Software Project

Team Name: TeamBot
Project Name: Cave Examination Bot
Group Number: 9

Student Names and IDs:
Adrian Portal Calcines - n01489363
Alfred Dowuona - N01490404
Ali Mohebi - N01477361
Hassan Noorani - N01485518

Table Of Contents:

Project Description:

Signatures Table:

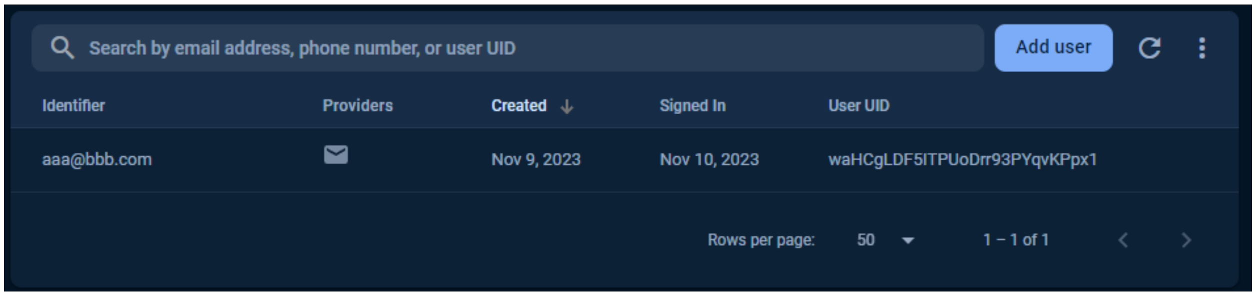
Name	Id	Signature	Effort
Adrian Portal C	n01489363	A.P.	100%
Alfred Dowuona	n01490404		
Ali Mohebi	n01477361		
Hassan Noorani	n01485518		

Github Repo Link:

https://github.com/HassanNoorani5518/CaveExaminationBot

Account Creation in Database:

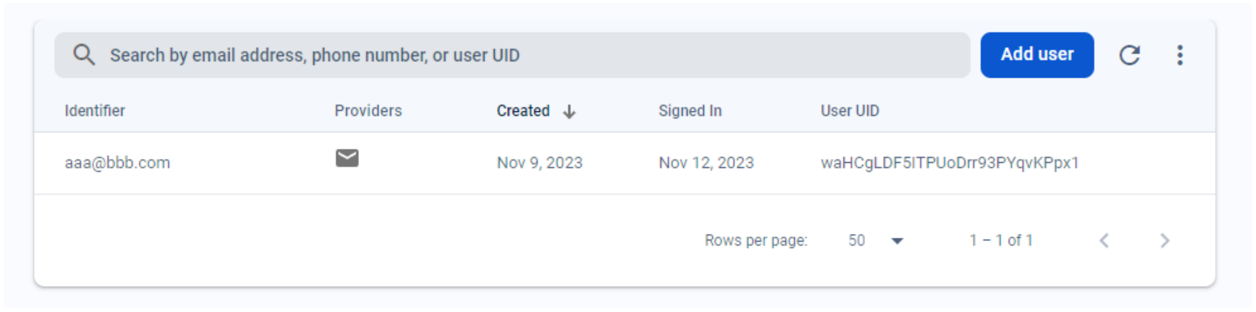
Admin Account Screenshot:



Identifier	Providers	Created	Signed In	User UID
aaa@bbb.com	✉	Nov 9, 2023	Nov 10, 2023	waHCgLDF5ITPUoDrr93PYqvKPpx1

Rows per page: 50 1 – 1 of 1

Logged in Users Screenshot:



Identifier	Providers	Created	Signed In	User UID
aaa@bbb.com	✉	Nov 9, 2023	Nov 12, 2023	waHCgLDF5ITPUoDrr93PYqvKPpx1

