

ANKARA ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ



BLM 4061-N PROJE RAPORU

TAMİRCİM.COM Mobil Uygulaması

Duygu Delice,Oğuzhan Sezer

19290232,19290786

Ayhan Aydın

27.12.2022

ÖZET

Bu çalışmada kullanıcılara Tamirci bulmaları konusunda yardımcı olacak ve firmaların müşterilerine karşı firma tanıtımını yapmasını sağlayarak müşterileri ile daha fazla etkileşime geçmelerini sağlayacak bir Tamirci uygulaması anlatılmaktadır.

.Projenin adı TAMİRCİNDEN.COM mobil uygulamasıdır. Projenin kullanımı herkese açıktır.

TAMİRCİNDEN.COM mobil uygulaması Flutter SDK kullanılarak Dart Programlama dili ile geliştirilmiştir.

TAMİRCİNDEN.COM uygulaması Android Studio IDE 'si kullanılarak geliştirilmiştir.

Veri tabanı olarak verilerin internet üzerinde depolanabildiği, bulut tabanlı FireStore kullanılmıştır.

Firestore, her biri veri içeren belge koleksiyonlarından oluşan NoSQL belge tabanlı bir veri tabanıdır.

Projede temel amaç kullanıcılara hizmet almak istedikleri firmayı daha etkin daha hızlı bir şekilde bulmalarını sağlamak ve onların firmadan daha hızlı ve etkin hizmet almalarını sağlamaktır. Diğer yandan Firmaların kendi firmalarını kullanıcılara tanıtmaları amaçlanmaktadır. Bu sayede firmalar müşterileri ile daha fazla etkileşim içerisinde olabilecektir.

İÇİNDEKİLER

ÖZET	2
İÇİNDEKİLER.....	3
1. GİRİŞ	4
2.TAMİRCİNDEN.COM UYGULAMASI	5
2.1. <u>Proje Açılması ve Projenin Veri Tabanının Oluşturulması</u>	5
2.2. <u>Proje Emülatörünün Oluşturulması</u>	9
2.3. <u>Splash Screen Oluşturulması</u>	10
2.4. <u>Ana Sayfa Sayfası Oluşturulması</u>	11
2.5. <u>İletişim Sayfası Oluşturulması</u>	15
2.6. <u>Hakkımızda Sayfası Oluşturulması</u>	17
2.7 <u>Tüm Tamirciler Sayfası Oluşturulması</u>	18
2.8. <u>Tamirci Detay Sayfası Oluşturulması</u>	19
2.9. <u>Tamirci CRUD(İşlemleri) Sayfası Oluşturulması</u>	21
2.10. <u>Haritalar Sayfası</u>	25
2.11. <u>Oturum Aç Sayfası</u>	27
3. SONUÇ	31
4.KAYNAKLAR	32

1.GİRİŞ

Bu raporda kullanıcılara tamirci firmalarını tanıtmak için kullanılan ve kendilerine uygun tamirciyi bulmalarını sağlayan diğer yandan da firmaların kendi tanıtımlarını yaparak kullanıcılar le daha fazla etkileşime geçmelerini sağlayan bir mobil uygulamanın geliştirilme aşamaları anlatılmaktadır.

Teknolojinin gelişmesiyle birlikte mobil uygulamalar yaygınlaşmaya başlamış ve zamanla büyük bir gelişme göstermiştir. Özellikle Flutter ile geliştirilen mobil uygulamaların sayısı oldukça fazladır.

Mobil uygulama geliştirme için Flutter, tüm ekran boyutlarında daha duyarlı davranan özellikleri sayesinde artık birçok çevrimiçi uygulamayı daha az maliyetle etkin bir şekilde geliştirmeye olanak tanımaktadır. Ayrıca Flutter'ın en büyük avantajlarından biri, geliştiricilerin aynı anda iOS ve Android platformlarında çekici uygulamalar oluşturabileceği çapraz platform geliştirmedir.

TAMİRCİNDEN.COM mobil uygulaması Flutter SDK kullanılarak Dart programlama dili ile geliştirilmiştir.

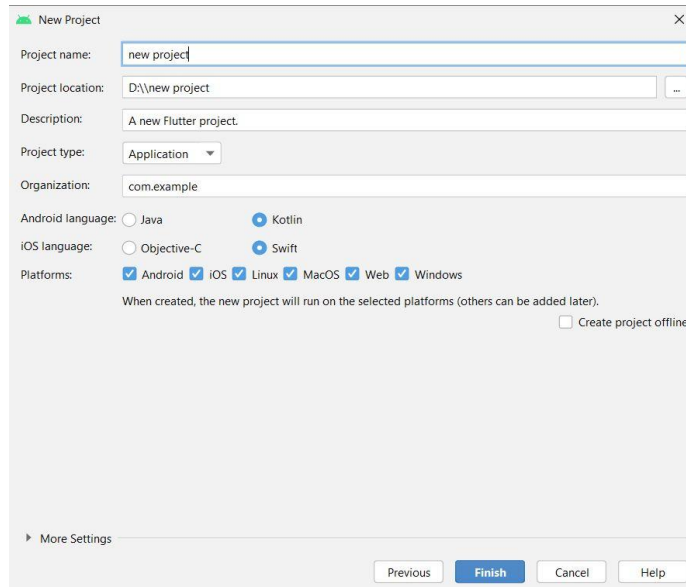
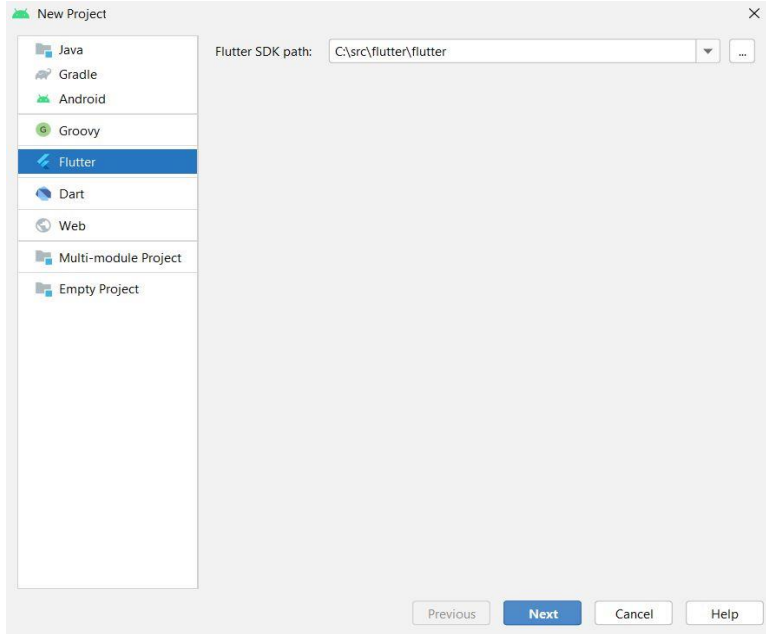
Proje temelde 7 sayfadan oluşmaktadır. Bunlar; Ana Sayfa, Hakkımızda Sayfası, İletişim Sayfası, Tüm Tamirciler Sayfası, Tamirci Detay Sayfası, Tamirci CRUD işlemleri sayfası ve Harita Erişim sayfasıdır.

Projede süreç ve değişiklik takibi için GİT sistemi kullanılmıştır. İlerleyen bölümlerde proje geliştirme detaylarından adım adım bahsedilecektir.

2. TAMİRCİM.COM FLUTTER UYGULAMASI

2.1.Projenin açılması ve Proje Veri tabanının Oluşturulması

Projeyi geliştirmek için Android Studio IDE'si kullanılmıştır.Projeyi açmak için Android Studio açılmıştır ve New Project butonuna tıklanarak aşağıdaki işlemler gerçekleştirilmiştir:

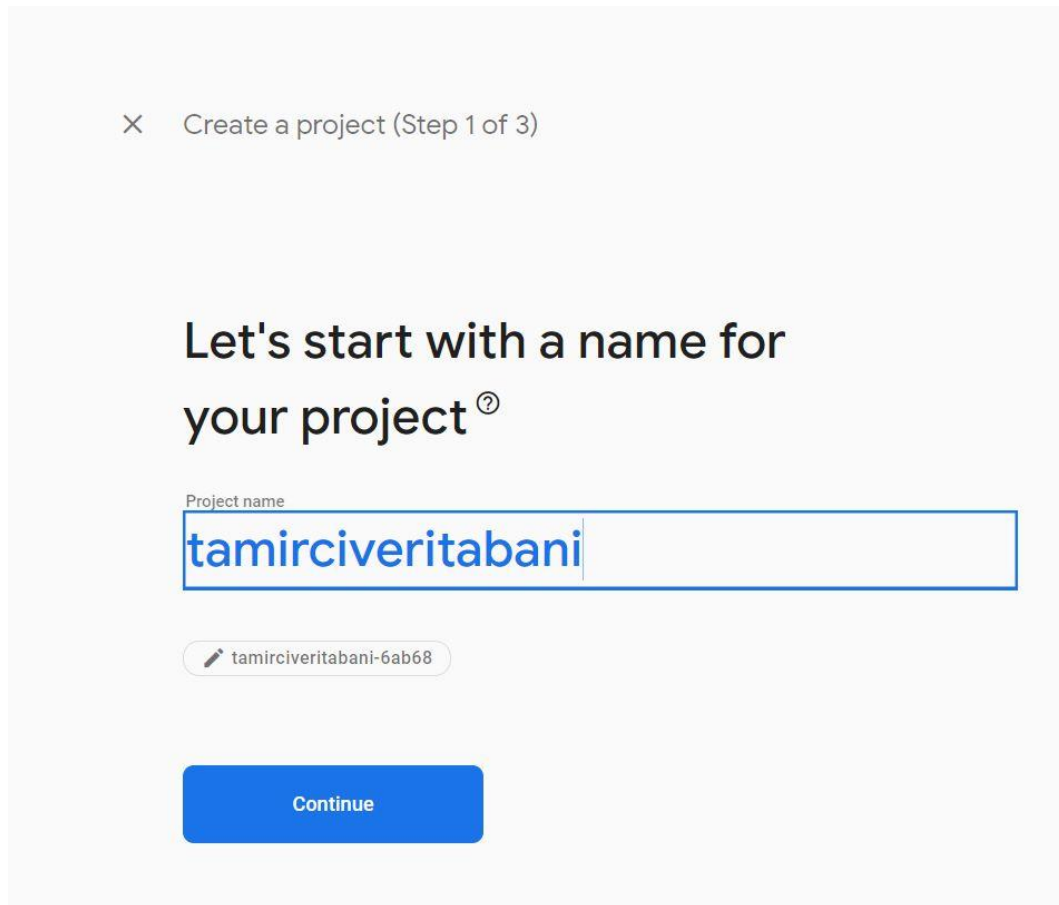


Şekil 2.1.1 Proje Açılması

Projede veri tabanı olarak verilerin internet üzerinde depolanabildiği, bulut tabanlı Firestore kullanılmıştır.

