AOSSIE

Google Summer of Code 2021 Proposal Idea (P2P messaging)

PROJECT SUMMARY :

The main objective of the project is to create a messaging app that provides peer to peer connection without the internet or without any backend support. It makes connections more secure and private. This can be done by using device wifi, Bluetooth as a source of connection. There are some drawbacks as well as like :

1. Users cannot connect to other users who are outside the range of Bluetooth and WiFi. Therefore, the connection will depend on the distance.

2. Transfer of data will be slow.

3. Data will not be centralized in one place. Because the app will not use an internet connection. So data will be stored in the local storage of the device.

4. Users will lose all the data whenever they change their device or lose their local files.

ABOUT :

I am a third-year undergraduate student pursuing B.Tech from the **Indian Institute of Technology, BHU.**

I have been doing web Development & App Development since my first Year. I have had experience in working with large codebases as well as making something from scratch through previous internships. Along with that, I have a stronghold on Data Structures and Algorithms.

I believe I am the right choice for this project and will work on this full time over the summer. I will give weekly updates about my progress and ensure I deliver according to the timeline set. I started my open source contribution 3 months back from the same Organization and continuously try to solve issues, as well as find bugs on different projects.

Some of my other relevant *achievements/ experience* are:

- Worked as an **App development intern** at Jashara Private Limited :

  During this internship, my work is to understand the whole source code, find bugs, their solutions, Writing logic part, refactor codes, improving UI and helping the team to deploy the application successfully on the Playstore. Link: Tax Whizzer.

- Members of **Mindby (college Open Source organization group),** there we developed the Android native app, which spread awareness and learning about menstrual cycle among tribal people and girls through this app having videos, pdf, and giving tests according to module read. Link: Mindby

- Already implemented a peer to peer messaging app through Bluetooth. Link: Demo.

- Worked as a **Backend Developer Intern** at Billbolo Private Limited (January 2021 to current) :

  During this internship, my job is to create a backend from scratch for the merchant/customer android app in which I had created authentication APIs, payment-related APIs, merchant & user banks related APIs as well as store this information into MongoDB database and successfully deploying these APIs on the AWS EC2 instance.

- Successfully cleared Uber Hashtag First Round.

- Institute Rank under 30 on Geeks for Geeks (Data structure and Algorithm platform). Link: Link.

BASIC INFORMATION :

Contact Information:

- Name: Avinash Kumar.
- Email:avinashkumar.met18@itbhu.ac.in
- IRC/Gitter/Github: Kumaravinash9
- Phone number: +91 6204840373
- Country/ Region:Varanasi, India
- University : Indian Institute of Technology, Varanasi .

Skills:

- Experienced in Native Android.
- Experienced in Flutter.
- Strong command on Backend (FrameWork : NodeJs (express)) .
- Strong command on HTML and CSS
- Proficient in Data Structures and Algorithms.
- Fluent in C++ .
- Databases : Firebase , SQL , Sqlite , MongoDb.

Version Control

- Strong command of Git. I used Github and Gitlab a lot.

PRE-GSOC INVOLVEMENTS ON THIS ORGANIZATION :

Some of my contributions are:

# AOSSIE Monumento : **(Cross-platform app that provides new learning opportunities for popular monuments through AR(augmented reality) & Cloud Vision API).**

Merge Requests:

- **1.1** #54 :(**merged**) Feature Request:Added Google Maps screen, which takes the user to the location of the monument with the appropriate animation when tapping the location icon in the details screen using the **google_map** package. Fix issue: #58.

  1.2 . Fix bug: Each popular monument was being saved in the bookmark collection as many times as you tap the bookmark icon in the details Screen. I fixed that issue and also implemented a logic code to toggle (add /remove) the bookmark items from its collections. Fix issue : #46.

- #55 :(**merged**) Feature Request: Added visibility icon and its logic code in the password textField that toggles the visibility of the password in both the signup screen and the login screen. Fix issue: #60.

- #44 :(**merged**) Feature Request: WebView takes some time to load the Url. so, I added CircularProgressIndicator Until the Url gets reloaded. Fix issue: #52.

- #59:**(merged)** Fix bug : Fixed pixel overflow error in the HomeScreen by Using **FittedBox Widget** . Fix issue : #61.