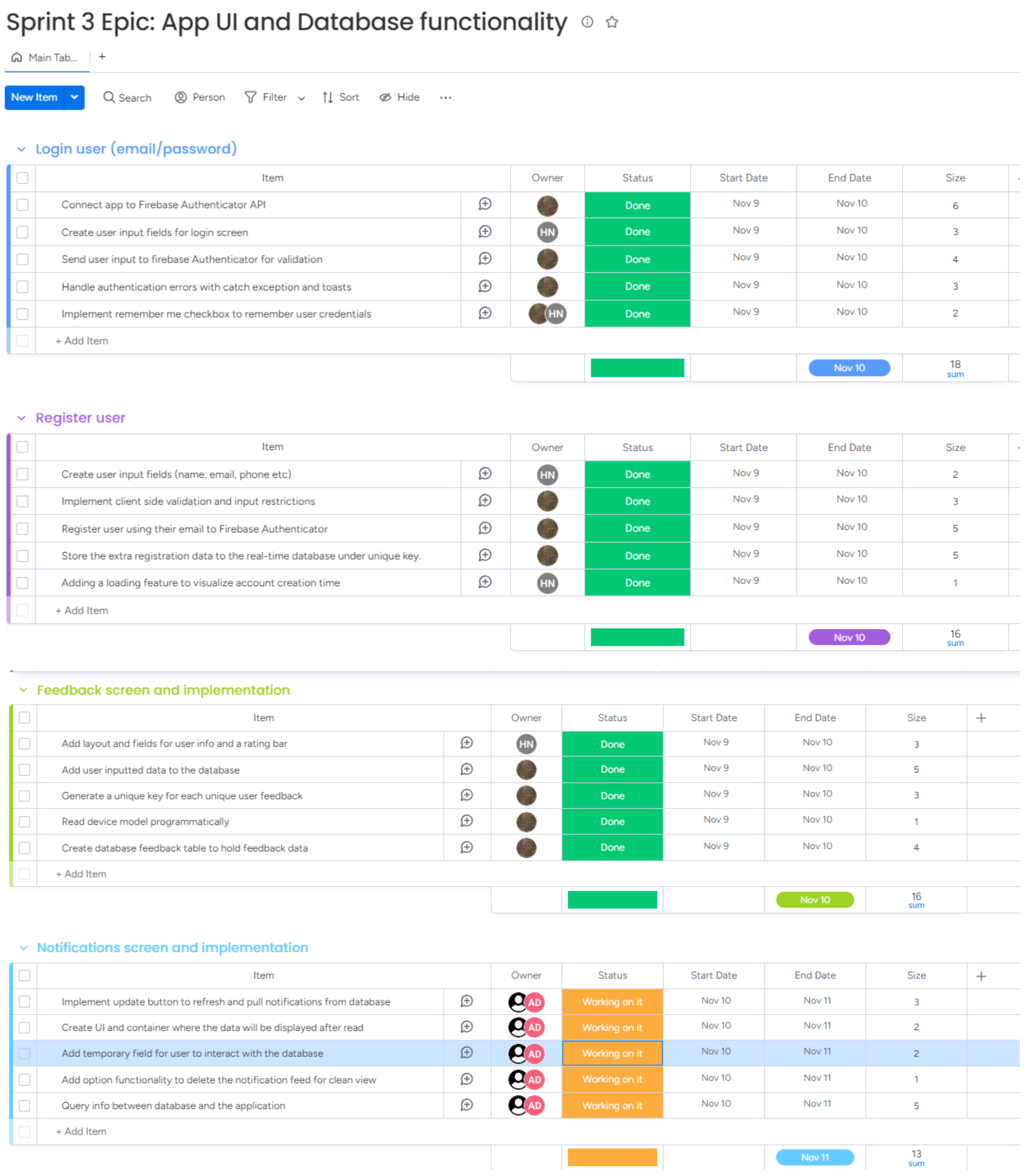
Rows per page: 50 1 – 1 of 1

Sprint Goals:

The sprint goals for sprint 3 are getting the following done:

- Fix and improve the UI and design of the following screens: dashboard, notifications, home.
- Create login, registration, feedback and about me page and all its functionalities.
- Connect the database to the app to allow login validation, registration of users, data gathering from user registration and user feedback.
- Implement a functioning UI for the settings screen and add 4 functionalities, such as switch theme (dark/light), lock screen orientation etc.
- Finish all functionalities of the menu items, “help” open’s the browser, about button goes to an about us fragment screen with our info, the feedback button and fragment and lastly a logout button for the user to log out from.
- Implement the google maps runtime permission that we failed to complete last sprint showing a location on the map.
- Further improving the app by using design principles that will help our code be clearer and create room for easy scalability.

