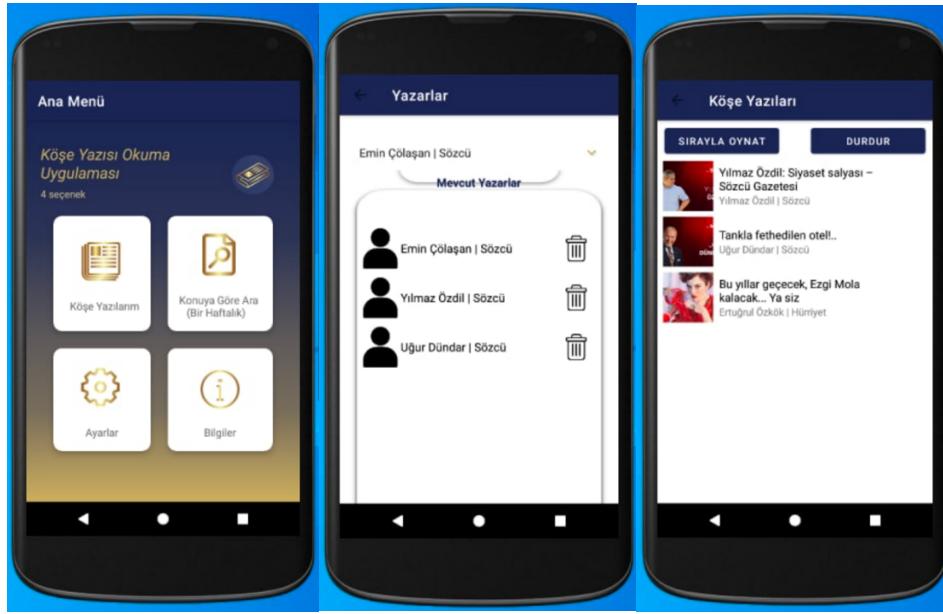


Şekil 3.1. Pixel 3a Emülatöründeki Çalışma

3.1.1.2. Nexus 4 (Emülatör)

- Ekran Boyutu: 4.7 İnç
- Dahili Depolama: 16 GB.
- Bellek (RAM): 2 GB.
- 4.7 inç, 768 x 1280 (HD+) ekran.
- Qualcomm Snapdragon S4 pro
- Android Oreo 5.1 (API level 22) x86

Bu emülatörde tüm özellikler ve arayüz olması gerekiği şekilde çalıştı. Fakat yazarlar kısmında arayüzde Mevcut Yazarlar kısmını kapsayan kısımda tam yerleşim sağlanamadığı için o kısım kaldırıldı. Sadece alttaki kısım olacak şekilde düzenlendi. Aşağıya örnek ekran görüntüleri eklendi.

