Technical Project Report - Android Module

# SaveNight

Group 09

| | |
|---|---|
| Subject: | Computação Móvel |
| Date: | Aveiro, 03/01/2023 |
| Students: | 81020: Diogo Aguiar<br>98385: Tiago Coelho |

Project abstract:

The application concept is a social networking app for users to connect with each other and share their location, photos, and messages. The key achievements of the app include the use of NearbyConnections to allow communication between nearby devices, the implementation of a map feature to show the user's and their friends' locations, and the creation of a mini-game to determine a user's drunkenness level. The app also includes features such as a chat function and the ability to share photos in a feed.

Report contents:

# 1 Application concept

This app is a social networking app that allows users to connect with their friends and see their locations in real-time on a map. It also has features such as a chat function, a "drunk test" game, and a camera function that allows users to take and share photos. The target users are people who want to keep in touch with their friends and stay updated on their locations, during a night out. The app would benefit users by providing an easy way to connect with friends and know their whereabouts, as well as providing entertainment through the drunk test game and the ability to share photos.

# 2 Implemented solution

## Architecture overview (technical design)

The application starts with the SignIn/SignUp page to identify the users, and then goes to the main activity that contains the navigation bar at the bottom of the screen.

Here the user can navigate between the fragments in the app:

- map,
- groups,
- drunk test,
- camera
- and feed.

The SignIn/SignUp page is an activity implemented with firebase to save the email/passsword and do the authentication of the user.

The main activity is the first activity that is called when the app is opened and is the activity that is always visible to the user. The main activity is the only activity visible to the user and is the only activity that is not destroyed when the user navigates between fragments.

Whenever the user clicks a button on the navigation bar, the main activity is called and the fragment that the user wants to see is called.

It is necessary if the user is in a group to maintain the connection with the group and the other users of the group. Thus, the lobby fragment is always kept in the fragments pile and never destroyed unless the user leaves the group. In this case, the lobby fragment is destroyed, and the user is redirected to the groups fragment.

- **Maps**

The Maps fragment is responsible for showing the user their current location and the location of their friends who are in the same group as them. The Maps fragment is called when the user clicks the "Map" button in the bottom menu of the application. The Maps fragment is called through the "onMapReady" method which is called when the map is ready to be shown to the user.

If the user does not have the necessary permissions to show their location, the Maps fragment displays an error message and asks the user to grant the necessary permissions. The Maps fragment displays the user's location through a blue marker and the location of their friends through red markers that are placed on the map according to the current location of their friends, and the name of the friend is placed on the marker.

To get the location of friends, the list of friends' locations and their names are retrieved from the Shared Preferences.

- **Groups**

In terms of groups, there are four fragments: Groups(), which is the main one where the user chooses whether they want to create a group or join one; Create_group(), where the user starts advertising so that other users can join; Join_group(), which is the fragment where the user does discovering of the hosts and can request to join the group; and finally, Lobby(), which has the list of members and the chat.

Throughout the group process, NearbyConnections is used to for discover, advertise, and exchange data. NearbyConnections is a Google service that allows communication between nearby devices without the need for a network, such as Bluetooth. NearbyConnections is used for communication between devices and for exchanging data, such as the location of group members. This was our biggest challenge because NearbyConnections led us to various bugs that took a lot of time and research to fix.

NearbyConnections is used in three fragments: Create_group(), Join_group(), and Lobby(). Create_group() is the fragment that starts advertising, Join_group() is the fragment that starts discovering, and Lobby() is the fragment that receives data from group members, such as location. It was necessary to be very careful with the calls to the fragments because if it was not done correctly, NearbyConnections did not work properly and was unable to receive data from group members, leading to disconnections and bugs.

The location exchange is always made when there is a message exchange in the chat and is saved in the Shared Preferences in order to be used in the map. Two adapters were created, one for the list of advertisers and one for the chat messages. The advertiser list adapter is used in Join_group(), and the chat message adapter is used in Lobby().

- **DrunkTest**

DrunkTest is a fragment that is like a mini game that gives a drunkenness percentage to the user. The game basically has a beer on the center of the screen and the user need to maintain the beer in the center for 5 seconds, keeping the phone practically immobile parallel to the floor.

