Multi-Tenant Chat application with email and Forum inbuilt

B. Tech Project Report

by

Meet Mehta – B16CS016

Manvendra Singh Kushwah – B16CS015

Under the supervision of

Dr. Sumit Kalra

Submitted in partial fulfilment of the requirements of the degree of Bachelor of Technology

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY JODHPUR

Acknowledgement

Firstly, we would like to show our gratitude to our respected professor Dr. Sumit Kalra sharing their pearl of wisdom with us during the course of this project.

We would also like to thank our institute Indian Institute of Technology Jodhpur for providing us such opportunity and the platform and resources for performing the experiment successfully.

In the end we would like to thank our friends who helped us in understanding the technology and designing the app in limited time frame.

Thank You.

Abstract

In this project we have made an android app that has features of Chat, Email and Forum. It is not just a simple app, it is an app where a single message and be represented using Chat UI, Email UI or Forum UI. In our app a user can sign up and while signing up he has to set his preference of UI. Thus a user selects Chat UI, Email UI or Forum UI based on his liking. Suppose a user selects Chat UI than it is not necessary that other user with whom he/she needs to communicate should also be on the chat UI. They can carry out conversation on any UI of their choice. We are storing data on Firebase Realtime Database.

Contents

1. Introduction

1.1. Objective

The term "**software multi tenancy**" refers to a software architecture in which a single instance of software runs on a server and serves multiple tenants. A tenant is a group of users who share a common access with specific privileges to the software instance. With a multitenant architecture, a software application is designed to provide every tenant a dedicated share of the instance - including its data, configuration, user management, tenant individual functionality and non-functional properties. Multi tenancy contrasts with multi-instance architectures, where separate software instances operate on behalf of different tenants.

With the growing technology in the market importance of chat, email and forum has been increased. People rely on many of such application for their daily living. There are Millions of user base for many of such popular apps. But there are many cases where user might feel comfortable in using chat application but not email or forum, or many be comfortable in using email or forum and not chat application.

There are no difference between email, chat and forum from developer point of view, neglecting the fact that email must use set of protocols for inter-domain transfer of message, i.e. to transfer message from gmail to yahoo or to any email of other service provider. While chat all chat application or forum application use there own set of protocols to transfer message so inter-domain message transfer is not available, for instance, message sent on whatsapp can't be seen on fb messenger. So even if a person is comfortable using chat application, if he want to read the message sent over email he needs to compulsory use email UI.

To overcome this problem we came up with this project where a single message can be viewed as chat, email or forum based on user preferences.

1.2. Platform

Currently we have developed app for mobile users as mobile users are increasing day by day. We have developed it for android operating system, which is the most popular among others. We aim at expanding this project for other platforms also.

2. Technologies Used

For building the app we have used Android Studio which the widely used IDE among android developers. We have used JAVA Language to built the app because of the reference and support available for JAVA on web. We have used Firebase to get complete backend services.

2.1. Android Studio

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. Because of the below given features Android Studio becomes the most widely used IDE among android developers.

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where developers can develop for all Android devices
- Instant Run to push changes to running app without building a new APK
- Code templates and GitHub integration to help build common app features and import sample code
- Extensive testing tools and frameworks
- Lint tools to catch performance, usability, version compatibility, and other problems

2.2. JAVA

Android app are best built on JAVA/Kotlin. Although C++ is faster than JAVA, it's the support provided from Google that gives JAVA an edge over C++ or any other Languages. We have used JAVA over Kotlin because Kotlin is relatively new language and also documentation of Firebase for JAVA is better than that for Kotlin.

2.3. Firebase

Firebase is a Backend-as-a-Service - BAAS. Firebase frees developers to focus crafting fantastic user experiences. Developers don't need to manage servers. You don't need to write APIs. Firebase is server, API and datastore, all written so generically that developer can modify it to suit most needs. Firebase provides many services like Realtime Database, Cloud Database, User Authentication, ML Kit.

We have used Firebase because of the below given reasons:

- Best documentation. Google has really worked on documentation and made the world class documentation for the support of developers.
- Firebase has great support available over web on sites like stackoverflow, github etc.
- Firebase provides complete backend solution. We needed database and User Authentication for our project.
- Firebase provides listeners, so that user gets realtime updates on there app without refreshing the app. This feature is must for Chat applications. Many database service providers don't have this facility.
- Firebase has APIs for iOS developers and for web developers, thus making it easy to expand the project to other platforms.

