IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Android Messenger Application Using Flutter

[1]Vijaykumar Dudhanikar, [2]Lakshmikanth, [3]Mohammed Shamal, [4]Pratheek B C Kulal, [5]Omkaresh

[1]Assistant Professor, [2,3,4,5]Student

[1]Department of Computer Science and Engineering, A.J. Institute of Engineering & Technology, Mangaluru, India

[2,3,4,5]Department of Computer Science and Engineering, A.J. Institute of Engineering & Technology, Mangaluru, India

Abstract: Being already integrated into most smartphone users`daily routines, providing easy-to-use solutions for the sharing of rich and context-sensitive data as well as an instant feedback channel, messaging apps make the documentation of one`s information/news use a familiar and convenient experience for participants that can easily be incorporated into their day-to-daylife. An online communication allows the users to communicate with other people in platforms such as iOS. The system developed on android will enable the users to communicate with another users through text messages with the help of internet a fast and convenient way. Considering this,the online communication application must be able share the textsor images or any other files in a faster way with minimum delay or with no delay. Firebase is one of the platforms which providesa real-time database and cloud services which allows the developer to make these applications with ease. Android providesbetter platform to develop various applications for instant messaging compared to other

Index Terms – Social media platform, messenger, cloud messaging, video communication.

I. INTRODUCTION

Mobile applications have completely changed how we engage with one another in recent years, during which time communication has become more crucial than ever. One such technology that has grown commonplace in our everyday lives and allows us to interact with friends, family, and coworkers from anywhere in the globe are messenger programmers. The Flutter framework, which offers a quick and effective approach to create mobile apps for both the Android and iOS platforms, will be discussed in this paper as we examine a messaging application created with it. Users may connect with one another easily thanks to the application's text chat, media sharing, and video calling functions. Users may send and receive text messages in real-time using the text chat function, enabling them to hold brief chats with their connections. Users may send files to their friends using the media sharing tool, making it simpler to share memories or critical information.

No matter where their contacts are located, users may conduct in-person chats with them using the video calling function. This function has become in significance since it enables people to maintain social distance while being connected, notably during the COVID-19 epidemic. Overall, the Flutter-built messaging app offers customers a full suite of communication tools. It is a perfect alternative for those who wish to keep in touch with their loved ones or coworkers since it is simple to use, dependable, and secure. The technical details of the programme will be examined in further detail in this article, along with its efficacy and usefulness. Mobile applications called "messaging apps" are made to make it easier for people to communicate with one another or with groups. They enable real-time text, picture, video, and other file transmission and reception for users. Additionally, messenger apps may offer group chats, voice and video calls, and other features.

Messenger programmers often provide text chat capabilities like typing indications, read receipts, and the capacity to send and receive messages even while disconnected. End-to-end encryption is another feature that many messaging applications offer to make sure that only the sender and receiver may access messages.

Mobile applications called "messaging apps" are made to make it easier for people to communicate with one another or with groups. They enable real-time text, picture, video, and other file transmission and reception for users. Additionally, messenger apps may offer group chats, voice and video calls, and other features.

