CS102	**Summer 2021/22**		**MIYAV**
Instructor:	**Lori Rae Russel Dağ**	Project Group	
Assistant:	**Mert Kara**		

Criteria	TA/Grader	Instructor
Presentation	**Mert Kara**	**Lori Rae Russel Dağ**
Overall		

~ BILCAT~

miyav

Beyza Ural, Dilara Kıymaz, Mehmet Emre Güneş, Ege Cenberci

Detailed Design Report

(final)

26 June 2022

1. Introduction

This project aims to develop an android mobile app that aims to track the cats around Bilkent campus and store their information. The application will allow its users to track the campus' cat's health status and locate them easily if the need arises.

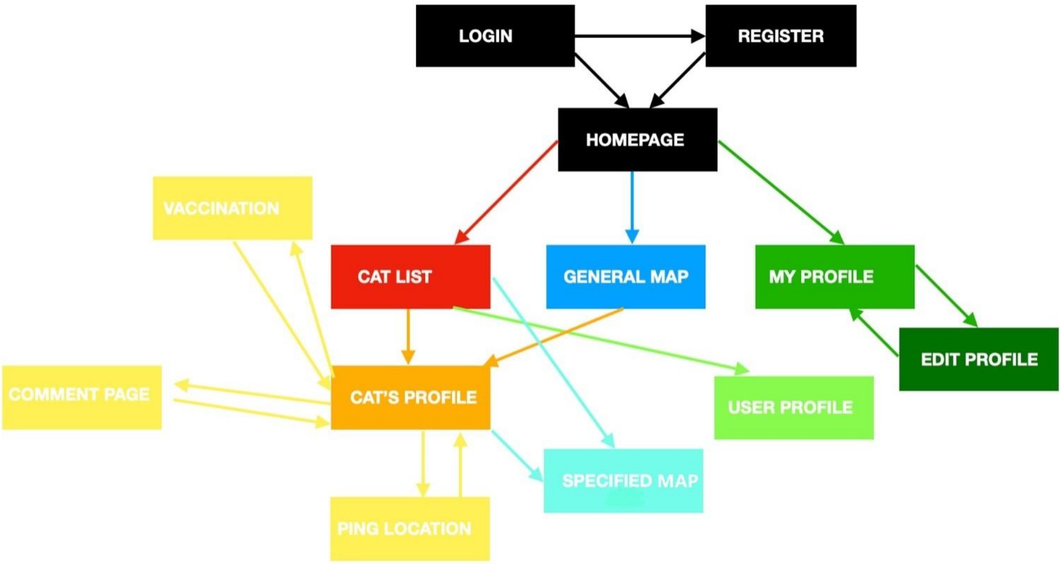
2. System Overview

The project's application will be an Android mobile application. The technologies utilized to create this application are:

- Java [4]

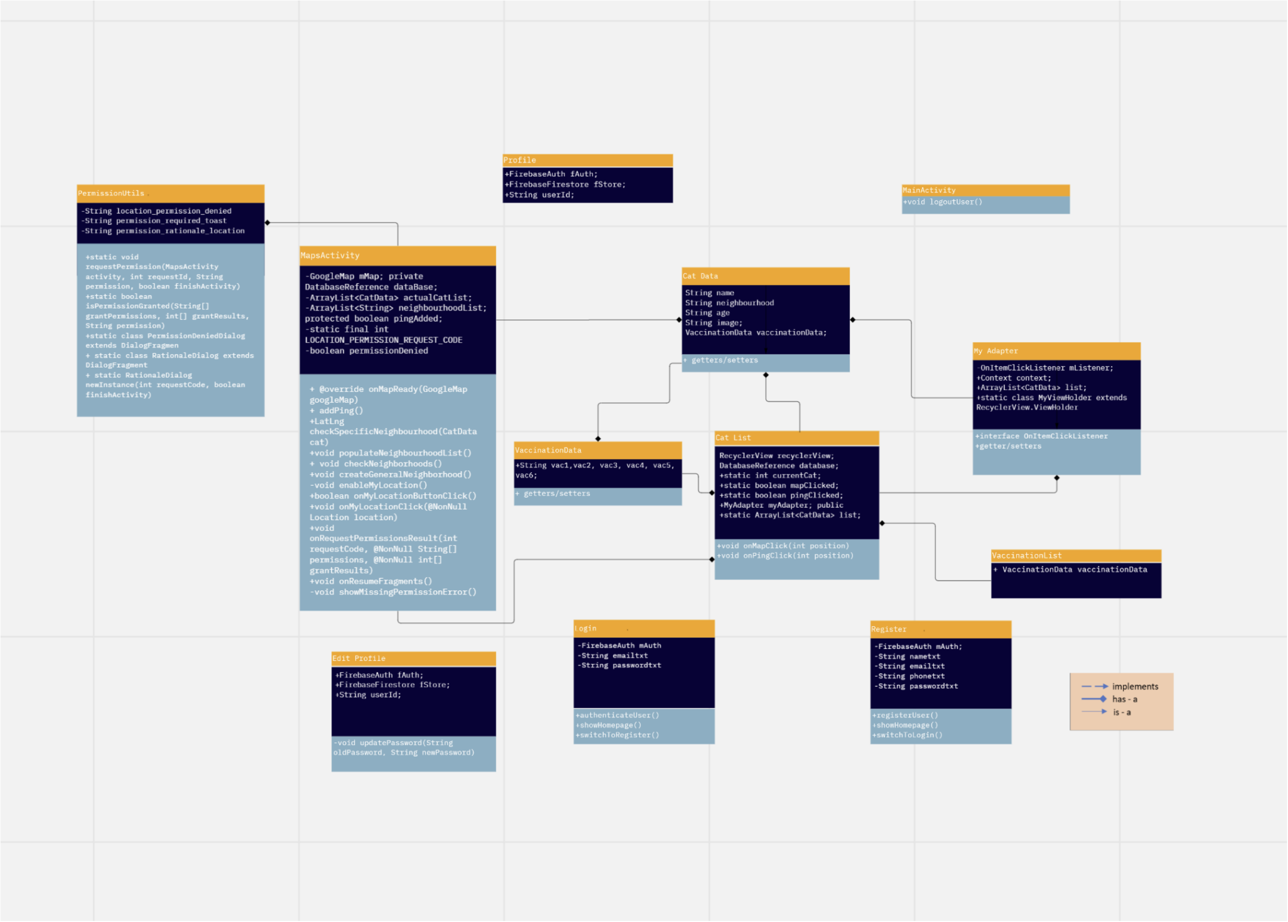
- Android [1]
- XML
- Google Maps SDK for Android [3]
- Firebase [2]