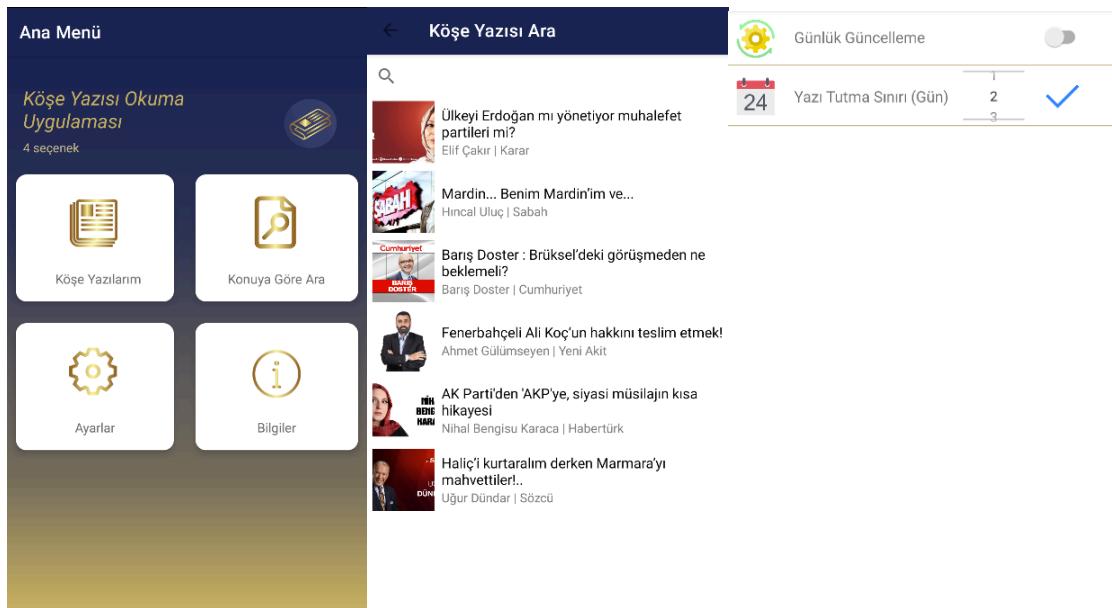


Şekil 3.2. Nexus 4 Emülatöründeki Çalışma

3.1.1.3. *sdk_gphone_arm64 (Geliştirme Ortamı)*

- Ekran Boyutu: 5.5 İnç
- Android 11 R (API level 30) aarch64

Bu emülatörde tüm özellikler ve arayüz olması gerekiği şekilde çalıştı. Aşağıya örnek ekran görüntüleri eklendi.

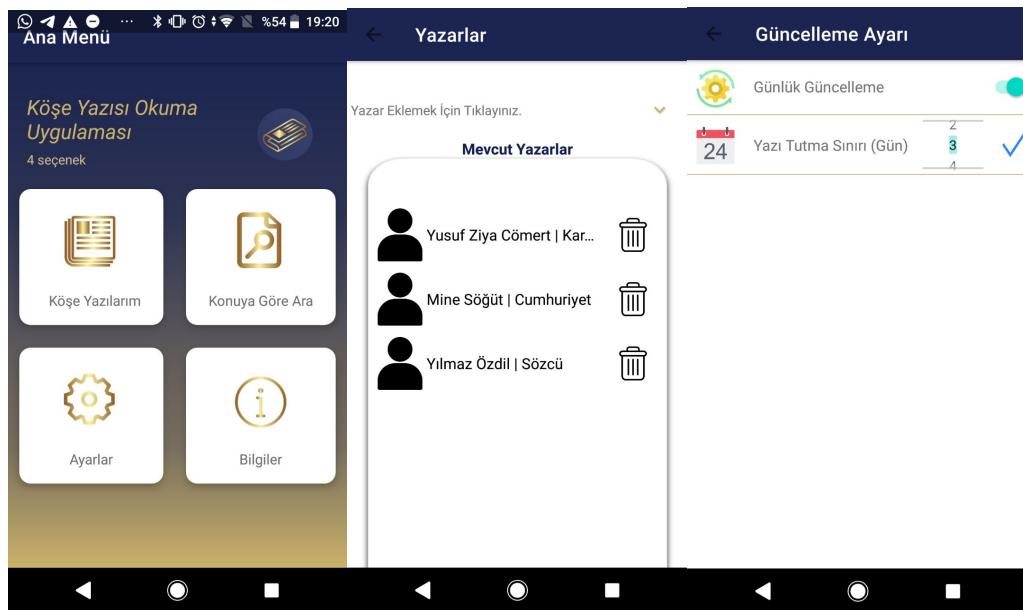


Şekil 3.3. *sdk_gphone_arm64* Emülatöründeki Çalışma

3.1.1.4. Sony Xperia X (Gerçek Cihaz)

- Ekran: 5 inç, 1080 x 1920 piksel.
- İşletim Sistemi: Android 8.0 Oreo
- İşlemci: Qualcomm Snapdragon 650.
- RAM: 3 GB.
- Dahili Hafiza: 32 GB.

Bu cihazında tüm özellikler ve arayüz olması gereki̇ği şekilde çalışı̇. Aşağıya örnek ekran görüntüleri eklendi.