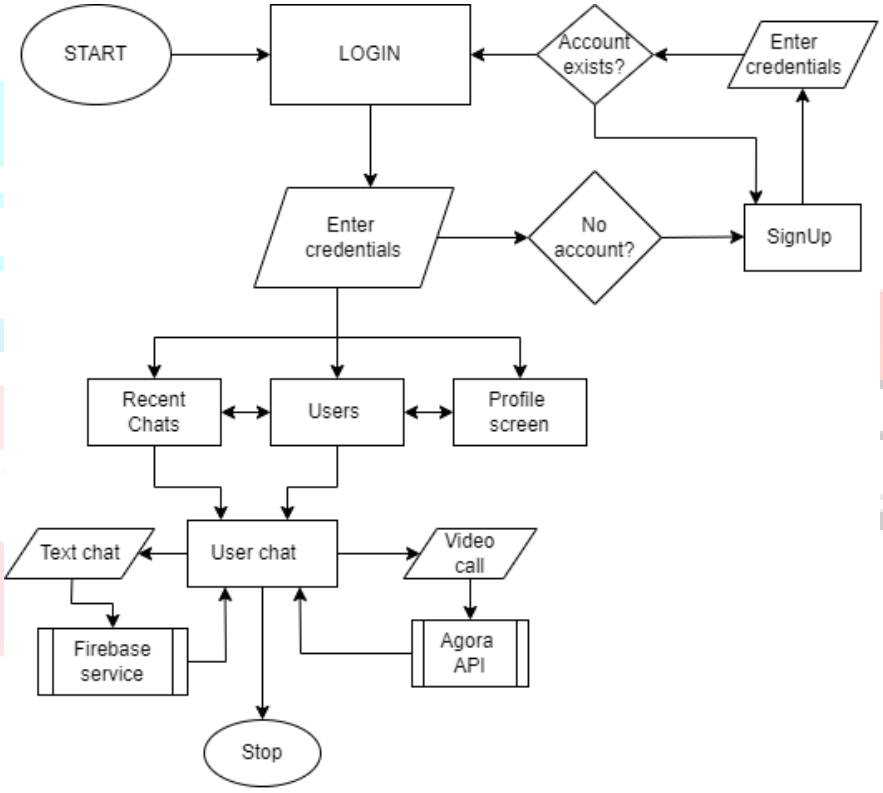
Messenger programmers often provide text chat capabilities like typing indications, read receipts, and the capacity to send and receive messages even while disconnected. End-to-end encryption is another feature that many messaging applications offer to make sure that only the sender and receiver may access messages.

II. METHODOLGY

Chatify messenger is a messaging application that uses Flutter for the frontend and Firebase for the backend. Designing the user interface for the text-chatting capability is the initial phase. Designing the layout and visual components entails determining the necessary components, such as the chat window, message entry box, and user profiles.

After the user interface has been created, the text-chatting feature will be added using Flutter and Firebase as the backend. This entails establishing the chat feature, setting up the database structure, and integrating it with the user interface.

Using Firebase Cloud Messaging, a push notification system can be put in place to guarantee that the messaging feature works in real-time. Users will always be notified of new messages as a result, even when the app is not being used. The next stage is to design and create the messenger chat application utilizing Flutter and Firebase for the backend and AGORA API for video chatting when the requirements have been acquired. This calls for the implementation of the video and chat capabilities, their integration, testing, and app optimization.



III. RESEARCH METHODOLOGY

A. "Developing an End-to-End Secure Chat Application by Noor Sabah, Jamal M. Kadhim and Ban N. Dhannoon "

In this paper [1] An application for client-server chat is the intended use of the suggested architecture. When configuring an application on the client side, a user has the option of choosing between registration and log-in. A message server and a users' server make up the chat server on the server side. server for users that controls users' login information. Users' communications are handled by the message server using Firebase Cloud Messaging (FCM). When the receiver becomes online, the messages are forwarded to him and subsequently removed from the FCM queue. If the recipient is offline, the messages are temporarily retained on the FCM queue for a certain amount of time.

B. "Android Based Instant Messaging Application Using Firebase by Sai Spandhana Reddy Emmadi, Sirisha Potluri"

This paper [2] demonstrates how Building the infrastructure for storing data like video, text, and photographs would be challenging and expensive for a new developer, but firebase offers the platform of cloud storage. Current database: It is a NoSQL database stored in the cloud. Along with cloud services, real-time databases, and authentication, Firebase also offers a service for crash reporting. This service helps Firebase deal with crashes. The software application for the creation of real-time communication services between operators/users is the subject of this study. A chat application is a form of many-to-many communication system where users may communicate with one another.