2.3.1. User Authentication

Firebase Auth offers multiple methods to authenticate, including email and password, third-party providers like Google or Facebook. Developers can build their own interface, or take advantage of open source, fully customizable UI from Firebase. Firebase Auth integrates directly into Firebase Database, so developer can use it to control access to his/her data. Building custom Authentication system is very difficult. Also to make it secure like that for Firebase Auth is nearly impossible in short time Frame.

Firebase Auth gives unique ID to each app and keeps track of the user, so that user need not to sign in every time he opens the app.

2.3.2. Realtime Database

The Firebase Realtime Database is a cloud-hosted NoSQL database that lets developers store and sync data between their users in realtime. It stores data in JSON format. Realtime syncing makes it easy for users to access their data from any device: web or mobile, and it helps your users collaborate with one another. When users go offline, the Realtime Database SDKs use local cache on their device to serve and store changes. When the device comes online, the local data is automatically synchronized.

Most databases require HTTP calls to get and sync your data. Most databases give data only when you ask for it.

By connecting app to Firebase, normal HTTP connection is not made. Connection si made through Websockets. WebSockets are much, much faster than HTTP. Developers don't have to make individual WebSocket calls, because one socket connection is plenty. All of the data syncs automagically through that single WebSocket as fast as the client's network can carry it.

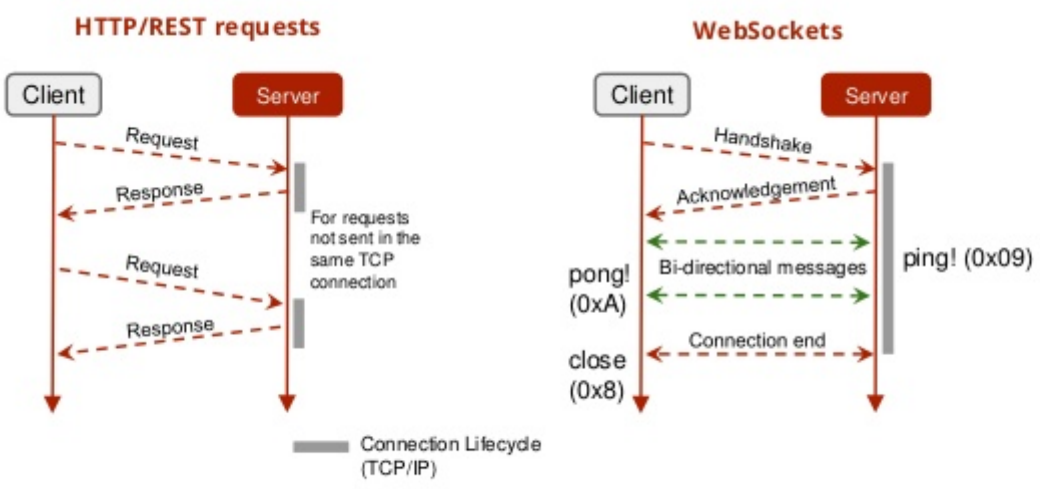


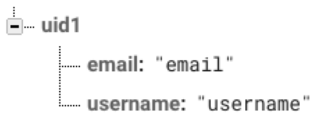
Figure 1 – HTTP Requests vs. WebSockets

Firebase sends new data as soon as it's updated. When the client saves a change to the data, all connected clients receive the updated data almost instantly.

2.3.2.1. Database Structure

Information of Users has to stored in database. Also all the messages are to be stored in Database. Structure of both the data is shown below.

2.3.2.1.1. User's Information

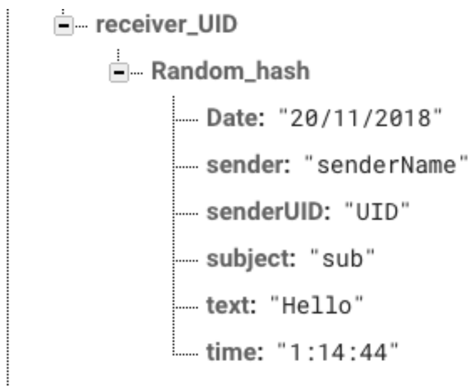


```
uid1
    email: "email"
    username: "username"
```

Figure 2 – User's Information JSON tree

Here UID is the unique ID provided to every user. This UID is provided by firebase itself. Thanks to Firebase Auth, we need not to store password in encrypted form. We have stored username and email ID. Username is stored to provided complete chat and forum experience to the users and email ID is stored to provide mail experience to the users.

