



I'm Here

**Android Application
CMPE 277 Fall 2018 - Project Report
November 29, 2018**

Team 7

Deepak Talwar

Chen-Feng Huang

Donghao Su

Executive Summary	3
Introduction	3
Motivation and Background	4
Android Application Structure	4
Major Activities	5
Initiate Activity	5
Login Page Activity	6
Signup Page Activity	7
Home Activity	8
User Discovery Activity	9
Chatter Activity	10
AR Activity	11
Other Classes and Components	12
WiFiDirectBroadcastReceiver	12
WiFiP2pService	12
CustomAdapter	12
ChatModel	13
Future scope	13
Conclusion	13
References	13

Executive Summary

Have you ever tried to contact your friends or loved ones when you're out on a hike? The only way you can do it right now is either by shouting out their name loudly, or by using an expensive satellite based IoT devices. But what about our smartphones? They are full of expensive sensors and hardware, can we make use of them to solve this use case?

In this project, we tackle this problem by using offline capabilities of our smartphones. Specifically we study Wi-Fi direct, the peer-to-peer protocol that uses Wi-Fi hardware as the physical layer as the communication platform, and use Augmented Reality features to locate other users where maps may not be available. This allowed us to create an app that can be of utmost utility in times of need. Locating and communicating with your friends in the forest on a hike, at the beach, on a boat, on a plane, literally anywhere where you don't have access to the internet. It can also be used in crowded places such as music concerts and sports events where the network may be impacted due to heavy usage.

The objective of making this application is to show different offline capabilities of our phones and create a use case for Augmented reality. This report details the various activities we created to make this app work, its flow and the screenshots.

Introduction

"I'm here" android application is developed by Deepak Talwar, Chen-Feng Huang and Donghao Su for the final group project assignment for CMPE 277 - Fall 2018 course. This application introduces a unique new way of using our modern smartphones as chatting devices in the absence of availability of any external network, such as cellular, 2G, 3G, 4G LTE, 5G or Wi-Fi Access Point. In addition, this application also allows for users to find their friends using Augmented Reality (AR) without the use of any data from the internet, such as map files. Hence, the name "I'm here", as if your friend is letting you know where they are through the phone without any network. In fact, the entire application can run on the phone completely in Airplane Mode.

This application leverages the power of peer-to-peer connectivity features of modern smartphones to achieve this connectivity without external networks. Specifically, we use Android's Wi-Fi peer-to-peer package, or Wi-Fi Direct as it is more commonly known. Wi-Fi direct allows a smartphone to locate another smartphone advertising a local service. It can then connect to that service and communicate with it. It also provides maximum range of up to 2 km without the use of any external hardware!

The Augmented Reality (AR) location of friends is also completely enabled by sensors inside the phones. On successful connection over Wi-Fi Direct, the devices exchange their location

information and use in-built GPS, camera and compass sensors to point in the direction of the friend. The app also shows a pin at the location of the friend. This gives the locator a clear idea of where the friend is relative to them and where they need to be headed. The app also provides a direct GPS distance (as the crow flies) to the location of the friend, giving them an idea of how far they need to go.

The use cases of this app are not difficult to imagine. Locating and communicating with your friends in the forest on a hike, at the beach, on a boat, on a plane, literally anywhere where you don't have access to the internet. It can also be used in crowded places such as music concerts and sports events where the network may be impacted due to heavy usage. Just ask your friends to install and you're ready to go. Another major use case could be to broadcast or contact friends in disaster or emergency situations when traditional networks may be down or not working. In addition, it could be used by lost hikers and skiers to find their friends. All this, without any data costs.

