# Project Report

## Android Application Development

# ChatConnect

(A Real-Time Chat and Communication App)

## Team ID: 593383

# List of Contents



**Source Code :** https://drive.google.com/file/d/11ZBHQzJY4vnnHzyvdXNko1ilAlE8kXsO/view

**Project Demo Link :**

https://drive.google.com/file/d/11ZBHQzJY4vnnHzyvdXNko1ilAlE8kXsO/view

# 1. INTRODUCTION

## 1.1 Project Overview:

The ChatConnect Application, a real-time chatting application, aims to provide a seamless communication platform for users across various devices. This Application With a surge in the use of digital communication, our project aims to stand out by providing a seamless and secure chatting experience. Key goals include improving messaging speed, offering multimedia sharing options, and ensuring user data privacy. User authentication and data encryption are prioritized to maintain a secure environment. It's not just an application but a comprehensive solution for enhancing digital interactions. The application targets users across a wide range of devices, from smartphones and tablets to desktop computers. Our project aims to bridge the gap between existing chat applications and the increasing demand for a more seamless and efficient mode of communication. This project is an answer to the ever-expanding need for advanced, user-centric chat applications.

## 1.2 Purpose:

The purpose of this project is to create a space where individuals can communicate with a sense of security and trust, free from data privacy concerns. The primary aim is to reduce communication obstacles by enhancing message delivery speed and reliability. Our purpose is to revolutionize how individuals communicate in the modern digital landscape. the critical role that communication plays in various aspects of modern life, we have undertaken this project to provide a robust and user-friendly platform that offers a holistic chatting experience.ChatConnect aim is to bring people closer, regardless of geographical distances, by offering a reliable and intuitive chat platform. Our goal is to eliminate barriers to entry, making digital communication accessible to a wide range of individuals. Ultimately, our vision is to set new standards, making digital interactions faster, safer, and more enjoyable for users across the globe, and enhancing the quality of online conversations.

# 2. LITERATURE SURVEY

## 2.1 Existing problem:

In our comprehensive literature survey, we explored the numerous challenges and issues that exist within the current landscape of digital communication applications. One of the most pervasive problems is the growing demand for secure, seamless, and user-centric chatting experiences. Existing chat applications often fall short of these expectations. Users frequently encounter issues related to messaging speed, data privacy, and efficient multimedia sharing.

## 2.2 References:

Our literature survey was informed by a wide range of authoritative sources, including academic studies, industry reports, and expert analyses. Some of the prominent references that shaped our understanding of the existing problems in digital communication include [List notable references and sources you used for the survey.

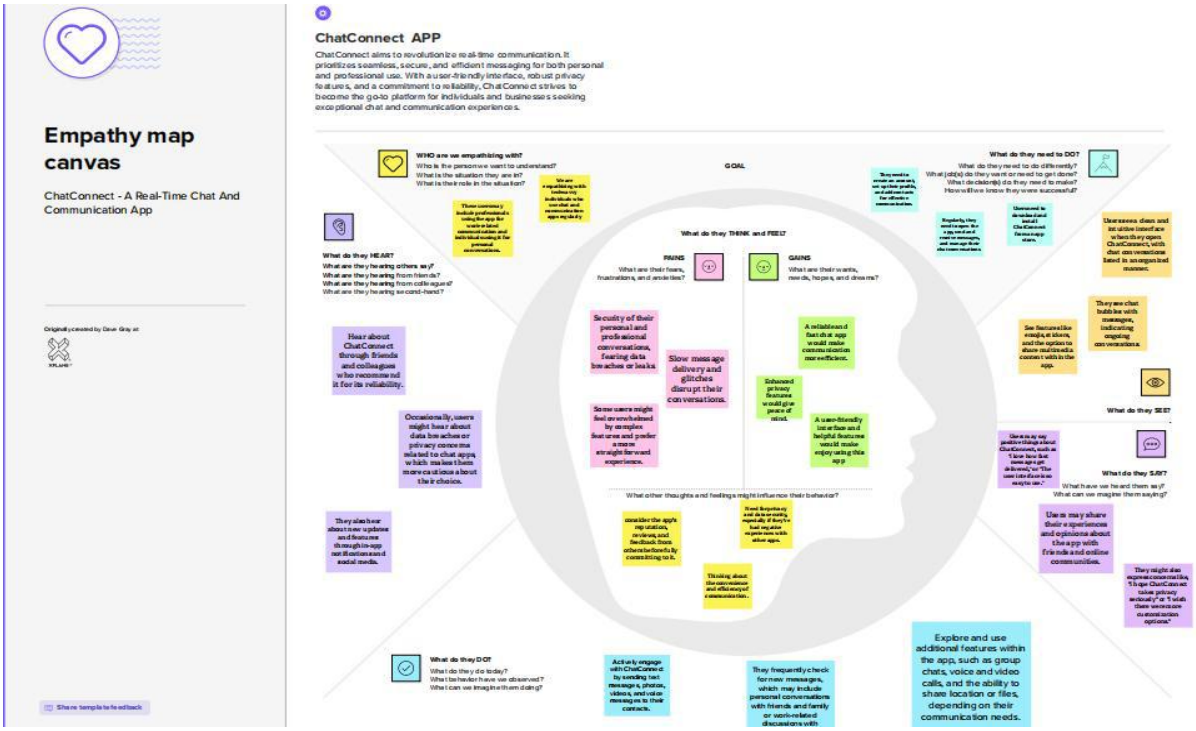
## 2.3 Problem Statement Definition:

Through our research, we have identified a clear statement that is central to our work. The problem stems from shortcomings in the current network, such as slow messaging, data privacy issues, and a lack of userfocused features like good collaboration. As digital communications continue to play an increasingly important role in today's society, these issues are becoming increasingly evident.

Our project ChatConnect emerged as a solution to these challenges to offer users a unique, secure and userfriendly chat. The definition of the problem can be summarized as the need for a new and powerful communication application that can bridge the gap between the limitations of existing solutions and the growing need for interactive, user-centered applications. By solving these problems, we aim to fundamentally change the way people communicate in today's digital environment, removing barriers to access and promoting trust, safety and community.

