GSoC 2023 Project Proposal for

AOSSIE - Resonate

Contact Information

Name: *Chandan S Gowda*

Email: chandansgowda167@gmail.com

GitHub: https://github.com/chandansgowda

Skype: chandansgowda1

LinkedIn: https://www.linkedin.com/in/chandan-s-gowda-4b2913183/

Phone: +91 8197796262

Address: #2155, Mahalakshmi Layout, Hinkal, Mysore, Karnataka, IN

Education

University: *JSS Science and Technology University (Formerly SJCE), Mysore, Karnataka, IN*

Degree: *Bachelor of Engineering (BE CSE)*

Expected Graduation Date: May 2024

Courses and Your Performance:

I am pursuing my third year in Computer Science and Engineering and till now I have studied C Programming, Data Structures and Algorithms, Computer Organization, Operating System, Database Management System, Computer Networks, Cryptography and Network Security and some other courses with a CGPA of 9.4 till the end of Fifth Semester.

Commitment

How many hours will you work per week on your GSoC project?

I will be able to work on the GSoC project for 30 hours per week which will sum up to 360 hours for the whole project.

Other Commitments:

I will have my 6th Semester examination from 17th to 30th July. I'm very good at managing my academics as I have been learning things since my childhood. I always study for the exams on the last day and even then, manage to score above 9 GPA. I might go to 15 hours per week during this period but I will manage to compensate for that in the rest of the standard coding period.

Project Summary

Chosen Project: *Resonate*

Chosen Idea: *Clubhouse, but Open Source. With the social voice platforms getting so popular like Clubhouse, Twitter Spaces, etc It's time we have an Open Source version of this. Having an engagement platform like this will create a lot of credibility in the open-source community, drive traction and growth.*

Vision

My vision is to create an open-source, social voice platform that fosters collaboration and community among open-source contributors. By providing a free and accessible space for developers to connect, share ideas, and collaborate on projects, I aim to promote the values of open-source software and drive growth in the open-source community.

The platform will be built with a user-centric design that prioritizes ease of use, accessibility, and inclusivity. It will support a range of features that enable users to connect and engage with each other, including voice chat rooms, messaging and more.

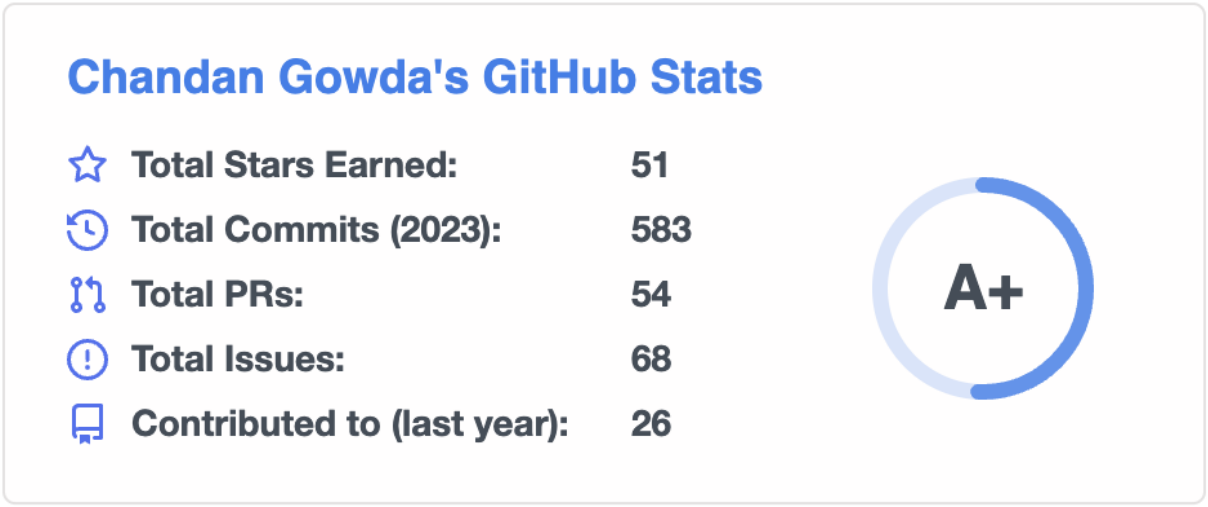
By creating an open-source alternative to proprietary social voice platforms like Clubhouse and Twitter Spaces, I hope to empower developers like me to take ownership of their online communities and build a more diverse, inclusive, and sustainable open-source ecosystem. I believe that this platform has the potential to become a leading destination for open-source contributors, providing us with a powerful tool for collaboration, learning, and growth.