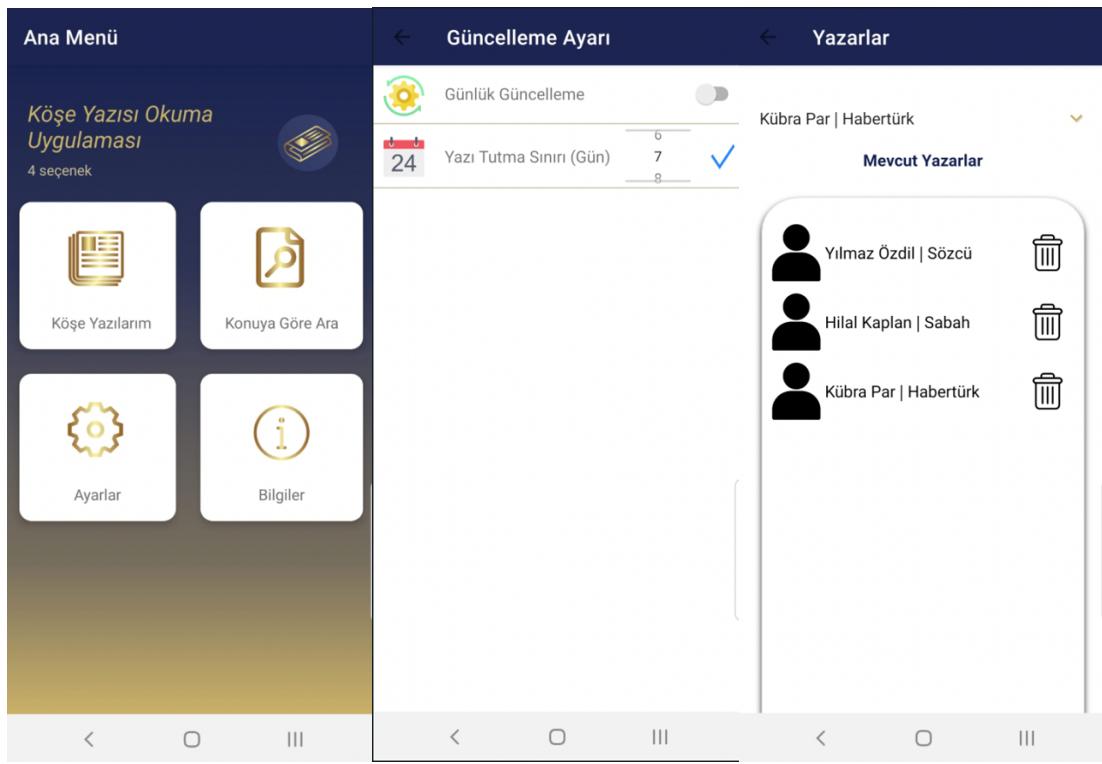


Şekil 3.4. Xperia X Cihazındaki Çalışma

3.1.1.5. Samsung Galaxy S8 (Gerçek Cihaz)

- Ekran Boyutu: 5.8 İnç
- İşletim Sistemi: Android 9
- İşlemci: Samsung Exynos 8895
- Ekran Çözünürlüğü: 1440 x 2960
- Dahili Depolama: 64 GB.
- Bellek (RAM): 4 GB.

Bu cihazında tüm özellikler ve arayüz olması gerektiği şekilde çalıştı. Aşağıya örnek ekran görüntüleri eklendi.

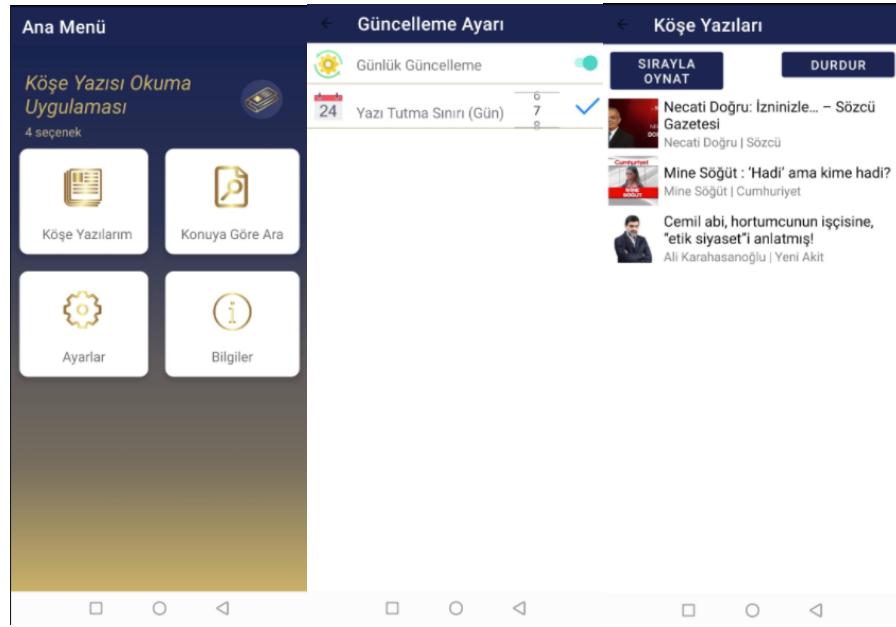


Şekil 3.5. Samsung Galaxy S8 Cihazındaki Çalışma

3.1.1.6. Huawei Mate 20 lite (Gerçek Cihaz)

- Ekran Boyutu: 6.3 İnç
- Ekran Çözünürlüğü: 1080 x 2340
- İşletim Sistemi: Android 10
- İşlemci: Huawei Kirin 710
- Dahili Depolama: 64 GB.
- Bellek (RAM): 4 GB.