# 3. IDEATION & PROPOSED SOLUTION

## 3.1 Empathy Map Canvas



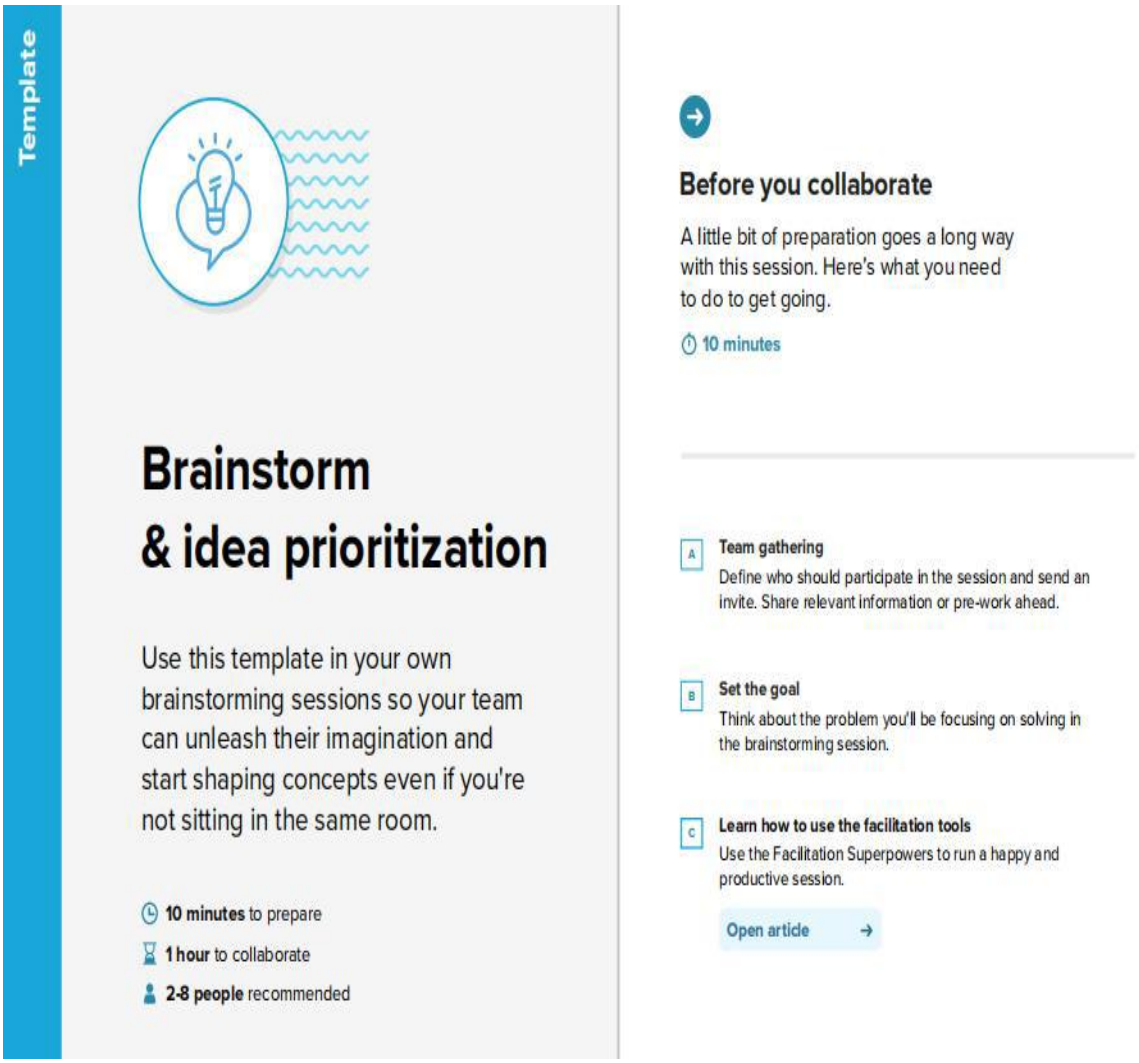

Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

**Link:**
https://app.mural.co/t/chatconnectapp7674/m/chatconnectapp7674/1696732146225/a0216f5267d6d9c4c8260a34a83bce6771b2160f?sender=ue2e1b0c643ed6453c4012001

## 3.2 Ideation & Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

### Step-1: Team Gathering, Collaboration and Select the Problem Statement



1

## Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

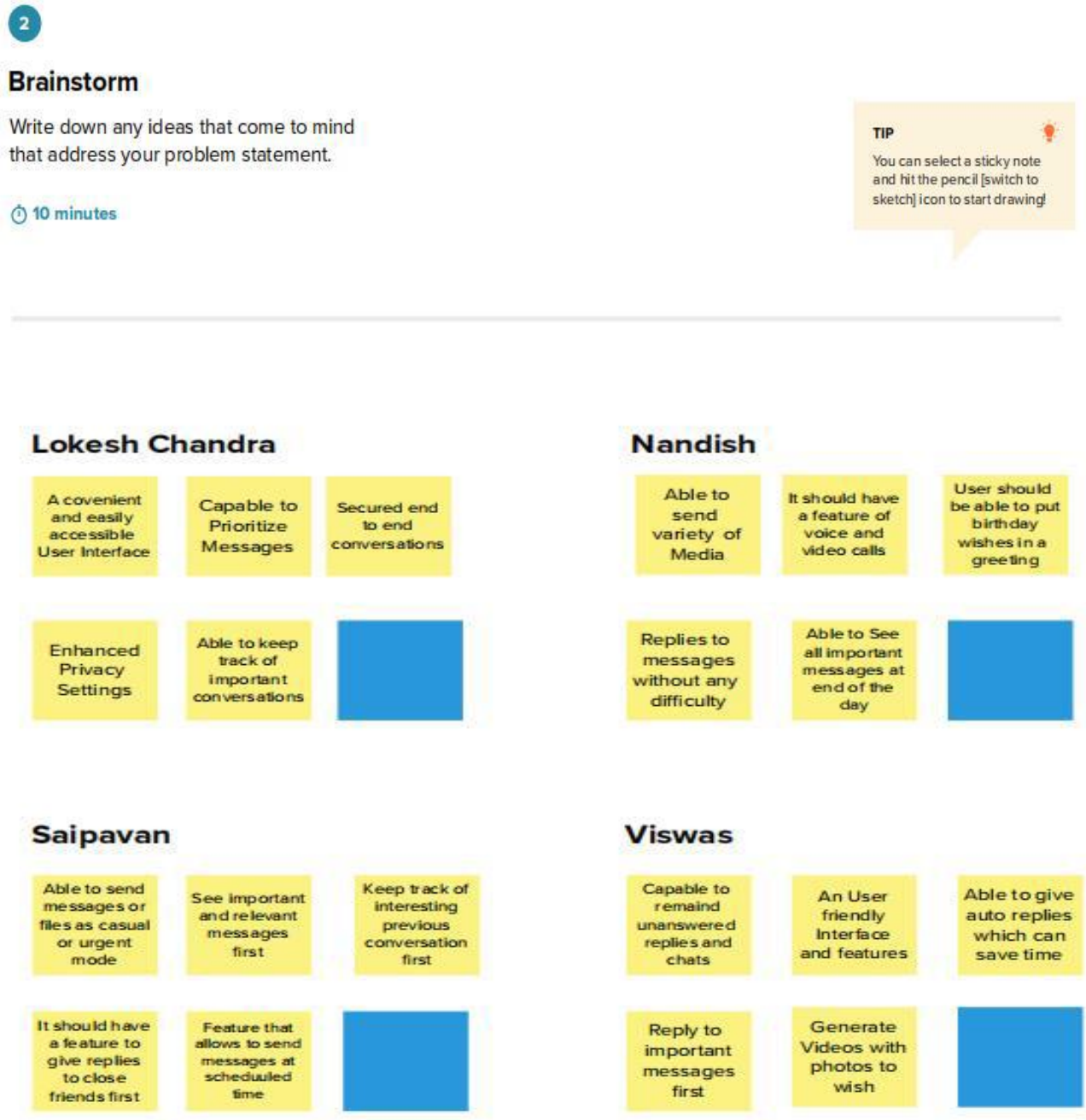
**How might we create a chatting application in real time ?**

### Key rules of brainstorming

To run an smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

## Step-2: Brainstorm, Idea Listing and Grouping



3

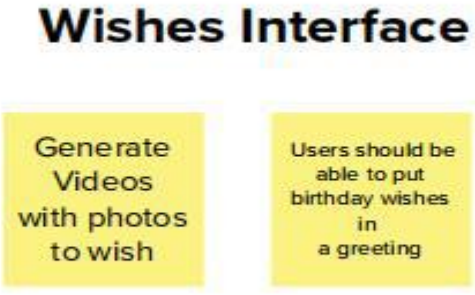
## Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

## Step-3: Idea Prioritization

4

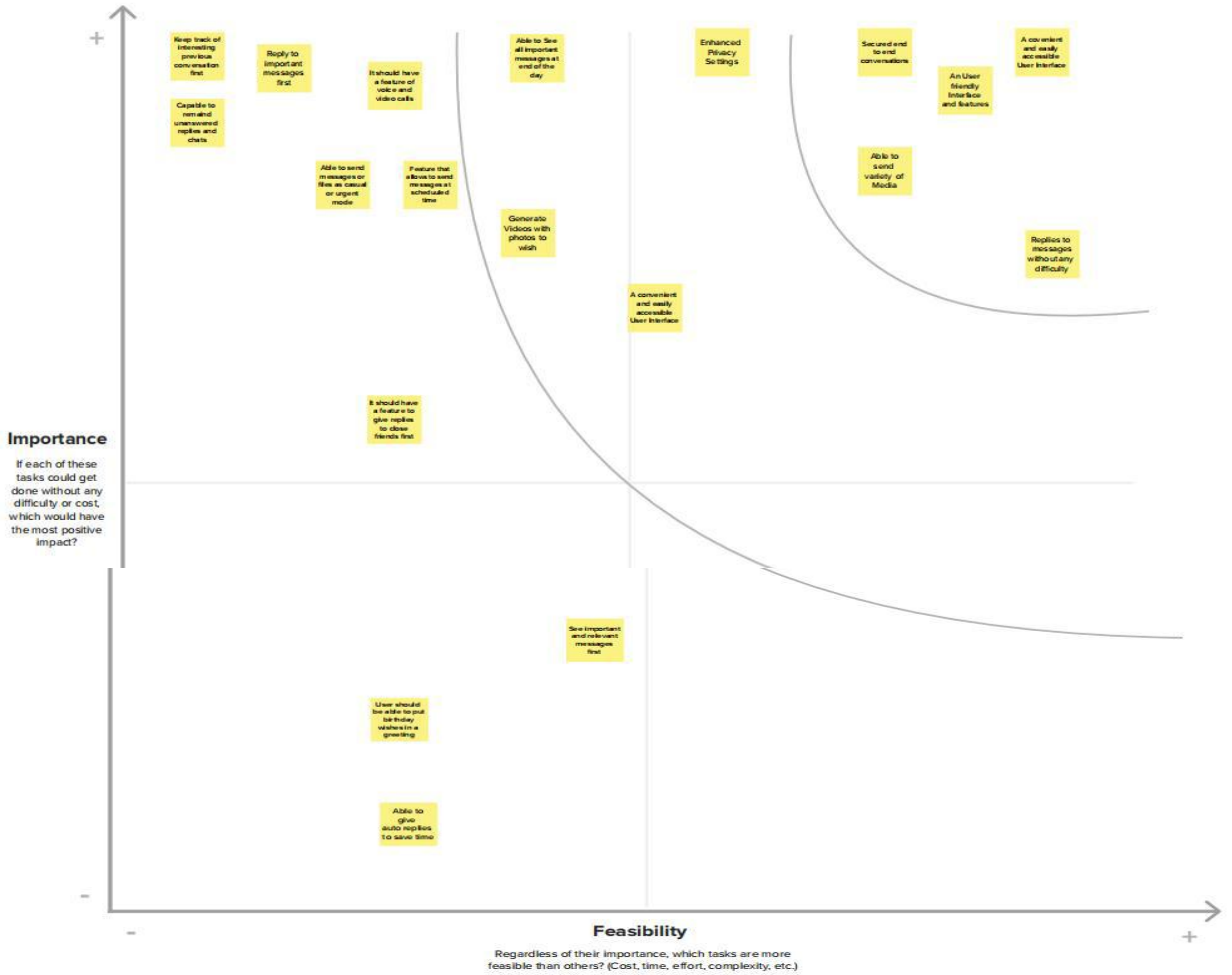
### Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