The game is based on the accelerometer sensor, so the user needs to keep the phone parallel to the floor and not moving it. For that purpose, the game has a timer that starts when the user touches the screen and stops when the user maintains for 5 seconds the beer centered or when the max time passed.

When game starts the user can see a red circle in the center of the screen, that is the area where the beer should be. The user can see the beer moving in the screen and when the beer is in the red circle the user can see a green circle around the beer.

- **Camera**

Camera is a fragment that allows the user to take pictures, or select pictures from the gallery, and share them with other users.

It starts by asking permission to use the dispositive camera. After granted the user can start using the fragment.

It uses the firebase storage to save the pictures from the users, and also uses the firebase real time database to save the respective url of the pictures, to update the Feed page.

- **Feed**

Feed fragment is used to see the pictures shared by all user, and has the logout button on top of the page. It uses one adpater with recyclerView to display the pictures,and Glide to load the picture url, that was previously saved when the user shared the picture.

## Implemented interactions

The first time the app is opened, the user is asked to register/login, as shown below respectively.

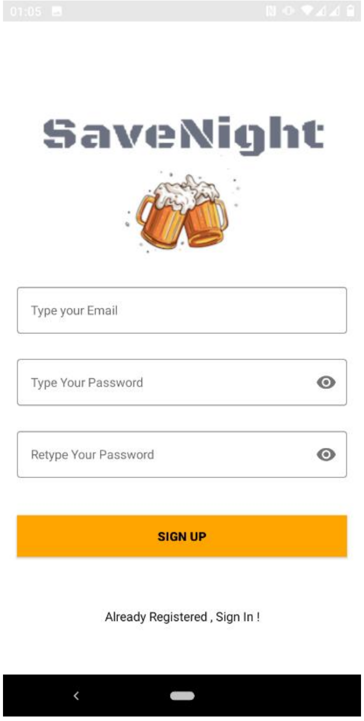


*Figure 1 - Regist*

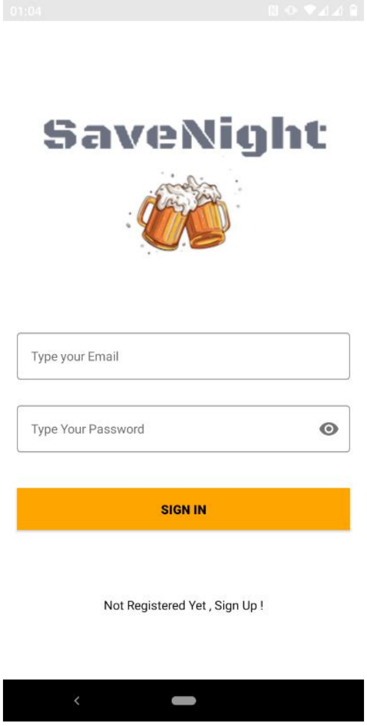


*Figure 2 - Login*

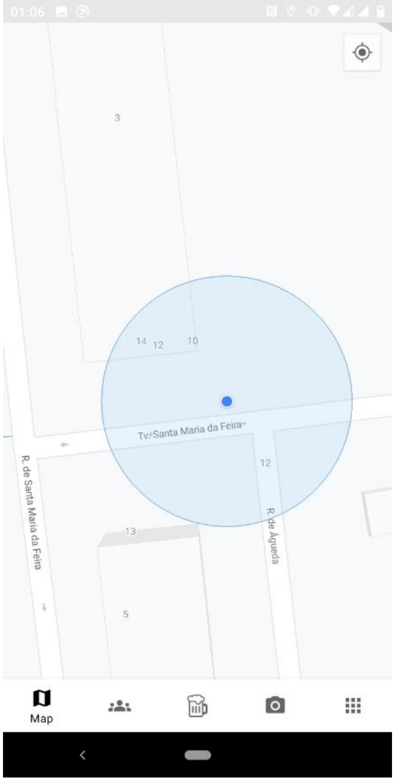
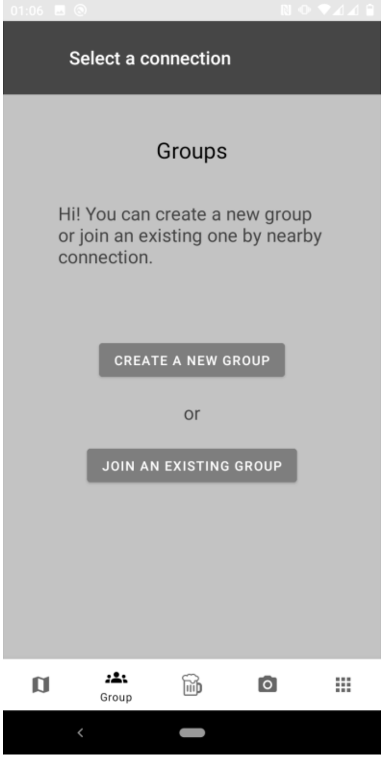