Bu cihazında tüm özellikler ve arayüz olması gerektiği şekilde çalıştı. Aşağıya örnek ekran görüntüleri eklendi.



Şekil 3.6. Huawei Mate 20 Lite Cihazındaki Çalışma

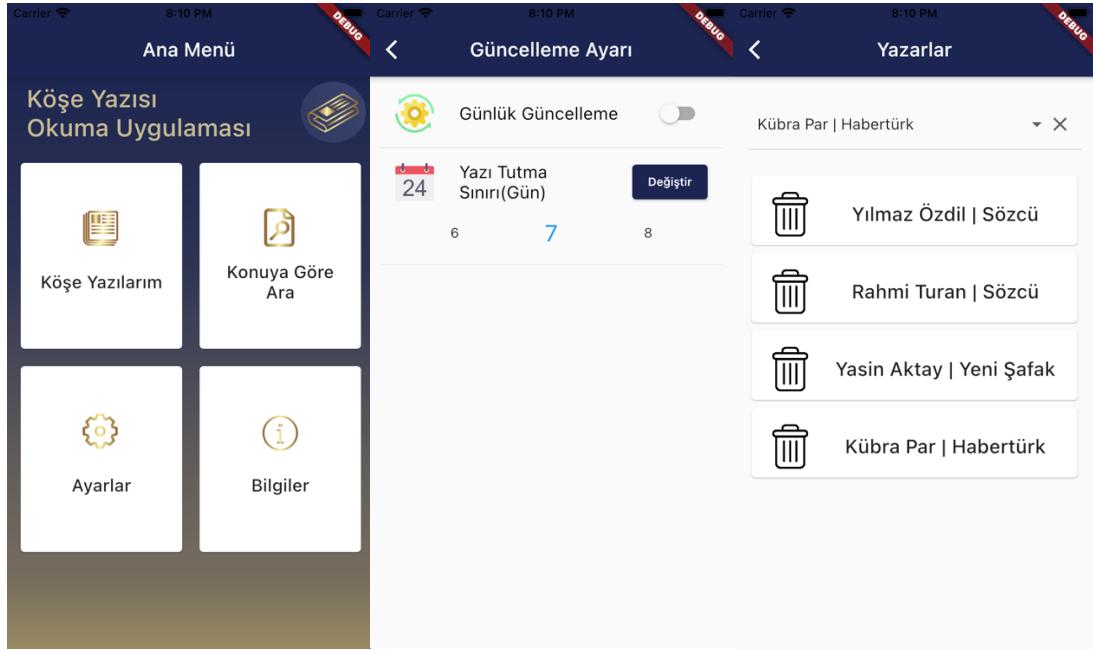
3.1.2. Flutter Tarafındaki Testler

Flutter uygulamanın iOS tarafının çıkarılması için kullanıldı. Bu yüzden buradaki testler iOS emülatörü ve iOS cihazla yapıldı.

3.1.2.1. Apple iPhone 8 (Emülatör)

- IPS teknolojisine sahip 4.7 inç (diyagonal) geniş ekran LCD Multi-Touch
- 326 ppi yoğunlukta 1334 x 750 piksel çözünürlük
- A11 Bionic Chip
- iOS 14.4

Bu simülatörde tüm özellikler ve arayüz olması gereki̇ği şekilde çalışı̇. Aşağıya örnek ekran görüntüleri eklendi.