Firebase veri tabanı kullanmak için Google üzerinden oturum açmak gerekmektedir. Sonrasında Oturum açmak için <https://firebase.google.com> adresine gidilmiştir.

Siteye girdikten sonra öncelikle oluşturulacak veri tabanı için bir proje oluşturulmuştur. Add project diyerek 3 adımlı proje ekleme adımına başlanmıştır. Veri tabanı oluşturmak için “tamirciveritabani” proje adı belirlenmiştir. Aşağıda gösterilmektedir:



Şekil 2.1.2 Veri tabanı Projesi Oluşturulması

Veri tabanı bu şekilde oluşturulmuştur. Veri tabanına tablolar eklenmeye başlanmıştır. Örnek bir tablo ekleme aşağıda gösterilmektedir:

Add a document
Parent path
/Tamirciler

Document ID ?
MEEId9JmNP6VCdcJcaf9

Field	Type	Value
adi	string	
adres	string	
aktiflik	string	
fotograf	string	
Field	Type	Value

Cancel Save

Şekil 2.1.3 Tamirciler Tablosunun Oluşturulması

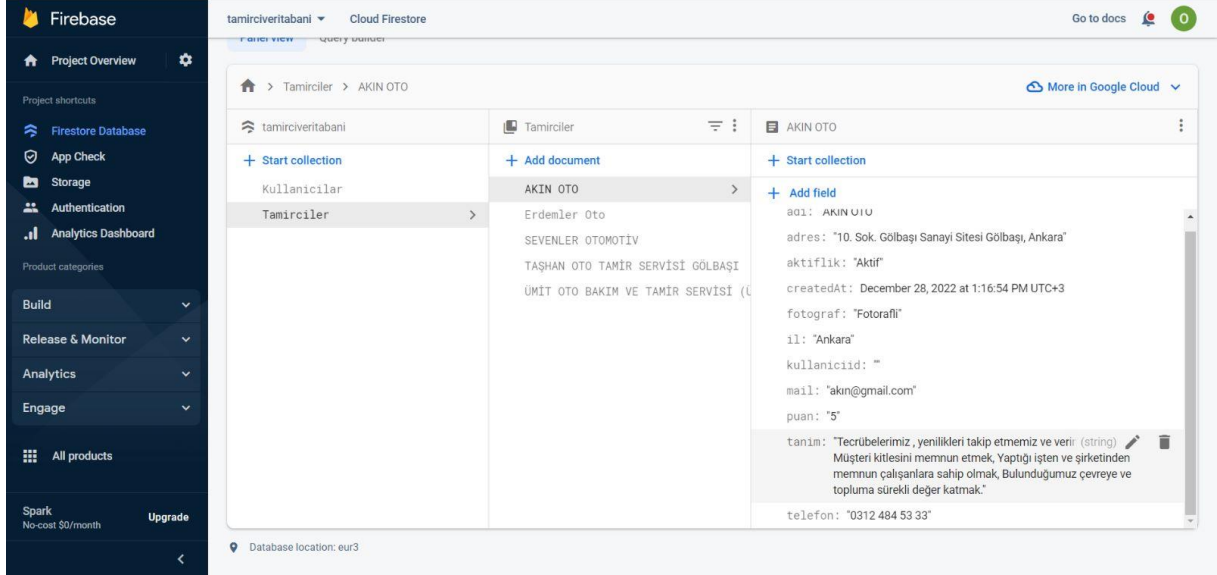
Add a document
Parent path
/Tamirciler

Field	Type	Value
il	string	
mail	string	
puan	string	
tanim	string	
telefon	string	
+ Add field		

Cancel Save

Şekil 2.1.4 Tamirciler Tablosunun Oluşturulması

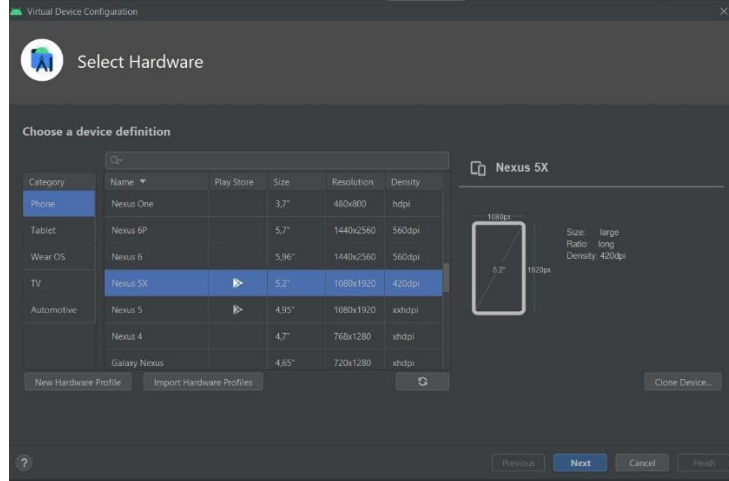
Veri Tabanı tablolarının son hali aşağıdaki gibidir:



Şekil 2.1.5 Veri Tabanının Son Görüntüsü

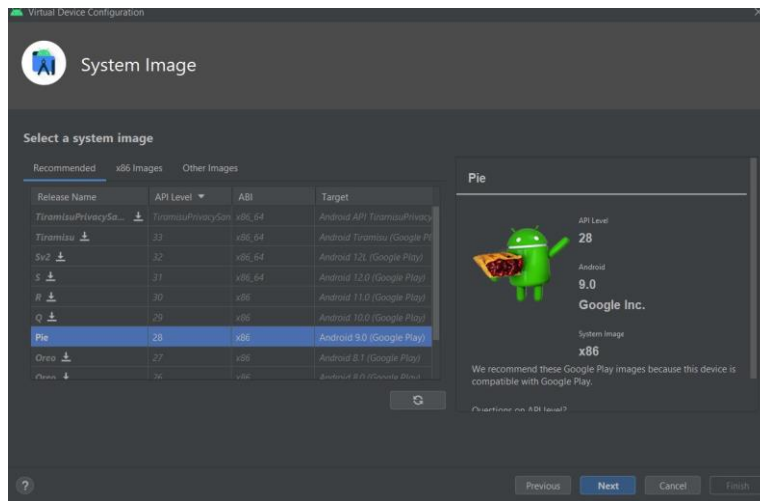
2.2.Proje Emülatörünün Oluşturulması

Yazılan Flutter kodlarının yansımalarının canlı olarak görebilmesi için bir emülatör kurulmalıdır. İlk olarak Android Studio'nun üst kısmındaki menüden “Tools -> AVD Manager” seçeneği seçilir.Burada da ortadaki “Create Virtual Device...” butonuna tıklanır. Ve aşağıdaki gibi bir ekran açılır.



Şekil 2.2.1 Emülatör Modelinin Seçilmesi

Bu adımda oluşturulmak istenen emülatörün modeli seçilir. Sonrasında cihazda kullanılacak işletim sistemi seçilir ve indirilir.



Şekil 2.2.2 İşletim Sistemi Seçilmesi

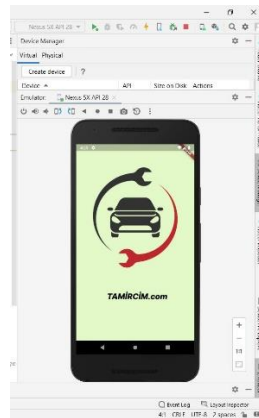
2.3.Splash Screen Oluşturulması

Birçok uygulamayı açtığımız zaman karşımıza genelde logo çıkar ve birkaç saniye içinde uygulama açılır. Bu karşılama ekranına splash screen adı verilir. Türkçe olarak ifade etmek istersek karşılama ekranı, açılış ekranı ifadeler kullanılabilir.