Sprint Dashboard:



Sprint 3 Epic: App UI and Database functionality

Main Tab... | +

New Item | Search | Person | Filter | Sort | Hide | ...

Login user (email/password)

Item	Owner	Status	Start Date	End Date	Size
Connect app to Firebase Authenticator API	●	Done	Nov 9	Nov 10	6
Create user input fields for login screen	HN	Done	Nov 9	Nov 10	3
Send user input to firebase Authenticator for validation	●	Done	Nov 9	Nov 10	4
Handle authentication errors with catch exception and toasts	●	Done	Nov 9	Nov 10	3
Implement remember me checkbox to remember user credentials	● HN	Done	Nov 9	Nov 10	2
+ Add Item					
		🟩		Nov 10	18 sum

Register user

Item	Owner	Status	Start Date	End Date	Size
Create user input fields (name, email, phone etc)	HN	Done	Nov 9	Nov 10	2
Implement client side validation and input restrictions	●	Done	Nov 9	Nov 10	3
Register user using their email to Firebase Authenticator	●	Done	Nov 9	Nov 10	5
Store the extra registration data to the real-time database under unique key.	●	Done	Nov 9	Nov 10	5
Adding a loading feature to visualize account creation time	HN	Done	Nov 9	Nov 10	1
+ Add Item					
		🟩		Nov 10	16 sum