**TIP**

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.



# 4. REQUIREMENT ANALYSIS :

## 4.1 Functional Requirement:

Functional requirements are essential for defining the specific features and functionalities that the Chat-Connect app should have. Here are some functional requirements for the Chat-Connect project, which is a chatting app developed using Firebase, Android Studio, Kotlin, and Jetpack Compose:

**1. User Registration and Authentication**:

- Users must be able to register and log in securely using their email or phone number.
- The system should support third-party authentication methods (e.g., Google, Facebook).

**2. User Profile Management:**

- Users should be able to create and update their profiles with a display name and profile picture.
- User profiles should display basic information such as status and last online time.

**3. Chat Features**:

- Users can initiate one-on-one or group chats.
- Messages should support text, images, and emojis.
- Real-time message delivery and read receipts (message status) must be implemented.
- Users can view and edit message history.

**4. Notification and Alerts:**

- Users should receive push notifications for new messages.
- Notifications should be customizable based on user preferences.

**5. Friend Management:**

- Users can send and accept friend requests.
- A user's friend list should be easily accessible.
- Block and report functionalities for managing contacts should be included.

**6. Search and Discovery**:

- Users can search for other users based on display names and interests.
- A recommendation system can suggest new contacts or groups based on user behavior.

**7. Integration with Firebase:**

- The application should use Firebase for real-time database and cloud storage.

**8. Android Studio and Kotlin:**

- The app will be developed using Android Studio and written in Kotlin.

- Utilize Jetpack Compose for building the user interface.

**9. Cross-Platform Compatibility:**

- Ensure compatibility with various Android devices and versions to reach a wide user base.

## 4.2 Non Functional Requirements

Non-functional requirements are equally important as they define the qualities and characteristics that the Chat-Connect app should exhibit beyond its core functionalities. Here are non-functional requirements for the project:

**1. Performance:**

- Response Time: The app must respond to user actions within 1 second for a seamless experience.

- Scalability: The app should be able to handle a growing user base and chat volume without significant performance degradation.

**2. Security:**

- Data Encryption: All user data and messages must be end-to-end encrypted for privacy and security.

- Authentication: User authentication should be robust, protecting against unauthorized access.

- Data Backup: User data backups should be stored securely to prevent data breaches.

**3. Reliability:**

- The app should have a 99.9% uptime, ensuring users can access their messages and contacts reliably.

**4. Compatibility:**

- The app must be compatible with a range of Android devices and screen sizes.

- Ensure compatibility with the latest Android OS versions.

**5. User Experience:**

- The app should provide an intuitive and user-friendly interface, following Material Design guidelines.

- Support for dark mode should be consistent and visually appealing.

**6. Accessibility:**

- The app must meet accessibility standards, ensuring it's usable by people with disabilities.

**7. Resource Usage:**

- Optimize resource usage, including battery and data, to ensure efficient operation..

**8. Maintainability:**

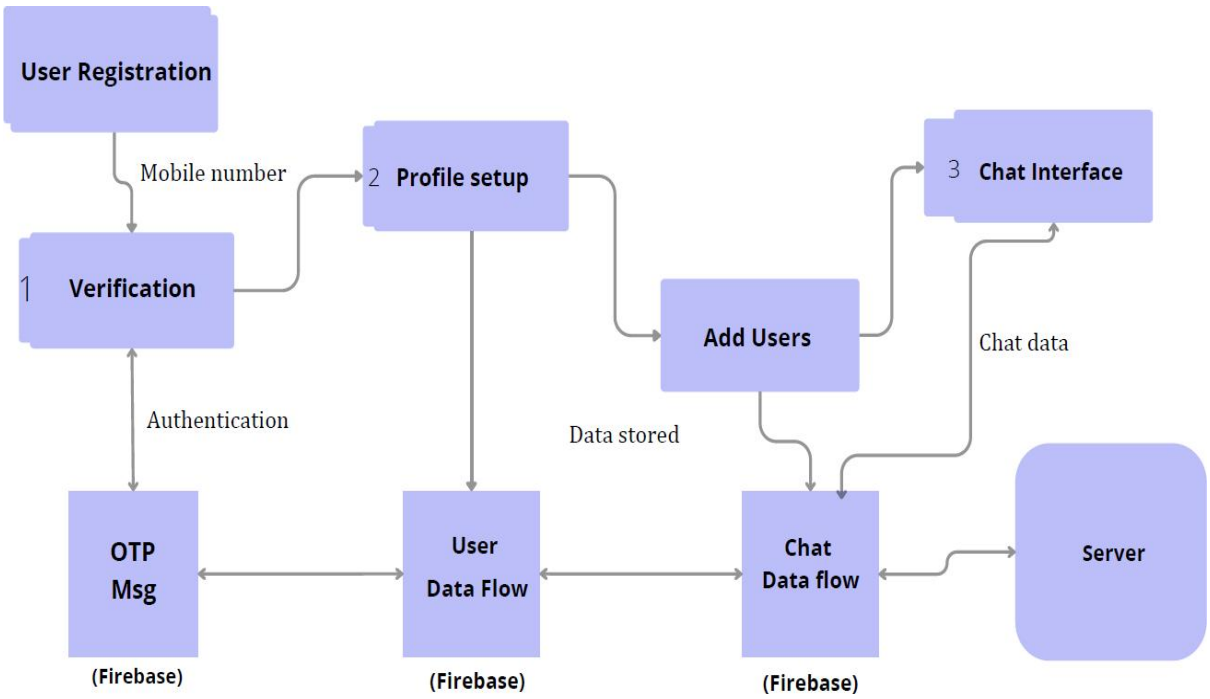
- Maintainable codebase with proper documentation to facilitate future updates and feature additions.

These non-functional requirements help ensure that Chat-Connect not only functions as intended but also delivers a secure, reliable, and enjoyable user experience while complying with industry standards and regulations.

# 5.PROJECT DESIGN

### 5.1 Data Flow Diagrams and User Stories:

A Data Flow Diagram (DFD) for the "Chat-Connect" app, developed in Firebase and Kotlin within Android Studio, illustrates the flow of data and interactions within the application. It showcases the primary components, including users, chat rooms, and Firebase services. User actions, such as sending messages, initiating chats, and managing profiles, are depicted as data flows between these components. Firebase serves as the central hub for real-time messaging, user authentication, and data storage. This DFD provides a clear visual representation of how data moves through the system, aiding in the understanding and design of the app's functionality and data management.