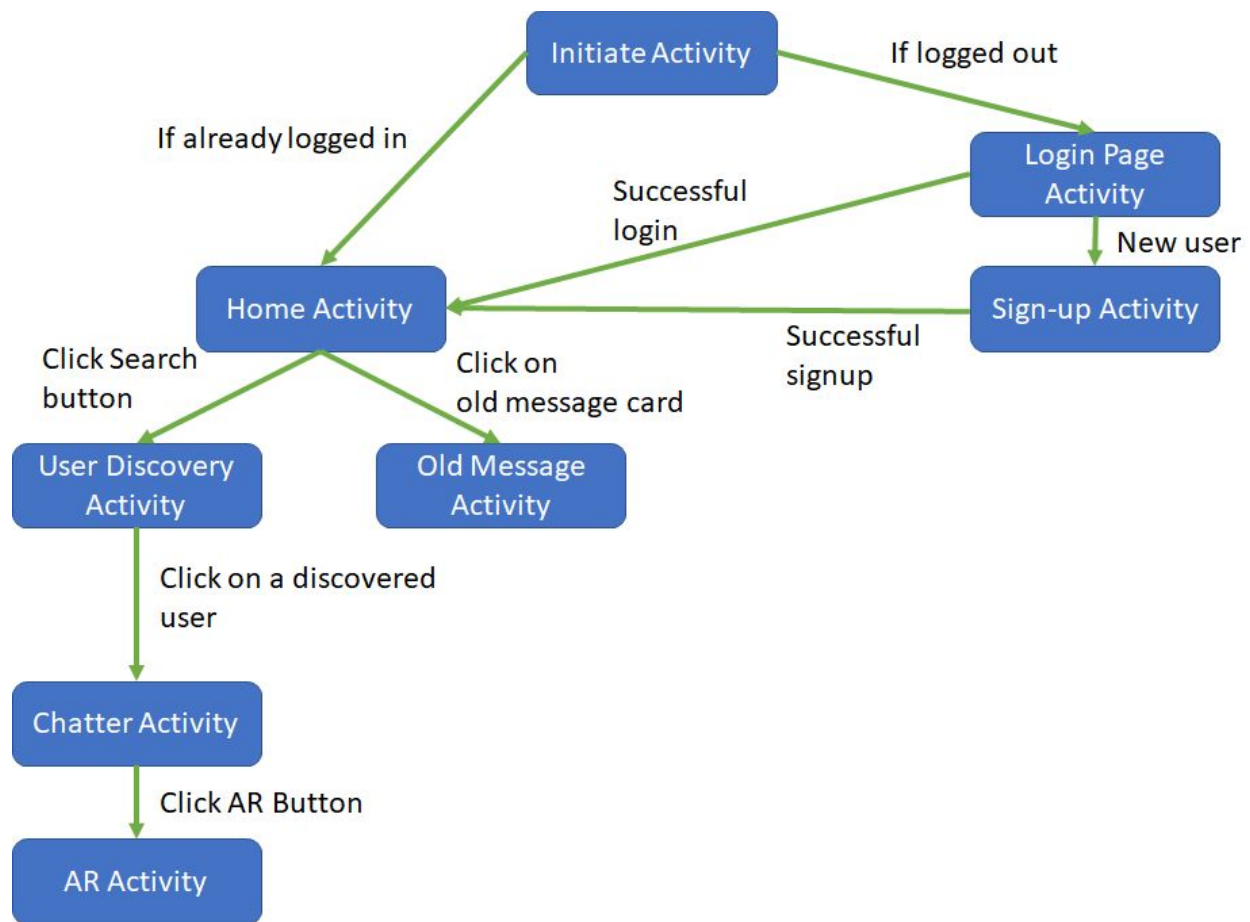
Motivation and Background

The motivation for such an app came from a personal experience. Last summer, I went on a trip to swimming hole with a few friends and we arrived in different cars. As we were deep into the wilderness, there was no cellular coverage 30 minutes before we arrived at our destination, so we didn't get a chance to collectively decide the swimming hole we wanted to go to. As a result, we all parked at different places and couldn't find each other for the entire day. Although, we still had fun, it would have been a lot better had all of us been together. This got me thinking if there was a way to leverage our phones' hardware to create a network to communicate between each other, and when I looked into it, I realized we could implement this using Wi-Fi Direct. This project is a step in producing a commercially viable and fully featured Android application.

On researching how hikers, skiers etc., communicate in the wilderness, we found that they use satellite connected IoT devices that are very slow, expensive and unreliable. They also don't provide full 24 hour coverage, they are very heavily dependent on whether or not the satellite is above you in the sky and there's no way to tell that visually. They also do not provide ways to locate friends or exchange GPS locations easily. This made us realize that we could solve many of these problems just by leveraging the capabilities of our smartphones.

Android Application Structure

Figure 1 below shows the Android Application Structure and Flow. It lists the relationship between activities and how a user navigates the app. Details on each activity will follow in following sections.



Major Activities

“I’m here” Android application has 8 activities in total. This section describes what these activities contain and what their roles are.