- #60 :**(merged)** Fix bug: Fixed pixel overflow error in the details screen that was appearing in the original container of the Webview. Fix issue : #62.

- #62 :(**mergeable**) Feature Request: Implemented a profile change screen to remove and add a profile picture using **ImagePicker** (to get the user's image through the camera, gallery) and **Firebase_Storage.** (storing the user's image, creating its host URL and Store the URL in the Firebase user collection). Fix issue: #50.

- #46 :(**mergeable**) Feature Request: Previously forget password logic was not defined in the source code. So, I implemented a forget password logic code along with designing and creating a forget password Screen. Fix issue : #44.

- #47 :(**mergeable**) Feature Request: Added animation to the carousel of popular monuments providing a better user experience (UI) using the **carousel_sider** package.

- #58 :(**mergeable**)Feature Request : Added merge_request template and issue template.

- #64 :(**mergeable**) Fix: The previously plain password was being stored in the firestore user collection. To make it more private, First I hashed the plain password then stored it in the Firestore User collection by using the decrypt package (creating a hashed password with a salt). Fix issue : #68.

- #48 :(**mergeable**) Feature Request: Fix: Previously **remember me** checkbox logic was not defined in the source code in the login screen. So I implemented the logic code for the **remember me** by using the **Shared Preference package.** Fix issue : #45.

Issues created :

(Note all the issues which are mentioned above in MR requests were also created by me.)

- #55 : **(Issue)**: In the Login Screen & Sign Up screen, Form validation is to be there for the validation of the textField like email, name and password.

- #59 : **(Issue)**: App is not specifying the proper reason why login fails. for example: "User already exists ", "Password is wrong".

- #66 : **(Issue):** Users are not able to edit their profile credentials like name, about, status and profile picture.

- #73 : **(Issue):** Push notification feature is not defined in the source code.

# **ASSOIE SCHOLAR PROJECT** :(chrome extension that calculates proper metrics of Scholars)

# MERGE REQUEST :

- #104 :(**merged**): Added loading spinner in the popup.html until the web scraping is completed. changes I committed in popup.html,style.css and popup.js. Fix issue : #52.

- #97 : (**merged**) : Assoie Scholar extension is not working with url having **.co.in** domain or **.ca** domain . so i fixed it.Fix issue : #45.

- #96 :(**merged):** Added loading spinner for the profile, starred and Search tab until the page completely reloaded. Fix issue : #43.
- #103 : **(merged):** On the starred page, the "Google Scholar Page" is not completely inside its respective container. So, I fixed this by doing some changes in style.css. Fix issue : #51.

- #101 : (**merged**): On the scholar profile page, "Profile pic of the scholar is not completely inside its tag". So, I made some changes to my profile.css file and fixed it. Fix issue : #49.

- #100 :(**merged):** Added document title for all the tabs.

- #99 : (**merged**): Added description tag in manifest.json file which provides a brief description to the user about the project's goal in Chrome extension tab. Fix issue : #47

ISSUE CREATED :

- (Note all the issues mentioned above in each MR was also created by me.)

- #54 : (**Issue**): UI improvement is needed on the starred page.

- #50 : (**Issue**): showing an error that module is not defined in the profile .js which is showing in the chrome extension debugging.

TODO LIST :

Deciding FrameWork for the implementation of App :

I have decided to use **Native Android** for the production of this app. Some keys points which support why native android:-

- Native android provides a Wifi Direct P2P and Bluetooth APIs feature. Using this feature we can easily implement an app that does not require any internet connection for texting and sharing files. Using the Wifi Direct feature, a group chat can also be possible where two or more are able to talk.

- In future, If I will face any difficulty/issue in implementing the app. I can easily get to the solution of the issue over the internet because it has strong community support as well as popularity.

I made a demo Bluetooth chat application in android native using Android Bluetooth API. Link: Demo

Selecting Database :

Because we are not using internet connection. So, we need to choose a database that can store data in the local device's storage. There are too many databases available which fulfil our requirements. But I will go for SQLite because It has some extra advantages.