The following diagram depicts the architecture of the application and the pathways the users can follow to navigate through the app.



3. Core Design Details

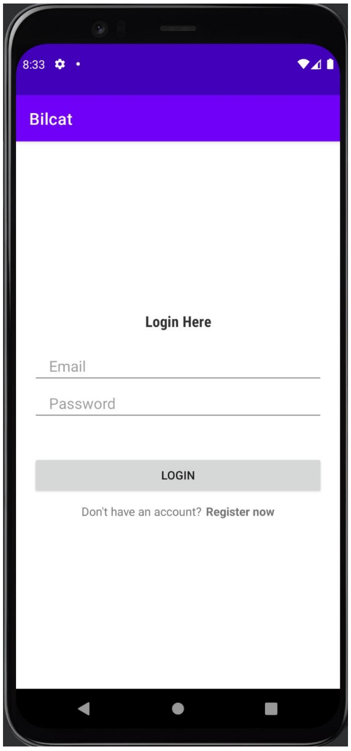
3.1 UML Design

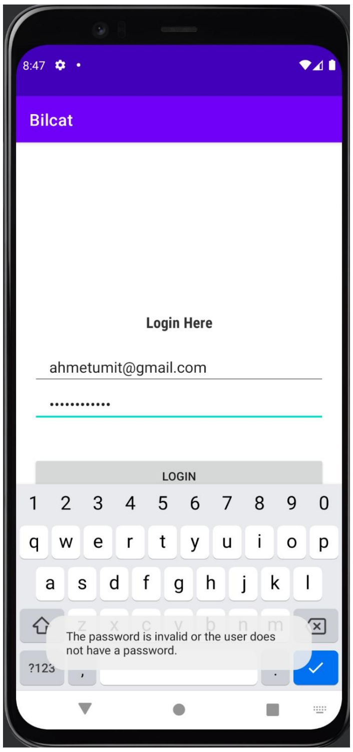

PermissionUtils
-String location_permission_denied
-String permission_required_toast
-String permission_rationale_location
+static void requestPermission(MapsActivity activity, int requestId, String permission, boolean finishActivity)
+static boolean isPermissionGranted(String[] grantPermissions, int[] grantResults, String permission)
+static class PermissionDeniedDialog extends DialogFragmen
+ static class RationaleDialog extends DialogFragment
+ static RationaleDialog newInstance(int requestCode, boolean finishActivity)
Profile
+FirebaseAuth fAuth;
+FirebaseFirestore fStore;
+String userId;
MainActivity
+void logoutUser()
MapsActivity
-GoogleMap mMap; private DatabaseReference dataBase;
-ArrayList<CatData> actualCatList;
-ArrayList<String> neighbourhoodList;
protected boolean pingAdded;
-static final int LOCATION_PERMISSION_REQUEST_CODE
-boolean permissionDenied
+ @override onMapReady(GoogleMap googleMap)
+ addPing()
+LatLng checkSpecificNeighbourhood(CatData cat)
+void populateNeighbourhoodList()
+ void checkNeighborhoods()
+void createGeneralNeighborhood()
-void enableMyLocation()
+boolean onMyLocationButtonClick()
+void onMyLocationClick(@NonNull Location location)
+void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults)
+void onResumeFragments()
-void showMissingPermissionError()
Cat Data
String name
String neighbourhood
String age
String image;
VaccinationData vaccinationData;
+ getters/setters
My Adapter
-OnItemClickListener mListener;
+Context context;
+ArrayList<CatData> list;
+static class MyViewHolder extends RecyclerView.ViewHolder
+interface OnItemClickListener
+getter/setters
VaccinationData
+String vac1,vac2, vac3, vac4, vac5, vac6;
+ getters/setters
Cat List
RecyclerView recyclerView;
DatabaseReference database;
+static int currentCat;
+static boolean mapClicked;
+static boolean pingClicked;
+MyAdapter myAdapter; public
+static ArrayList<CatData> list;
+void onMapClick(int position)
+void onPingClick(int position)
VaccinationList
+ VaccinationData vaccinationData
Edit Profile
+FirebaseAuth fAuth;
+FirebaseFirestore fStore;
+String userId;
-void updatePassword(String oldPassword, String newPassword)
Login
-FirebaseAuth mAuth
-String emailtxt
-String passwordtxt
+authenticateUser()
+showHomepage()
+switchToRegister()
Register
-FirebaseAuth mAuth;
-String nametxt
-String emailtxt
-String phonetxt
-String passwordtxt
+registerUser()
+showHomepage()
+switchToLogin()
implements
has - a
is - a