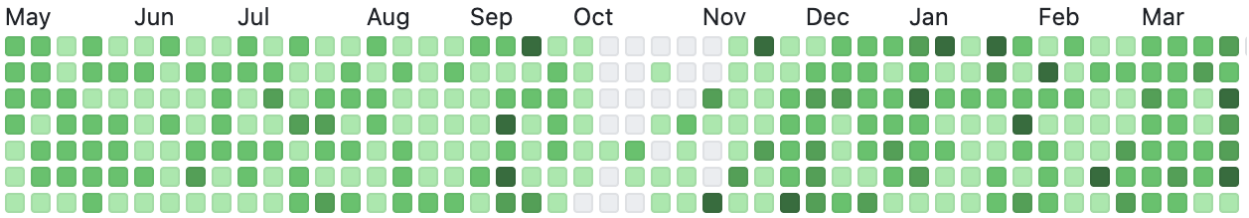
Motivation

*I am a passionate **Flutter App developer** with a strong background in **backend development**. I have developed several full stack mobile applications for my clients as a **freelancer** as well as for **startups as an intern**. I have worked with various state management approaches like **GetX, Provider and BLoC** and also have the experience in **developing and integrating REST APIs** with mobile applications. I have **hands-on experience** with **deploying** servers on cloud using GCP and AWS and I am a skilled **Penetration Tester** which will be helpful to strengthen the security of our application. I am excited about the opportunity to work on this GSoC project because it aligns with my interests and expertise.*

*I have also built a **functional prototype of Resonate** and since I have **designed the architecture** of the project, I have a better idea of the architecture, the flow and the implementation.*

*Moreover, I am **committed to open-source** development and have contributed to several open-source projects in the past. I am familiar with the **Git version control** system and have experience collaborating with other developers using GitHub. I have won a few hackathons including **JP Morgan & Chase Code for Good** and also been a technical mentor in many hackathons.*

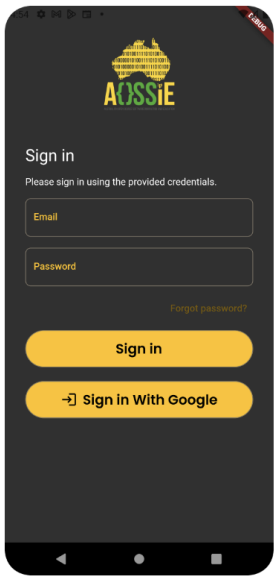
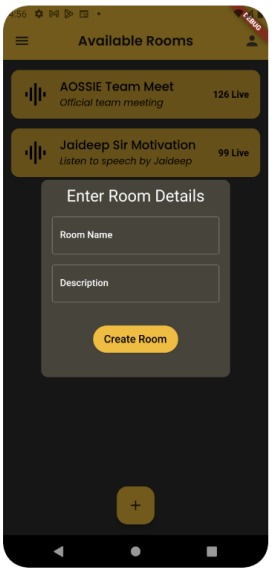
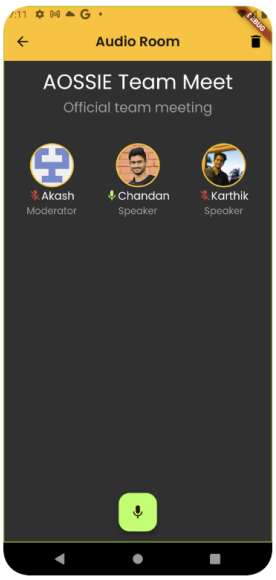




In summary, I believe that my experience in software development, my passion for the project, and my commitment to open-source development makes me the best person to execute this proposal. I am excited about the opportunity to work on this project and contribute to the community.

Past Experience in Software Development

- **AudioRoom:**
    - A Flutter app powered with Firebase and LiveKit WebRTC Stack.
    - This is a fully functional **prototype of Resonate**. I have implemented CreateRoom, JoinRoom, DeleteRoom and LeaveRoom functionalities.
    - Links: Source Code

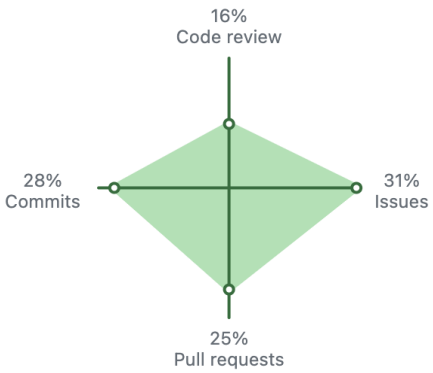
SignIn Screen	Rooms Screen	Create Room Popup	Single Room Screen
			