The advantages :

- Reading and writing operations are very fast for SQLite databases. It is almost 35% faster than the File system.
- It updates contents continuously, so little or no work is lost in a case of power failure or crash.

Because of its better performance and reliability, I will choose SQLite as a database for this app.

DETAILED DESCRIPTION :

WIFI DIRECT (P2P) :

(supported in all devices which have wifi support)

Wifi Direct allows devices to connect directly to each other via Wi-Fi without an intermediate access point. Wifi Direct provides a **Wifi2P2**

Manager class that provides a lot of pre-definitions functions which helps me to discover, request, and connect to peers as well as providing a facility for transferring files as well as messages. Wifi Direct sends files & messages through **ServerSocket**. Whenever we connect the devices through Bluetooth and Wifi Direct, a server is created by a device that is sending files. And another connected device listens to that server and receives data. This is how wifi-Direct and Bluetooth communicate with other devices. Most popular apps like Share it, Xender, gaming apps are using the same feature for sending files and communicating.

For creating a messaging app, we need to find some solution to how a sender device also works as a receiver device at the same instance. For that, there is a popular concept called " **Multithreading**" which facilitates the possibility that our app can do multiple tasks at the same time. Using this concept, our app listens or sends data with the connected device at the same instance.

Some Important Functions/Code which will help me to create Wifi Direct functionality :

- **Checking the wifi Direct options enable/supported in the device or not :**

Wifi2P2Manager class provides a function that checks if the device is on or support the Wifi Direct.

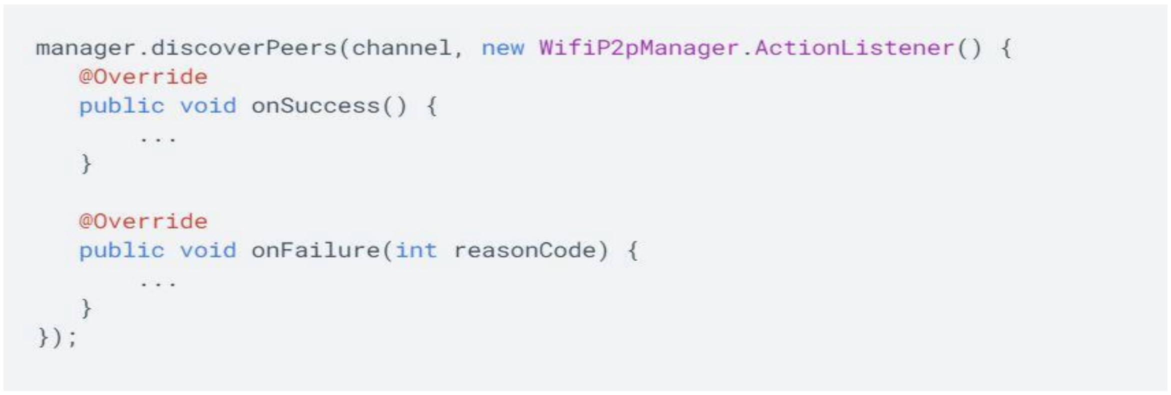
```
    val state = intent.getIntExtra(WifiP2pManager.EXTRA_WIFI_STATE, -1)
    when (state) {
        WifiP2pManager.WIFI_P2P_STATE_ENABLED -> {
            // Wifi P2P is enabled
        }
        else -> {
            // Wi-Fi P2P is not enabled
        }
    }
```

- After checking the compatibility of the device, I will initialize the app by WifiDirectBroadcastReceiver (which allows me to register applications for Wifi Direct events and also handle different intents) . so that app will be able to fetch all the nearby available devices.

```
WifiP2pManager manager;
Channel channel;
BroadcastReceiver receiver;
...
@Override
protected void onCreate(Bundle savedInstanceState){
    ...
    manager = (WifiP2pManager) getSystemService(Context.WIFI_P2P_SERVICE);
    channel = manager.initialize(this, getMainLooper(), null);
    receiver = new WiFiDirectBroadcastReceiver(manager, channel, this);
    ...
}
```