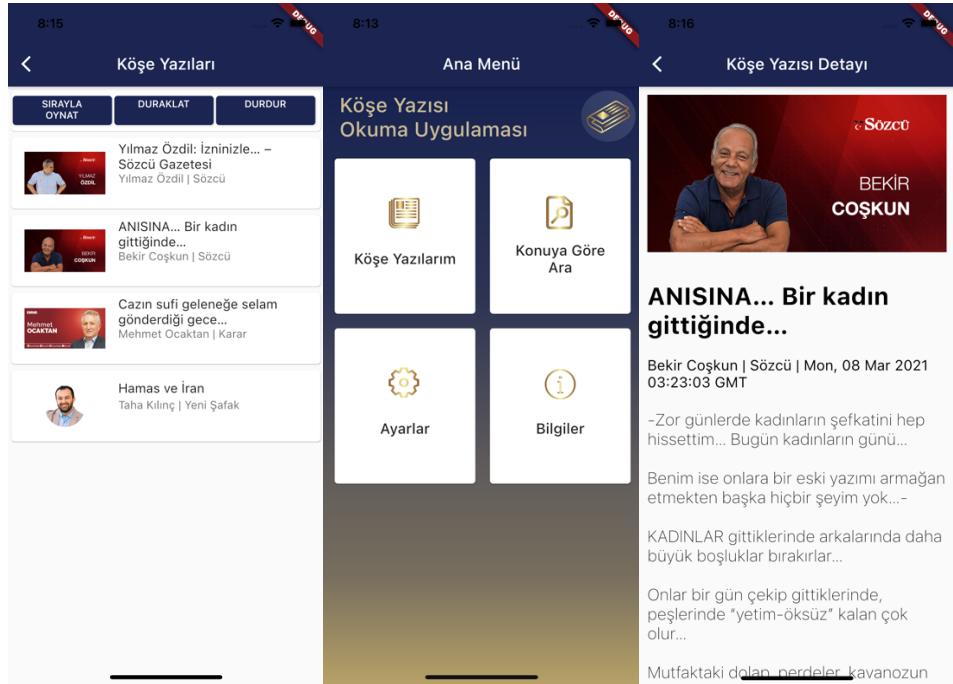


Şekil 3.7. iPhone 8 Simülatöründeki Çalışma

3.1.2.2. Apple iPhone 12 Pro (Emülatör)

- Ekran Boyutu: 6.1" OLED.
- Çözünürlük: 2532 x 1170.
- Boyut: 146,7 x 71,5 x 7,4 mm.
- iOS 14.4
- A14 Bionic Chip

Bu simülatörde tüm özellikler ve arayüz olması gerekiği şekilde çalıştı. Aşağıya örnek ekran görüntüleri eklendi.