2.3.2.1.2. Message Data



```
receiver_UID
    Random_hash
        Date: "20/11/2018"
        sender: "senderName"
        senderUID: "UID"
        subject: "sub"
        text: "Hello"
        time: "1:14:44"
```

Figure 3 – Message Data JSON tree

- receiver_UID is the unique UID of the receiver, again this UID is provided by the firebase.
- Random_hash is the unique key provided by the firebase. It is randomly provided. We need this key because in firebase data is stored in key-value format.
- Remaining fields are of message class, like Date and Time of the message sent. This will be used to sort the messages.
- Sender key corresponds to the username of the sender.
- senderUID key corresponds to the UID i.e. unique ID of the sender.
- Subject key corresponds to the subject of the message. This field will be used for email and forum purposes.
- Text key corresponds to the real message being sent by the sender to the receiver.

3. App Functionality and Design

3.1. Login Screen

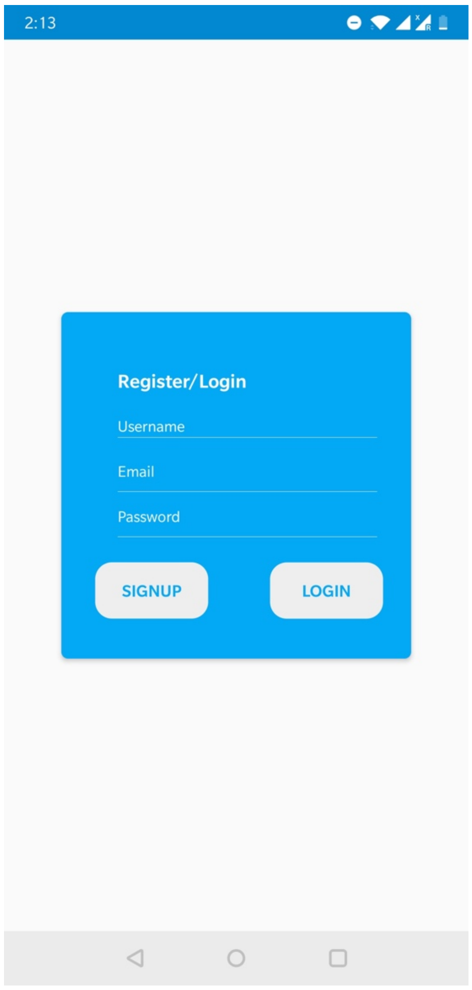


Figure 4 – Login Screen

It is a simple Login/Register Screen where user needs to enter their email ID and password to login and to Register all three fields are needed. When user registers he is prompted to select any option from Chat, Email or Forum. He/She will be always redirected to the respective screen.

3.2. Chat Screen

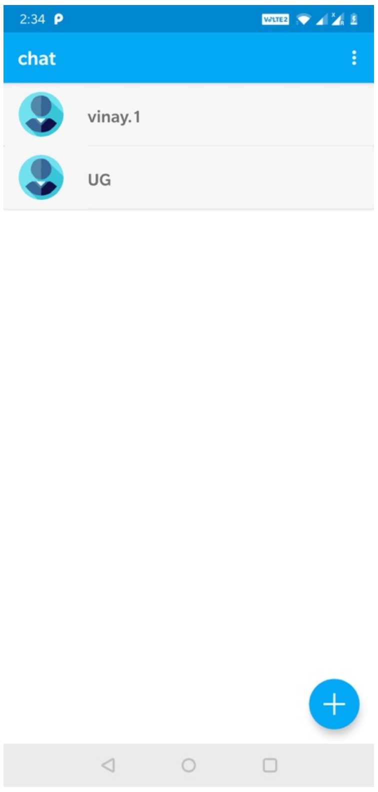


Figure 5 – Chat Home Screen

This screen displays usernames of all user with whom this particular user has started any conversation. He can click on + button to start a conversation with any other user, that are not listed.