- **Finding all the peers devices :**

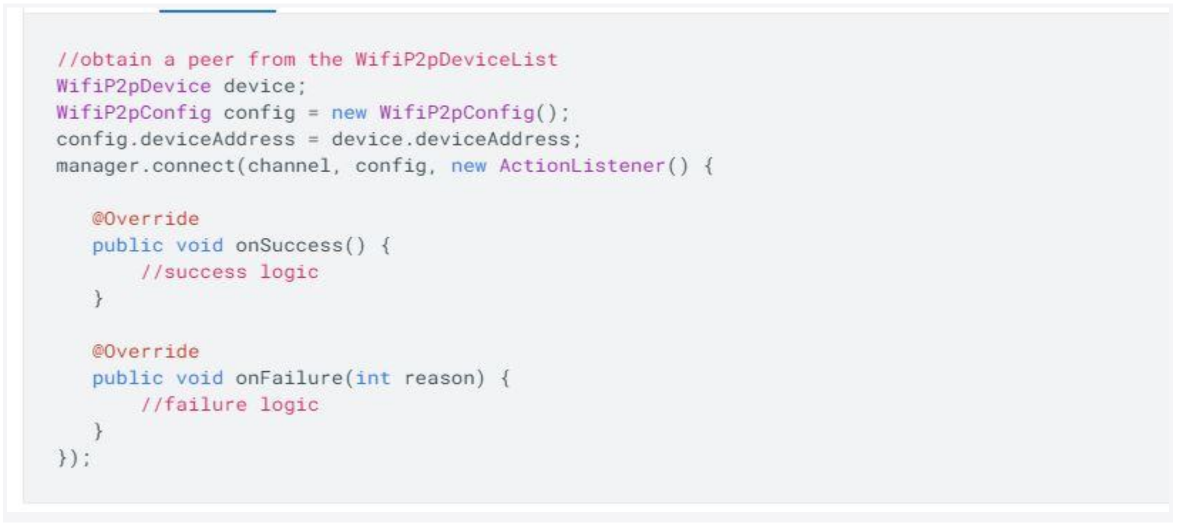
WifiP2pManager class which provides discoverPeers() function that will provide asynchronously a list of all devices available nearby. After fetching all the devices, I will render it on the screen.



```
manager.discoverPeers(channel, new WifiP2pManager.ActionListener() {
    @Override
    public void onSuccess() {
        ...
    }

    @Override
    public void onFailure(int reasonCode) {
        ...
    }
});
```