3.2 Classes

3.2.1 Login

Login class contains the FirebaseAuth object as a variable for the firebase authentication. The method authenticateUser() makes sure that user login correctly with appropriate input and adds data of the user to the database. The method switchToRegister() is used for the users who enter the app for the first time and don't have an account, it directs the user to the registration page.





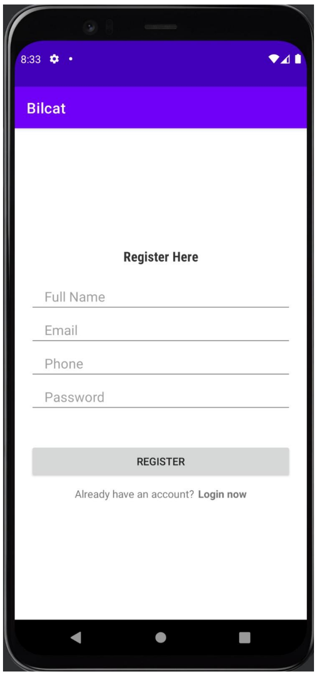
ahmetumit@gmail.com

••••••••••••

The password is invalid or the user does not have a password.

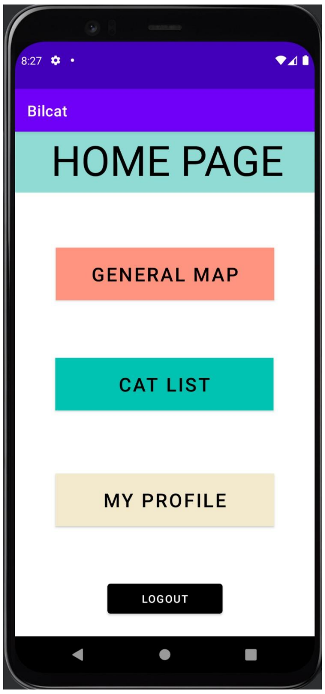
3.2.2 Register

Registration class also contains a FirebaseAuth object attribute for the firebase authentication. The registerUser() method makes sure that the user registers correctly. If the user registered correctly using email and password it adds data of the user to the database and utilizes the User class to create a user. Then the FirebaseDatabase object variable of the class is used to save the data of the user to the database. The switchToLogin() method is used to take the users who have already registered back to the Login page.



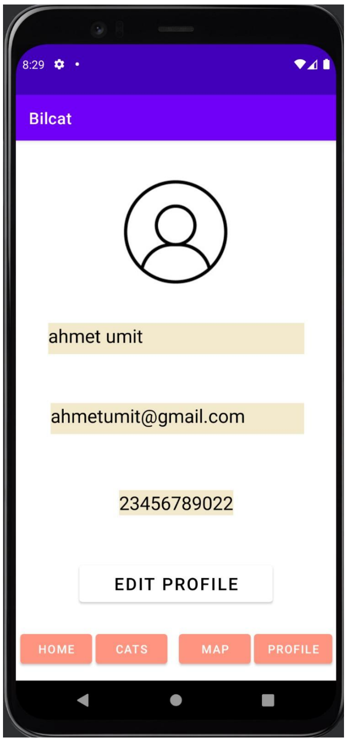
3.2.3 Home Page

The Home Page class allows for transitions between pages. From here you can pass on to the general map, cat list and user profile. All these transitions were made by the click listeners that Android Studio has provided. Additionally, there is a private method called logoutUser(), this method is connected to the firebase and uses its signOut() method before starting the Login page.



3.2.4 Profile

The profile is a class that is the user connection of the user. Inside the onCreate(Bundle savedInstanceState) method a firebase instance has been created and the information of the user has been taken from the firebase via this instance to be displayed.



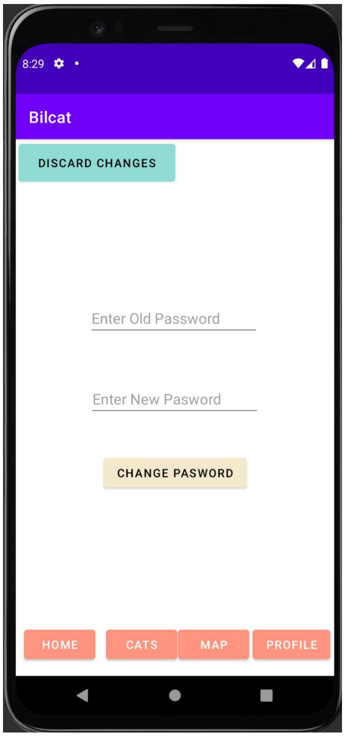
ahmet umit

ahmetumit@gmail.com

23456789022

3.2.5 Edit Profile

Edit profile is displayed through profile page only. This is a class that changes user password and there are options to discard or save. In the case of choosing to change the password, there is a method called updatePassword(). Inside this method, first, the current logged in user is found. Then, the authentication process is applied with the user's email and current password. After that, a toast message is displayed depending on whether it is successful or not. Finally, the user's password is updated with the firebase method called updatePassword(String newPassword).