*Figure 3- Map*

After logging in or creating an account, we are directed to the Map, which also already displays the nav bar, allowing us to navigate to the group tab, drunk test, camera, or feed. As for the map, we can see our real-time location and can also zoom in or out and explore the map. There is also a button to return to our position in case we have been exploring the map.

In the group tab, the user is greeted and given an explanation of groups. They can then decide whether they want to create a group and be the host, or join a new group by searching for hosts. If they want to create a group, they can start advertising and wait for requests to join their group. If they want to join a group, they can start discovering and wait for available hosts in the area to appear.

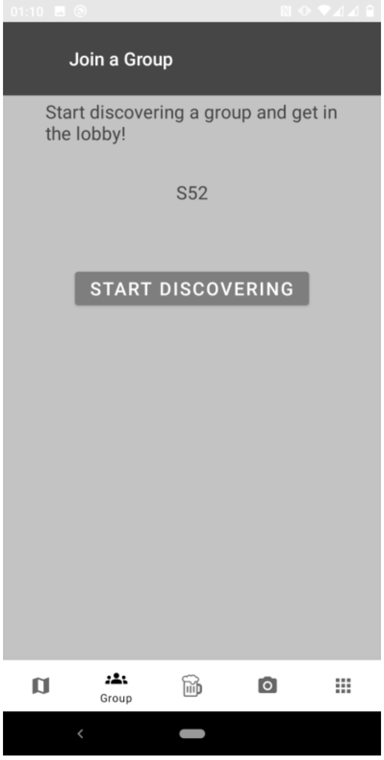


Select a connection

Groups

Hi! You can create a new group or join an existing one by nearby connection.

*Figure 4 - Inicial Page*

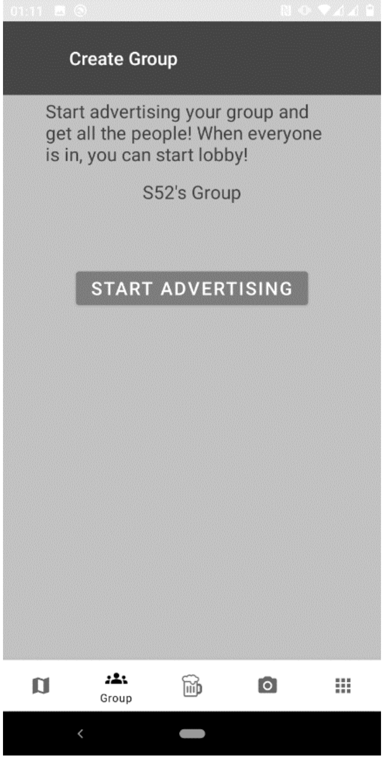


Join a Group

Start discovering a group and get in the lobby!

S52

*Figure 5 - Join page*



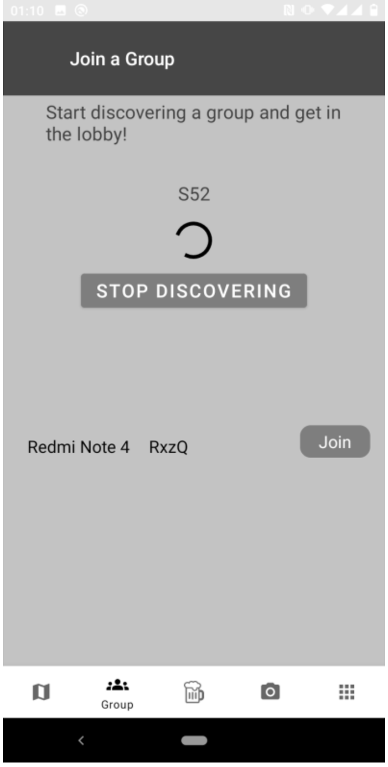
Create Group

Start advertising your group and get all the people! When everyone is in, you can start lobby!