- **College Attendance/Evaluation Manager:**
    - A Flutter app powered with Django Backend to manage attendance and examinations in our college which will be used by over 5000+ students and staff.
    - I was responsible for the overall system design and API development.
    - Links: Source Code
- **Billmi POS App** :
    - A Point-Of-Sales Windows and Android application built using Flutter and Firebase for Xiaomi OdeToCode Hackathon (Pre-Finalists).
    - Razorpay API was used to handle payments and Sendinblue API was used for sending mails.
    - Links: Source Code , Demo Video

- **Bunkwell Android App**:
    - An offline mobile app developed using Flutter to manage school/college attendance. Provider is used for state management and Hive database is used to store data.
    - It has over 3500+ downloads on Google Play Store with a rating of 4.8
    - Links: Play Store Listing

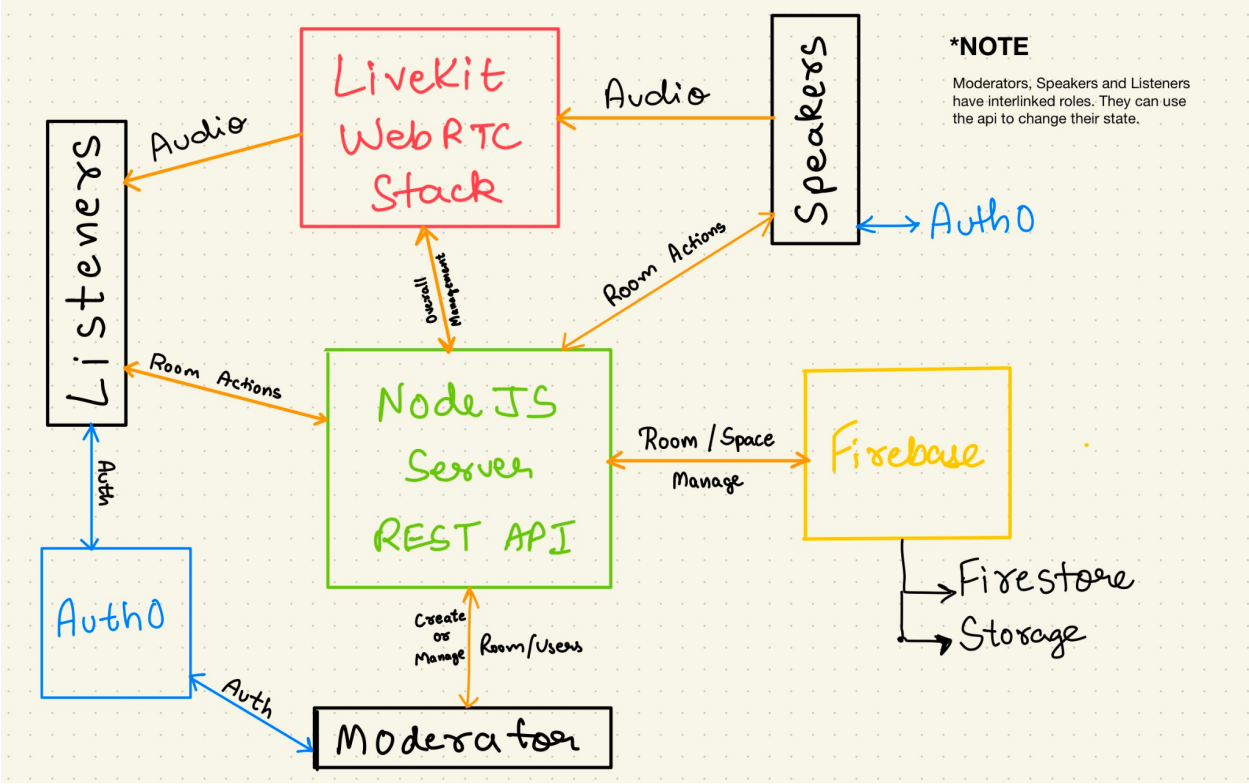
Contributions

- **Pull Request #**20: Added architecture documentation : *Designed the architecture of the whole system and got it finalized by the mentor and created a readme file with the finalized architecture.*
- **Pull Request #**25: Implemented Google Sign In Feature + MVC Structure : *Implemented Google Sign In without using Webview/Firebase. Also setup a basic MVC file structure with example files.*
- **Pull Request #**2: Initialized project with basic codebase : *Initial NodeJS and Express codebase was set up for the backend.*
- **Pull Request #**12: REFACTOR: disabled desktop and web support : *Since our goal as of now is just to develop an app for Android and iOS, we can disable web and desktop support. This makes our projects a little cleaner and lighter.*
- *Also, raised 10+ issues, 3+ PRs, 3+ Code Reviews including other projects under AOSSIE.*
- *The contribution graph for AOSSIE is attached beside.*