Splash Screen için yazılan kodlar aşağıda verilmiştir:

```
main.dart
1 import 'package:flutter/material.dart';
2 import 'package:tamircim/anasayfa.dart';
3 import 'package:firebase_core/firebase_core.dart';
4
5 Future<void> main() async {
6   WidgetsFlutterBinding.ensureInitialized();
7   await Firebase.initializeApp();
8   runApp(MaterialApp(home: MyApp2()));
9 }
10 class MyApp2 extends StatefulWidget {
11   @override
12   _MyApp2State createState() => _MyApp2State();
13 }
14 class _MyApp2State extends State<MyApp2> {
15   @override
16   void initState() {
17     super.initState();
18     Future.delayed(const Duration(seconds: 3), () {
19       Navigator.pushReplacement(
20         context, MaterialPageRoute(builder: (context) => const HomePage()));
21     }); // Future.delayed
22   }
23   @override
24   Widget build(BuildContext context) {
25     return Scaffold(
26       body: Container(
27         decoration: const BoxDecoration(
28           image: DecorationImage(
29             image: AssetImage("assets/splash.jpg"), fit: BoxFit.cover)), // DecorationImage, BoxDecoration
30       ), // Container
31     ); // Scaffold
```

Şekil 2.3.1 Splash Screen Yazılan Kodlar



Şekil 2.3.2 Splash Screen Sayfasının Emulatordeki Görünümü

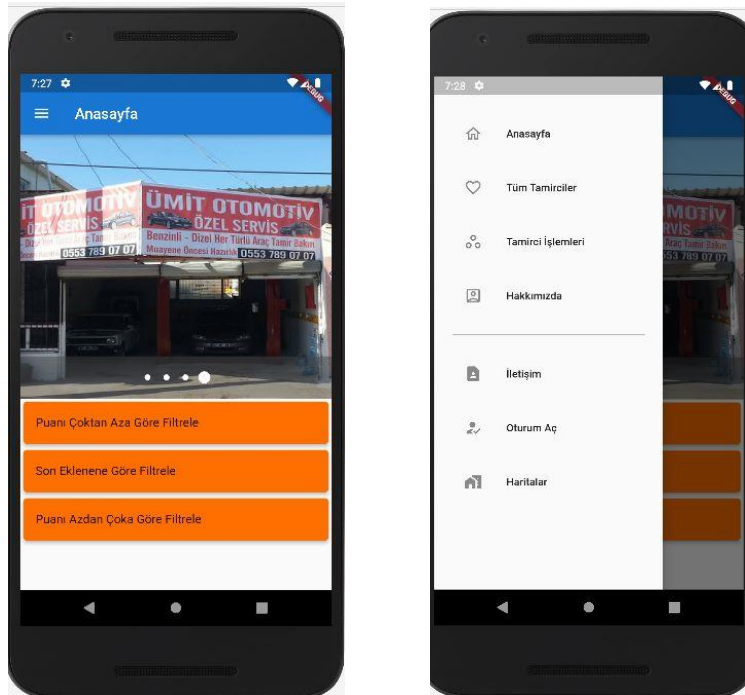

```

127 title: Text('Tamirci İşlemleri'),
128 onTap: () {
129   Navigator.pop(context);
130   Navigator.of(context).push(MaterialPageRoute(
131     builder: (context) => MyApp(),
132   )); // MaterialPageRoute
133 },
134 ), // ListTile
135 ListTile(
136   leading: Icon(Icons.account_box_outlined),
137   title: Text('Hakkımızda'),
138   onTap: () {
139     Navigator.pop(context);
140     Navigator.of(context).push(MaterialPageRoute(
141       builder: (context) => hakkimizda(),
142     )); // MaterialPageRoute
143   },
144 ), // ListTile
145 Divider(color: Colors.black54),
146 ListTile(
147   leading: Icon(Icons.contact_page),
148   title: Text('İletişim'),
149   onTap: () {
150     Navigator.pop(context);
151     Navigator.of(context).push(MaterialPageRoute(
152       builder: (context) => iletisim(),
153     )); // MaterialPageRoute
154   },
155 ), // ListTile
156 ListTile(
157   leading: Icon(Icons.how_to_reg),
158   title: Text('Oturum Aç'),
159   onTap: () {
160     Navigator.pop(context);
161     Navigator.of(context).push(MaterialPageRoute(
162       builder: (context) => kullanici(),
163     )); // MaterialPageRoute
164   },
165 ), // ListTile
166 ListTile(
167   leading: Icon(Icons.maps_home_work),
168   title: Text('Haritalar'),
169   onTap: () {
170     Navigator.pop(context);
171     Navigator.of(context).push(MaterialPageRoute(
172       builder: (context) => locationAndMapLauncher(),
173     )); // MaterialPageRoute
174   },
175 ), // ListTile
176 ],
177 ) // Wrap
178 ); // Container
179 }

```

Şekil 2.4.1 Ana Sayfa için Yazılan Kodlar

Emülatördeki görünüm aşağıda verilmiştir:



Şekil 2.4.2 Ana Sayfanın Emülatördeki Görünümü

Yukarıdaki görünüme bakıldığında filtreleme button yapıları görülür. Örnek olarak “Puanı Çoktan Aza Göre Filtrele” butonuna basıldığında

onTap: () {

Navigator.of(context).push(MaterialPageRoute(

builder: (context) => tamircidetaypuan(),

)); },

ifadesi ile “tamircidetaypuan” StatelessWidget çağırılır. Bu yapı puan.dart sayfası içerisinde yer alır.

Bu sayfada yer alan;

**“final Stream<QuerySnapshot> _usersStream =
FirestoreFirestore.instance**

.collection('Tamirciler')

.where('puan')

.orderBy('puan', descending: true)

.snapshots(); “

komutu ile tamirciler puan özelliklerine göre çoktan aza olacak şekilde veri tabanından çekilir ve listelenir.

Bu “puan.dart” sayfasına yazılan kodlar ve emülatördeki görünüm aşağıda verilmiştir:

```

1 import 'package:flutter/material.dart';
2 import 'package:cloud_firestore/cloud_firestore.dart';
3
4 class tamircidetaypuan extends StatelessWidget {
5   const tamircidetaypuan({Key? key}) : super(key: key);
6   @override
7   Widget build(BuildContext context) {
8     return Scaffold(
9       appBar: AppBar(
10        title: Text("Tamirci Detay Çoktan Aza Göre Puan Filtrelenmesi"),
11        // AppBar
12        body: tunTamirciler(),
13      ); // Scaffold
14    }
15  }
16
17 class tunTamirciler extends StatefulWidget {
18   @override
19   _tunTamircilerState createState() => _tunTamircilerState();
20 }
21
22 class _tunTamircilerState extends State<tunTamirciler> {
23   final Stream<QuerySnapshot> _usersStream = FirebaseFirestore.instance
24     .collection('tamirciler')
25     .where('puan')
26     .orderBy('puan', descending: true)
27     .snapshots();
28   @override
29   Widget build(BuildContext context) {
30     return StreamBuilder<QuerySnapshot>(
31       stream: _usersStream,
32       builder: (BuildContext context, AsyncSnapshot<QuerySnapshot> snapshot) {
33         if (snapshot.hasError) {
34           return Text('Something went wrong');
35         }
36         if (snapshot.connectionState == ConnectionState.waiting) {
37           return Text("Loading");
38         }
39         return ListView(
40           children: snapshot.data!.docs.map((DocumentSnapshot document) {
41             Map<String, dynamic> data =
42               document.data()! as Map<String, dynamic>;
43             return Card(
44               color: Colors.amber[900],
45               shape: RoundedRectangleBorder(
46                 borderRadius: BorderRadius.circular(5)), // RoundedRectangleBorder
47               elevation: 4,
48               child: ListTile(
49                 textColor: Colors.white,
50                 leading: CircleAvatar(
51                   radius: 40,
52                   backgroundImage: AssetImage(
53                     "assets/default.jpg",
54                   ), // AssetImage
55                   // CircleAvatar
56                   tileColor: Colors.lightBlue,
57                   title: Text(
58                     data['adi'],
59                     style: TextStyle(fontWeight: FontWeight.bold, fontSize: 20),
60                   ), // Text
61                   subtitle: Column(
62                     crossAxisAlignment: CrossAxisAlignment.start,
63                     children: [
64                       Text(data['adres']),
65                       Text(data['aktiflik']),
66                       Text(data['fotograf']),
67                       Text(data['il']),
68                       Text(data['mail']),
69                       Text(data['puan']),
70                       Text(data['tanim']),
71                       Text(data['telefon']),
72                     ],
73                   ), // Column
74                 ), // ListTile
75               ), // Card
76             ).toList(),
77           ); // ListView
78         },
79       ); // StreamBuilder
80     }
81   }

```

Şekil 2.4.3 Puana Göre Filtreleme Sayfası

Bu sayfanın emülatördeki görünümü aşağıda verilmiştir:

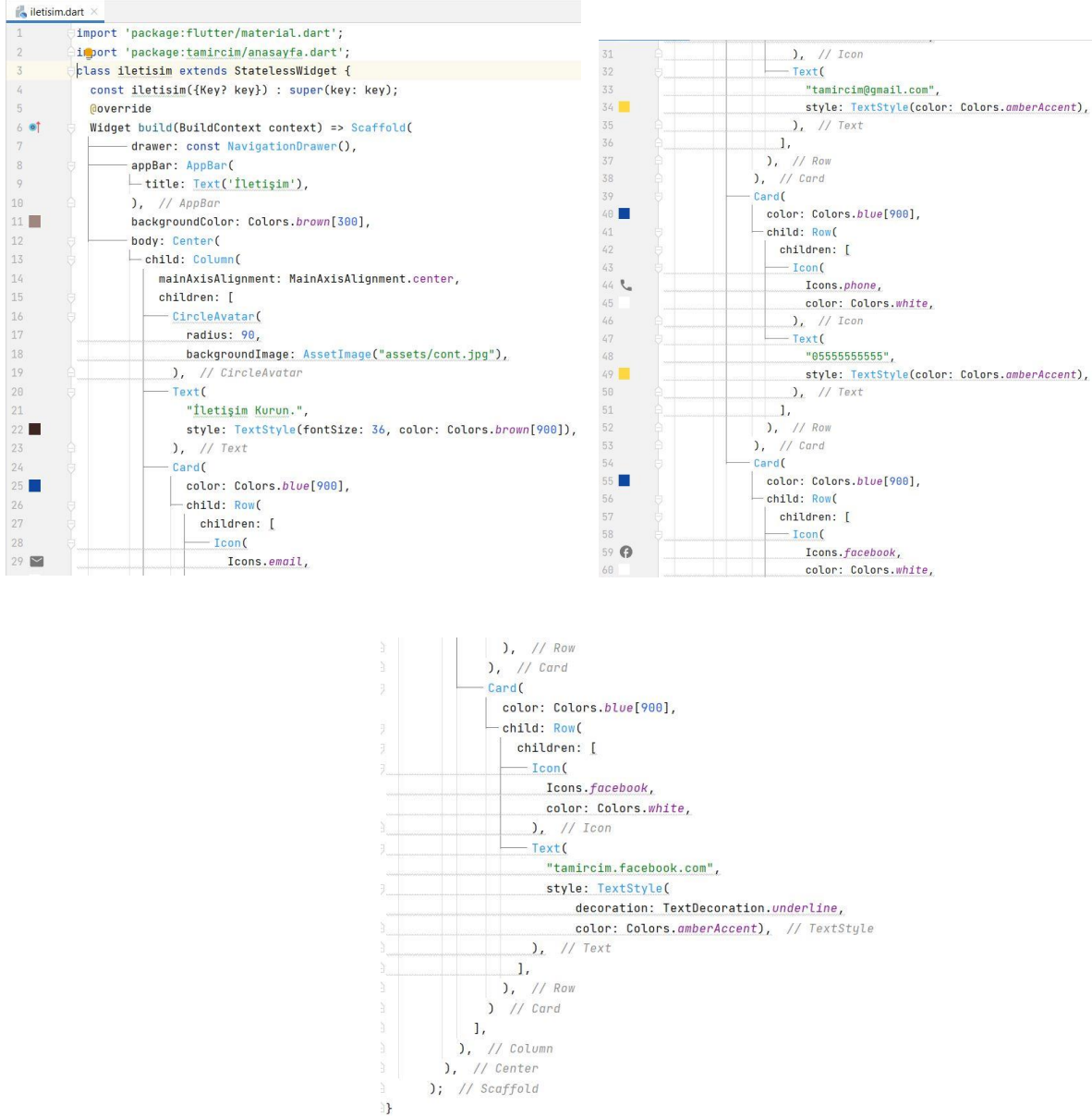


Şekil 2.4.4 Puana Göre Çoktan Aza Filtreleme Sayfasının Emülatörde Görünümü

2.5. İletişim Sayfası Oluşturulması

Bu sayfada kullanıcıların uygulama oluşturucuları ile iletişim kurabilmeleri için çeşitli iletişim bilgileri yer almaktadır.

Yazılan kodlar ve sayfanın emülatördeki görünümünü aşağıda gösterilmiştir:



Şekil 2.5.1 İletişim Sayfası İçin Yazılan Kodlar

Emülatördeki görünüm aşağıda verilmiştir:



Şekil 2.5.2 İletişim Sayfasının Emülatördeki Görünümü

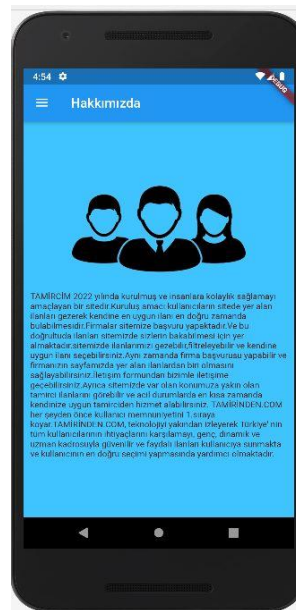
2.6. Hakkımızda Sayfası Oluşturulması

Bu sayfada kullanıcılara Mobil uygulama hakkında bilgiler verilmekte ve uygulamanın amacı anlatılmaktadır.

Yazılan kodlar ve sayfanın emülatördeki görünümü aşağıda gösterilmiştir:

```
hakkimizda.dart
1 import 'package:flutter/material.dart';
2 import 'package:tamircim/anasayfa.dart';
3 class hakkimizda extends StatelessWidget {
4   const hakkimizda({Key? key}) : super(key: key);
5   @override
6   Widget build(BuildContext context) => Scaffold(
7     drawer: const NavigationDrawer(),
8     appBar: AppBar(
9       title: Text('Hakkımızda'),
10    ), // AppBar
11    backgroundColor: Colors.lightBlueAccent,
12    body: Center(
13      child: Column(
14        mainAxisAlignment: MainAxisAlignment.center,
15        children: [
16          ClipRRect(
17            borderRadius: BorderRadius.all(Radius.circular(10.0)),
18            child: Image.asset('assets/about.jpg'),
19          ), // ClipRRect
20          Padding(
21            padding: EdgeInsets.all(10.0),
22            child: Text(
23              "TAMİRCİM 2022 yılında kurulmuş ve insanlara kolaylık sağlamayı amaçlayan bir sitedir."
24              "Kuruluş amacı kullanıcıların sitede yer alan ilanları gezerek kendine en uygun ilanı en doğru zamanda bulabilmesidir."
25              "Ve bu doğrultuda ilanları sitemizde sizlerin bakabilmesi için yer almaktadır.sitemizde ilanlarımızı gezebilirsiniz,filtreleyebilirsiniz."
26              "Aynı zamanda firma başvurusu yapabilir ve firmanızın sayfamızda yer alan ilanlardan biri olmasını sağlayabilirsiniz."
27              "TAMİRCİM.com her şeyden önce kullanıcı memnuniyetini 1.sıraya koyar."
28              "TAMİRCİM.com, teknolojiyi yakından izleyerek Türkiye' nin tüm kullanıcılarının ihtiyaçlarını karşılamayı,"
29              "genc, dinamik ve uzman kadrosuyla güvenilir ve faydalı ilanları kullanıcıya sunmakta ve kullanıcının en doğru seçimi yapmasını desteklemektedir."
30            style: TextStyle(fontSize: 13, color: Colors.brown[900]),
31          ),
32        ],
33      ),
34    ),
35  );
```

Şekil 2.6.1 Hakkımızda Sayfası Yazılan Kodlar



Şekil 2.6.2 Hakkımızda Sayfasının Emülatördeki Görünümü

2.7 Tüm Tamirciler Sayfası Oluşturulması

Bu sayfada veri tabanının da yer alan tüm tamircilerin son eklenenden ilk eklenene olacak sırada sadece ad ve adres bilgisi bir sorgu ile çekilir ve kullanıcılara liste şeklinde gösterilir. Burada tamircilerin sadece ad ve adres bilgisi çekilmektedir. ListView yapısının Card özelliğine “OnTap();” özelliği verilerek Tamircilerin tüm detaylarının gösterildiği sayfaya gidilmektedir.

Yazılan kodlar ve sayfanın emülatördeki görünümü aşağıda gösterilmiştir:

```
favourites_pages.dart
1 import 'package:cloud_firestore/cloud_firestore.dart';
2 import 'package:flutter/cupertino.dart';
3 import 'package:flutter/material.dart';
4 import 'package:tamircim/anasayfa.dart';
5 import 'package:tamircim/tamircidetay.dart';
6
7 class FavouritesPage extends StatelessWidget {
8   const FavouritesPage({Key? key}) : super(key: key);
9   @override
10  Widget build(BuildContext context) {
11    return Scaffold(
12      drawer: const NavigationDrawer(),
13      appBar: AppBar(
14        title: Text("Tamirci Listeleri"),
15      ), // AppBar
16      body: tumTamirciler(),
17    ); // Scaffold
18  }
19 }
20
21 class tumTamirciler extends StatefulWidget {
22   @override
23   _tumTamircilerState createState() => _tumTamircilerState();
24 }
25
26 class _tumTamircilerState extends State<tumTamirciler> {
27   final Stream<QuerySnapshot> _usersStream = FirebaseFirestore.instance
28     .collection('Tamirciler')
29     .where('adi')
30     .orderBy('createdAt', descending: true)
31     .snapshots();
32   @override
33   Widget build(BuildContext context) {
34     return StreamBuilder<QuerySnapshot>(
35       stream: _usersStream,
36       builder: (BuildContext context, AsyncSnapshot<QuerySnapshot> snapshot) {
37         if (snapshot.hasError) {
38           return Text('Something went wrong');
39         }
40
41         if (snapshot.connectionState == ConnectionState.waiting) {
42           return Text("Loading");
43         }
44
45         return ListView(
46           children: snapshot.data!.docs.map((DocumentSnapshot document) {
47             Map<String, dynamic> data =
48               document.data()! as Map<String, dynamic>;
49             return Card(
50               color: Colors.amber[900],
51               shape: RoundedRectangleBorder(
52                 borderRadius: BorderRadius.circular(5)), // RoundedRectangleBorder
53               elevation: 4,
54               child: ListTile(
55                 title: Text(data['adi']),
56                 subtitle: Text(data['adres']),
57                 onTap: () {
58                   Navigator.pop(context);
59                   Navigator.of(context).push(MaterialPageRoute(
60                     builder: (context) => tamircidetay(),
61                   )); // MaterialPageRoute
62                 }
63               ),
64             );
65           }).toList(),
66         );
67       },
68     );
69   }
70 }
```

Şekil 2.7.1 Tüm Tamirciler Sayfası Yazılan Kodlar



Şekil 2.7.2 Tüm Tamirciler Sayfası Emülatördeki Görünüm

2.8. Tamirci Detay Sayfası Oluşturulması

Bu sayfada veri tabanında yer alan tamircilerin tüm bilgileri gösterilmektedir.