S52's Group

*Figure 6 - Create page*

Continuing in the group tab, after clicking to connect, both the host and discover are asked to accept the connection, where a code can be seen if it is necessary to confirm. If both accept, the discover is redirected to the lobby, and the host waits for everyone to enter the group, being able to enter the lobby when everyone has entered. In the lobby, you can see the members and exchange messages.



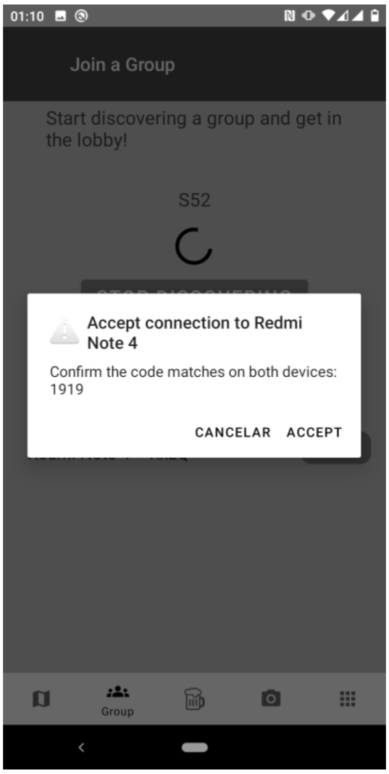
Join a Group

Start discovering a group and get in the lobby!

S52

Redmi Note 4 RxzQ

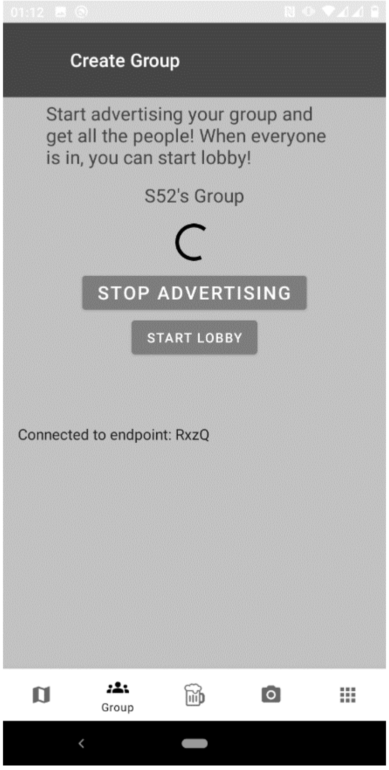
*Figure 7 - Discover page*



Accept connection to Redmi Note 4

Confirm the code matches on both devices: 1919

*Figure 8 - Message to accept*



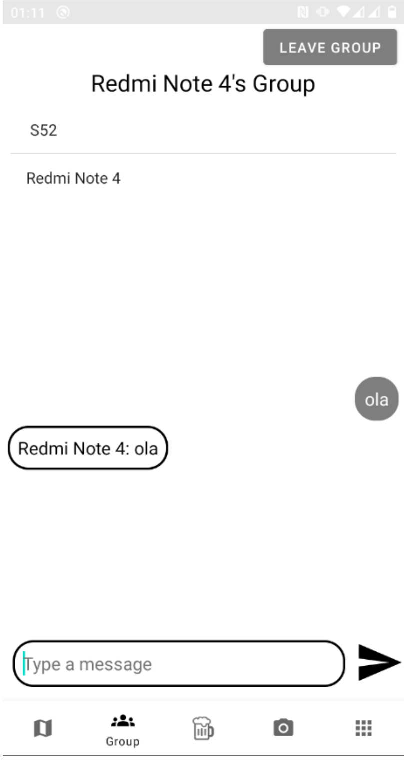
Create Group

Start advertising your group and get all the people! When everyone is in, you can start lobby!