Initiate Activity

Initiate activity is the first splash page that shows up when the user opens the app. This activity uses the `activity_initiate.xml` layout. This layout shows the “I’m here” logo front and center and stays open for 2 seconds. The whole purpose of this activity is to check if the user is already logged in or not and choose which activity to open thereafter. If the user is already logged in, this activity opens the Home Activity, otherwise it opens the Login Page Activity where the user can log back in. This activity provides a good user experience by choosing the correct activity to start depending on the situation.

To determine whether the user is logged in or not, the activity queries the Couchbase Lite NoSQL database.

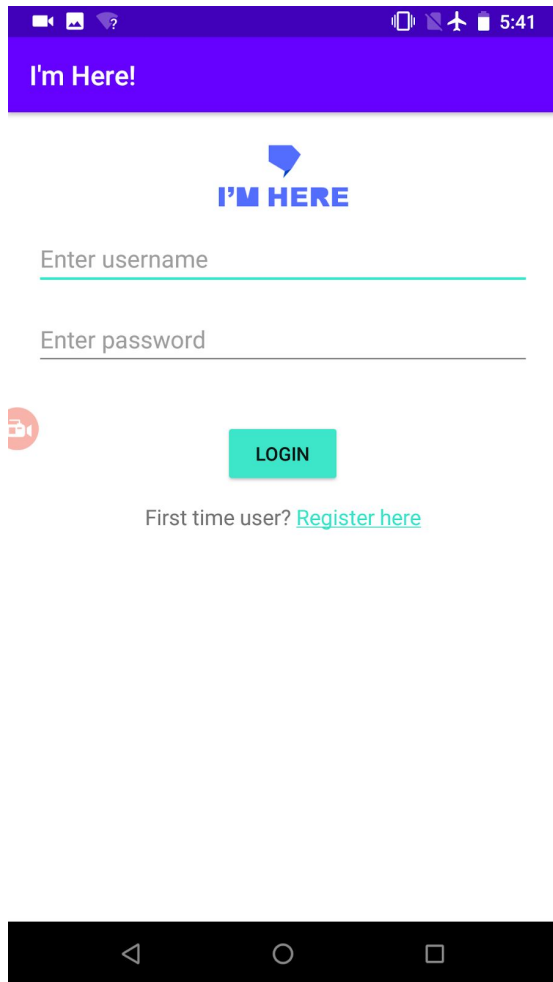


Login Page Activity

Login Page activity is started by the Initiate activity if the user is not logged in. This activity uses the `activity_login_page.xml` layout file. This activity shows two fields for username and password respectively, has a login button and also a Register here link for new users to register and sign up for the app.

When a user enters their credentials and presses login, this activity queries the Couchbase Lite NoSQL database for login credentials and logs the user in if found. Else, it throws a Toast explaining that no such user was found. On successful login, the user is taken to Home Activity.

First time users who do not have a login must click on “Register here” to go to Sign-up Activity. There they can create a new account.