C. "Design & implementation of real time chat application, by Teena Verma, Dishant Solanki, and Rahul Khandelwal":

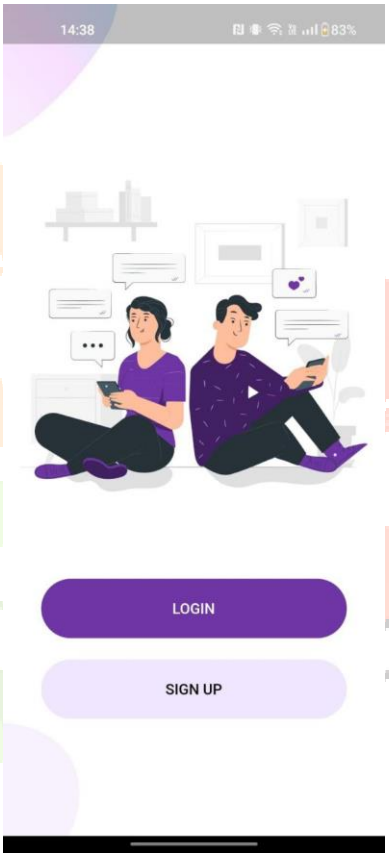
This paper [3] demonstrates both the client side and the server side of the programme are each constructed separately. The server gets messages from clients, but the server is powerless after it receives them. The server transfers data from the server to the client, who may then display the message through their browser, ensuring that everyone in the chat room sees it. By doing this, it is made sure that everyone in the chat room can see the message the client has sent and when it was sent.

D. "Designing a large-scale video chat application":

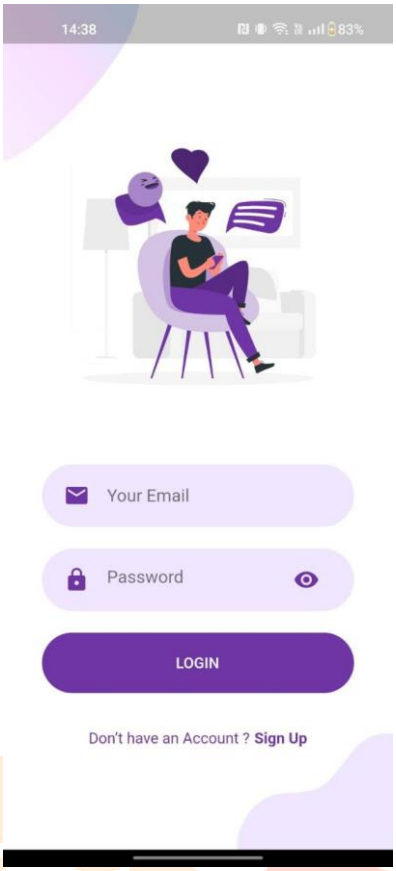
Described in this paper [4] uneven bandwidth sharing is a technique used in video conferencing to reduce the drawbacks of equal bandwidth sharing. It entails designating some users as being more significant than others and allocating them a higher portion of the resources. The simplest kind of uneven bandwidth sharing is Jacobson's video silence suppression, although more complex strategies have been investigated to optimize how much bandwidth each sender needs. They have not been shown useful through user studies.

IV. RESULTS AND DISCUSSION

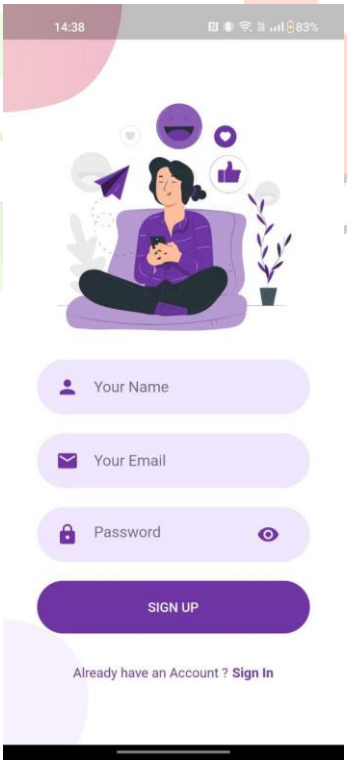
The current chapter deals with the analysis of results of the proposed framework. In this section we'll show the different screens according to our implementation of the project.