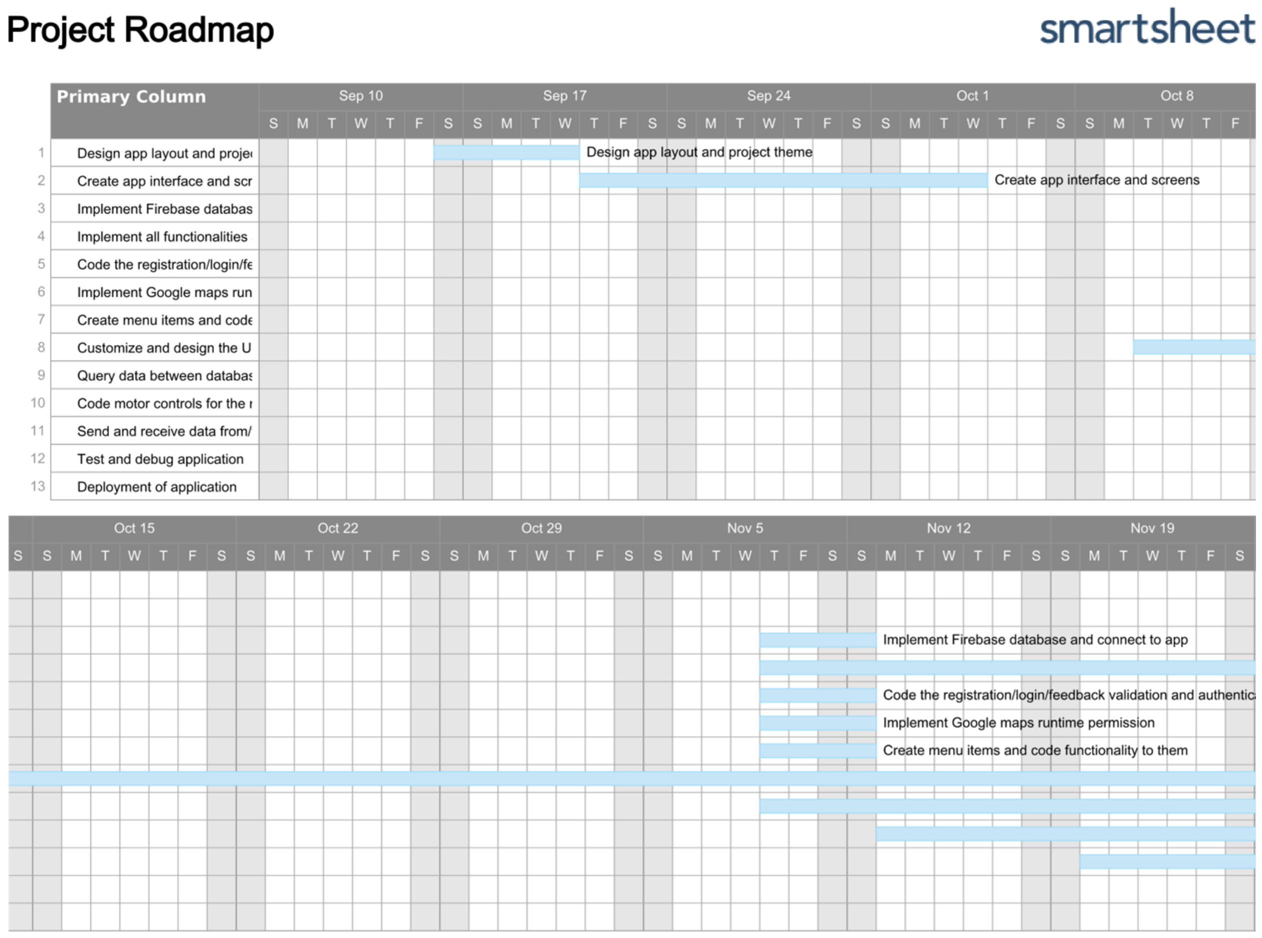
Feedback screen and implementation

Item	Owner	Status	Start Date	End Date	Size
Add layout and fields for user info and a rating bar	HN	Done	Nov 9	Nov 10	3
Add user inputted data to the database	●	Done	Nov 9	Nov 10	5
Generate a unique key for each unique user feedback	●	Done	Nov 9	Nov 10	3
Read device model programmatically	●	Done	Nov 9	Nov 10	1
Create database feedback table to hold feedback data	●	Done	Nov 9	Nov 10	4
+ Add Item					
		🟩		Nov 10	16 sum

Notifications screen and implementation

Item	Owner	Status	Start Date	End Date	Size
Implement update button to refresh and pull notifications from database	● AD	Working on it	Nov 10	Nov 11	3
Create UI and container where the data will be displayed after read	● AD	Working on it	Nov 10	Nov 11	2
Add temporary field for user to interact with the database	● AD	Working on it	Nov 10	Nov 11	2
Add option functionality to delete the notification feed for clean view	● AD	Working on it	Nov 10	Nov 11	1
Query info between database and the application	● AD	Working on it	Nov 10	Nov 11	5
+ Add Item					
		🟧		Nov 11	13 sum

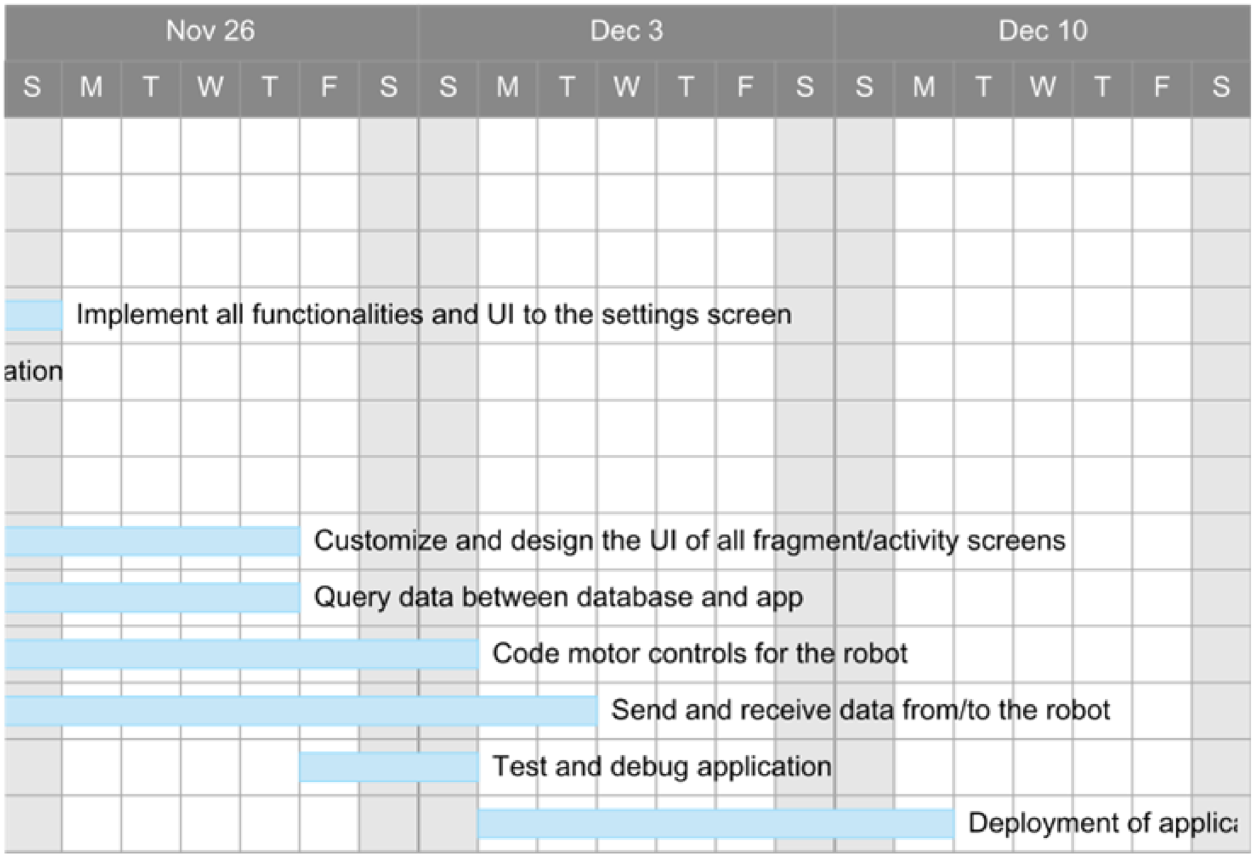
Gantt Chart:



Project Roadmap

	Primary Column
1	Design app layout and proje
2	Create app interface and scr
3	Implement Firebase databas
4	Implement all functionalities
5	Code the registration/login/fe
6	Implement Google maps run
7	Create menu items and code
8	Customize and design the U
9	Query data between databas
10	Code motor controls for the r
11	Send and receive data from/
12	Test and debug application
13	Deployment of application

smartsheet
Sep 10
Sep 17
Sep 24
Oct 1
Oct 8
Oct 15
Oct 22
Oct 29
Nov 5
Nov 12
Nov 19
S M T W T F S
Design app layout and project theme
Create app interface and screens
Implement Firebase database and connect to app
Code the registration/login/feedback validation and authentic
Implement Google maps runtime permission
Create menu items and code functionality to them

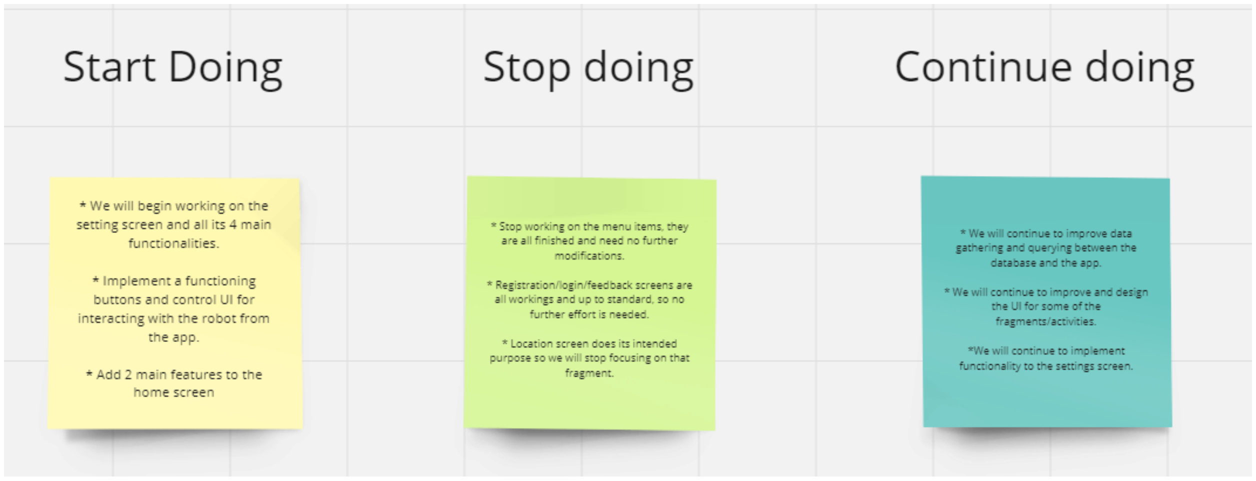


Daily Standup:

Date	Info	Who missed the meeting
04-10-2023	- Meeting to discuss what database to use - How we were going to implement said database - Overall our thoughts and ideas on the project so far	
07-10-2023	- Meeting to discuss the dashboard look/functionality - What the configuration screen was going to have - Expectations on when the code should be done	
10-10-2023	- Meeting to talk about the last features needed to be done	

	- Work on the in class presentation and powerpoint - Testing and quality control of the app	
8-11-2023	- Yesterday everyone scanned the deliverable and noted down what they were comfortable doing - Today we will assign tasks and stories for each group member with an expected finish date - Nothing is blocking progress	
10-11-2023	- Yesterday members worked on All screen UI and database setup, along with the login/registration/feedback functionality and storing on the database. - Today team members will continue working on login functionality, UI design for the dashboard and implement the settings screen with its features - Nothing is blocking the progress	
TBD	TBD	

Sprint Retrospective:



Start Doing

* We will begin working on the setting screen and all its 4 main functionalities.

* Implement a functioning buttons and control UI for interacting with the robot from the app.

* Add 2 main features to the home screen

Stop doing

* Stop working on the menu items, they are all finished and need no further modifications.

* Registration/login/feedback screens are all workings and up to standard, so no further effort is needed.

* Location screen does its intended purpose so we will stop focusing on that fragment.