Sayfaya yazılan kodlar ve emülatördeki görünümü aşağıdaki gibidir:

```
tamircidetay.dart
1 import 'package:flutter/material.dart';
2 import 'package:cloud_firestore/cloud_firestore.dart';
3 import 'package:flutter/cupertino.dart';
4 import 'package:tamircim/puan.dart';
5
6 class tamircidetay extends StatelessWidget {
7   const tamircidetay({Key? key}) : super(key: key);
8   @override
9   Widget build(BuildContext context) {
10     return Scaffold(
11       appBar: AppBar(
12         title: Text("Tamirci Detay Listeleri"),
13       ), // AppBar
14       body: tumTamirciler(),
15     ); // Scaffold
16   }
17 }
18
19 class tumTamirciler extends StatefulWidget {
20   @override
21   _tumTamircilerState createState() => _tumTamircilerState();
22 }
23
24 class _tumTamircilerState extends State<tumTamirciler> {
25   final Stream<QuerySnapshot> _usersStream = FirebaseFirestore.instance
26     .collection('Tamirciler')
27     .where('adi')
28     .orderBy('createdAt', descending: true)
29     .snapshots();
30   @override
31   Widget build(BuildContext context) {
32     return StreamBuilder<QuerySnapshot>({
33       stream: _usersStream,
34       builder: (BuildContext context, AsyncSnapshot<QuerySnapshot> snapshot) {
35         if (snapshot.hasError) {
36           return Text('Something went wrong');
37         }
38         if (snapshot.connectionState == ConnectionState.waiting) {
39           return Text("Loading");
40         }
41         return ListView(
42           children: snapshot.data!.docs.map((DocumentSnapshot document) {
43             Map<String, dynamic> data =
44               document.data()! as Map<String, dynamic>;
45             return Card(
46               color: Colors.amber[900],
47               shape: RoundedRectangleBorder(
48                 borderRadius: BorderRadius.circular(5)), // RoundedRectangleBorder
49               elevation: 4,
50               child: ListTile(
51                 textColor: Colors.white,
52                 leading: CircleAvatar(
53                   radius: 40,
54                   backgroundImage: AssetImage(
55                     "assets/default.jpg",
56                   ), // AssetImage
57                 ), // CircleAvatar
58                 title: Text(
59                   data['adi'],
60                   style: TextStyle(fontWeight: FontWeight.bold, fontSize: 20),
61                 ), // Text
62                 subtitle: Column(
63                   crossAxisAlignment: CrossAxisAlignment.start,
64                   children: [
65                     Text(data['adres']),
66                     Text(data['aktiflik']),
67                     Text(data['fotograf']),
68                     Text(data['il']),
69                     Text(data['mail']),
70                     Text(data['puan']),
71                     Text(data['tanim']),
72                     Text(data['telefon']),
73                   ],
74                 ), // Column
75               ), // ListTile
76             ).toList(),
77           ); // ListView
78         },
79       ); // StreamBuilder
80     }
81   }
82 }
```

Şekil 2.8.1 Tamirci Detay Sayfasına Yazılan Kodlar



Şekil 2.8.2 Tamirci Detay Sayfası Emülatördeki Görünümü

Yukarıdaki şekilde de görüldüğü üzere tamircilerin veri tabanında yer alan tüm bilgiler sayfada listelenmektedir.

```
“final Stream<QuerySnapshot> _usersStream = FirebaseFirestore.instance  
.collection('Tamirciler')  
.where('adi')  
.orderBy('createdAt', descending: true)  
.snapshots();” komutu ile tamirciler son eklenen tamirciden ilk eklenen tamirciye  
şeklinde sıralanarak çekilir.
```

2.9. Tamirci CRUD Sayfası Oluşturulması

Bu sayfada veri tabanında yer alan tamirciler üzerinde CRUD işlemleri yapılmaktadır. Bu sayfada veri tabanına tamirci ekleme,güncelleme ve tamirci silme işlemleri yapılmaktadır.

Sayfaya yazılan kodlar ve emülatördeki görünümü aşağıda gösterilmiştir:

```
tamirci.dart
1 import 'package:cloud_firestore/cloud_firestore.dart';
2 import 'package:firebase_auth/firebase_auth.dart';
3 import 'package:firebase_core/firebase_core.dart';
4 import 'package:firebase_storage/firebase_storage.dart';
5 import 'package:flutter/material.dart';
6 import 'dart:io';
7
8 Future<void> main() async {
9   WidgetsFlutterBinding.ensureInitialized();
10  await Firebase.initializeApp();
11  runApp(const MyApp());
12 }
13
14 class MyApp extends StatelessWidget {
15   const MyApp({Key? key}) : super(key: key);
16
17   @override
18   Widget build(BuildContext context) {
19     return MaterialApp(
20       home: Iskele(),
21     ); // MaterialApp
22   }
23 }
24
25 class Iskele extends StatefulWidget {
26   const Iskele({Key? key}) : super(key: key);
27
28   @override
29   State<Iskele> createState() => _IskeleState();
30 }
31
32 class _IskeleState extends State<Iskele> {
33   TextEditingController t1 = TextEditingController();
34   TextEditingController t2 = TextEditingController();
35   TextEditingController t3 = TextEditingController();
36   TextEditingController t4 = TextEditingController();
37   TextEditingController t5 = TextEditingController();
38   TextEditingController t6 = TextEditingController();
39   TextEditingController t7 = TextEditingController();
40   TextEditingController t8 = TextEditingController();
41   TextEditingController t9 = TextEditingController();
42
43   get onPressed => null;
44   get child => null;
45
46   FirebaseAuth auth = FirebaseAuth.instance;
47
48   tamirciekle() {
49     FirebaseFirestore.instance.collection("Tamirciler").doc(t1.text).set({
50       'kullaniciid': auth.currentUser?.uid,
51       'adi': t1.text,
52       'adres': t2.text,
53       'aktiflik': t3.text,
54       'fotograf': t4.text,
55       'il': t5.text,
56       'mail': t6.text,
57       'puan': t7.text,
58       'tanim': t8.text,
59       'telefon': t9.text,
60       'createdAt': DateTime.now()
61     }).whenComplete(() => print("Tamirci Eklendi"));
62   }
63
64   tamirciguncelle() {
65     FirebaseFirestore.instance.collection("Tamirciler").doc(t1.text).update({
66       'adi': t1.text,
67       'adres': t2.text,
68       'aktiflik': t3.text,
69       'fotograf': t4.text,
70       'il': t5.text,
71       'mail': t6.text,
72       'puan': t7.text,
73       'tanim': t8.text,
74       'telefon': t9.text,
75       'createdAt': DateTime.now()
76     }).whenComplete(() => print("Tamirci Guncellendi"));
77   }
78
79   tamircisilme() {
80     FirebaseFirestore.instance.collection("Tamirciler").doc(t1.text).delete();
81   }
82
83   @override
84   Widget build(BuildContext context) {
85     return Scaffold(
86       margin: EdgeInsets.all(50),
87       child: Center(
88         child: SingleChildScrollView(
89           child: Column(
90             children: [
91               TextField(
92                 controller: t1,
93                 decoration: InputDecoration(
94                   labelText: "Tamircinin Adı Giriniz",
95                   prefixIcon: Icon(Icons.people),
96                   enabledBorder: OutlineInputBorder(
97                     borderSide: BorderSide(width: 3, color: Colors.greenAccent),
98                     borderRadius: BorderRadius.circular(40.0),
99                   ), // OutlineInputBorder
100                 ), // InputDecoration
101               ), // TextField
102               TextField(
103                 controller: t2,
104                 decoration: InputDecoration(
105                   labelText: "Tamircinin Adresini Giriniz.",
106                   prefixIcon: Icon(Icons.note_alt_outlined),
107                   enabledBorder: OutlineInputBorder(
108                     borderSide: BorderSide(width: 3, color: Colors.greenAccent),
109                     borderRadius: BorderRadius.circular(40.0),
110                   ), // OutlineInputBorder
111                 ), // InputDecoration
112               ), // TextField
113               TextField(
114                 controller: t3,
115                 decoration: InputDecoration(
116                   labelText: "Tamirci Aktivlik Durumunu Giriniz.",
117                   prefixIcon: Icon(Icons.notifications_active),
118                   enabledBorder: OutlineInputBorder(
119                     borderSide: BorderSide(width: 3, color: Colors.greenAccent),
120                   ), // OutlineInputBorder
121                 ), // InputDecoration
122               ), // TextField
123             ],
124           ),
125         ),
126       ),
127     );
128   }
129 }
```



```

120 enabledBorder: OutlineInputBorder(
121   borderSide:
122     BorderSide(width: 3, color: Colors.greenAccent),
123   borderRadius: BorderRadius.circular(40.0),
124 ), // OutlineInputBorder
125 ), // InputDecoration
126 ), // TextField
127
128 TextField(
129   controller: t4,
130   decoration: InputDecoration(
131     labelText: "Tamirci Fotoğrafını Giriniz.",
132     prefixIcon: Icon(Icons.photo_library_outlined),
133     enabledBorder: OutlineInputBorder(
134       borderSide:
135         BorderSide(width: 3, color: Colors.greenAccent),
136       borderRadius: BorderRadius.circular(40.0),
137     ), // OutlineInputBorder
138   ), // InputDecoration
139 ), // TextField
140
141 TextField(
142   controller: t5,
143   decoration: InputDecoration(
144     labelText: "Tamircinin İlini Giriniz",
145     prefixIcon: Icon(Icons.home_outlined),
146     enabledBorder: OutlineInputBorder(
147       borderSide:
148         BorderSide(width: 3, color: Colors.greenAccent),
149       borderRadius: BorderRadius.circular(40.0),
150     ), // OutlineInputBorder
151   ), // InputDecoration
152 ), // TextField
153
154 TextField(
155   controller: t6,
156   decoration: InputDecoration(
157     labelText: "Tamircinin Mailini Giriniz",
158     prefixIcon: Icon(Icons.mail_outline),
159     enabledBorder: OutlineInputBorder(
160       borderSide:
161         BorderSide(width: 3, color: Colors.greenAccent),
162       borderRadius: BorderRadius.circular(40.0),
163     ), // OutlineInputBorder
164   ), // InputDecoration
165 ), // TextField
166
167 TextField(
168   controller: t7,
169   decoration: InputDecoration(
170     labelText: "Tamircinin Puanını Giriniz",
171     prefixIcon: Icon(Icons.control_point),
172     enabledBorder: OutlineInputBorder(
173       borderSide:
174         BorderSide(width: 3, color: Colors.greenAccent),
175       borderRadius: BorderRadius.circular(40.0),
176     ), // OutlineInputBorder
177   ), // InputDecoration
178 ), // TextField
179
180 TextField(
181   controller: t8,
182   decoration: InputDecoration(
183     labelText: "Tamircinin Tanımını Giriniz",
184     prefixIcon: Icon(Icons.description),
185   ), // InputDecoration
186 ), // TextField
187
188 TextField(
189   controller: t9,
190   decoration: InputDecoration(
191     labelText: "Tamircinin Telefonunu Giriniz",
192     prefixIcon: Icon(Icons.phone_in_talk),
193     enabledBorder: OutlineInputBorder(
194       borderSide:
195         BorderSide(width: 3, color: Colors.greenAccent),
196       borderRadius: BorderRadius.circular(40.0),
197     ), // OutlineInputBorder
198   ), // InputDecoration
199 ), // TextField
200
201 Row(
202   children: [
203     ElevatedButton(
204       child: Text("Ekle"),
205       onPressed: tamirciekle,
206     ), // ElevatedButton
207     ElevatedButton(
208       child: Text("Güncelle"),
209       onPressed: tamirciguncelle,
210     ), // ElevatedButton
211     ElevatedButton(
212       child: Text("Sil"),
213       onPressed: tamircisilme,
214     ), // ElevatedButton
215   ],
216 ), // Row
217
218 ), // Column
219
220 ), // SingleChildScrollView
221
222 ), // Center
223
224 ), // Container