User stories encapsulate the essence of their interactions with the app, such as the seamless registration process, intuitive messaging, real-time notifications, and personalized profile management. Users, from diverse backgrounds, are at the heart of our design, enjoying the convenience of securely connecting with friends and colleagues. These stories reflect the essence of our application, making it an indispensable tool for modern communication, further highlighting how Firebase and Kotlin in Android Studio come together to create a reliable and user-centric chat app.

| User Type | Functional Requirement (Epic) | User Story Number | User Story / Task | Acceptance criteria | Priority | Release |
|---|---|---|---|---|---|---|
| Customer (Mobile user) | Registration | USN-1 | As a user, I can register on the application by using my mobile number. | I can access my account | High | Sprint-1 |
| | Privacy | USN-2 | As a user, I need end-to-end encryption of chats | Chats are end-to-end encrypted | High | Sprint-1 |
| | UI/UX | USN -3 | User friendly interface | The UI/UX is well designed | High | Sprint-1 |
| | | USN-4 | As a user I expect all multi media is shared on the app | All types of files and media are shareable | medium | Sprint-2 |
| | Chat Back up /storage | USN-5 | As a user, I should back up my chat data | Chats are backed up to the cloud. | medium | Sprint-2 |
| | Dashboard | USN-6 | As a user, I can access a personalized dashboard | The dashboard is accessible from the application's main menu | Medium | Sprint 2 |
| Customer Care Executive | User Support and Assistance | USN-7 | I can help the users regarding the steps they should take for respective issues they have come across | The CCE should provide clear and detailed instructions or steps for the user to take in order to address or resolve their issue. | medium | Sprint 3 |
| Administrator | Analytics and Reporting | USN -8 | As an Administrator, I can access analytics and reporting tools to monitor application usage, user engagement, and performance metrics. | have access to analytics dashboards that provide insights into user activity, chat usage, and system performance. | Medium | Sprint 4 |

## 5.2 Solution Architecture:

A messaging app developed using Firebase, Android Studio, Kotlin, and Jetpack Compose, is designed to provide a robust and efficient platform for seamless real-time communication.

At the core of this architecture is Firebase, which handles user authentication, real-time database services, and cloud storage for chat history and media sharing. Firebase Authentication ensures secure user access, while Firebase Realtime Database or Cloud Firestore manages real-time message delivery and chat room creation. Firebase Cloud Messaging enables push notifications for timely message alerts.

The app's frontend is developed in Kotlin and Jetpack Compose, offering a modern, responsive, and user-friendly interface. Jetpack Compose simplifies UI development with its declarative approach, making it easier to create dynamic, interactive user interfaces.

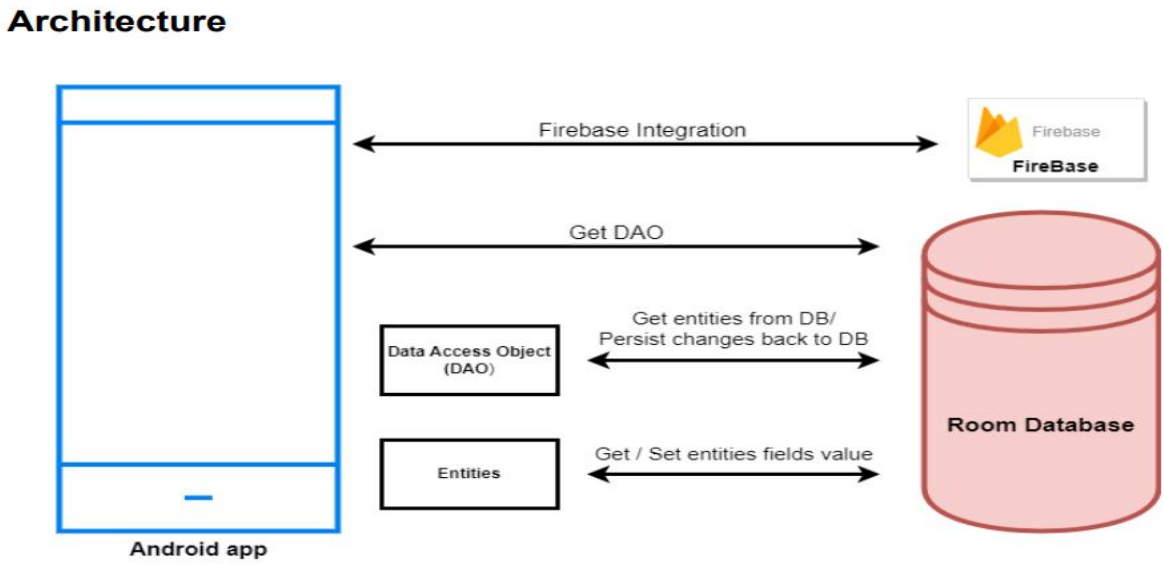
The solution architecture emphasizes scalability, security, and real-time capabilities, ensuring that "Chat-Connect" delivers a reliable and engaging chat experience while benefiting from the power of Firebase and the flexibility of Kotlin and Jetpack Compose.

| S.No. | Parameter | Description |
|---|---|---|
| 1. | Problem Statement (Problem to besolved) | In today's fast-paced digital world, effective and efficient communication is very important for individuals, businesses, and organizations alike. However, the existing landscape of chat and communication apps presents several challenges and shortcomings that need to be addressed.<br>ChatConnect is a real-time chat and communication app that aims to revolutionize the way people interactand connect online. This provides a wide range of features and functionalities to facilitate seamless communication and collaboration.Here's an overview of what ChatConnect provides Connectivity, Collaboration, Convenience, Expression Conversations, Privacy and Security. |
| 2. | Idea / Solution description | Whether you're looking to chat with friends, collaborate with colleagues, or engage with customers, ChatConnect provides a comprehensive platform to meet all your communication needs.<br>ChatConnect strives to provide an intuitive and user- friendly experience, making communication effortless and enjoyable. Whether you're chatting with friends, collaborating with colleagues, or engaging with customers, this aims to be your go-to<br>communication app, keeping you connected in real-time and fostering meaningful interactions. |
| 3. | Novelty / Uniqueness | Use of ChatConnect application is to bring together various communication tools like messaging, voice calls, video calls, and file sharing into one integrated platform for convenience and efficiency. This will offer advanced collaboration features such as document sharing, screen sharing, and collaborative editing, enabling teams to work together seamlessly from anywhere in real-time. This provide a stable and fast communication experience, minimizing downtime and delays. |
| 4. | Social Impact / Customer Satisfaction | ChatConnect brings people closer and makes communication easier, fostering connections and collaboration among individuals and groups while respecting privacy and security."<br>In simpler terms, ChatConnect helps people connect and work together more easily and safely, which can improve relationships and teamwork in our digitalworld. |

| | | |
|---|---|---|
| 5. | Business Model (Revenue Model) | ChatConnect's business model revolves around a freemium approach. We offer a free version of the app with basic features and generate revenue by selling premium subscriptions with advanced functionalities. Additionally, we display targeted advertisements to users, collaborate with partners for integrated services, and may offer in-app purchases and data analytics insights. This combination of monetization strategies ensures sustainability and growth for the ChatConnect application. |
| 6. | Scalability of the Solution | We designed the app architecture to be horizontally scalable, allowing seamless expansion of server capacity as the user base grows.We utilized cloud-based services or distributed systems to handle increased traffic and ensure optimal performance even during peak usage.Scalability for ChatConnect means making sure that as more and more people use the app, it doesn't slow down or break. We do this by setting up the app in a way that it can handle lots of users and messages without any problems, like adding more computers when needed and making sure everything runs smoothly. So, no matter how many people use ChatConnect, it keeps working well. |

# 6.PROJECT PLANNING & SCHEDULING

## 6.1 Technical Architecture



## 6.2 Sprint Planning & Estimation

Creating a sprint delivery schedule for the "Chat-Connect" Android application, built using Firebase, Android Studio, Kotlin, and Jetpack Compose, involves planning the work to be completed in each sprint. Here's a sample sprint delivery schedule:

| Sprint | Functional Requirement (Epic) | User Story Number | User Story / Task | Story Points | Priority | Team Members |
|---|---|---|---|---|---|---|
| Sprint-1 | Project Setup | USN-1 | Set up the environment needed to do this project by collecting and knowing all the required plugins and library dependencies.Also we should add them in gradle Script files. | 10 | High | Viswas |
| Sprint-2 | User Interface Design | USN-2 | For every application User Interface plays very important role.So User Interface should be simple and stylish at the same time when looking into the screen users should be able to understand.<br>a. Login Interface<br>b. Registration Interface<br>c. OTP Authentication Interface<br>d. Chatting Interface | 40 | Moderate to High | Nandish<br>Viswas<br>Lokesh<br>Saipavan |
| Sprint-3 | Database Integration | USN-3 | Whenever a new user registers,his login credentials should be stored in the database.So that if a new user enters the same data and try to register "User Already Exists" message should be notified. | 15 | High | Pavan |
| Sprint-4 | Firebase Authentication | USN-4 | Whenever a user enters username or email id or mobile number and password they should be Verified and login action should be done. | 10 | High | Nandish |
| Sprint-5 | Navigation Features | USN-5 | In every application clear and visible navigation buttons should be there.So that User can have Better experience of the App. | 10 | Moderate | Viswas & Pavan |
| Sprint-6 | Robustness | USN-6 | Whenever users use the application concurrently we should be able handle all of them.The app should run perfectly without any lag and difficulty. | 15 | Low | Nandish & Lokesh |