Detailed Proposal Description

High Level Architecture



LiveKit *- It is an open source WebRTC stack that gives everything needed to build scalable and real-time video, audio, and data experiences in our application. This will be responsible for audio streaming. It will be easier to implement and deploy it since they have great documentation too. To get started, we can use their free cloud service and then we can host the open-source version on our own server.*

Firebase *- We can use it for storing user data, managing rooms/spaces (Firestore Database).*

Auth0 & GCP *- Will be used for authentication of users.*

NodeJS + Express *-Will be used to develop a web server which will interact with the mobile app, firebase as well as LiveKit. The flutter app will interact with the server using REST API endpoints.*

Flutter *- Will be used to develop the application for various platforms. Listeners, moderators and speakers are devices having the flutter application.*

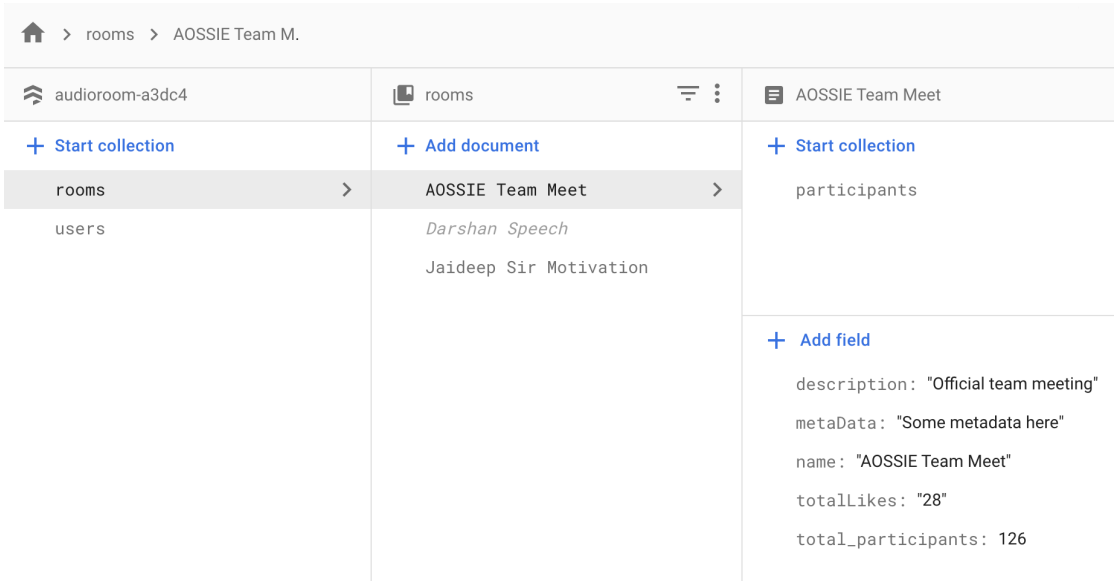
Flow / Implementation

1. ***Authentication***
    - *User chooses between Auth0 Email or Google for authentication.*
    - *If the user is registering, create a user doc inside Firebase Firestore to store user data.*
    - *Create a common app level User Model and store it in shared prefs.*
    - *Navigate to the next screen.*
2. ***Create New Room***
    - *User clicks on the create room button in the app.*
    - *Get room details like name, description, metadata etc. from the user.*
    - *Post it to NodeJS CreateRoom Endpoint*
        1. *Generate and sign the JWT token associated with user's email and room_name with room_admin set to true*
        2. *Post data to LiveKit's CreateRoom Endpoint using the token.*
        3. *Create room on Firebase Firestore*
        4. *Forward the response to the app along with token*
    - *Store the token in shared preferences and use LiveKit Flutter SDK to connect to the room.*
3. ***Join Room Flow***
    - *User clicks on the join room button corresponding to the required room.*
    - *Hit the NodeJS JoinRoom API endpoint*
        1. *Generate and sign the JWT token associated with the user's email and room_name with room_admin and publish_data_to_stream set to false.*
        2. *Add the user to participants list on firestore and then return a successful response along with the generated token.*
    - *User can now use the token to connect to the room using the Livekit Flutter SDK.*