3.2.6 Comment

The Comment class was created for use in cat profiles. Calendar class is used to show the time that comments are discarded.

3.2.7 CatData

The CatData class is used to access the cats created in Firebase and on other classes such as the CatList class to access the properties of a specific cat. CatData class contains constructors and getters/setters.

3.2.8 Cat List

The main feature of the CatList class is to display cats. The CatList class works with the MyAdapter class because, in the CatList class, cats are displayed to users with RecyclerView and CardView. For the cat cards to be displayed correctly, the MyViewHolder class has been created as the inner part of the MyAdapter class. In Cat List, cats are displayed to users with their cards. These cards show the cats' photos, names, ages, and neighborhoods. All these data are accessed using the Realtime Database. In addition, there are four buttons on each cat card. In order to activate these buttons, the interface is defined as inner in MyAdapter class. The methods in this interface are used in the CatList where necessary. One of the primary purposes of the methods inside this OnItemClickListener interface is to determine which cat was clicked in the Cat list, so it takes a position value which is int as a parameter. The CatList class has a currentCat which is int variable that indicates which cat was last clicked. It is defined as static in order to use this variable easily in Map and other classes. There is also a static ArrayList<CatData> variable inside this CatList class. This variable contains the cats in the Realtime Database. In addition, since it is static, it can be easily used in other classes.

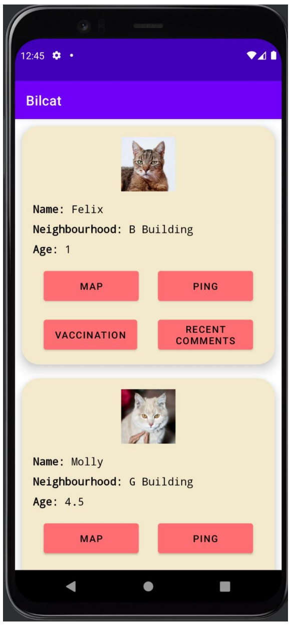
In the cat list, each cat has four buttons. What these buttons do is listed below:

Map -> Shows only the neighborhood where the selected cat is located.

Ping -> Allows marking the last location of the selected cat. Users can comment after marking.

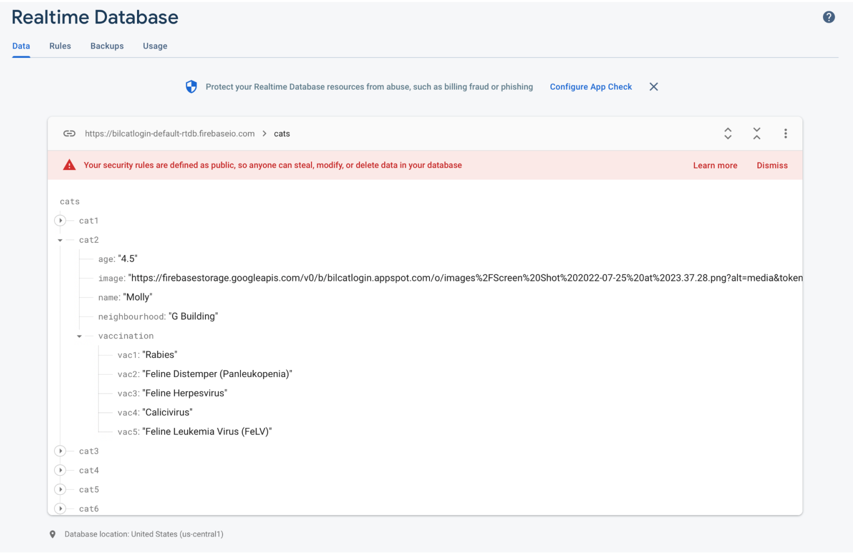
Vaccination -> Shows the vaccinations of the selected cat.

Recent Comments -> Shows comments made via Ping. The comments shown are only those for the selected cat.



Name: Felix
Neighbourhood: B Building
Age: 1

Name: Molly
Neighbourhood: G Building
Age: 4.5



Protect your Realtime Database resources from abuse, such as billing fraud or phishing

https://bilcatlogin-default-rtdb.firebaseio.com > cats

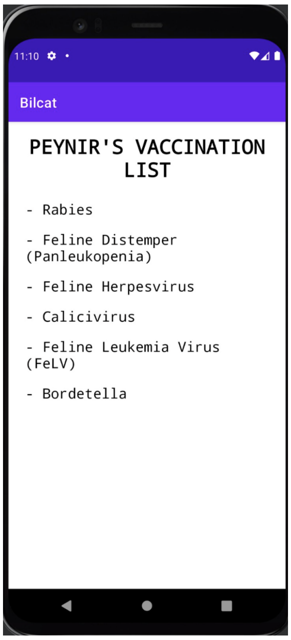
Your security rules are defined as public, so anyone can steal, modify, or delete data in your database

```
cats
  cat1
  cat2
    age: "4.5"
    image: "https://firebasestorage.googleapis.com/v0/b/bilcatlogin.appspot.com/o/images%2FScreen%20Shot%202022-07-25%20at%2023.37.28.png?alt=media&token
    name: "Molly"
    neighbourhood: "G Building"
    vaccination
      vac1: "Rabies"
      vac2: "Feline Distemper (Panleukopenia)"
      vac3: "Feline Herpesvirus"
      vac4: "Calicivirus"
      vac5: "Feline Leukemia Virus (FeLV)"
  cat3
  cat4
  cat5
  cat6
```

Database location: United States (us-central1)

3.2.9 VaccinationList and VaccinationData

VaccinationData allows using data in Realtime Database via constructor and getters/setters. The VaccinationList class allows displaying the vaccines of specific cats.