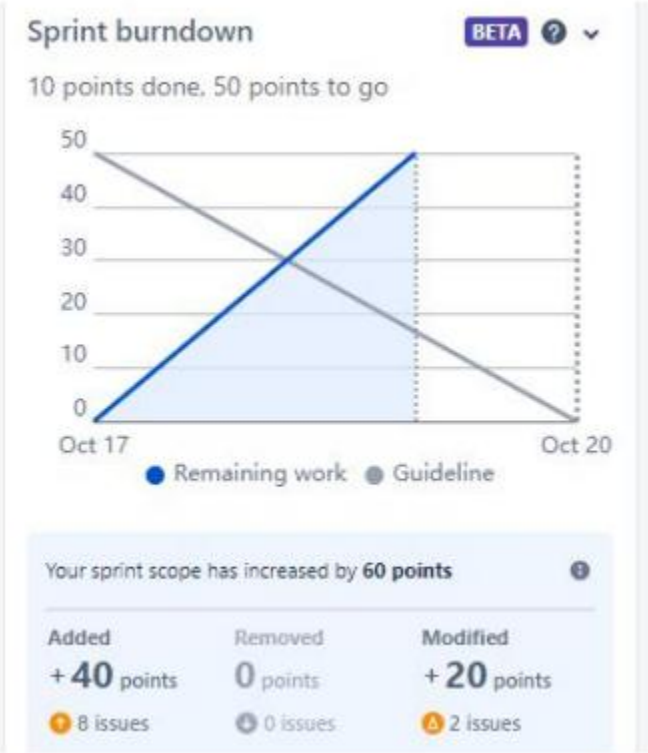
## 6.3 Sprint Delivery Schedule

| Sprint | Total Story Points | Duration | Sprint Start Date | Sprint End Date (Planned) | Story Points Completed (as on Planned End Date) | Sprint Release Date (Actual) |
|---|---|---|---|---|---|---|
| Sprint-1 | 10 | 1 Day | 16 Oct 2023 | 16 Oct 2023 | 10 | 17 Oct 2023 |
| Sprint-2 | 40 | 4 Days | 17 Oct 2023 | 20 Oct 2023 | 40 | 20 Oct 2023 |
| Sprint-3 | 15 | 1 Days | 21 Oct 2023 | 21 Oct 2023 | 15 | 21 Oct 2023 |
| Sprint-4 | 10 | 1 Days | 22 Oct 2023 | 22 Oct 2023 | 10 | 22 Oct 2023 |
| Sprint-5 | 10 | 2 Days | 23 Oct 2023 | 24 Oct 2023 | 10 | 23 Oct 2023 |
| Sprint-6 | 15 | 1 Day | 24 Oct 2023 | 25 Oct 2023 | 15 | 23 Oct2023 |

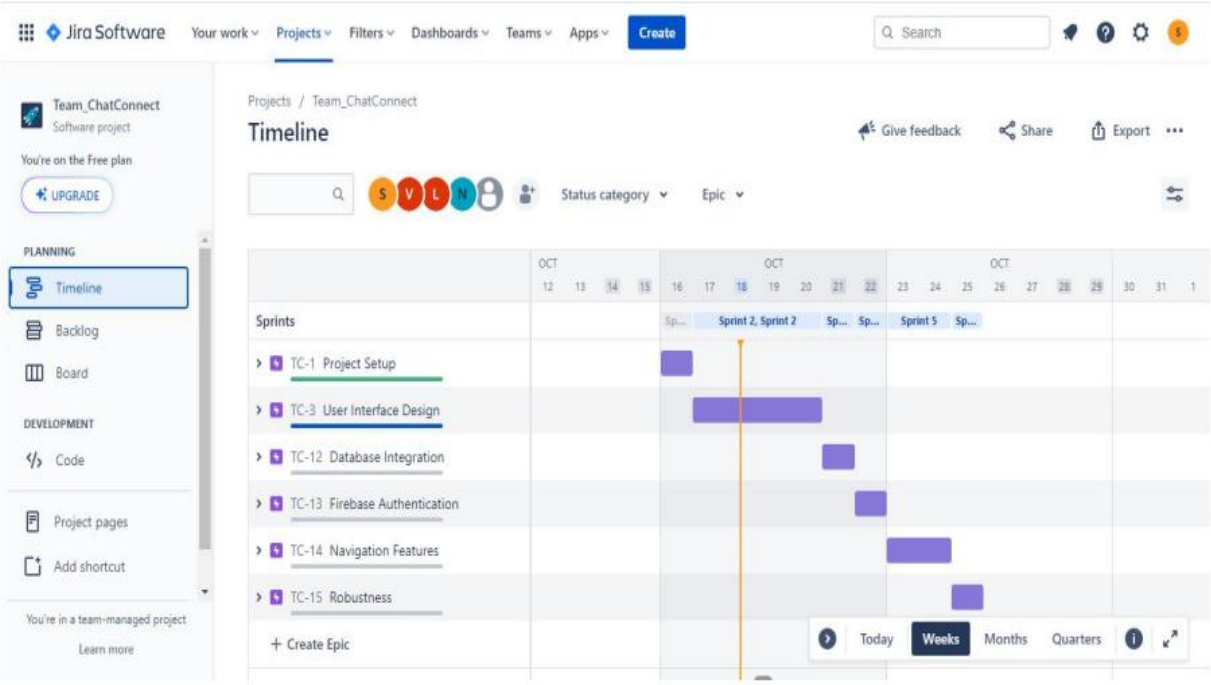
**Velocity:**

We have a 10-day sprint duration, and the velocity of the team is 16.67 (average points per sprint). If we calculate the team's average velocity (AV) per iteration unit (story points per day)

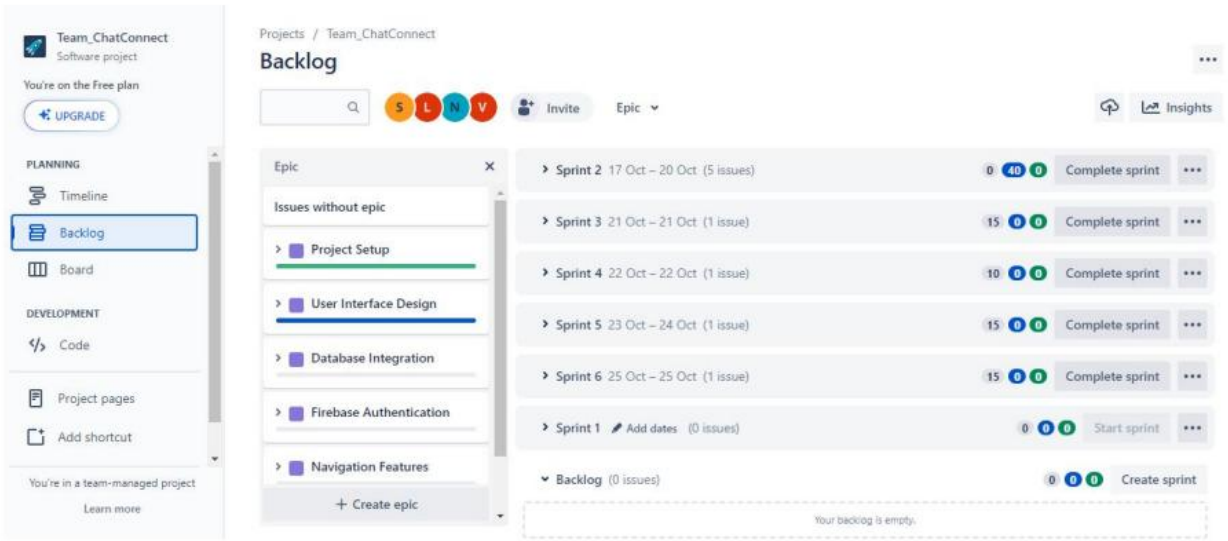
$$AV = (\text{sprint duration}) / (\text{velocity}) = 16.67/10 = 1.667$$


Sprint burndown
BETA
10 points done. 50 points to go
50
40
30
20
10
0
Oct 17
Oct 20
Remaining work
Guideline
Your sprint scope has increased by 60 points
Added
+40 points
8 issues
Removed
0 points
0 issues
Modified
+20 points
2 issues