```

Şekil 2.9.1 Tamirci CRUD Sayfası Yazılan Kodlar

Sayfanın emülatördeki görünümü aşağıda gösterilmiştir:



Şekil 2.9.2 Tamirci CRUD Sayfası Emülatördeki Görünümü

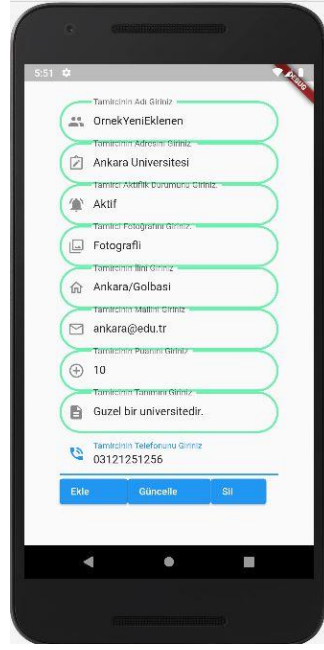
Yukarıda görüldüğü üzere

```
“Row(  
    children: [  
        ElevatedButton(  
            child: Text("Ekle"),  
            onPressed: tamirciekle,  
        ),  
        ElevatedButton(  
            child: Text("Güncelle "), onPressed: tamirciguncelle),  
        ElevatedButton(child: Text("Sil "), onPressed: tamircisilme), ],)”
```

İfadesi ile sayfaya button yapıları eklenmiştir. Bu button yapılarının onPressed özelliği kullanılarak her birine basıldığında hangi fonksiyonun çağırılacağı eklenmiştir. Örnek olarak ilk button' basıldığında “tamirciekle” fonksiyonu çağrılır.

```
“tamirciekle() {  
    FirebaseFirestore.instance.collection("Tamirciler").doc(t1.text).set({  
        'kullaniciid': auth.currentUser?.uid,  
        'adi': t1.text,  
        'adres': t2.text,  
        'aktiflik': t3.text,  
        'fotograf': t4.text,  
        'il': t5.text,  
        'mail': t6.text,  
        'puan': t7.text,  
        'tanim': t8.text,  
        'telefon': t9.text,  
        'createdAt': DateTime.now()  
    }).whenComplete(() => print("Tamirci Eklendi"));  
}”
```

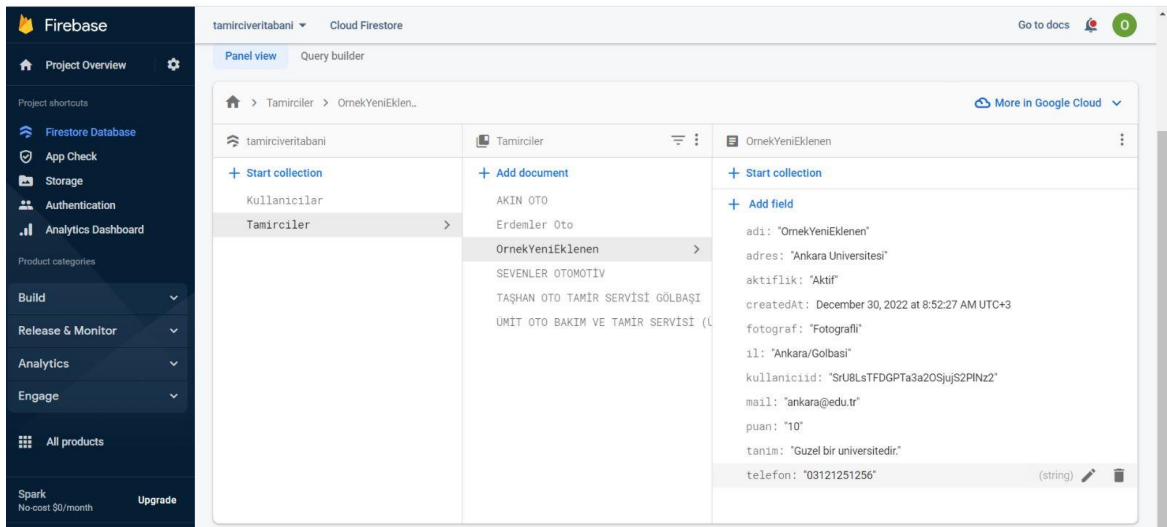
Bu fonksiyon ile sayfada yer alan Text elemanlarına girilen değerler tek tek veri tabanında yer alan tamircilerin özellikleri ile eşleştirilerek veri tabanına yeni tamirci eklemesi yapılır. Örnek bir veri ekleme işlemi ve sonrasında eklenen yeni tamircinin Firestore veri tabanında görünümü aşağıda gösterilmiştir:



Şekil 2.9.3 Örnek Tamirci Ekleme İşlemi

```
Run: main.dart x
Console
W/IInputConnectionWrapper(20993): getSelectedText on inactive InputConnection
W/IInputConnectionWrapper(20993): getTextBeforeCursor on inactive InputConnection
W/IInputConnectionWrapper(20993): getTextBeforeCursor on inactive InputConnection
W/IInputConnectionWrapper(20993): getTextAfterCursor on inactive InputConnection
W/IInputConnectionWrapper(20993): getSelectedText on inactive InputConnection
W/IInputConnectionWrapper(20993): getTextBeforeCursor on inactive InputConnection
W/IInputConnectionWrapper(20993): getTextBeforeCursor on inactive InputConnection
I/TextInputPlugin(20993): Composing region changed by the framework. Restarting the input method.
I/AssistStructure(20993): Flattened final assist data: 764 bytes, containing 1 windows, 3 views
I/flutter (20993): Tamirci Eklendi
```

Şekil 2.9.4 “tamirciekle” Fonksiyonu Çalışması



Şekil 2.9.5 Yeni Eklenen Ögenin FireStore Veri Tabanında Görünümü

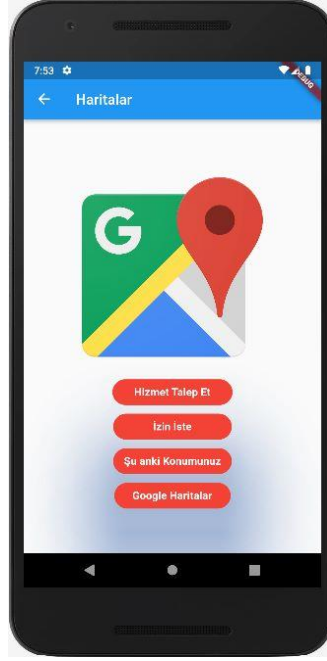
2.10. Haritalar Sayfası

Bu sayfa haritalar üzerinden tamirci araması yapılabilmesini sağlar. Aynı zamanda konum izni açılarak o anda bulunan konum bilgilerini döndürür ve size yakın tamircileri listeler.