11:10

Bilcat

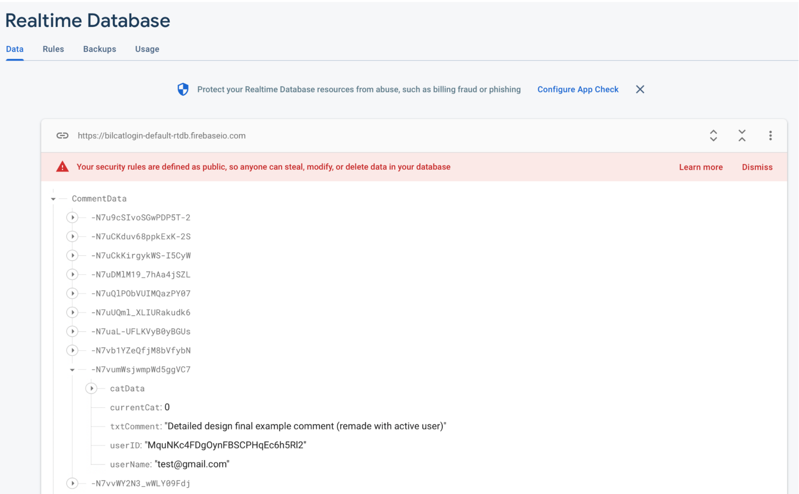
PEYNIR'S VACCINATION LIST

- Rabies
- Feline Distemper (Panleukopenia)
- Feline Herpesvirus
- Calicivirus
- Feline Leukemia Virus (FeLV)
- Bordetella

3.2.10 CommentList, CommentData, DAOComment, CommentAdapter

As in CatList, RecyclerView and CardView are used in the CommentList class. The CommentAdapter class enables RecyclerView and CardView to work in a coordinated manner. In the CommentData class, there are constructors and getters/setters to use the data in the Realtime Database. The DAOComment class is used to communicate with the Database. The ability to comment on the ping screen is available thanks to the add() method in the DAOComment class. In the CatList class, the data from the Realtime Database is retrieved, and here it shows the specific cat's comments that the Recent Comments button was clicked on.

Through the Realtime Database, we can reach which cat the comment was made and by which user. In this way, users can see the cat's Recent Comments section of whichever cat the comment belongs to. In addition, under the comment, the e-mail address of that user is written by whichever user it was made.



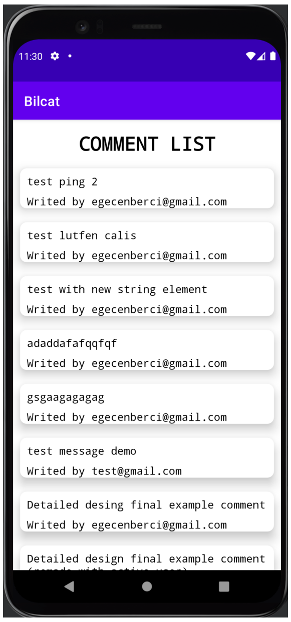
Realtime Database

Protect your Realtime Database resources from abuse, such as billing fraud or phishing

https://bilcatlogin-default-rtdb.firebaseio.com

Your security rules are defined as public, so anyone can steal, modify, or delete data in your database

```
CommentData
  -N7u9cSIvoSGwPDP5T-2
  -N7uCKduv68ppkExK-2S
  -N7uCkKirgykWS-I5CyW
  -N7uDM1M19_7hAa4jSZL
  -N7uQ1PObVUIMQazPY07
  -N7uUQml_XLIURakudk6
  -N7uaL-UFLKVyB0yBGUs
  -N7vb1YZeQfjM8bVfybN
  -N7vumWsjwmpWd5ggVC7
    catData
    currentCat: 0
    txtComment: "Detailed design final example comment (remade with active user)"
    userID: "MquNKc4FDgOynFBSCPHqEc6h5Rl2"
    userName: "test@gmail.com"
  -N7vvWY2N3_wWLY09Fdj
```