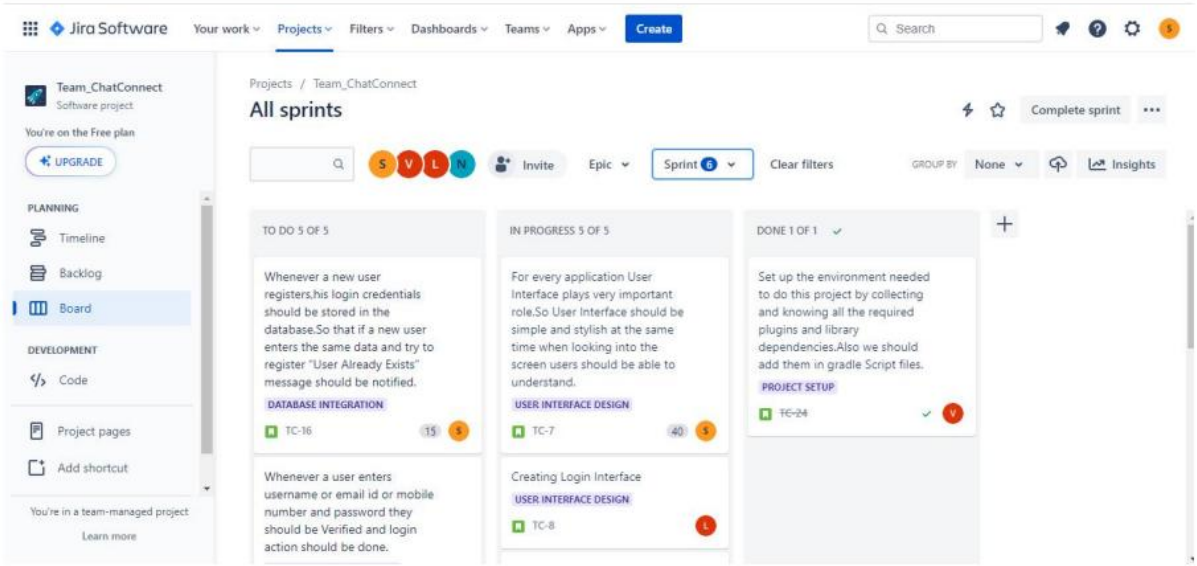
**Timeline :**


Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Team_ChatConnect
Software project
You're on the Free plan
UPGRADE
PLANNING
Timeline
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
You're in a team-managed project
Learn more
Projects / Team_ChatConnect
Timeline
Give feedback
Share
Export
Status category
Epic
OCT
Sprints
Sprint 2, Sprint 2
Sprint 5
TC-1 Project Setup
TC-3 User Interface Design
TC-12 Database Integration
TC-13 Firebase Authentication
TC-14 Navigation Features
TC-15 Robustness
+ Create Epic
Today
Weeks
Months
Quarters

**Backlog :**




**Board :**



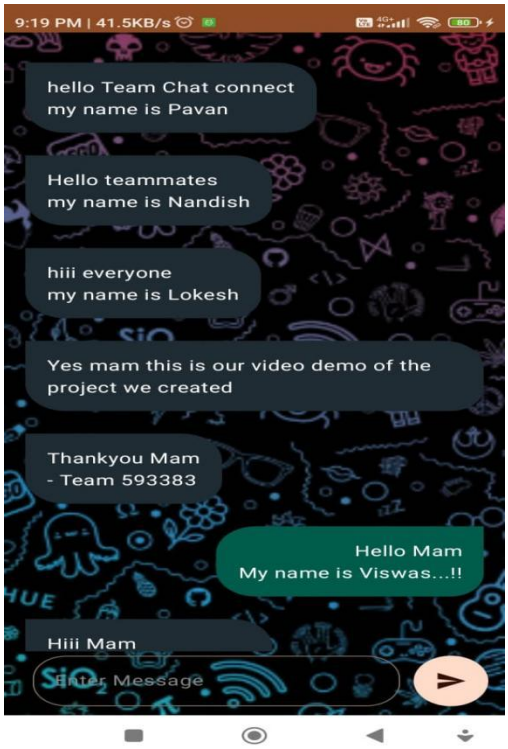

# 7. CODING & SOLUTIONING

## 7.1 Feature 1

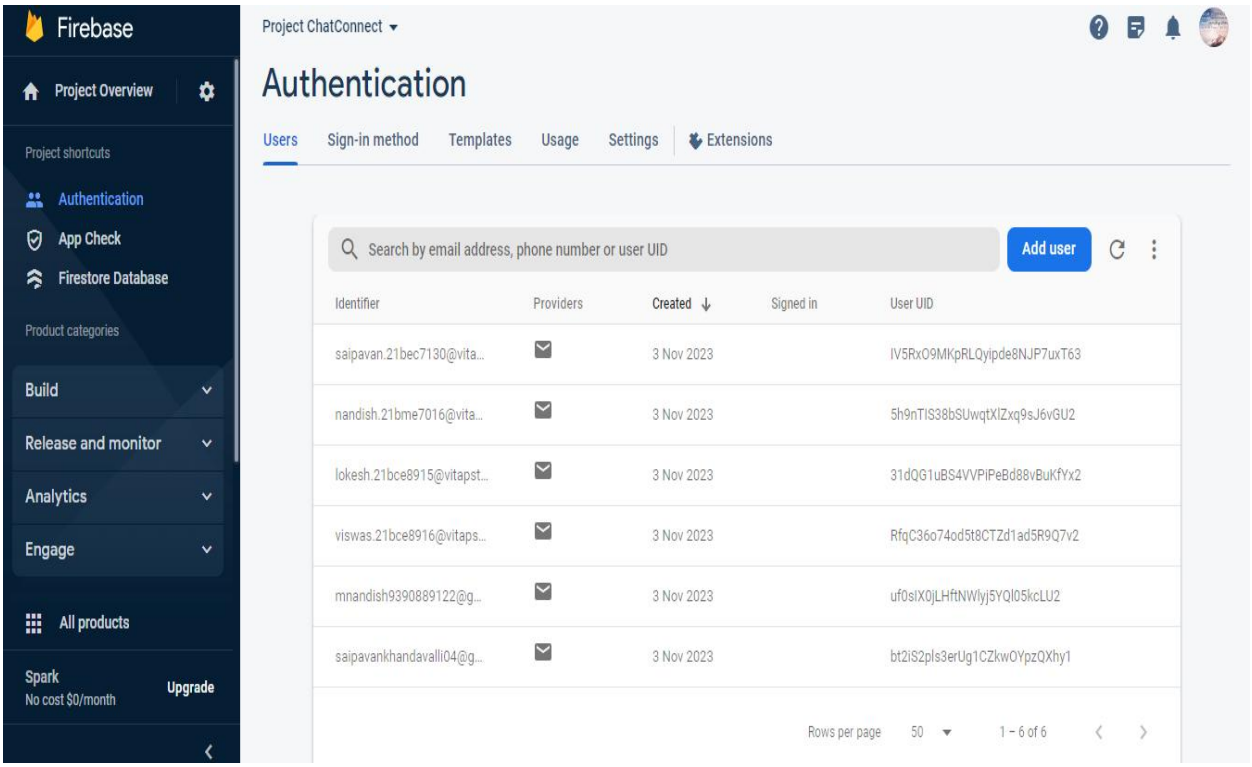
The users can have group discussions through text messages after logging into the app.