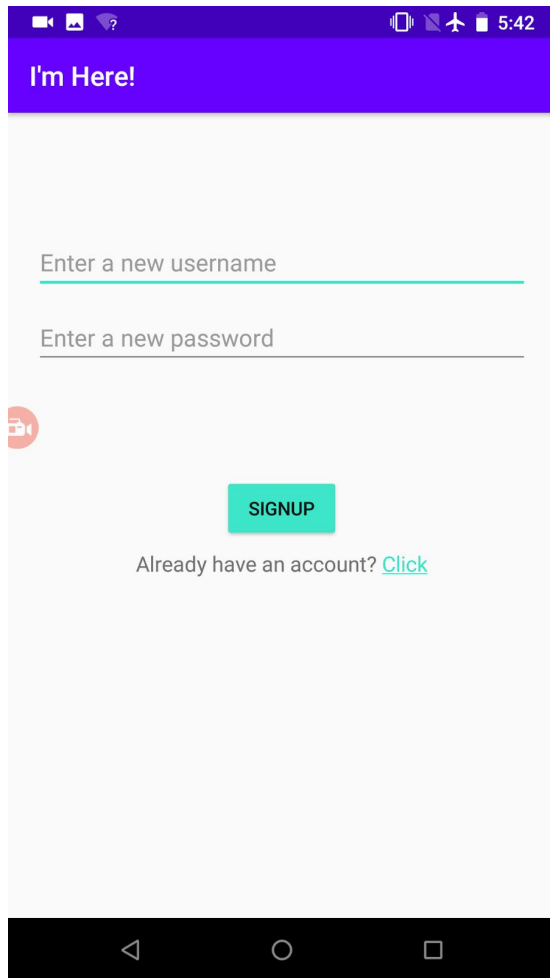


The screenshot shows a mobile application interface for a login page. At the top, there is a status bar with icons for camera, location, Wi-Fi, and battery, along with the time 5:41. Below the status bar is a purple header with the text "I'm Here!". Underneath the header is a blue speech bubble icon containing the text "I'M HERE". Below this are two input fields: "Enter username" and "Enter password". To the left of the "Enter password" field is a red circular icon with a white camera symbol. Below the input fields is a green "LOGIN" button. At the bottom, there is a link that says "First time user? [Register here](#)". The bottom of the screen shows a black navigation bar with three white icons: a back arrow, a circle, and a square.

Signup Page Activity

This activity is started by clicking on "Register here" on Login Page Activity. This activity is responsible for accepting new credentials and creating a user account for a new user. It has two EditText fields, for username and password specifically, and a Sign Up button that the user must press after choosing the username and password. It also has a button that takes the user back to the Login Page Activity in case the user accidentally came to Signup Page activity but intends to login to their own account.

On successful sign up, a unique ID (called UUID) is generated for the user and stored along with the login credentials in the database. The user is then taken to Home Activity. If the username entered already exists, a Toast is thrown that asks users to login instead.

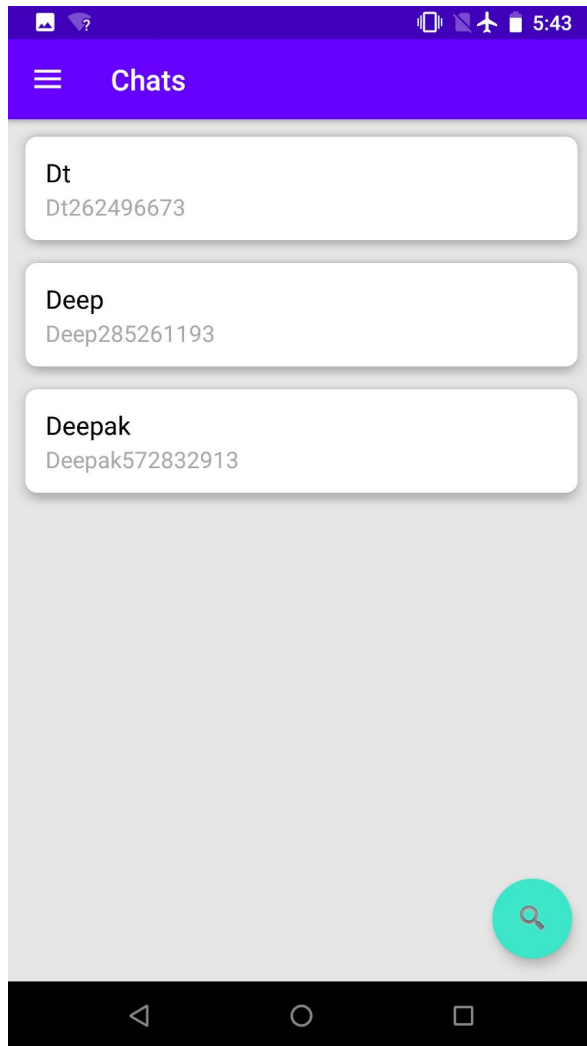


Home Activity

Home Activity is the place where most actions are initiated. It inflates the `activity_home.xml` layout, includes a Navigation drawer, a floating action button to initiate User Discovery Activity and a RecyclerView that lists all the past conversations. The RecyclerView populates items as Card Views and each card view opens Old Message Activity, which populates old chats as view only.

On creation, Home Activity creates some objects that are crucial to the functioning of the app. First, it checks if Wi-Fi is enabled, if not, it will turn on the Wi-Fi. Then it creates a local service specific to the "I'm here" application and registers it on Wi-Fi direct so that other users can discover this device. Finally, it queries the database to search for all older chats and displays them in the recycler view.

The navigation drawer shows the username and UUID for the user logged in and provides the ability to log out of the app.