11:30

Bilcat

COMMENT LIST

test ping 2
Writed by egecenberci@gmail.com

test lutfen calis
Writed by egecenberci@gmail.com

test with new string element
Writed by egecenberci@gmail.com

adaddafafqqfqf
Writed by egecenberci@gmail.com

gsgaagagagag
Writed by egecenberci@gmail.com

test message demo
Writed by test@gmail.com

Detailed desing final example comment
Writed by egecenberci@gmail.com

Detailed design final example comment

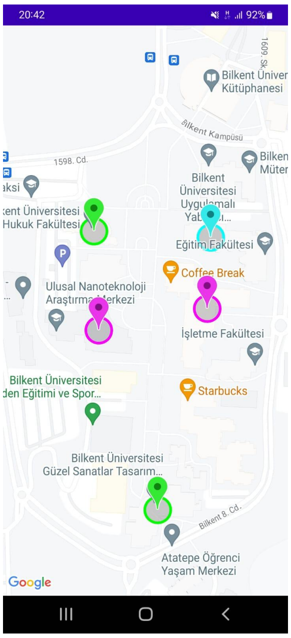
3.2.11 MapsActivity

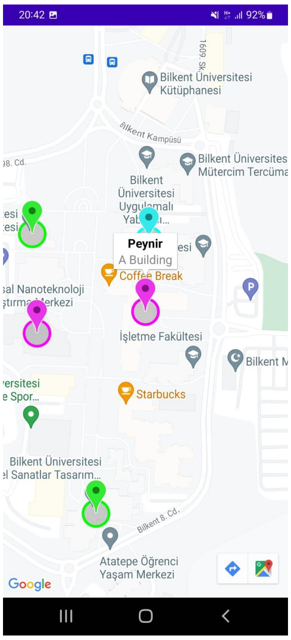
The MapsActivity class extends FragmentActivity, implements OnMapReadyCallBack, GoogleMap.OnMyLocationButtonClickListener, GoogleMap.OnMyLocationClickListener, ActivityCompat.OnRequestPermissionsResultCallback and was used to build the integrated Google Map of the General, Specific and Ping map pages. The class' attributes will consist of a GoogleMap object, an ArrayList containing the data of the cats on the database, an ArrayList containing the neighborhood information of all the cats on the database, numerous boolean flags to track specific parts of the program, and final Latlng objects to store coordinates of buildings in Bilkent Campus.

The MapsActivity class overrides the onMapReady() method to initialize the map that is going to be displayed. By utilizing boolean flags, the creation of all three maps was established inside the onMapReady() method. Since some of the maps differ in their user interface, the relevant buttons and their listeners too were created inside the onMapReady() method. The "pinging" of the cats' location was also done through the use of button listeners and MarkerOptions objects declared as attributes in the class file, making the Ping class that was originally designed redundant. The methods to manipulate the map with user clicks and drags were in-built and therefore didn't need any adjustments.

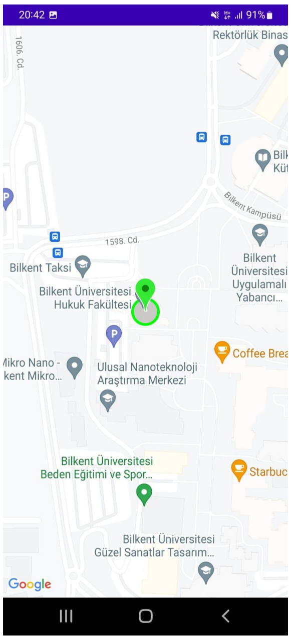
The MapsActivity class also has methods such as the override onRequestPermissionResult() to check if the application has access to the devices' location services. To effectively use such permission checker methods like onRequestPermissionResult(), the java class PermissionUtils, originally created by Google, was slightly modified and added to the project. Since the PermissionUtils class only deals with android permission activities, it does not require further discussion.

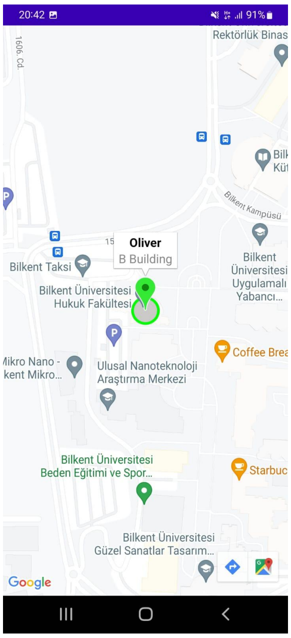
The General Map's initial build and the message displayed when any cat neighborhood marker is clicked can be seen in the following pictures.


20:42
92%
Bilkent Üniver Kütüphanesi
Bilkent Kampüsü
Bilkent Üniversitesi Uygulamalı Yab...
Eğitim Fakültesi
Coffee Break
Ulusal Nanoteknoloji Araştırma Merkezi
İşletme Fakültesi
Bilkent Üniversitesi Beden Eğitimi ve Spor...
Starbucks
Bilkent Üniversitesi Güzel Sanatlar Tasarım...
Atatepe Öğrenci Yaşam Merkezi
Google

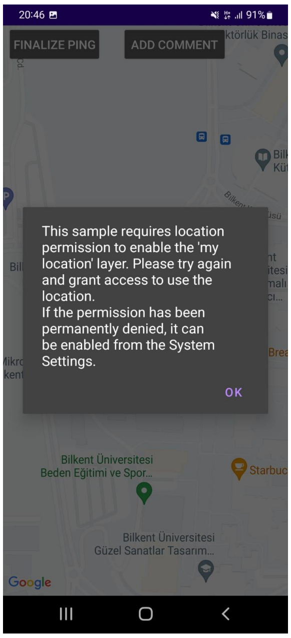

20:42
92%
Bilkent Üniversitesi Kütüphanesi
Bilkent Kampüsü
Bilkent Üniversitesi Mütercim Tercüme
Peynir
A Building
Coffee Break
İşletme Fakültesi
Starbucks
Atatepe Öğrenci Yaşam Merkezi
Google

The Specific Map that is unique to each cat, displays a map similar to the General Map. The only difference between them is that the Specific Map sports only one cat's neighborhood. The Specific Map can be accessed through the Map button on the Cat List page and looks like the following:

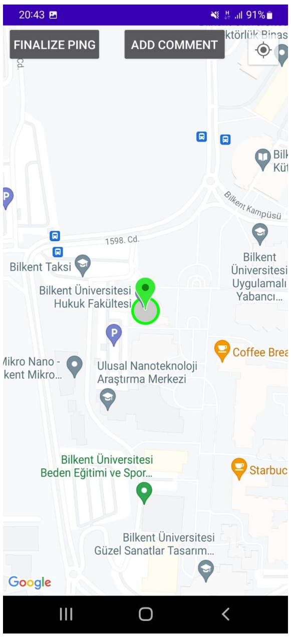

20:42
91%
Rektörlük Binas
1606. Cd.
1598. Cd.
Bilkent Taksi
Bilkent Üniversitesi Hukuk Fakültesi
Bilkent Kampüsü
Bilkent Üniversitesi Uygulamalı Yabancı...
Coffee Brea
Mikro Nano - kent Mikro...
Ulusal Nanoteknoloji Araştırma Merkezi
Bilkent Üniversitesi Beden Eğitimi ve Spor...
Starbuc
Bilkent Üniversitesi Güzel Sanatlar Tasarım...
Google

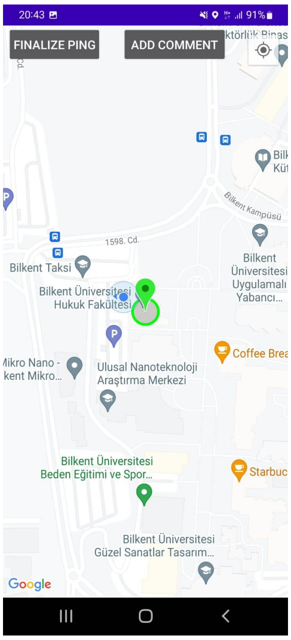

20:42
91%
Rektörlük Binas
Oliver
B Building
Bilkent Taksi
Bilkent Üniversitesi Hukuk Fakültesi
Bilkent Kampüsü
Bilkent Üniversitesi Uygulamalı Yabancı...
Coffee Brea
Mikro Nano - kent Mikro...
Ulusal Nanoteknoloji Araştırma Merkezi
Bilkent Üniversitesi Beden Eğitimi ve Spor...
Starbuc
Bilkent Üniversitesi Güzel Sanatlar Tasarım...
Google

Similarly to the Specific Map, the Ping Map can be accessed through the Cat List page. It displays the neighborhood of the clicked cat and allows functionality for the user to “ping” the location of the cat with an appended comment in String form. If the application’s access to the location services was not enabled by the user, Ping Map displays a relative error message. After enabling the location permission for the application, the user can access the Ping Map. If the user turns the location services on for their mobile phone, they can observe their fine location on the Ping Map. The described pages look like the following:



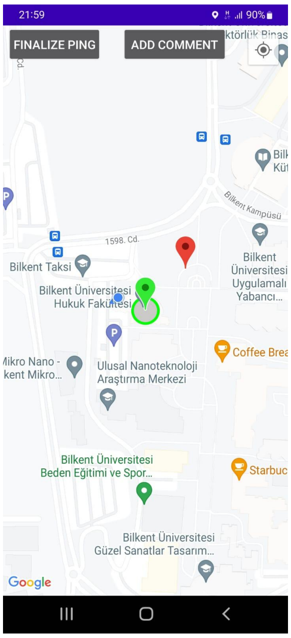
This sample requires location permission to enable the 'my location' layer. Please try again and grant access to use the location.
If the permission has been permanently denied, it can be enabled from the System Settings.

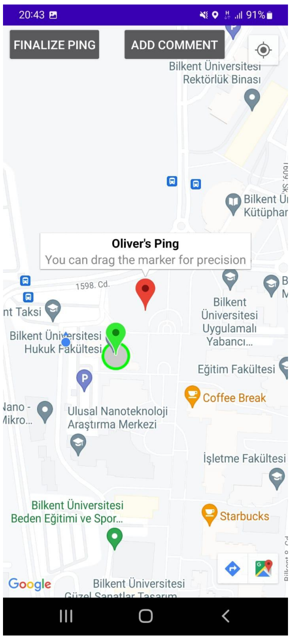

20:43
91%
FINALIZE PING
ADD COMMENT
Bilkent Taksi
Bilkent Üniversitesi Hukuk Fakültesi
Coffee Brea
Ulusal Nanoteknoloji Araştırma Merkezi
Starbuc
Google


20:43
91%
FINALIZE PING
ADD COMMENT
Bilkent Taksi
Bilkent Üniversitesi Hukuk Fakültesi
Coffee Brea
Ulusal Nanoteknoloji Araştırma Merkezi
Starbuc
Google

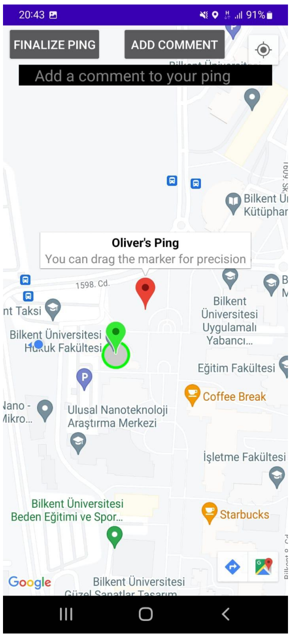
The user can click their own representation on the map represented by the blue dot to get a pop-up message, or click the location button on the top right corner of the Ping Map to center the map on their live location. However, considering that these functionalities are trivial and don’t display essential information to the user, their images were not added for better viewing.