## 7.2 Feature 2

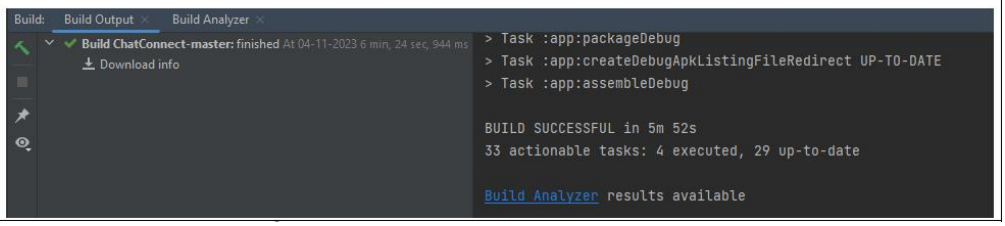
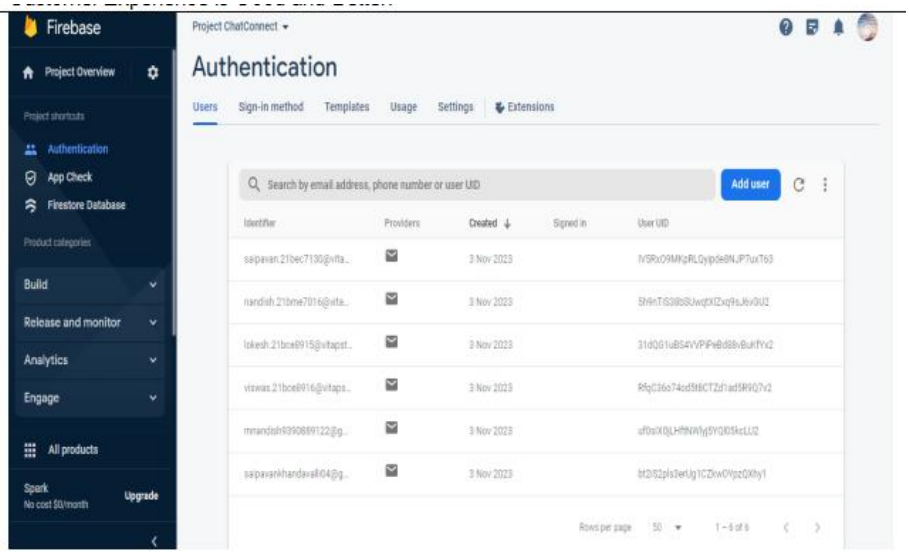
We can register add  new users at Starting by registering.



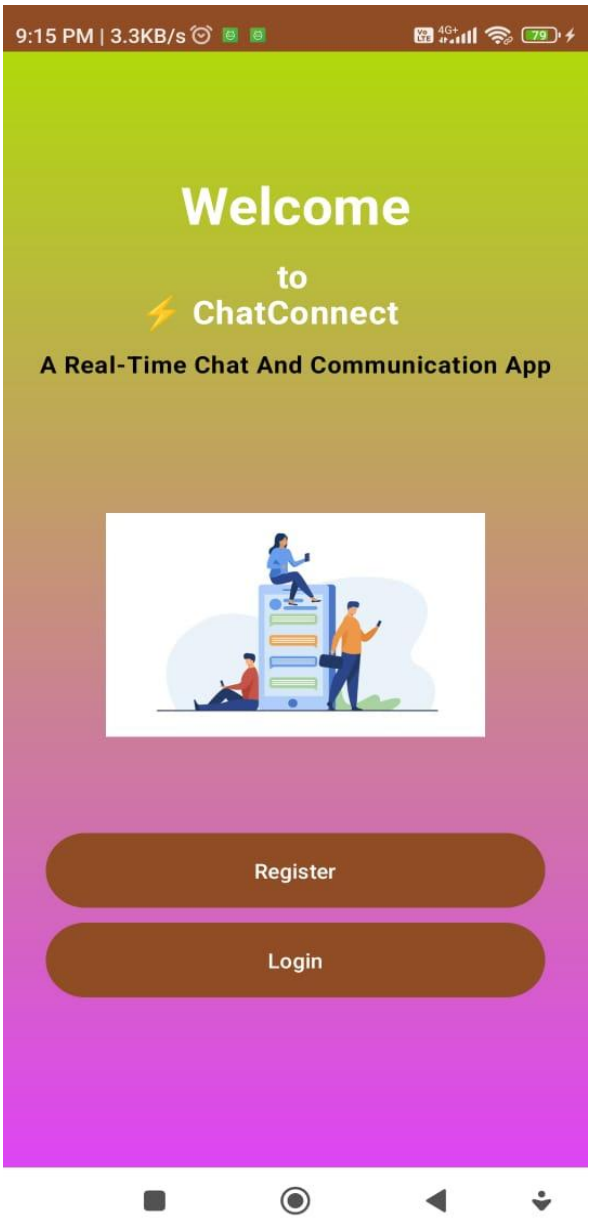

## 7.3 Database Schema



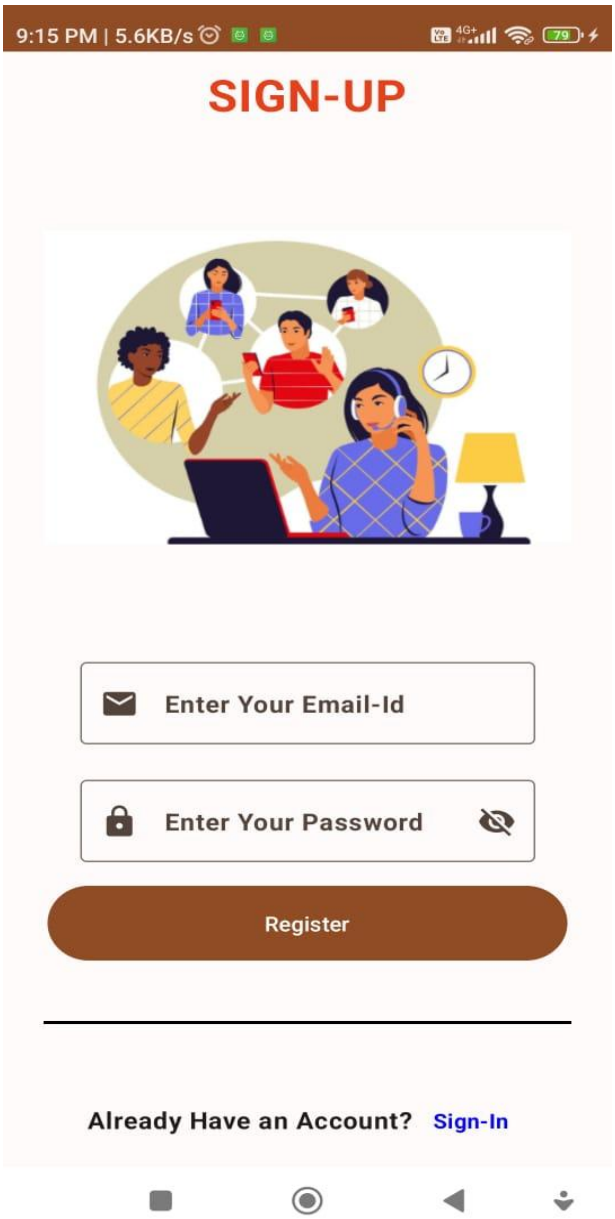
# 8. PERFORMANCE TESTING

| S.No. | Parameter | Values | Screenshot |
|---|---|---|---|
| 1. | Metrics | App Launch Time-<br>Screen Render Time-<br>Code Quality- |  |
| 2. | Usage | App Size-<br>Customer Experience- |  <br>Customer Experience is Good and Better. |
| 3. | Performance | Error and Crash Rates-<br>Database Query Performance- |  <br>Error crash rates are very less so far. |