3.3 Profile Screen

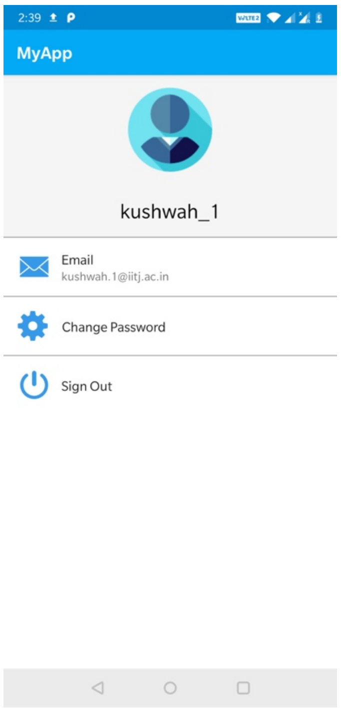
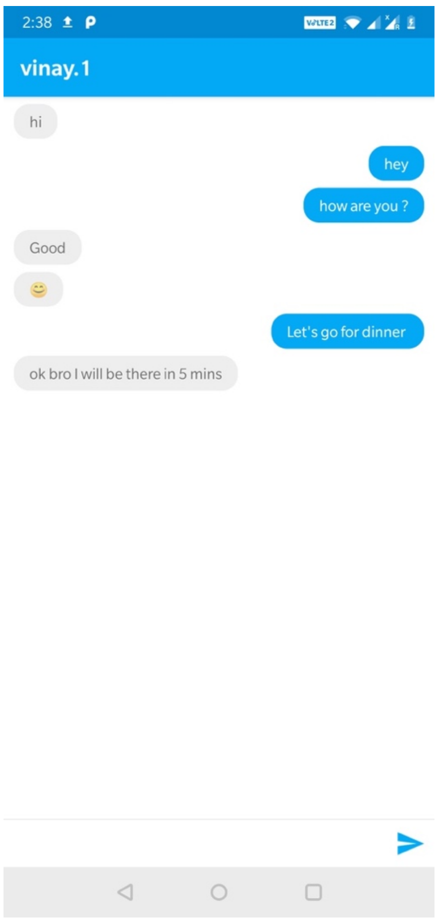


Figure 6 – Profile Screen

Figure 6 shows the profile screen of the app. It can be seen that kushwah_1 user is logged in. User can change password from here or sign out from the app.

3.4 Chatting screen


2:38

vinay.1

hi

hey

how are you ?

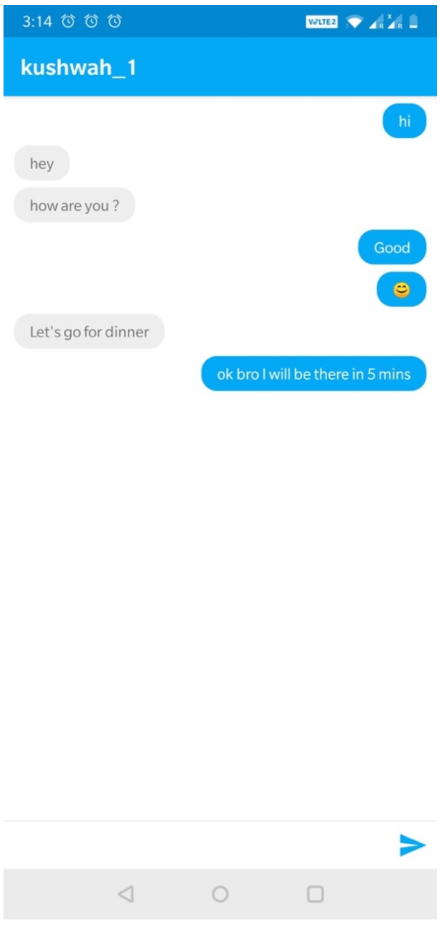
Good

😊

Let's go for dinner

ok bro I will be there in 5 mins

Figure 7 – Chatting screen of kushwah_1


3:14

kushwah_1

hi

hey

how are you ?