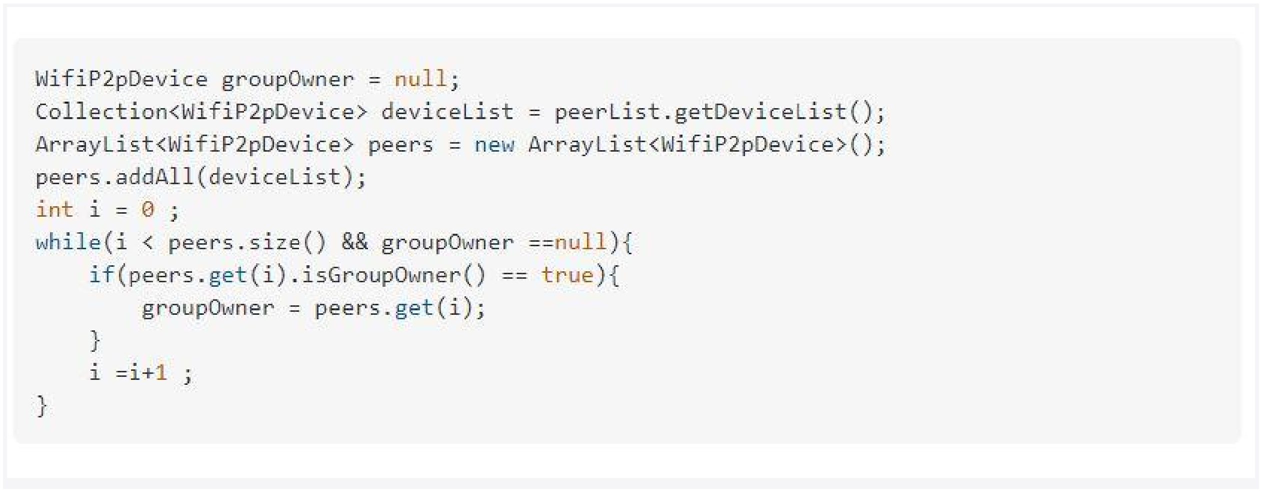
- **Connecting Peers :**

  After Discovering all the peers/devices, I will easily connect the devices using the connect function. Below, I added some code snippets which will help me to create one to one connection and group connections.



```
//obtain a peer from the WifiP2pDeviceList
WifiP2pDevice device;
WifiP2pConfig config = new WifiP2pConfig();
config.deviceAddress = device.deviceAddress;
manager.connect(channel, config, new ActionListener() {

    @Override
    public void onSuccess() {
        //success logic
    }

    @Override
    public void onFailure(int reason) {
        //failure logic
    }
});
```

Code for connecting one to one



```
WifiP2pDevice groupOwner = null;
Collection<WifiP2pDevice> deviceList = peerList.getDeviceList();
ArrayList<WifiP2pDevice> peers = new ArrayList<WifiP2pDevice>();
peers.addAll(deviceList);
int i = 0 ;
while(i < peers.size() && groupOwner ==null){
    if(peers.get(i).isGroupOwner() == true){
        groupOwner = peers.get(i);
    }
    i =i+1 ;
}
```

Code for connecting peers for a group chat

- **Creating server socket :**

Wifi Direct P2P shares the files, text and images by using a **server socket** as well as the concept of **MultiThreading**.
This is the most crucial part of our project. So, I wrote the entire codebase in which I implemented a wifi-direct demo project to send messages.

I have created a GitHub repository, in which I wrote code for the implementation of creating a server between two peers through wifi direct. **Git hub Link**

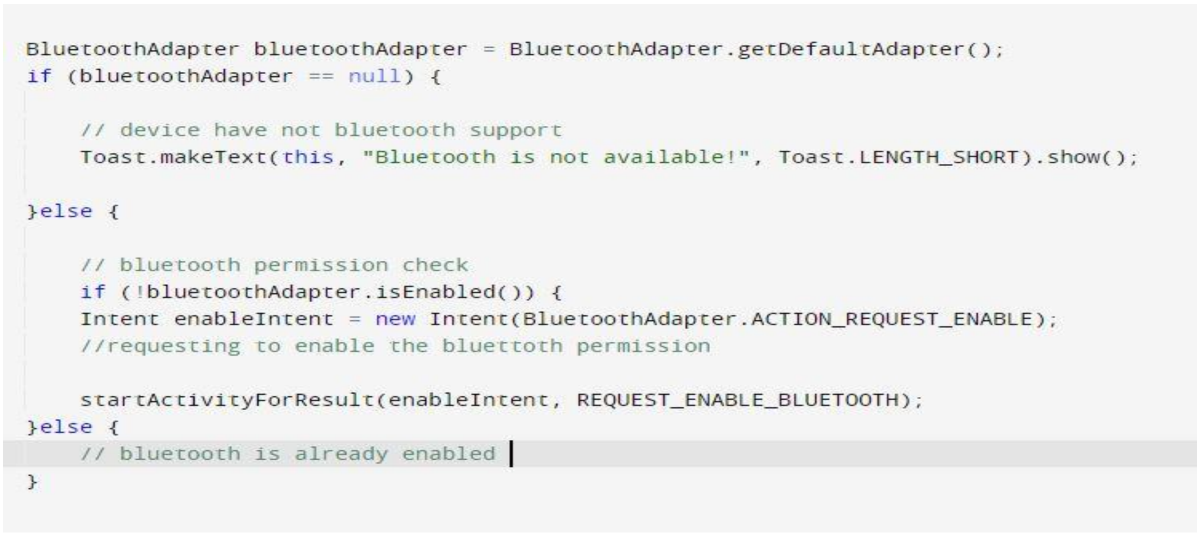
ANDROID BLUETOOTH   :

The Android platform includes support for the Bluetooth network stack, which allows one device to exchange data wirelessly with other Bluetooth devices. The Android framework provides access to Bluetooth functionality through the Android Bluetooth API. These APIs enable wirelessly connecting to other Bluetooth devices, enabling point-to-point and wireless features so that we can transfer data to other devices in a network circle. So, I will use these Bluetooth Apis for creating a peer to peer connection.