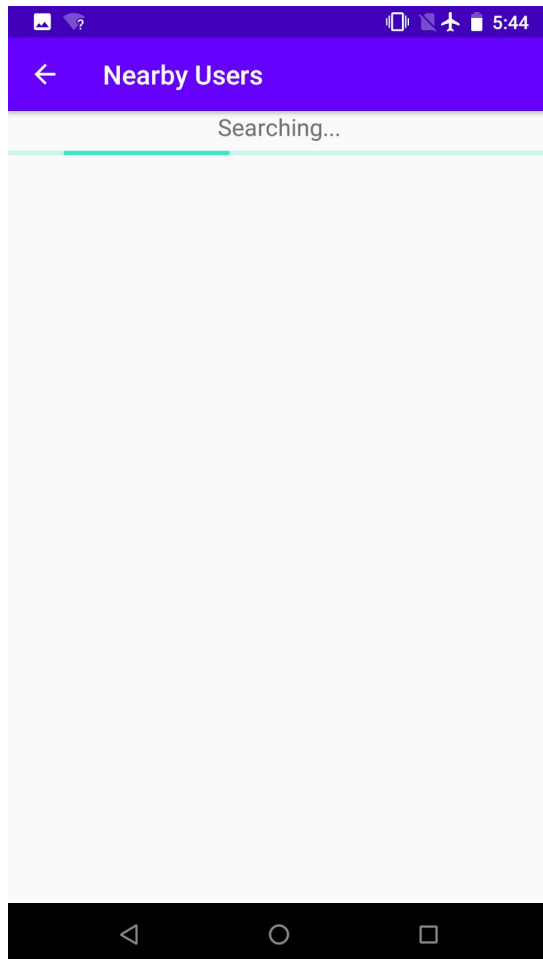
User Discovery Activity

User Discovery Activity is started from the Home Activity on pressing the Floating Action Button. This activity goes through the following process to search for other devices that are advertising this app's service on Wi-Fi Direct:

1. Removes any existing connections of Wi-Fi Direct.
2. Initiates `onConnectionInfoAvailableListener` to listen for incoming connections from other devices.
3. Initiates discovery of Wi-Fi Direct services with names `_imhere`.
4. Collects the Username, Device Name, Device Address and UUID of the services and puts them in Hashmaps.
5. Populates Device names and UUIDs as items in the ListView.
6. Sets `onClickItemListeners` on all the items in the list to initiate connection and open `ChatterActivity`

If the user clicks on an item, it sends an invite to connect with the selected device and opens `ChatterActivity` to finish connecting.

If the device receives an invitation to connect, a pop up will prompt the user to accept the invitation. As soon as the connection is formed, `Chatter Activity` is started and users can start chatting with each other.

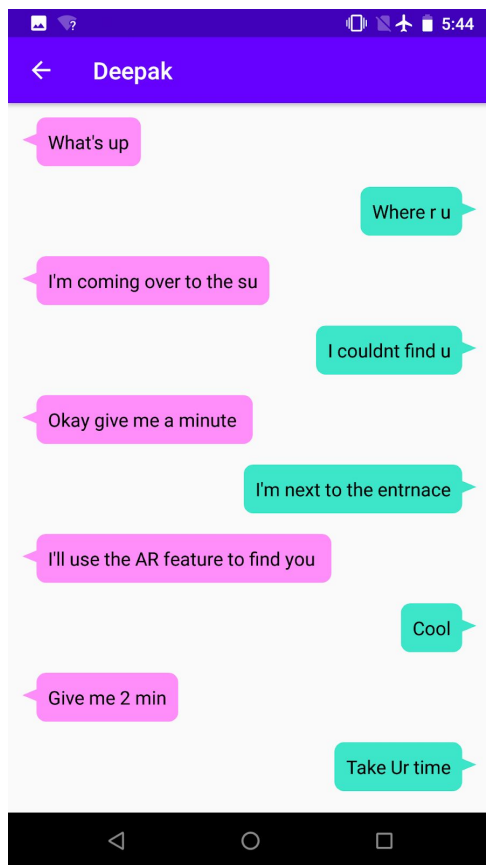


Chatter Activity

This is the main chatting activity. This activity is opened when a successful connection is established over Wi-Fi Direct. This activity has a layout similar to many chat apps, with a chat entry box, a send button, and a `ListView` for showing the chats. The chats are displayed in bubbles of different colors to differentiate between sent and received messages. It also includes a button for AR which opens the AR activity.

When the users send their first messages, they also share their Usernames and UUIDs, which are populated in the `ActionBar` and stored in the database. Every message sent also includes the location of the sender. This location is what is used by the AR activity to point in the correct

direction of the other user. All the messages sent and received are stored in the database under the UUID of the friend as the key. This allows for OldMessageActivity (image below) to populate and display these messages later as view only.