Sayfaya yazılan kodlar ve emülatördeki görünümü aşağıdaki gibidir:

```
harita.dart
1 import 'package:flutter/material.dart';
2 import 'package:location/location.dart';
3 import 'package:map_launcher/map_launcher.dart';
4
5 class LocationAndMapLauncher extends StatefulWidget {
6   const LocationAndMapLauncher({Key? key}) : super(key: key);
7
8   @override
9   State<LocationAndMapLauncher> createState() => _LocationAndMapLauncherState();
10 }
11
12 class _LocationAndMapLauncherState extends State<LocationAndMapLauncher> {
13   Location location = Location();
14   @override
15   Widget build(BuildContext context) {
16     return Scaffold(
17       appBar: AppBar(title: Text("Haritalar"),),
18       body: Container(
19         child: Center(
20           child: Column(
21             mainAxisAlignment: MainAxisAlignment.center,
22             children: [
23               ClipRRect(
24                 borderRadius: BorderRadius.all(Radius.circular(10.0)),
25                 child:
26                   Image.asset('assets/maps.jpg',
27                     width: 250, height: 300,
28                   ), // Image.asset
29               ), // ClipRRect
30
31               ElevatedButton(
32                 onPressed: () {
33                   location.requestService().then((value) {
34                     print(value);
35                   });
36                   location.serviceEnabled().then((value) {
37                     print(value);
38                   });
39                 },
40                 child: Text("Hizmet Talep Et"),
41                 style: ElevatedButton.styleFrom(
42                   primary: Colors.red,
43                   padding: EdgeInsets.symmetric(horizontal: 30),
44                   elevation: 100,
45                   shadowColor: Colors.blueAccent,
46                   textStyle: TextStyle(fontSize: 15, fontWeight: FontWeight.bold),
47                   shape: RoundedRectangleBorder(borderRadius: BorderRadius.all(Radius.circular(30))),
48                 ), // ElevatedButton
49               ElevatedButton(
50                 onPressed: () async {
51                   if (await location.hasPermission() !=
52                     PermissionStatus.granted) {
53                     location.requestPermission().then((value) {
54                       print(value);
55                     });
56                   } else {
57                     print("Zaten Açık");
58                   }
59                 },
60               ),
61
62               child: Text("İzin iste"),
63               style: ElevatedButton.styleFrom(
64                 primary: Colors.red,
65                 padding: EdgeInsets.symmetric(horizontal: 55),
66                 elevation: 100,
67                 shadowColor: Colors.blueAccent,
68                 textStyle: TextStyle(fontSize: 15, fontWeight: FontWeight.bold),
69                 shape: RoundedRectangleBorder(borderRadius: BorderRadius.all(Radius.circular(30))),
70               ), // ElevatedButton
71               ElevatedButton(
72                 onPressed: () {
73                   location.getLocation().then((value) {
74                     print(value.longitude);
75                     print(value.latitude);
76                   });
77                 },
78                 child: Text("$u anki Konumunuz"),
79                 style: ElevatedButton.styleFrom(
80                   primary: Colors.red,
81                   padding: EdgeInsets.symmetric(horizontal: 15),
82                   elevation: 100,
83                   shadowColor: Colors.blueAccent,
84                   textStyle: TextStyle(fontSize: 15, fontWeight: FontWeight.bold),
85                   shape: RoundedRectangleBorder(borderRadius: BorderRadius.all(Radius.circular(30))),
86                 ), // ElevatedButton
87               ElevatedButton(
88                 onPressed: () async {
89                   var value = await location.getLocation();
90                   var value = await location.getLocation();
91                   if (await MapLauncher.isMapAvailable(MapType.google)!=null) {
92                     MapLauncher.showMarker(
93                       mapType: MapType.google,
94                       coords: Coords(value.latitude!, value.longitude!),
95                       title: 'Koordinatlar');
96                   }
97                 },
98                 child: Text("Google Haritalar"),
99                 style: ElevatedButton.styleFrom(
100                   primary: Colors.red,
101                   padding: EdgeInsets.symmetric(horizontal: 25),
102                   elevation: 100,
103                   shadowColor: Colors.blueAccent,
104                   textStyle: TextStyle(fontSize: 15, fontWeight: FontWeight.bold),
105                   shape: RoundedRectangleBorder(borderRadius: BorderRadius.all(Radius.circular(30))),
106                 ), // ElevatedButton
107             ], // Column
108           ), // Center
109         ), // Container
110       ), // Scaffold
111     );
112 }
```

Şekil 2.10.1 Haritalar Sayfasına Yazılan Kodlar



Şekil 2.10.2 Haritalar Sayfasının Emülatördeki Görünümü

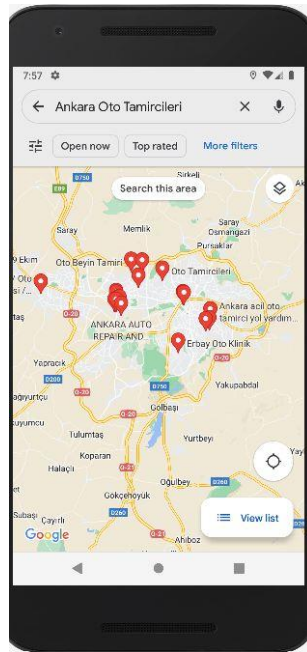
Hizmet Talep Et: Konum hizmetlerinin açılmasını sağlar.

İzin İste: Konum açarken telefonda konumu açmak için izin ister.

Şu anki Konumunuz: Konsol Ekranına şuanda bulunduğunuz koordinatların yazılmasını sağlar.

Google Haritalar: Google Haritaları kullanarak size yakın olan tamircilerini listeler.

Örnek İşlem aşağıda gösterilmiştir:



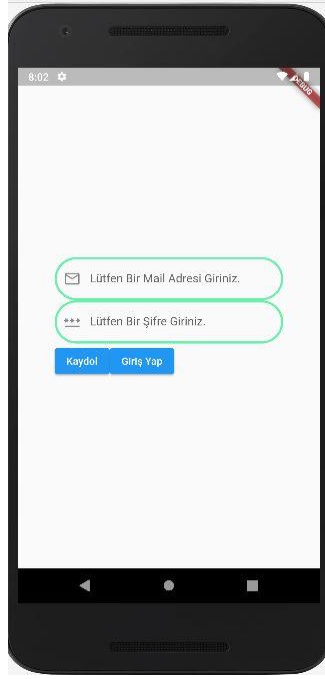
Şekil 2.10.3 Örnek Bir Harita İşlemi

2.11. Oturum Aç Sayfası

Oturum açma sayfasında kullanıcılar için bir giriş paneli bulunmaktadır. Burada mail adresi ve şifreyle kaydolduktan sonra giriş yap kısmına tıkladıklarında kendi sayfalarına geçtiklerini görürler.

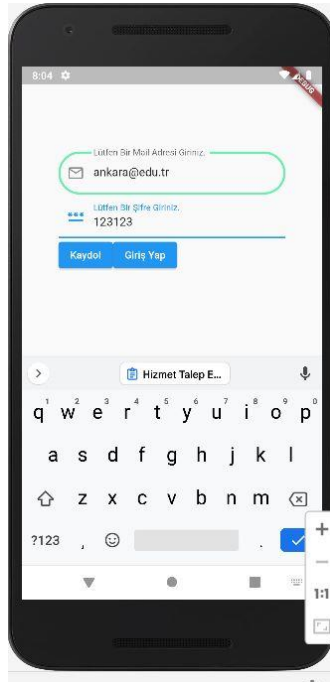
```
kullanici.dart
1 import 'package:cloud_firestore/cloud_firestore.dart';
2 import 'package:firebase_auth/firebase_auth.dart';
3 import 'package:firebase_core/firebase_core.dart';
4 import 'package:flutter/material.dart';
5 import 'package:tamircim/profil.dart';
6
7 Future<void> main() async {
8   WidgetsFlutterBinding.ensureInitialized();
9   await Firebase.initializeApp();
10  runApp(const MyApp1());
11 }
12
13 class MyApp1 extends StatelessWidget {
14   const MyApp1({Key? key}) : super(key: key);
15
16   @override
17   Widget build(BuildContext context) {
18     return MaterialApp(
19       home: Kullanici(),
20     ); // MaterialApp
21   }
22 }
23
24 class Kullanici extends StatefulWidget {
25   const Kullanici({Key? key}) : super(key: key);
26
27   @override
28   State<Kullanici> createState() => _KullaniciState();
29 }
30
31 class _KullaniciState extends State<Kullanici> {
32
33   TextEditingController t1 = TextEditingController();
34   TextEditingController t2 = TextEditingController();
35
36   Future<void> kayitOl() async {
37     await FirebaseAuth.instance
38       .createUserWithEmailAndPassword(email: t1.text, password: t2.text)
39       .then((kullanici) {
40         FirebaseFirestore.instance.collection("Kullaniciilar").doc(t1.text).set({
41           "KullaniciEposta": t1.text,
42           "KullaniciSifre": t2.text
43         }).whenComplete(() => print("Kullanici Firestore veritabanina eklendi"));
44       }).whenComplete(() => print("Kullanici Firebase'e kaydedildi."));
45   }
46
47   girisYap() {
48     FirebaseAuth.instance
49       .signInWithEmailAndPassword(email: t1.text, password: t2.text)
50       .then((kullanici) {
51         Navigator.push(
52           context,
53           MaterialPageRoute(
54             builder: (context) => ProfilEkrani(),
55           ), // MaterialPageRoute
56         );
57       });
58   }
59
60   get onPressed => null;
61
62   @override
63   Widget build(BuildContext context) {
64     return Scaffold(
65       body: Container(
66         margin: EdgeInsets.all(50),
67         child: Center(
68           child: SingleChildScrollView(
69             child: Column(
70               children: [
71                 TextFormField(
72                   controller: t1,
73                   decoration: InputDecoration(
74                     labelText: "Lütfen Bir Mail Adresi Giriniz.",
75                     prefixIcon: Icon(Icons.email_outlined),
76                     enabledBorder: OutlineInputBorder(
77                       borderSide:
78                         BorderSide(width: 3, color: Colors.greenAccent),
79                       borderRadius: BorderRadius.circular(40.0),
80                     ), // OutlineInputBorder
81                   ), // TextFormField
82                 TextFormField(
83                   controller: t2,
84                   decoration: InputDecoration(
85                     labelText: "Lütfen Bir Şifre Giriniz.",
86                     prefixIcon: Icon(Icons.password_outlined),
87                     enabledBorder: OutlineInputBorder(
88                       borderSide:
89                         BorderSide(width: 3, color: Colors.greenAccent),
90                       borderRadius: BorderRadius.circular(40.0),
91                     ), // OutlineInputBorder
92                   ), // TextFormField
93                 Row(
94                   children: [
95                     ElevatedButton(
96                       child: Text("Kaydol"),
97                       onPressed: kayitOl,
98                     ), // ElevatedButton
99                     ElevatedButton(
100                      child: Text("Giriş Yap"),
101                      onPressed: girisYap,
102                    ), // ElevatedButton
103                  ], // Row
104                ), // Column
105              ], // SingleChildScrollView
106            ), // Center
107          ), // Container
108        ), // Scaffold
109      );
110    }
111  }
112 }
113 }
```