Good

😊

Let's go for dinner

ok bro I will be there in 5 mins

Figure 8 – Chatting screen of vinay.1

3.5 Email Home Screen

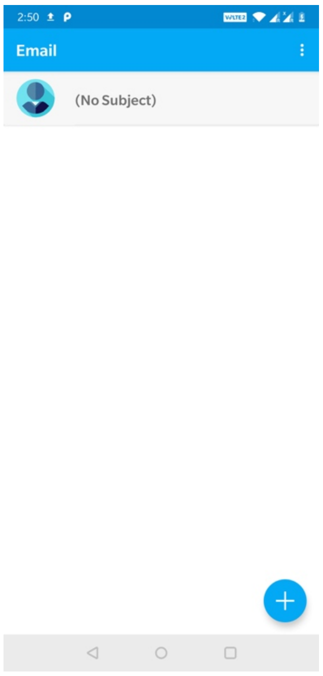
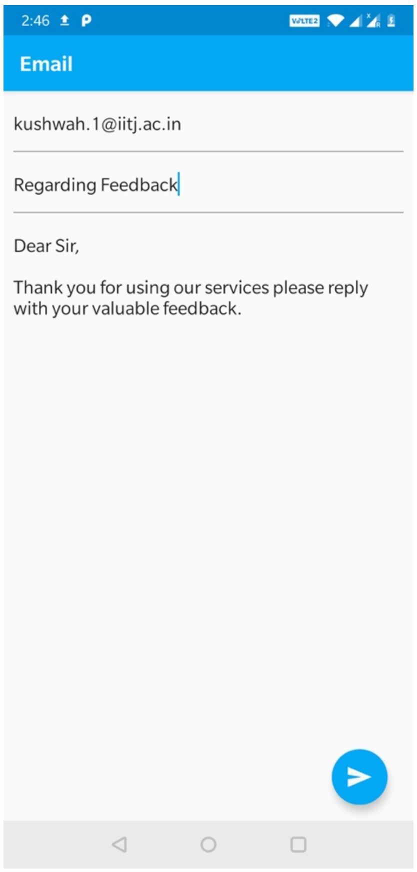


Figure 9 – Email Screen

Figure 9 Shows Email Home Screen of other user. This shows the inbox. User can click on + button to go to the compose email screen.

3.6 Compose Email



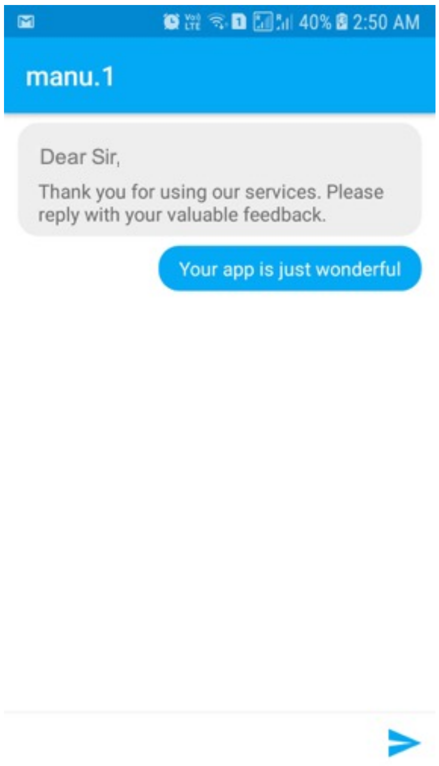
kushwah.1@iitj.ac.in

Regarding Feedback

Dear Sir,

Thank you for using our services please reply with your valuable feedback.

Figure 10 – Compose Email Screen

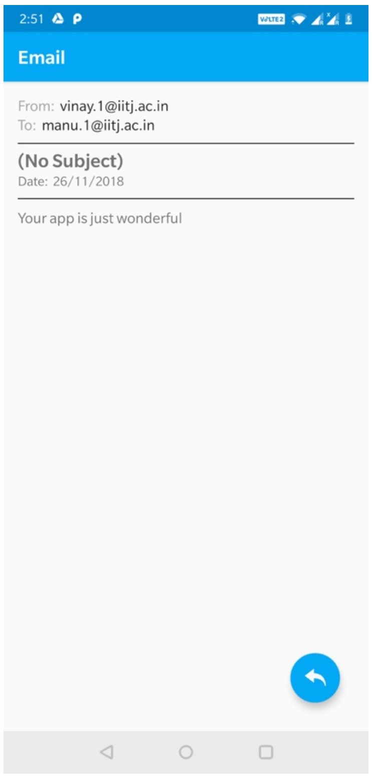

manu.1

Dear Sir,

Thank you for using our services. Please reply with your valuable feedback.