SOME IMPORTANT FUNCTION/CODES (That will help me to implement the peer to peer connection through Bluetooth) :

- **Checking Bluetooth compatibility and their state(ON or off):**

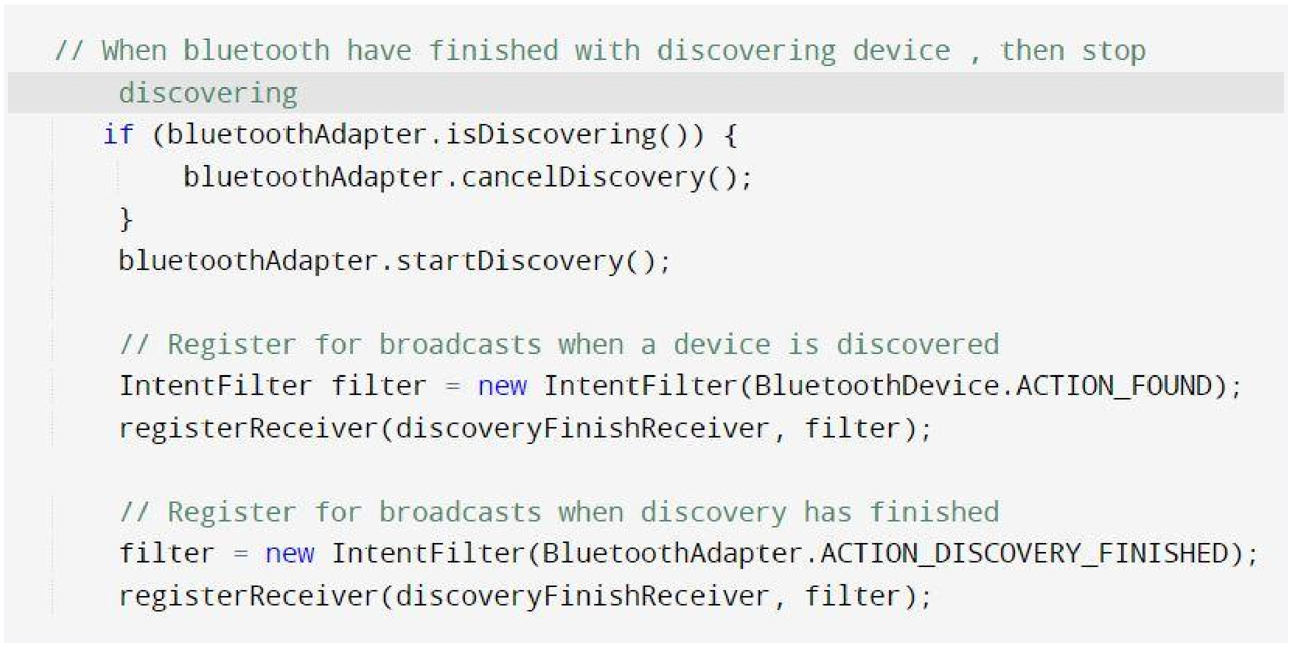
Whenever a user will want to search peers through Bluetooth, then I will trigger the function to check the compatibility and availability of their device. If compatibility and availability will not present, I will show them an info message through a toast or show dialogue box widget.



```
BluetoothAdapter bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
if (bluetoothAdapter == null) {

    // device have not bluetooth support
    Toast.makeText(this, "Bluetooth is not available!", Toast.LENGTH_SHORT).show();

}else {

    // bluetooth permission check
    if (!bluetoothAdapter.isEnabled()) {
    Intent enableIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
    //requesting to enable the bluettoth permission

    startActivityForResult(enableIntent, REQUEST_ENABLE_BLUETOOTH);
}else {
    // bluetooth is already enabled
}
```