4. ***Delete Room Flow***
    - *Admin will have a delete room button.*
    - *Hit the NodeJS DeleteRoom API endpoint*
        1. *Check if the user is an admin using the token and the isAdmin variable value on Firebase Firestore.*
        2. *Send a request to the DeleteRoom endpoint on LiveKit Server.*
        3. *Delete room on firestore and save it to firebase realtime database if the room history has to be recorded.*
    - *App will be listening to the onRoomUpdate trigger and hence users will be automatically logged out from the room.*
5. ***When the last participant leaves***
    - *LiveKit handles the job of automatically deleting the audio room.*
    - *It then sends a signal to the NodeJS server using Webhooks.*
    - *The server will then delete the room on Firebase Firestore and the room will be stored on Firebase's realtime database to record the history.*

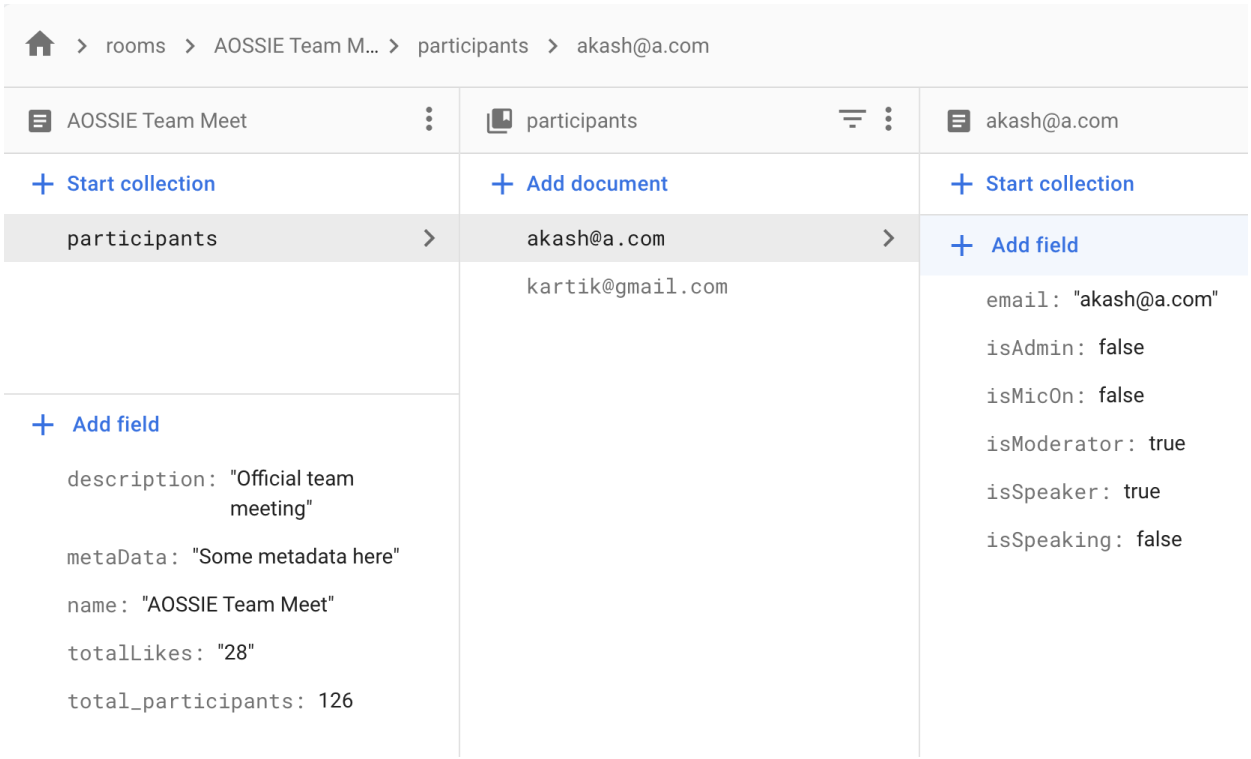
Firebase Firestore Structure

1. Room



Here Room Document has been assigned with the room name as the ID but this will be auto generated.

2. Participant



App Level Design Pattern

Flutter will be used to develop the application for various platforms. We will be designing the app using MVC (Model View Controller) architecture and we will be using GetX for state management. GetX offers several features, including reactive programming, dependency injection, and routing, which can help us build scalable and maintainable applications.