Your app is just wonderful

Figure 11 – Chatting Screen of vinay.1


Email

From: vinay.1@iitj.ac.in
To: manu.1@iitj.ac.in

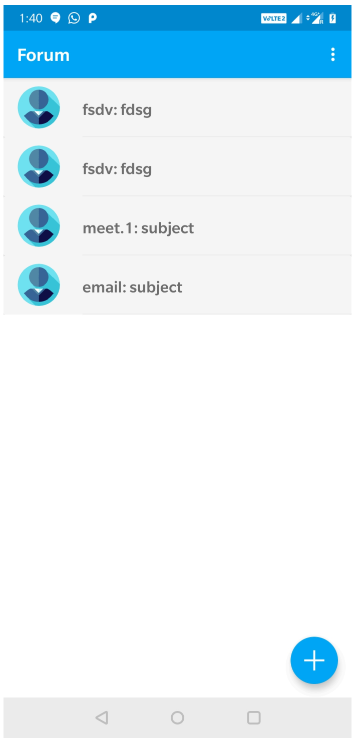
(No Subject)
Date: 26/11/2018

Your app is just wonderful

Figure 12 – Mail from vinay.1

Figure 11 shows user manu.1, who is using email UI and vinay.1, using chat UI. Yet they both are having conversation flawlessly.

3.7 Forum Home screen


Forum

fsdv: fdsg

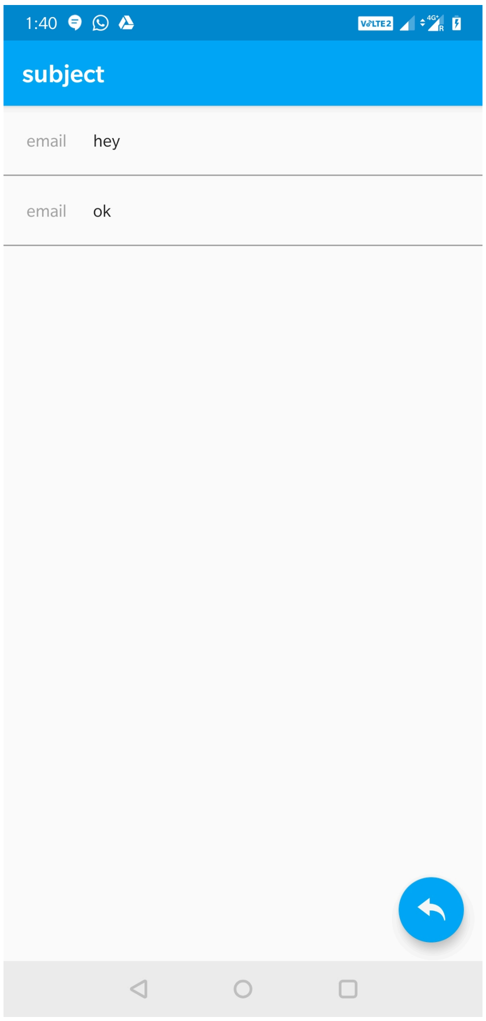
fsdv: fdsg

meet.1: subject

email: subject

Figure 13 – Forum Home Screen

3.8 Forum conversation with other user



email hey

email ok

Figure 14 – Conversation with other user

4. Challenges

- Working with Android Studio was difficult at first glance.
- Selecting Database Structure was a big issue as we need to provide features of chat, email and forum in a single app. Easy way was to create separate copies for email, chat and forum, but this would use lot of database space.
- Firebase has its own pros and cons. Realtime Database don’t have powerful queries that are available with SQL Databases. This also contributed towards difficulty of designing database structure.
- At beginning we had not used github, thus combining the parts of code was very difficult as we were not able to keep track of changes made. Github solved this problem to great extent.

5. Result

- Users are successfully authenticated using Firebase Auth.
- Users can update their password.
- Users can select UI based on their liking.

- Users can have conversation irrespective of the UI selected by both of them.
- Their conversation is stored on Firebase Realtime Database.

6. Future Scope

This single app can replace email, chat and forum apps. Populating this app with features like blocking users, group conversation, freedom of sending attachments can create great impact in the market.

7. References

- Firebase Documentation URL : https://firebase.google.com/docs/
- Android Studio Documentation URL : https://developer.android.com/docs/
- Introduction to Multi-Tenant Apps : https://quickleft.com/blog/introduction-to-designing-multi-tenant-web-applications/
- Introduction to web sockets : https://medium.com/@dominik.t/what-are-web-sockets-what-about-rest-apis-b9c15fd72aac