- **Finding all the available devices :**

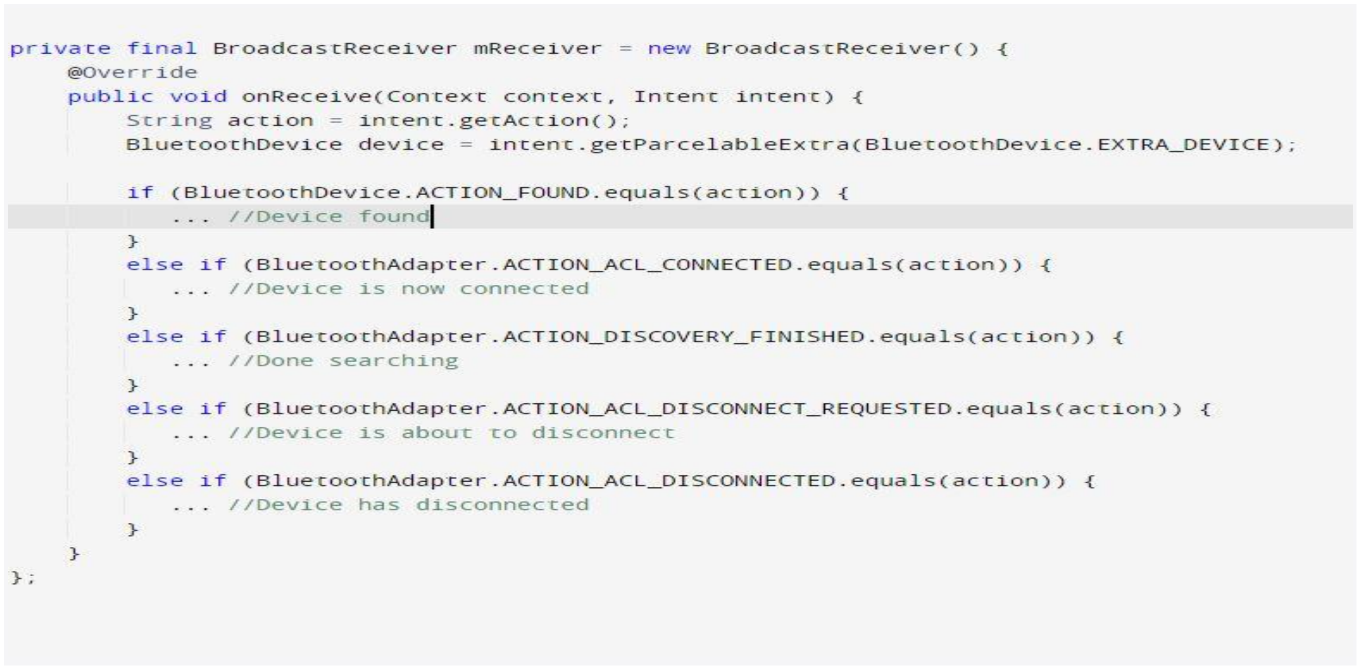
After checking Bluetooth compatibility and availability, I will show all available devices that exist around the user by using ListView Widget. This will be done by calling start discovering function.



```
// When bluetooth have finished with discovering device , then stop
    discovering
   if (bluetoothAdapter.isDiscovering()) {
       bluetoothAdapter.cancelDiscovery();
    }
    bluetoothAdapter.startDiscovery();

    // Register for broadcasts when a device is discovered
    IntentFilter filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
    registerReceiver(discoveryFinishReceiver, filter);

    // Register for broadcasts when discovery has finished
    filter = new IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_FINISHED);
    registerReceiver(discoveryFinishReceiver, filter);
```

- **Handling Events:**

BroadcastReceiver allows the app for handling different Bluetooth events and also handles different **intents** like ACTION_DISCOVERY_FINISHED, DEVICE_FOUND.



```
private final BroadcastReceiver mReceiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);

        if (BluetoothDevice.ACTION_FOUND.equals(action)) {
           ... //Device found
        }
        else if (BluetoothAdapter.ACTION_ACL_CONNECTED.equals(action)) {
           ... //Device is now connected
        }
        else if (BluetoothAdapter.ACTION_DISCOVERY_FINISHED.equals(action)) {
           ... //Done searching
        }
        else if (BluetoothAdapter.ACTION_ACL_DISCONNECT_REQUESTED.equals(action)) {
           ... //Device is about to disconnect
        }
        else if (BluetoothAdapter.ACTION_ACL_DISCONNECTED.equals(action)) {
           ... //Device has disconnected
        }
    }
};
```

- **Creating BluetoothServerSocket** :

Android Bluetooth uses **Bluetoothserversocket** for sharing files, images and text. It works almost like a Wifi direct server**.** So, I will implement it with the same logic as I will implement for Wifi Direct. I made a demo app. Link: Link

DATABASE :

I made a GitHub repository in which I added important query operations that I will use for the saving data of users as well as shared videos, images and chats. Link: SQLiteQuery

UI/UX :

Recyclerview :

This widget is used for displaying those data which frequently change. I will use this to show all the discovered devices.