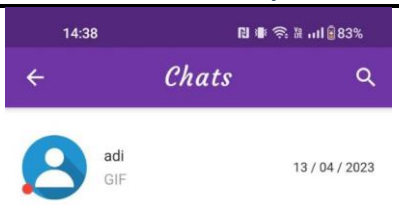
This is the initial landing screen of the project i.e., the first detail of the splash screen. This screen shows the different options that are available for the user i.e. Login and Sign Up .The login page is used by the user who is already having an account in the application and a new user can sign up to the application using Sign Up page.



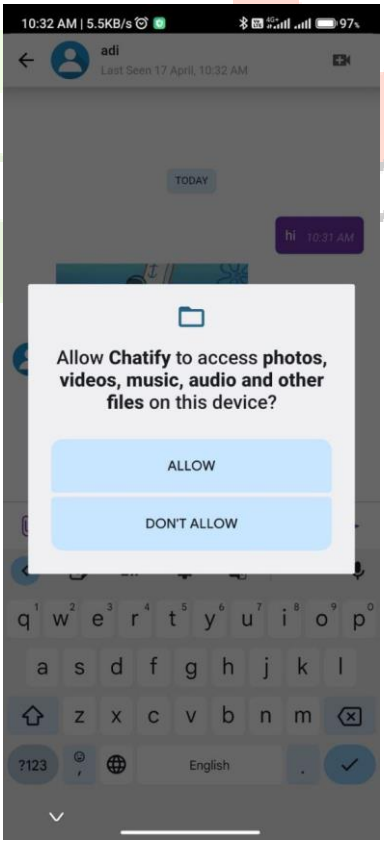
This screen shows the email and password field that the user needs to enter this password can then be used for logging in to the portal, finally after entering the password. If there is no account on entered cred credentials there is a signup option which redirects to Signup screen. Add your name, email and password and press on signup button.



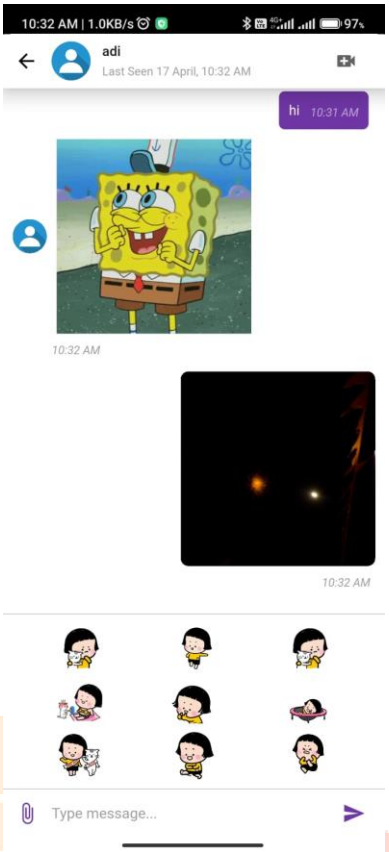
This screen shows the sign-up page where the user can register to the application by providing the credentials like name, email and password and to click on sign up to register to the application.