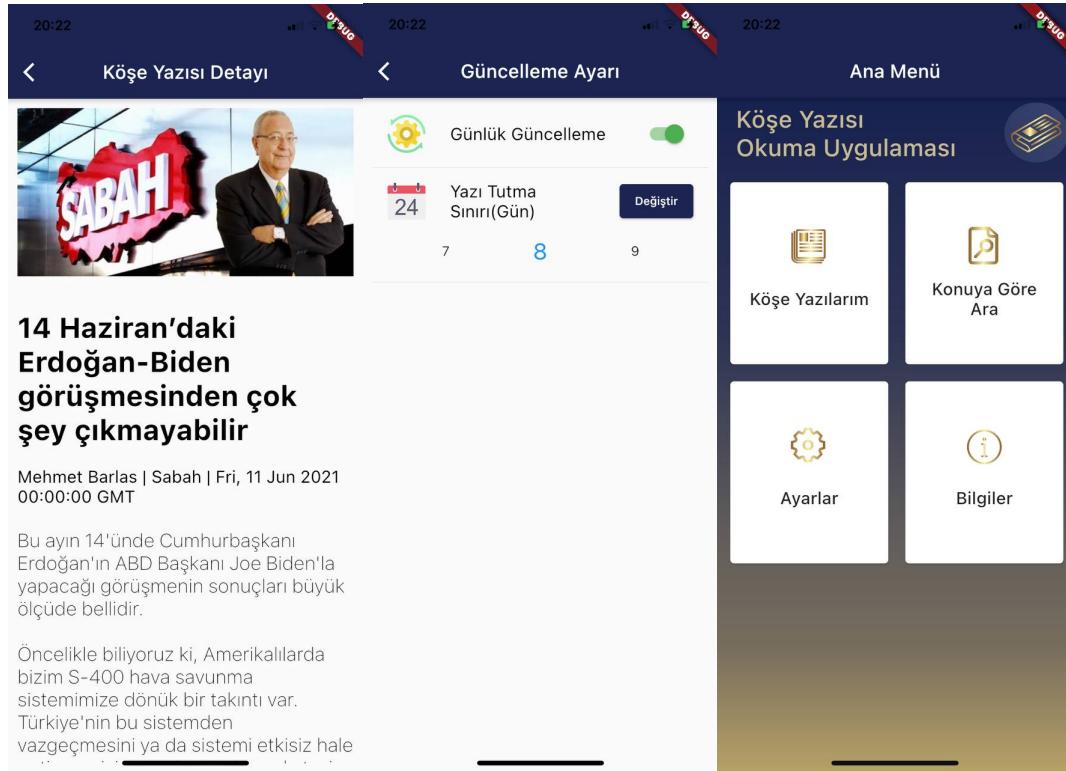


Şekil 3.8. iPhone 12 Pro Simülatöründeki Çalışma

3.1.2.3. Apple iPhone 11 (Gerçek Cihaz)

- 326 ppi yoğunlukta 1792 x 828 piksel çözünürlük
- IPS teknolojisine sahip 6.1 inç (diyagonal) tam ekran LCD Multi-Touch
- Dahili Depolama: 64 GB.
- Bellek (RAM): 4 GB.
- A13 Bionic Chip
- iOS 14.4

Bu cihazda tüm özellikler ve arayüz olması gerekiği şekilde çalıştı. Aşağıya örnek ekran görüntüleri eklendi.



Şekil 3.9. iPhone 11 Cihazındaki Çalışma

3.2. KULLANICI TESTLERİ

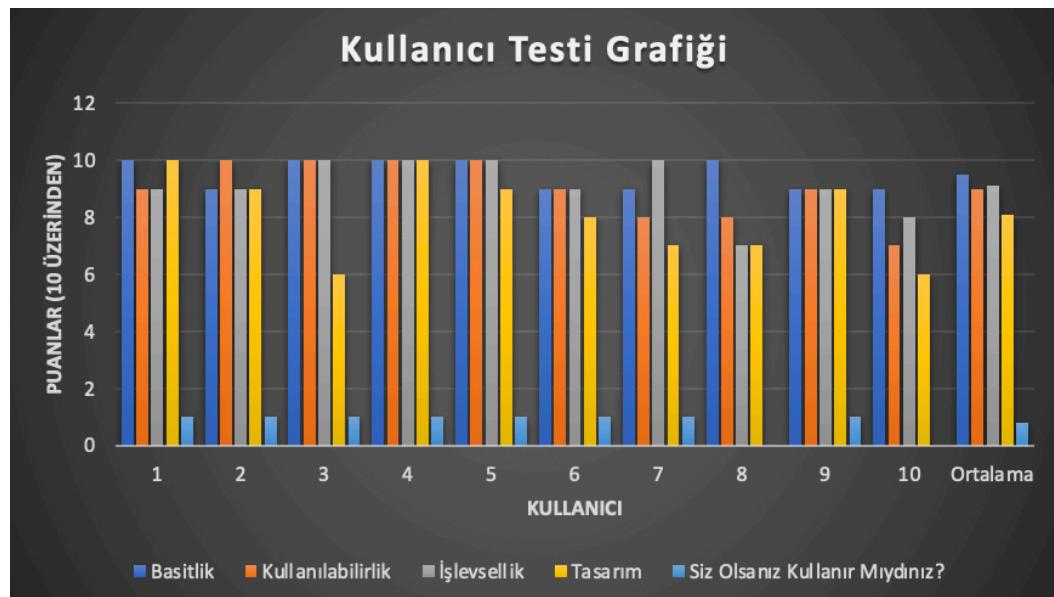
Bu kısımda 10 adet kullanıcıya uygulama sunuldu ve kullanıcılarla uygulama deneyimini bazı kriterlere göre değerlendirmesi istendi. Bu kriterler genel olarak:

- Basitlik (Anlaşılabilirlik)
- Kullanılabilirlik
- İşlevsellik
- Tasarım

Başlıklarından oluşmaktadır. Ayrıca tüm bunların sonucunda kullanıcıya “Siz olsanız kullanır mıydınız?” sorusu soruldu. Bu değerlendirmeler sonucu aşağıdaki tablo elde edildi.