Load Toast :

This widget works like loading a spinner with proper animation. When the user taps the connect button then I will show this widget until the device finds a nearby device.

Card View :

This widget is used to show data with an elevation. I will use this widget to show previous chats.

Navigation components :

This is used for navigation purposes to different screens from simple button clicks to more complex patterns, such as the app bar and navigation drawer. I will also use this library for navigators purposes.

shimmer Effect :

Shimmer is an android library that provides an easy way to add a shimmer effect to any view in the android app. I will use this effect to provide a better User experience.

I will use all these above libraries and Widget for the implementation of a better User Interface.

TIMELINE :

Duration	Milestone
May 17, 2021 -June 07, 2021.	**Community Bonding Period**
May 17, 2021 -June 07, 2021.	1 . Community bonding 2. Learning more about Wifi Direct , Bluetooth and Sqlite. 3. Discussing with the mentor about the possible features that we can add to the project. 4 . Start implementing Wifi Direct feature
May 17, 2021 -June 07, 2021.	**Coding Period Start**

Week- 1	Start implementing Wifi Direct feature
Week-2	Continue Implementing Wifi Direct feature
Week -3	Working on Screens and polishing their UI.
Week-4	Completing all the Wifi Direct Feature With Complete Screen
Week-5	Start implementing Bluetooth feature
July 12, 2021 -July 16, 2021	**Phase 1 evaluation**
Week-6	Continue Implementing Bluetooth feature

Week-7	Working on bluetooth Screens and polishing their UI.
Week-8	Start connecting database with the app
Week-9	1. Continue connecting databases with the app. 2. learning test
Week 10	Writing tests and polishing UI
Week 11 && Week-12	• Adding more feature suggested by the mentors • Completing pending work if any

August 16, 2021 -August 23, 2021	**Submit Code and Final Evaluations**
August 31, 2021	**Result Announced**

Apart from all these, I would be very much interested in implementing any ideas suggested to me by the mentors in addition to these features.

Why should I be selected for the project?

I have always been interested in open-source projects and have been passionately working on them. I can push myself beyond boundaries. I have a good team management skill and can learn things quickly. I have been developing Android apps for 2 years from now and have contributed to various open-source projects. I believe I have the required skills to finish the proposed goals of the proposal. Moreover, I believe I will follow the best practices in implementing the goals of the proposal. I have even made a working prototype that required the knowledge of app architecture. **Github link:Demo.**

MOTIVATION :

My inspiration for GSoC this year is making myself more familiar with open source Organization. GSoC is a great program to start open-source contribution and when I saw this project, I felt that it was something I could do. I believe this project can definitely help me expand my boundaries, as this is the first time I will be working on an open-source project. This possibility is very exciting for me.

AVAILABILITY :

The official GSoC period is from 17 May to 23 August. I can easily devote 25-35 hours a week till my college reopens and 20-25 hours per week after that I'm free on weekends. I intend to complete most of the work before my college reopens.

Other than this project, I have no commitments/ vacations planned for the summer. Also, I don't plan on doing any internships this summer. I shall keep my status posted to all the community members on a weekly basis and maintain transparency in the project.

AFTER GSOC :

Being a part of such a vast community is a great opportunity in itself and I would love to collaborate with others throughout my project timeline and even after that, as this is the true essence of Open Source culture.I'll be an active member in the community and keep contributing. My motivation would always be that I'd be able to contribute to something big and widely in use . This gives me a lot of satisfaction.

REFERENCES:

https://developer.android.com/guide/topics/connectivity/wifip2p.
https://developer.android.com/guide/topics/connectivity/bluetooth.
SQLite