Şekil 2.10.1 Oturum Aç Sayfası İçin Yazılan Kodlar



Şekil 2.10.2 Oturum Aç Sayfasının Emülatördeki Görünümü

Örnek bir giriş oturum açma örneği aşağıda gösterilmiştir:



Şekil 2.10.3 Örnek Oturum Açma İşlemi

Kendi sayfalarında AppBar'da yer alan + simgesine bastıklarında kendi ID'lerine ait olan gönderileri üzerinde ekleme, düzenleme, silme işlemlerini yapabilmektedir. Ayrıca burada yer alan listeleme sayesinde o kullanıcıya ait gönderiler de listelenmektedir.

```

1 import 'dart:io';
2 import 'package:cloud_firestore/cloud_firestore.dart';
3 import 'package:firebase_auth/firebase_auth.dart';
4 import 'package:flutter/material.dart';
5 import 'package:tamircim/main.dart';
6 import 'package:tamircim/tamirci.dart';
7
8 class ProfilEkranı extends StatelessWidget {
9   const ProfilEkranı({Key? key}) : super(key: key);
10
11   @override
12   Widget build(BuildContext context) {
13     return Scaffold(
14       appBar: AppBar(
15         title: Text("Kullanıcı Profil Sayfası"),
16         actions: <Widget>[
17           IconButton(
18             icon: Icon(Icons.exit_to_app),
19             onPressed: () {
20               FirebaseAuth.instance.signOut().then((deger){
21                 Navigator.pushAndRemoveUntil(
22                   context,
23                   MaterialPageRoute(builder: (.) => MyApp2()),
24                   (Route<dynamic> route) =>false);
25             }
26           ), // IconButton
27           FloatingActionButton(
28             child: Icon(Icons.add),
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

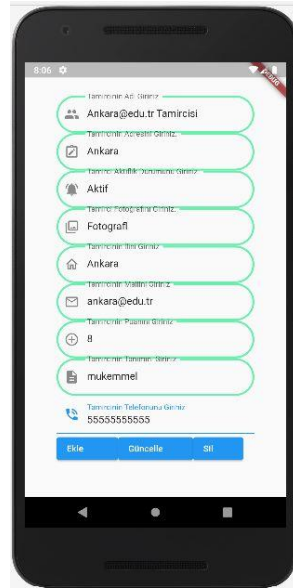
```

```

43 class TamirciVerileri extends StatelessWidget {
44   @override
45   Widget build(BuildContext context) {
46     FirebaseAuth auth = FirebaseAuth.instance;
47     final Stream<QuerySnapshot> _usersStream = FirebaseFirestore.instance
48       .collection('Tamirciler')
49       .where("kullaniciid", isEqualTo: auth.currentUser?.uid)
50       .snapshots();
51     return StreamBuilder<QuerySnapshot> (
52       stream: _usersStream,
53       builder: (BuildContext context, AsyncSnapshot<QuerySnapshot> snapshot) {
54         if (snapshot.hasError) {
55           return Text('Something went wrong');
56         }
57         if (snapshot.connectionState == ConnectionState.waiting) {
58           return Text('Loading');
59         }
60         return ListView(
61           children: snapshot.data!.docs.map((DocumentSnapshot document) {
62             Map<String, dynamic> data =
63               document.data()! as Map<String, dynamic>;
64             return Card(
65               color: Colors.amber[900],
66               shape: RoundedRectangleBorder(
67                 borderRadius: BorderRadius.circular(5)), // RoundedRectangleBorder
68               elevation: 4,
69               child: ListTile(
70                 textColor: Colors.white,
71                 titleColor: Colors.lightBlue,
72                 title: Text(
73                   data['adi'],
74                   style: TextStyle(fontWeight: FontWeight.bold, fontSize: 20),
75                 ), // Text
76                 subtitle: Column(
77                   crossAxisAlignment: CrossAxisAlignment.start,
78                   children: [
79                     Text(data['adres']),
80                     Text(data['aktiflik']),
81                     Text(data['fotograf']),
82                     Text(data['il']),
83                     Text(data['mail']),
84                     Text(data['puan']),
85                     Text(data['tanin']),
86                     Text(data['telefon']),
87                   ],
88                 ), // Column
89               ), // ListTile
90             ); // Card
91           }).toList(),
92         ); // ListView
93       },
94     ); // StreamBuilder
95   }
96 }
97
98
99
100

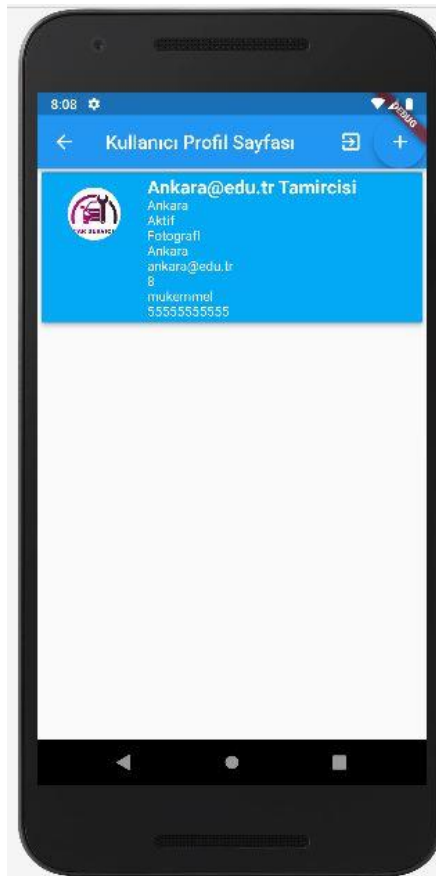
```

Şekil 2.10.4 Kullanıcı Sayfası için Yazılan Kodlar



Şekil 2.10.5 Kullanıcı Sayfasından Tamirci Ekleme

Eklenen tamirci kullanıcı sayfasında aşağıdaki gibi görülmektedir:



Şekil 2.10.6 Kullanıcı Sayfasında Eklenen Tamircinin Gözükmesi

The screenshot shows the Cloud Firestore console interface. At the top, there's a navigation bar with 'tamirciveritabani' and 'Cloud Firestore'. Below it, a sidebar contains 'Panel view' and 'Query builder'. The main area displays a collection named 'Kullanicilar' under the path 'ankara@edu.tr'. The collection contains documents with email addresses. The interface is clean and modern, with a light blue and white color scheme.

Şekil 2.10.7 Kullanıcılar Veri Tabanı Tablosu

3.SONUÇ

Sonuç olarak, önceki bölümlerde açıklandığı gibi, bilgiler elde edilmiş ve Flutter kullanarak Dart ile bir proje geliştirilmiştir. Flutter, Firebase, Firestore, Authencation gibi yapıların nasıl kullanıldığı öğrenilmiş ve bilgi birikimine sahip olunmuştur.

Proje geliştirilirken alınan çeşitli hatalarda nasıl yollar izlenmesi gerektiği ve ne gibi araştırılması gerektiği hakkında bilgi sahibi olunmuştur.

Proje değişikliklerini yönetmek ve takip etmek için kullanılan GiT sistemi hakkında tecrübe edinilmiştir.

Anlatılan tüm süreçlerin sonunda her yaştan her cinsiyetten tüm insanlara hitap eden, kolay kullanım sağlayan ve tüm cihazlara uygunluk sağlayan bir Mobil Uygulama Projesi oluşturulmuş ve bu proje sayesinde kullanıcıların en iyi hizmeti alabileceği tamirciyi bulmaları kolaylaştırılmış aynı zamanda kendi firmalarını tanıtmak ve reklamını yapmak isteyen kullanıcıların amaçlarına ulaşması sağlanmıştır.

4. KAYNAKLAR

<https://flutter.dev/>

<https://stackoverflow.com/>

<https://talentgrid.io/tr/firebase-nedir/>

<https://medium.com/@berkekurnaz/flutter-g%C3%BCnl%C3%BCkleri-5-drawer-menu-yapal%C4%B1m-7c030dd56b3d>

https://pub.dev/packages/flutter_native_splash

<https://www.webtekno.com/flutter-nedir-nasil-kullanilir-h115673.html>

<https://ceaksan.com/tr/firebase-nedir-nasil-kullanilir>

<https://www.youtube.com/watch?v=cMI3LETcxEM>

<https://gelecegiyazanlar.turkcell.com.tr/konu/egitim/android-401/firebase-realtime-database>

<https://onursahin.net/flutterda-acilis-ekrani-splash-screen/>