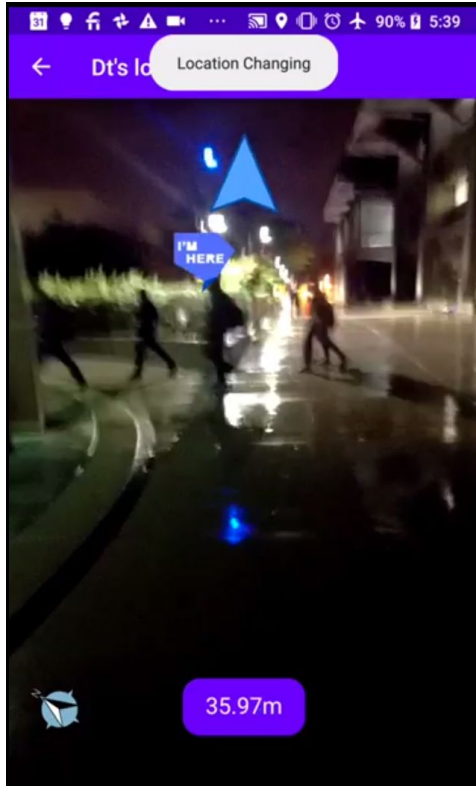


Clicking the AR Activity asks for the permissions to access location and camera of the device and passes the location of the friend as parameters of the intent.

AR Activity

AR Activity uses the camera, GPS location and compass to show the relative location of the friend from the user on the screen. Using the user's own GPS, friend's GPS and the user's own compass readings, distance between the two GPS points and the relative direction from one point to the other can be calculated. This is done using the Haversine formula.

The layout of AR Activity includes a full-screen camera SurfaceView, an arrow to point towards the user, a pin (which is our logo), a bar with the distance to the friend and a compass that points towards north. These UI elements help the user to move towards the friend and find them. The distance, arrows and compass are animated and move based on the location and compass direction of the user.



Other Classes and Components

WiFiDirectBroadcastReceiver

This class is responsible for receiving information on Wi-Fi Direct related events from the OS. In Android, Wi-Fi direct events are sent as broadcasts by the OS. This class receives those broadcasts and calls the appropriate callback functions to initiate some actions. The constructor of this class takes a `WifiP2pManager`, `WifiP2pManager.Channel` and an `Activity` to establish context.

WifiP2pService

This is a class that contains all the necessary attributes needed to recognize a Wi-Fi Direct service. This is a serializable object that is used to communicate service information between User Discovery Activity and Chatter Activity.

CustomAdapter

This adapter is for `ListView` that displays chat messages in Chatter Activity and Old Message Activity.

ChatModel

This class is an data structure to hold the message and its type (sent or received) which is then used by the CustomAdapter to show in the ListView.

Future scope

This application is the first step in creating an application that be posted on the Android Play Store. In the future, we could implement methods such as connection maintenance in the background, message notifications, Wi-Fi Direct file sharing and perhaps also voice and video sharing. We could also include Google's new ARCore APIs to improve the AR functionality.

The app could also be expanded as a very hyper-local social network where you can message with the people around you, share images, chats etc.

Conclusion

"I'm here" android application provides a new unique way for friends to communicate and locate each other in areas where cellular connectivity or Wi-Fi may be limited or unavailable. It uses Wi-Fi direct as the protocol for sending messages and location between devices. It utilizes the hardware ability of the smartphones to create a network for communication and message sending and does not rely on external networks or availability of offline maps.

Through this project, we are able to create an application that is not only novel but also extremely useful. We are grateful to Prof. Hungwen Li and CMPE 277 for providing us with this opportunity to work on this project and creating something novel.

References

- Java | Couchbase Doc. (n.d.). Retrieved November 29, 2018, from <https://docs.couchbase.com/couchbase-lite/2.1/java.html>
- <https://developers.google.com/maps/documentation/android-sdk/map>
- <https://www.blog.google/products/arcore/arcore-augmented-reality-android-scale/>
- <https://stuff.mit.edu/afs/sipb/project/android/docs/guide/topics/connectivity/wifip2p.html>
- <https://developer.android.com/guide/topics/connectivity/wifip2p>
- <https://dzone.com/articles/couchbase-lite-with-react-native-on-android>
- <https://android.googlesource.com/platform/development.git/+master/samples/WiFiDirect/ServiceDiscovery/>
- <https://android-developers.googleblog.com/2017/08/arcore-augmented-reality-at-android.html>