Why MVC Architecture ?

- *Separation of concerns*
- *Reusability*
- *Testability*
- *Flexibility*
- *Scalability*
- *Ease of maintenance*

Why GetX ?

GetX is a very lightweight and powerful state management solution for flutter and hence will be easy for us to get started with it. Features include:

- *High Performance*
- *Less Code*
- *No need to worry about context*
- *No unnecessary rebuilds*
- *Code organization is simple*

Timeline

Community Bonding:

- Extensive research about the proposed architecture to come up with any changes to the flow and implementation if required.
- Prepare documentation regarding the finalized architecture and flow and then create a readme file including the environment setup and contributor guidelines.
- Bonding with the mentor and other contributors in the community.
- Read Livekit's documentation and brush up my NodeJS skill.

Week 1:

- Design a minimalistic UI for the app using Figma and finalize it.
- Develop 25% of the User interface (views) using Flutter.
- Update the documentation with the designs and raise issues for developing the same.

Week 2:

- Develop the remaining 75% of the user interface (views).
- Refactor the code to ensure good code quality guidelines

Week 3:

- Setup Firebase and Livekit SDK on NodeJS and Flutter.
- Refactor NodeJS codebase to match the project requirements.
- Develop API endpoints to create and delete rooms on Livekit and Firebase.

Week 4:

- Develop API endpoints to join and leave rooms.
- Develop an APIService class on Flutter with methods to create, join, leave and delete rooms.

Phase 1 Evaluation

Week 5:

- Design models for Room, Participant and responses received from the API
- Develop a RoomService class and a LiveKitServiceController to manage room connections.

Week 6:

- Interlink the models, views, services and controllers.
- Make final changes and improvements to the User Interface.
- First working version of the application will be ready by the end of this week.

Week 7:

- Setup NodeJS to talk to Auth0 and GCP to verify user authentication and authorization.
- Setup authorization on NodeJS server so that only authenticated users can use the endpoints.
- Modify the Flutter's APIService class accordingly

Week 8:

- Add a feature to allow listeners to raise hands to join the conversation so that the admin can grant them access to become a speaker.

- Develop NodeJS RequestSpeakerAccess API Endpoint.
- Modify flutter app's views, controllers and the models accordingly.

Phase 2 Evaluation

Week 9:

- Add a feature to group rooms based on organization and/or project. A user can follow the Organization/Rooms to see particular rooms only on his/her feed.
- Design and develop the User Interface to accommodate this feature.
- Develop NodeJS FollowOrg, FollowProject, GetFollowed API endpoints.

Week 10:

- Develop GetOrgDetails and GetProjectDetails API endpoints.
- Modify the flutter's models, services, views and controllers to accommodate the grouping feature developed in the previous week.
- Implement a feature to share room links so that a new user can directly access the room on the app.

Week 11:

- Add a feature for the Admins to make moderators who will have the ability to manage and moderate a room.
- Intensive testing of the whole system to find bugs, security loopholes and performance issues.
- Resolve all security related bugs (like setting up Firebase Access Rules) etc.

Week 12:

- Resolve all bugs and performance issues and make improvements to the user interface.
- Deploy LiveKit Server to a self hosted cloud infrastructure.
- Deploy our app to Google Play Store and iOS app store.
- Document the journey of 12 weeks including all the contributions made to be submitted for final evaluation.

Final Evaluation

Future Scope

I would like to work on the project even after the GSoC season and plan to implement the following:

- Replacing Livekit with our own WebRTC implementation
- Replacing Auth0 with our own implementation for Authentication
- Replacing Firebase with our own implementation using MongoDB
- Building an effective CI/CD Pipeline
- Support for Web and Desktop

Any Other Relevant Information

*In addition to my consistent contributions to open source projects, I am also **passionate about teaching** Programming and Development related topics in Kannada, which is my mother tongue. I have found that teaching these topics in my native language not only helps me solidify my own understanding but also enables me to share my knowledge with a wider audience in a way that is accessible and inclusive.*

*I have contributed to the **Kannada-speaking community** by creating and sharing tutorials, blog posts, and videos in Kannada on various programming and development-related topics on my youtube channel - Engineering in Kannada which now has over **30k+ Subscribers**. These resources have been well-received by the community and have helped many individuals gain a better understanding of technical concepts.*

*By contributing to open source projects, I hope to not only improve my own technical skills but also share my knowledge and expertise with others and make a **positive impact by inspiring** the Kannada speaking community to contribute to Open Source.*