S52's Group

Connected to endpoint: RxzQ

*Figure 9 - Advertiser page*



Redmi Note 4's Group

S52

Redmi Note 4

ola

Redmi Note 4: ola

*Figure 10 - Lobby page*

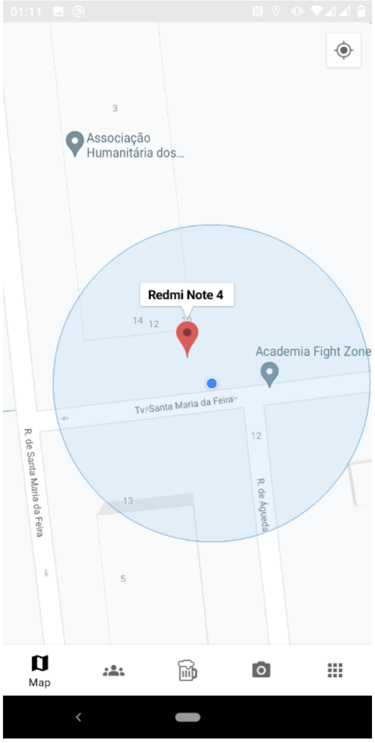
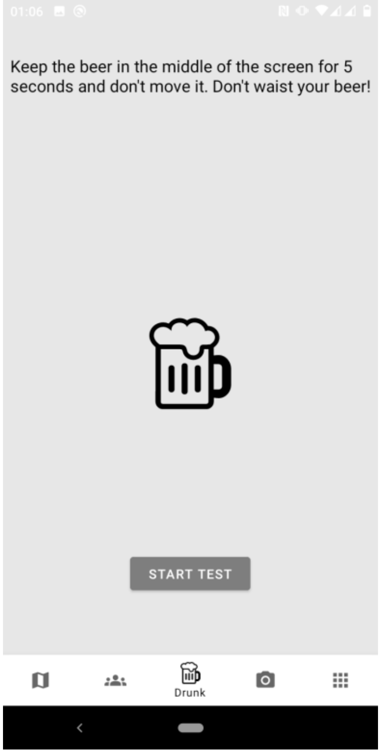


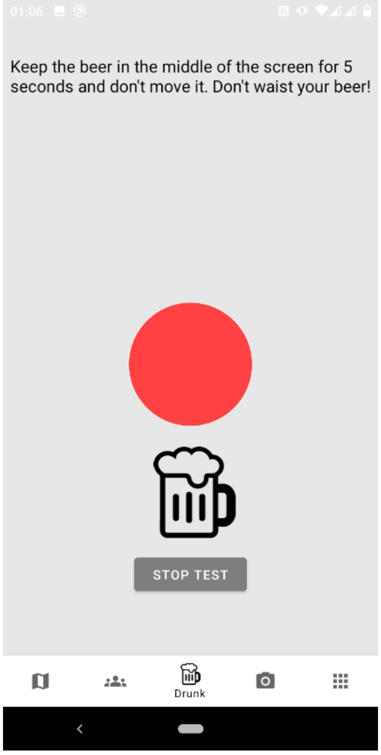
*Figure 2 - Map with location*

In the DrunkTest tab, the user can start the test by clicking on "Start" and after completing the goal, their drunkenness is presented to them. The test ends and the user can do it again if they wish.


01:06

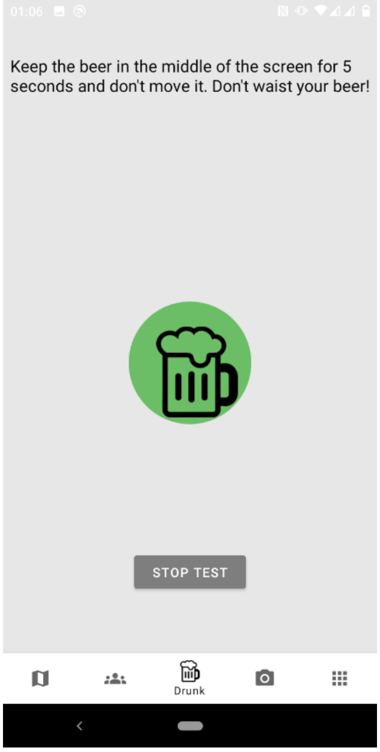
Keep the beer in the middle of the screen for 5 seconds and don't move it. Don't waist your beer!