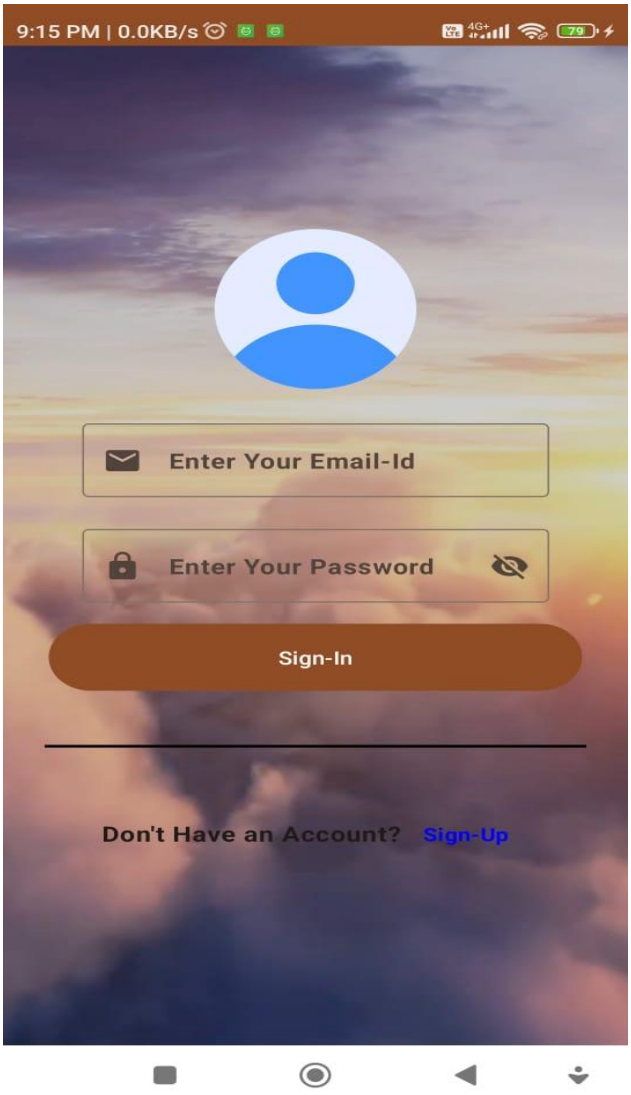
# 9.RESULTS

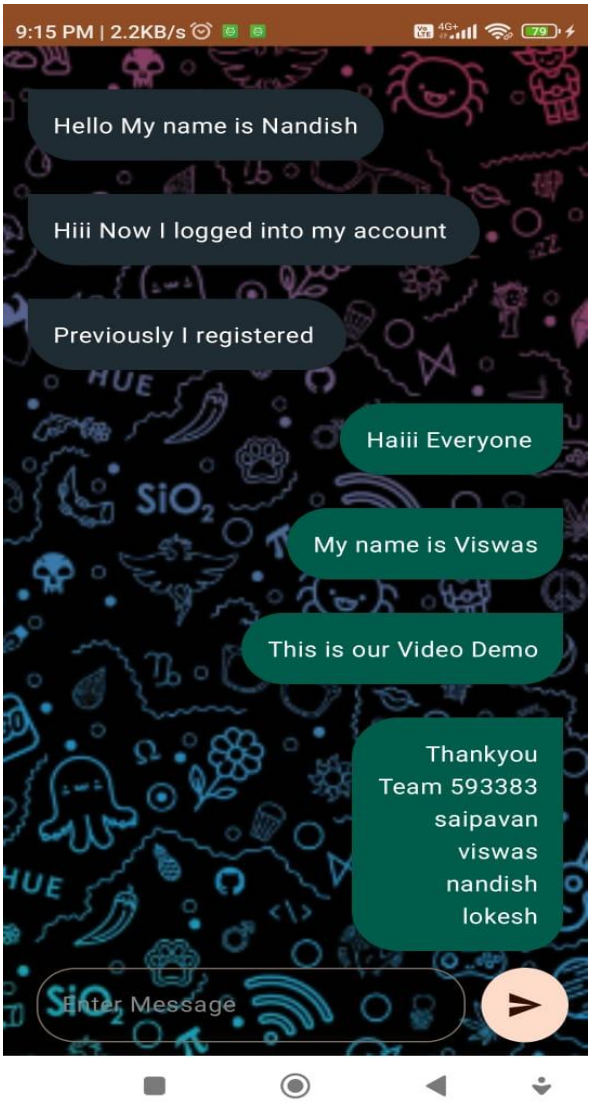



Welcome Screen




Register Screen




**Login Screen** **Chatting Screen**

# 10.ADVANTAGES & DISADVANTAGES

## Advantages:

• **Ease of communication:** ChatConnect allows users to communicate with their contacts in real-time, making it easier for them to stay

connected with friends, family, or colleagues.

• **User-friendly interface:** The user-friendly interface of ChatConnect makes it easy for users to navigate the app and find the features they need.

• **Secure communication:** The app uses Firebase backend for user authentication, which provides a secure way for users to log in and

protect their account information.

• **Personalization:** The user profile feature in ChatConnect allows users to personalize their profile picture, status, and other personal information to make the app more personalized and engaging.

• **Real-time messaging:** With Firebase Realtime Database, ChatConnect provides real-time messaging, ensuring that users receive messages as soon as they are sent, which can enhance communication and collaboration.

### Disadvantages:

• **Ease Limited functionality:** ChatConnect has limited functionality and is only designed for basic communication and messaging features. Users may prefer to use other apps for more advanced features.

• **Network connectivity:** ChatConnect requires a stable internet connection to function correctly. Poor network connectivity can result in slow message delivery or lost messages.

• **Dependency on Firebase:** Cabconnect's backend is dependent on Firebase. If Firebase experiences downtime or other issues, the app may not function correctly.

• **Limited platform support:** ChatConnect is an Android-native app, which means it is not available on other platforms like iOS or Windows, potentially limiting the user base.

• **Security risks:** As with any messaging app, there is a risk of users sharing sensitive or personal information. It is essential to have measures in place to protect user data and privacy.

# 10. CONCLUSION

ChatConnect is a chatting application developed for Android using Kotlin and Firebase backend. It is designed to enable real-time communication between users, allowing them to chat and exchange messages in a secure and user-friendly manner. ChatConnect provides features such as user authentication, real-time messaging, user profile, and a simple messaging interface.

Overall, ChatConnect can be used for a variety of applications, including personal communication, business communication, education, customer support, social networking, group communication, and community-building. With ongoing maintenance and updates, ChatConnect has the potential to be a useful tool for facilitating communication and collaboration among its users.

# 11. FUTURE SCOPE

**1. Location sharing:** By adding location sharing features, users can share their location with friends or family members, making it easier to meet up or find each other in real-time.

**2. File sharing:** The ability to share files through ChatConnect can be useful for businesses or individuals who need to share documents or images quickly and easily.

**3. Group messaging:** Group messaging allows users to communicate with multiple people at once, which can be useful for collaboration, event planning, or socializing.

**4. Voice and video calling:** By adding voice and video calling features, ChatConnect can compete with other popular messaging apps and offer users more ways to communicate with their contacts.

# 13.APPENDIX

## Chat Window Code :

Login.kt | Welcome.kt | Chat_Window.kt | Register.kt | MainActivity.kt

Code | Split | Design

```
package com.example.chatu.view.chat_menu
import android.annotation.SuppressLint
import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.foundation.lazy.items
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.foundation.text.KeyboardOptions
import androidx.compose.material.*
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.Send
import androidx.compose.material3.ExperimentalMaterial3Api
import androidx.compose.material3.FloatingActionButton
import androidx.compose.material3.Icon
import androidx.compose.material3.OutlinedTextField
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.runtime.getValue
import androidx.compose.runtime.livedata.observeAsState
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.layout.ContentScale
```

```
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.TextStyle
import androidx.compose.ui.text.input.KeyboardType
import androidx.compose.ui.unit.dp
import androidx.lifecycle.viewmodel.compose.viewModel
@SuppressLint("SuspiciousIndentation")
@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun HomeView(
    homeViewModel: HomeViewModel = viewModel()
) {
    Image(painter = painterResource(id = com.example.chatu.R.drawable.img_3), contentDescription ="", contentScale
    val message: String by homeViewModel.message.observeAsState(initial = "")
    val messages: List<Map<String, Any>> by homeViewModel.messages.observeAsState(
        initial = emptyList<Map<String, Any>>().toMutableList()
    )
    Column(
        modifier = Modifier
            .fillMaxSize()
            .fillMaxWidth(),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Bottom,
    ) { this: ColumnScope
        Spacer(modifier = Modifier.height(20.dp))
        LazyColumn(
```

```
            modifier = Modifier
                .fillMaxWidth()
                .weight(weight = 0.85f, fill = true),
            contentPadding = PaddingValues(horizontal = 16.dp, vertical = 8.dp),
            verticalArrangement = Arrangement.spacedBy(25.dp),
            reverseLayout = true
        ) { this: LazyListScope
            items(messages) { this: LazyItemScope message ->
                val isCurrentUser = message[Constants.IS_CURRENT_USER] as Boolean
                SingleMessage(
                    message = message[Constants.MESSAGE].toString(),
                    isCurrentUser = isCurrentUser
                )
            }
        }
        Row(
            modifier = Modifier
                .fillMaxWidth()
                .wrapContentSize()
                .padding(9.dp, 5.dp, 0.dp, 10.dp)
                .background(Color.Transparent),
            horizontalArrangement = Arrangement.spacedBy(10.dp)
        ) { this: RowScope
            OutlinedTextField(
                value = message,
```

```
            onValueChange = { it: String
                homeViewModel.updateMessage(it)
            },
            textStyle = TextStyle(Color.White),
            placeholder = {
                Text(text = "Enter Message", color = Color.Gray)
            },
            shape = RoundedCornerShape(25.dp),
            modifier = Modifier.width(275.dp),

            keyboardOptions = KeyboardOptions(
                keyboardType = KeyboardType.Text
            )
        )
        FloatingActionButton(
            onClick = {
                homeViewModel.addMessage()
                homeViewModel.updateMessage("")
            },
            shape = RoundedCornerShape(85.dp),
        ) {
            Icon(
                imageVector = Icons.Default.Send,
                contentDescription = "Send Button"
            )
```

```
            )
        }
    }
}
}
```

**- ( Viswas , Lokesh , Nandish , Saipavan ) @ ChatConnect**

**>>>>>>Thankyou<<<<<<<**