Upon clicking anywhere on the map, the user can create a draggable marker object, or a “ping”. Upon clicking on this marker, the appropriate information of the marker is displayed. When the add comment button is clicked, a text field is shown for the user to write their comment they wish to add to the “ping”.


21:59
FINALIZE PING
ADD COMMENT
Bilkent Taksi
Bilkent Üniversitesi Hukuk Fakültesi
Bilkent Üniversitesi Uygulamalı Yabancı...
Coffee Break
Ulusal Nanoteknoloji Araştırma Merkezi
Bilkent Üniversitesi Beden Eğitimi ve Spor...
Bilkent Üniversitesi Güzel Sanatlar Tasarım...
Google


20:43
FINALIZE PING
ADD COMMENT
Bilkent Üniversitesi Rektörlük Binası
Bilkent Ün Kütüphar
Bilkent Üniversitesi Uygulamalı Yabancı...
Eğitim Fakültesi
Coffee Break
Ulusal Nanoteknoloji Araştırma Merkezi
İşletme Fakültesi
Bilkent Üniversitesi Beden Eğitimi ve Spor...
Starbucks
Google

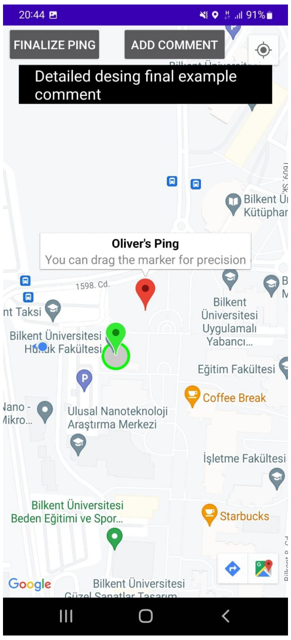
Oliver's Ping
You can drag the marker for precision



Add a comment to your ping

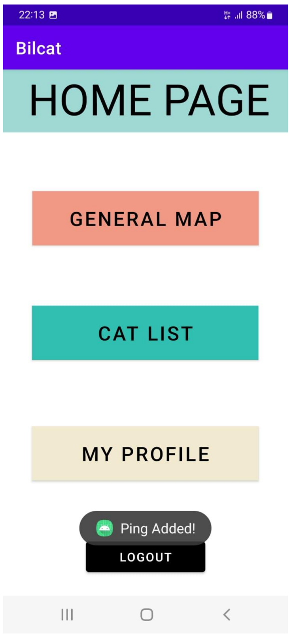
Oliver's Ping
You can drag the marker for precision

After writing their comment, the user can press the finalize ping button to create a marker with their comment attached to it. Upon finalizing the ping, the user is returned to the Home Page with an appropriate message displayed. The newly added comment can be seen from the relevant cat’s Recent Comments screen, accessible through the Cat List.

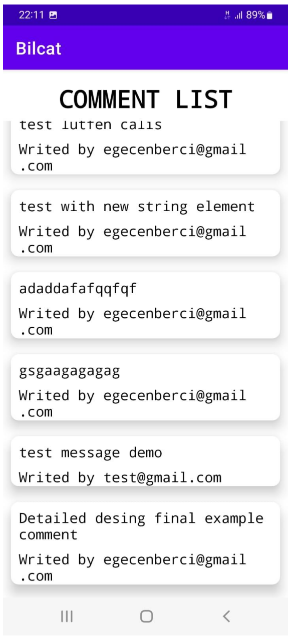


Detailed desing final example comment

Oliver's Ping
You can drag the marker for precision



Ping Added!



test lutfen calis
Writed by egecenberci@gmail.com

test with new string element
Writed by egecenberci@gmail.com

adaddafafqqfqf
Writed by egecenberci@gmail.com

gsgaagagagag
Writed by egecenberci@gmail.com

test message demo
Writed by test@gmail.com

Detailed desing final example comment
Writed by egecenberci@gmail.com

4. Task Assignment

Sena Beyza Ural: Login, Register, Main Activity classes

Dilara Kıymaz: User, Profile, EditProfile, HomePage classes

Ege Çenberci: MapsActivity and PermissionUtils classes

Mehmet Emre Güneş: CatList, CommentList, and VaccinationList activities; cardlistcard and commentlistcard layouts; CatData, CommentAdapter, CommentData, MyAdapter and VaccinationData classes.

References

[1] Google. (n.d.). *Android Studio and SDK tools: Android Developers*. Android Developers. Retrieved June 19, 2022, from https://developer.android.com/studio/

[2] Google. (n.d.). *Firebase*. Firebase-Google. Retrieved June 19, 2022, from https://firebase.google.com/

[3] Google. (n.d.). *Maps SDK for Android*. Google. Retrieved July 19, 2022, from https://developers.google.com/maps/documentation/android-sdk/overview

[4] Java, “Java,” Oracle, [Online]. Retrieved July 19, 2022 from https://www.java.com/tr/.

[5] Miro.com/. (n.d.). Retrieved July 19, 2022, from https://miro.com/