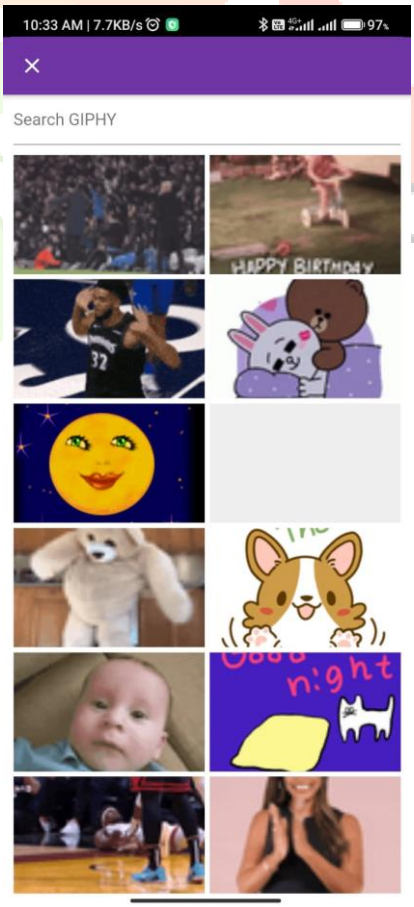
After login we land in the chats screen which shows the list of users, we had chat recently. Here the user can chat with the people who have already have an account in the application.



This screen represents the permission asked by the application in order to perform different operations like send images from the device to the other user.

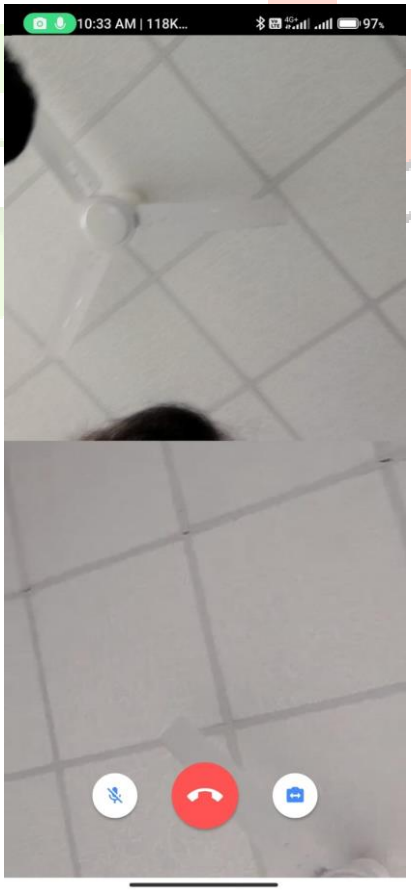


This screen displays the different stickers that are available for the user which is available for the user while chatting with another user. This package is taken from flutter package which makes the conversation more attractive and interesting.



This screen shows the gif's that are available for the user to use which will make the conversation more interesting. The gif's will be available to all users that are using the application.

This screen shows the call interface that the user can avail once the user has logged in to the application and decides to call and have a one-to-one talk with another user to have an more clear and efficient conversation.



This screen represents the audio and video call interface where the user can have an clear and one to one interaction with other user. The user can turn on and off as per the convenience of the user and the requirement.

V. CONCLUSION

In conclusion, a robust, safe, and top-notch user experience may be achieved by building a messenger application utilizing Flutter for the frontend, Firebase for the backend, and AGORA API for video calling. Planning, requirements collecting, user interface design, implementation, real-time messaging, media sharing, video calling, user authentication, testing, and optimization are all part of the development process. Cloud-based data storage and synchronization, real-time communication, and secure user authentication are just a few advantages that Firebase and AGORA API provide. Developers may produce a messaging application that fulfils the demands of their target market and offers a top-notch user experience by using a structured process and the advantages of these technologies.

VI. REFERENCES

[1] Developing an End-to-End Secure Chat Application by Noor Sabah, Jamal M. Kadhim and Ban N. Dhannoon
[2] Android Based Instant Messaging Application Using Firebase by Sai Spandhana Reddy Emmadi, Sirisha Potluri
[3] Design & implementation of real time chat application, by Teena Verma, Dishant Solanki, and Rahul Khandelwal
[4] Designing a large-scale video chat application, Jeremiah Scholl, John D. McCarthy, Angela Sasse,Peter Parnes
[5] https://www.agora.io/en/products/video-call/
[6] https://flutter.dev/ - Flutter Website
[7] https://firebase.google.com/ - Firebase Website
[8] https://www.figma.com/ - Figma
[9] https://flutterflow.io/ - FlutterFlow