Kullanıcı	Basitlik	Kullanılabilirlik	İşlevsellik	Tasarım	Siz Olsanız Kullanır Miyiniz?
1	10	9	9	10	1
2	9	10	9	9	1
3	10	10	10	6	1
4	10	10	10	10	1
5	10	10	10	9	1
6	9	9	9	9	1
7	9	8	10	7	1
8	10	8	7	7	0
9	9	9	9	9	1
10	9	7	8	6	0
Ortalama	9,5	9	9,1	8,1	0,8

Tablo 3.1. Kullanıcı Testi Tablosu



Şekil 3.10. Kullanıcı Testi Bar Grafiği

4. TARTIŞMA VE SONUÇ

Bu bitirme projesinde bir köşe yazısı uygulaması yapıldı. Bu projedeki asıl ve özgün olan amaç uygulamanın sadece gerçekten gerekli işlevleri için ağ erişimi kullanmak ve diğer ana işlevleri ise ağ erişimi olmadanda sağlayabilmekti. Köşe yazısı okuma, yazarların resminin tutulması, sesli dinleme, yazılar arasında arama, yazıları belirli gün dahilinde kaydetme, yazar ekleme ve çıkarma işlevleri için ağ erişimi sınırlayıcılığı olmadan yapılabilir hale geldi. Ayrıca tüm bu işlevler hem Android mobil cihazlar hem de iOS mobil cihazlar için sağlanacak şekilde geliştirme yapıldı. Bu bahsedilen özellikleri sağlamak amacıyla geliştirme aracı olarak Flutter ve Android Studio kullanılırken, veri tabanı sistemi olarak SQLite, sesli dinleme için Google TTS ve iOS öntanımlı TTS sistemi, Arka planda güncelleme için Android tarafında AlarmManager Flutter tarafında ise isolates kullanıldı. Yazıların ayrıştırılması için Python Flask kullanarak bir API yazıldı ve Heroku üzerinden web'de dağıtıımı sağlandı.

Sonuç olarak bu bitirme projesi bu alanda çalışacak olanlar için iyi bir örnek veya prototip olabilir. Ayrıca kullanıcılarından alınan ve EK-A' da belirtilen geri bildirimler ve çeşitli özelliklerle dahada geliştirilip gerçek bir ticari uygulamaya dönüştürülebilir. Fakat mevcut durumda da ,amaçlarını gerçekleştiren bir uygulama olduğu için, kullanıcı tarafından rahatlıkla kullanılabilir.

5. KAYNAKLAR

- [1] Google, Android Studio Developer [Çevrimiçi], https://developer.android.com/studio/* [Ziyaret Tarihi: 22 Mart 2021].
- [2] Stackoverflow [Çevrimiçi], https://stackoverflow.com/questions/* [Ziyaret Tarihi: 15 Mart 2021].
- [3] Youtube, Create SQLite Database [Çevrimiçi], <https://www.youtube.com/watch?v=UdNn4S60I3k> [Ziyaret Tarihi: 10 Mart 2021].
- [4] Youtube, Android XD Tutorial [Çevrimiçi], <https://www.youtube.com/watch?v=IVE5ETGEoA4> [Ziyaret Tarihi: 17 Mart 2021].
- [5] Lucidchart [Çevrimiçi], <https://lucid.app/lucidchart> [Ziyaret Tarihi: 22 Nisan 2021].
- [6] Geeksforgeeks [Çevrimiçi], https://www.geeksforgeeks.org/* [Ziyaret Tarihi: 7 Nisan 2021].
- [7] Youtube, News Feed In Android Studio [Çevrimiçi], https://www.youtube.com/watch?v=9oNZAzlhL7s&list=PLT3-dzFEBix16zontJjPJPeYUWtE_HReq [Ziyaret Tarihi: 5 Nisan 2021].
- [8] Looka, Free Logo Maker, https://looka.com/onboarding?company_name=kYOU [Ziyaret Tarihi: 15 Nisan 2021]
- [9] Flaticon[Çevrimiçi], <https://www.flaticon.com/> [Ziyaret Tarihi: 18 Nisan 2021]
- [10] Pub Dev[Çevrimiçi], <https://pub.dev/>, [Ziyaret Tarihi: 10 Nisan 2021]
- [11] Flutter[Çevrimiçi], <https://flutter.dev/docs/development> , [Ziyaret Tarihi: 10 Nisan 2021]
- [12] Heroku Dev Center[Çevrimiçi], <https://devcenter.heroku.com/articles/git>, [Ziyaret Tarihi: 06 Haziran 2021]

[13] Python Flask Tutorial[Çevirmiçi], <https://flask.palletsprojects.com/en/2.0.x/>,
[Ziyaret Tarihi: 05 Haziran 2021]

6. EKLER

A. Kullanıcı Geri Bildirimleri

a. Kullanıcı 1

Basitlik(anlaşılabilirlik) : İlk kurulum için ek olarak görsel öğretici olabilir, onun dışında ayarlaması ve kullanılması oldukça basit.