*Figure 11 - Drunk Test*


01:06

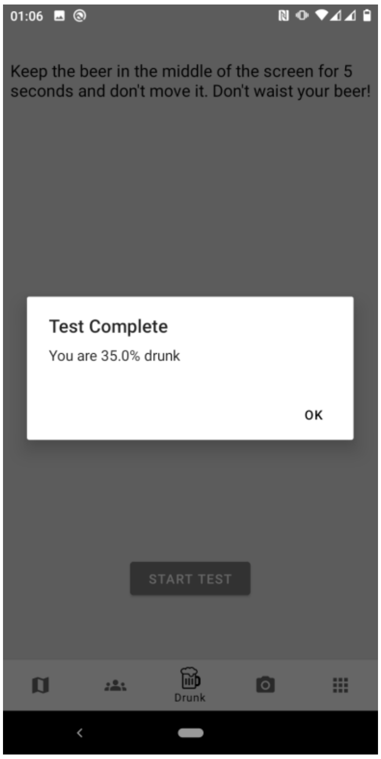
Keep the beer in the middle of the screen for 5 seconds and don't move it. Don't waist your beer!

*Figure 13 - Red circle*


01:06

Keep the beer in the middle of the screen for 5 seconds and don't move it. Don't waist your beer!

*Figure 14 - Green circle*

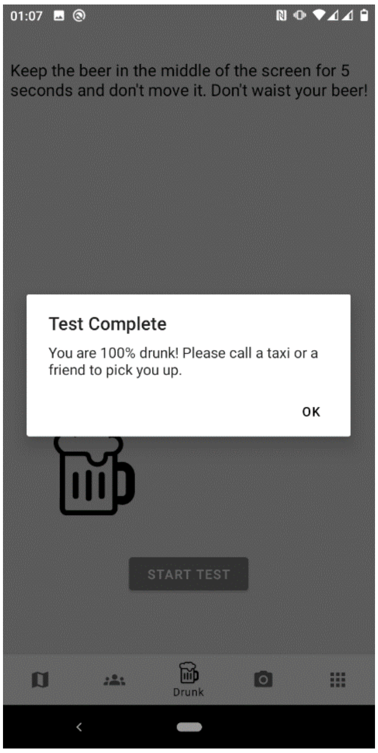

01:06

Keep the beer in the middle of the screen for 5 seconds and don't move it. Don't waist your beer!

Test Complete

You are 35.0% drunk

*Figure 15 - Test result*


01:07

Keep the beer in the middle of the screen for 5 seconds and don't move it. Don't waist your beer!

Test Complete

You are 100% drunk! Please call a taxi or a friend to pick you up.

*Figure 15 - Max Time*

On the camera tab, it is possible to take photos or select photos of the galery to upload. It is also possible to see a preview of the photo and share it to the feed.