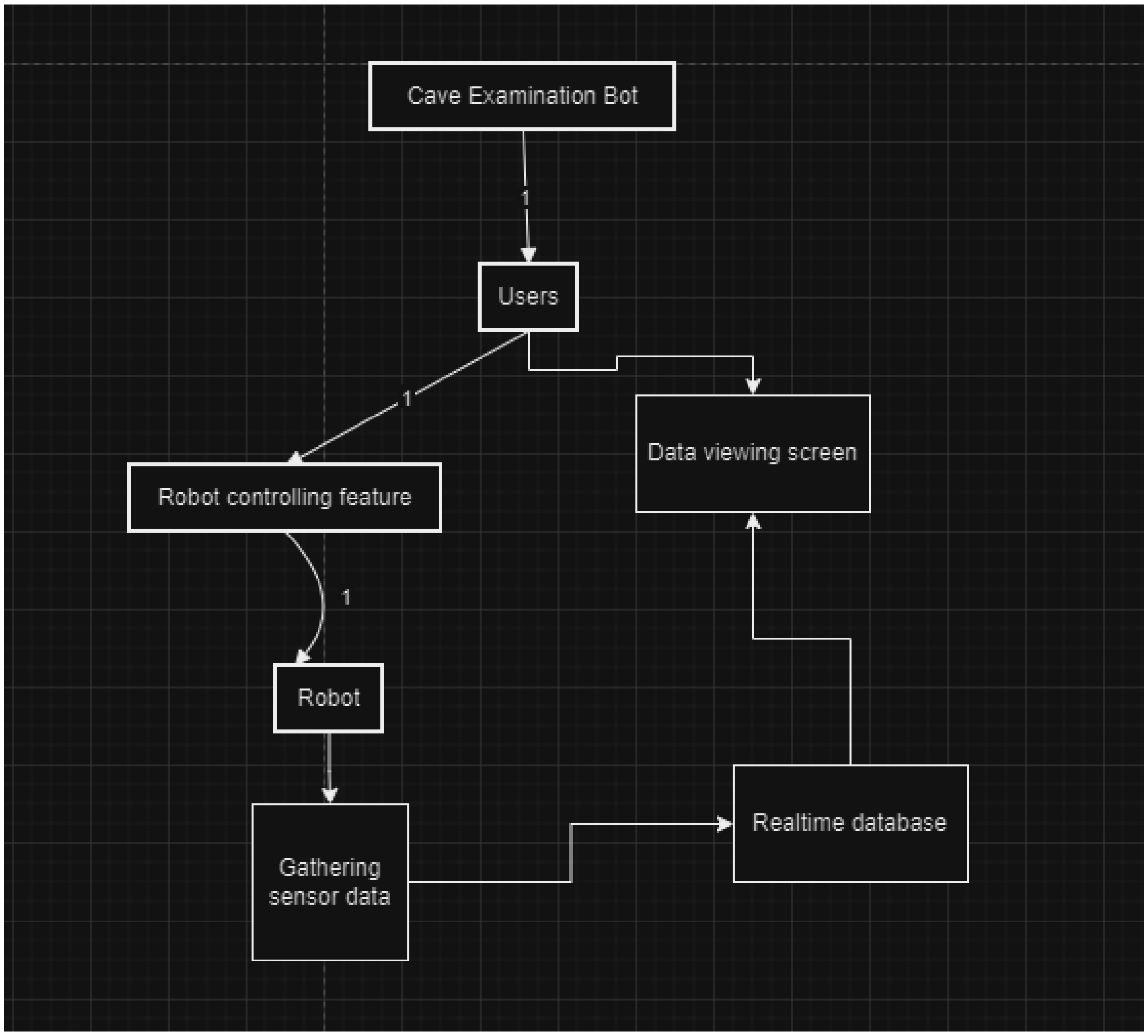
Continue doing

* We will continue to improve data gathering and querying between the database and the app.

* We will continue to improve and design the UI for some of the fragments/activities.

*We will continue to implement functionality to the settings screen.

C4 Model:



Design Principles Used:

We used the single responsibility principle in this code, all google map methods that interact with location of device and map interaction are under the location fragment.

```
public class LocationFragment extends Fragment implements OnMapReadyCallback {
```

```
@Override
public void onMapReady(@NonNull GoogleMap googleMap) {
   mMap = googleMap;
   LatLng ontario = new LatLng(51, 85);
```

```
    mMap.addMarker(new
MarkerOptions().position(ontario).title(getString(R.string.ontario)));
    mMap.moveCamera(CameraUpdateFactory.newLatLng(ontario));
}
```

We used closed/open design principle here since this alertdialog is open for extension by calling it in other methods and adding more features but its closed for modification so new errors are not introduces, potentially breaking the app since this function is called in all fragments.

```
public void showExitAlertDialog() {
    // Create an AlertDialog
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setIcon(android.R.drawable.ic_dialog_alert);
    builder.setTitle(R.string.exit_confirmation);
    builder.setMessage(R.string.do_you_want_to_exit_the_app);

    // Add buttons to the AlertDialog
    builder.setPositiveButton(R.string.exit, (dialog, which) -> {
        // Close the app
        finish();
        System.exit(0);
    });


    builder.setNegativeButton(R.string.stay, (dialog, which) -> {
        // User chose to stay, reset the flag
        backArrowPressed = false;
    });

    // Show the AlertDialog
    AlertDialog dialog = builder.create();
    dialog.show();
}
```

Design Patterns Used:

In this code we used builder pattern, we built on the alertDialog by chaining its methods and creating its properties in a simple way making it easy to customize.

```
public void showExitAlertDialog() {
    // Create an AlertDialog
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setIcon(android.R.drawable.ic_dialog_alert);
    builder.setTitle(R.string.exit_confirmation);
    builder.setMessage(R.string.do_you_want_to_exit_the_app);
```

```
   // Add buttons to the AlertDialog
   builder.setPositiveButton(R.string.exit, (dialog, which) -> {
       // Close the app
       finish();
       System.exit(0);
   });


   builder.setNegativeButton(R.string.stay, (dialog, which) -> {
       // User chose to stay, reset the flag
       backArrowPressed = false;
   });

   // Show the AlertDialog
   AlertDialog dialog = builder.create();
   dialog.show();
}
```

This code uses the observer design pattern, using the implements onMapReadyCallback observer, the LocationFragment notifies other classes when the onMapReady is ready for interaction.

```
public class LocationFragment extends Fragment implements OnMapReadyCallback {
@Override
public void onMapReady(@NonNull GoogleMap googleMap) {
   mMap = googleMap;
   LatLng ontario = new LatLng(51, 85);
   mMap.addMarker(new
MarkerOptions().position(ontario).title(getString(R.string.ontario)));
   mMap.moveCamera(CameraUpdateFactory.newLatLng(ontario));
}
```

Coding Work Progress Since Deliverable 2:

The progress that got done so far in sprint 3 from the last sprint:

- Login/registration screens with full implementation, UI firebase authentication and creation of users, data collecting of user's sign up information.
- Feedback screen with user info fields, rating bar and comment field, data is now collected and stored in the firebase realtime database.
- Menu items about, and logout buttons have been implemented, log out logs the user out and about button sends the user to a new fragment where it details our app info.
- Database creation and connection with the app.
- The UI on some screens were fixed and improved.