Kullanılabilirlik: Uygulama işlevini yerine getiriyor, istenilen yazar veya köşe yazıları okunabiliyor.

İşlevsellik: Daha fazla yazar ve köşe yazısı eklenebilir. Daha önce okuyup beğendiğim köşe yazılarını ayrı bir yerde kaydedebilsem güzel olur. Onun dışında okumuş olduğum köşe yazılarını otomatik gizlese güzel olur.

Tasarım: Tasarım minimal ve güzel, Butonların renkleri ayarlanabilse güzel olur.

Siz olsanız kullanır mıydınız?: Evet

Bir köşe yazısı olarak gayet iyi ve işlevini yerine getiriyor. Gündemi takip etmek istediği. Yazarları okuyabilmek için çok kullanışlı bir uygulama.

Sadece tek bir kaynaktan değil farklı medya organlarının yer olması da ayrıca güzel olmuş. Kullanır ve tavsiye ederim.

b. Kullanıcı 2

Basitlik(anlaşılabilirlik) : Bir uygulamanın anlaşılabilirliği için en önemli şeyin bilgilendirme ve kurulum dokümanı olduğunu düşünüyorum. Bu doküman gayet anlaşılır bir şekilde ayarlanmış.

Kullanılabilirlik: Eğer ekran boyutu ön tanımlı olarak maksimum seçildiyse bunu düzelterek uygulamayı açmak gerekiyor. Bu sorun için bir uyarı yazısı eklenebilir.

İşlevsellik: Okuma uygulamalarında sıkça kullandığımız haftalık okuma saatı, toplam okunan köşe yazısı sayısı vs. gibi özellikler eklenebilir. Bunun dışında gayet kullanışlı.

Tasarım: Sadelikten yana bir kullanıcı olarak tasarım kesinlikle yerinde.

Siz olsanız kullanır mıydınız?: Evet

Farklı gazetelerden birçok köşe yazarının takip eden biri olarak kesinlikle tavsiye ederim. Ayrıca haber site ve uygulamalarının görsel anlamda yoran yapısından kurtulmuş oluyorsunuz. Bunun yanında reklamları görmemek de tabii ki mutlu ediyor :)

c. Kullanıcı 3

Basitlik(anlaşılabilirlik): Basitlik konusu 10/10 bence bunu anlamayan köşe yazısı okumasın zaten yazıyı da anlayamaz.

İşlevsellik: 7/10, çünkü haber içermiyor sadece köşe yazısı içeriyor. Dolayısıyla kitle daralmış oluyor diye düşünüyorum.

Tasarım: konusunda butonların yerleri vs. her şey hoş ama şu altın sarısı şeysiler pek göze hitap etmiyor bence, yeni çıkan uygulamalarda daha yumuşak geçişler yapılıyor.

Kullanılabilirlik: Farklı ayarlar ve farklı özellikler zaman içinde kullanıcı deneyimine göre değişir diye düşünüyorum.

Siz olsanız kullanır mıydınız ?: Açıkçası günlük olarak takip ettiğim bir köşe yazarı yok ben denk geldikçe okuyorum ama bu uygulama olsa boş vaktlerimi değerlendirebilirim. Tabi bunun olması için bu uygulamadan haberdar olmam gerekiyor. Bu ihtiyacım için Google Play'de uygulama aramazdım.

Bu çalışma/..../2021 tarihinde aşağıdaki jüri tarafından Bilgisayar Mühendisliği Bölümü’nde Lisans Bitirme Projesi olarak kabul edilmiştir.

Bitirme Projesi Jürisi

Danışman Adı	Doç. Dr. Habil KALKAN	
Üniversite	Gebze Teknik Üniversitesi	
Fakülte	Mühendislik Fakültesi	

Jüri Adı	Prof. Dr. Erkan ZERGEROĞLU	
Üniversite	Gebze Teknik Üniversitesi	
Fakülte	Mühendislik Fakültesi	

Jüri Adı	Dr. Öğr. Görevlisi Alp Arslan BAYRAKÇI	
Üniversite	Gebze Teknik Üniversitesi	
Fakülte	Mühendislik Fakültesi	