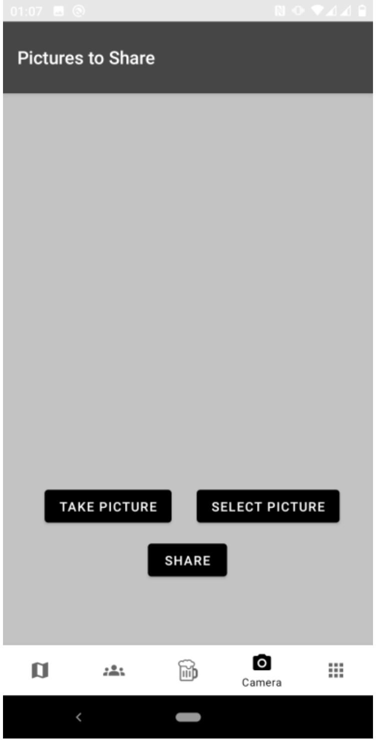


*Figure 16 - Camera page*

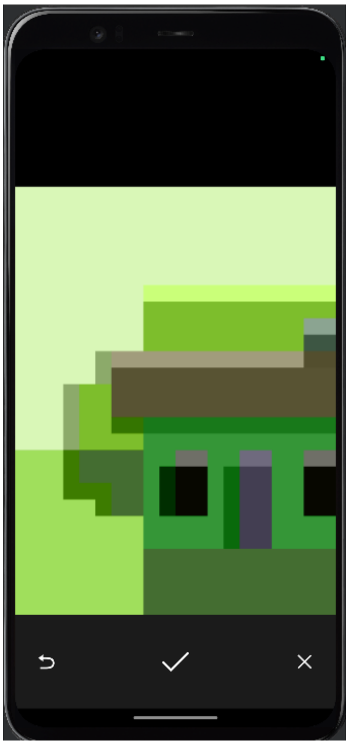


*Figure 17 - Photo taken*

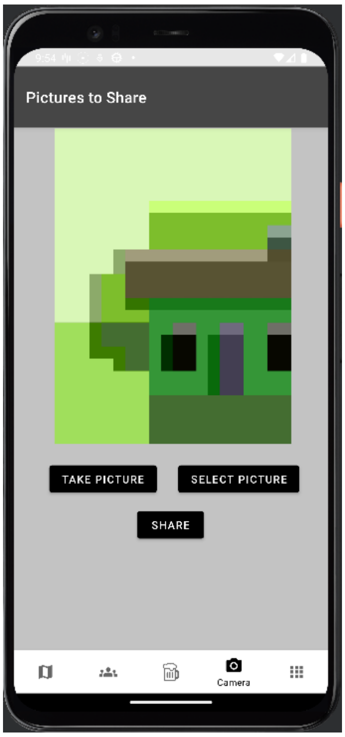


*Figure 18 - Preview*

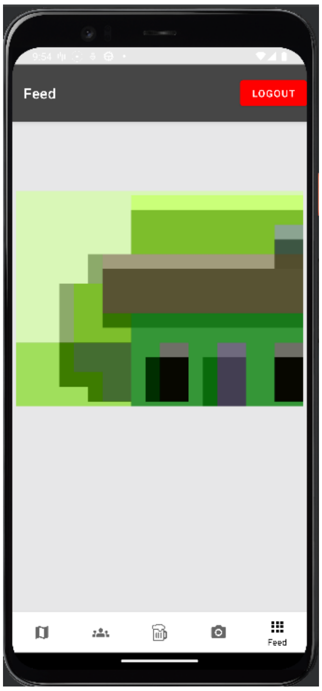


*Figure 19 - Feed page*

Finally, the Feed tab is where users can see the feed of all the photos.

### Project Limitations

There are several known limitations and unimplemented features in this project. These include:

- Improving the feed to make it more user-friendly
- Working around bugs in NearbyConnections to improve the reliability of the application
- Adding notifications for the chat group feature
- Saving group connection and other data when the user leaves the app

These are all planned features that are intended to enhance the overall user experience and make the app more useful and convenient for the target users.

## 3 Conclusions and supporting resources

### Lessons learned

One major "hard problem" we faced during the implementation process was dealing with bugs related to NearbyConnections. This service was used for communication between devices and for exchanging data, such as the location of group members. However, we encountered various bugs that took a lot of time and research to fix. In hindsight, we should have spent more time thoroughly testing NearbyConnections before implementing it in our app.

One aspect of Android development that was somewhat of a surprise to us was the ease of implementing user interfaces. The use of Fragments made it very easy for us to switch between different screens within the app, and the Android Studio layout editor made designing the layout of each screen very intuitive.

In terms of the Computação Móvel course, we would suggest that future students spend more time on debugging and testing their apps before finalizing them. It is important to thoroughly test all features and iron out any bugs as early as possible in the development process to save time and frustration in the long run.

### Work distribution within the team

Taking into consideration the overall development of the project, the contribution of each team member is distributed as follow:  Diogo Aguiar did 50% of the work, and Tiago Coelho contributed with 50%.

### Project resources

| **Resource:** | **Available at:** |
|---|---|
| Code repository: | https://github.com/trcoelhoo/CMProjectAndroid |
| Ready-to-deploy APK: | https://github.com/trcoelhoo/CMProjectAndroid/blob/main/app-debug.apk |

## Reference materials

https://developers.google.com/nearby/connections/overview

https://developer.android.com/training/data-storage/shared-preferences

https://developer.android.com/reference/android/hardware/Sensor#TYPE_ACCELEROMETER

https://github.com/bumptech/glide