- Google maps api and runtime permissions implemented showing the map with a location.

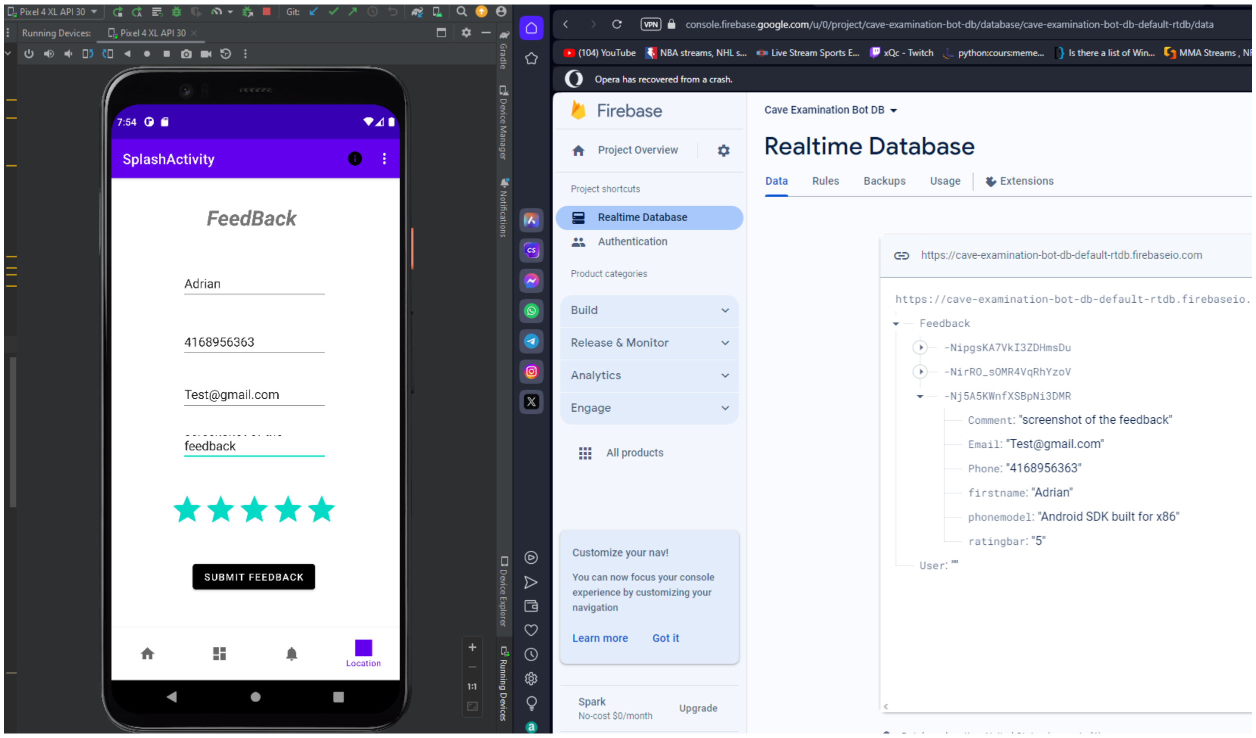
Runtime Permission Implemented:

The runtime permission implemented was the google maps API, it asks the user for location and shows on the map the current user location if the user agreed and has location on in their device.

Two main Functionalities Implemented:

The first main functionality implemented was the runtime permission of the google maps api allowing the user to see his location, or a fixed location if the user has location off.
The second was the authentication and database querying of the user's login/registration information with the Firebase database. The user's info gets stored in the database and retrieved when checking if the user's input matches the database user records.

Feedback Screen Info Database:



```
https://cave-examination-bot-db-default-rtdb.firebaseio.com

https://cave-examination-bot-db-default-rtdb.firebaseio.
  Feedback
    -NipgsKA7VkI3ZDHmsDu
    -NirRO_sOMR4VqRhYzoV
    -Nj5A5KWnfXSBpNi3DMR
      Comment: "screenshot of the feedback"
      Email: "Test@gmail.com"
      Phone: "4168956363"
      firstname: "Adrian"
      phonemodel: "Android SDK built for x86"
      ratingbar: "5"
  User: ""
```

Adrian
4168956363
Test@gmail.com
feedback