Teams Clone Engage 2021

|  |  |
| --- | --- |
| **Owner(s): Kavya Kandhway** | **Status:** Review |
| **Approvers**: Subhash Konda | **Last Updated:** 2021-07-11 |

Contents

[Summary 1](#_Toc76820708)

[Objective 1](#_Toc76820709)

[Success criteria 1](#_Toc76820710)

[Design 2](#_Toc76820711)

[Overview 2](#_Toc76820712)

[Experience 3](#_Toc76820713)

[Data flow 3](#_Toc76820714)

[Testing (Required) 4](#_Toc76820715)

[Timeline 4](#_Toc76820716)

[Appendix 4](#_Toc76820717)

# Summary

Prototype for Microsoft Engage 2021 challenge as individual contributor using Agile methodology.

# Objective

To develop Teams Clone with mandatory feature of 1:1 video calling. The solution could be either a mobile based application or web based.

# Success criteria

Fully functional prototype with at least **one mandatory functionality** - a minimum of two participants should be able connect with each other using your product to have a video conversation.

Using Agile Methodology, we should be able to divide the work in sprints.

In “adapt” phase a surprise feature was disclosed that is chatting within a video call which could be continued when the call ends.

# Design

## Overview

This is a mobile application developed using flutter and implemented various features using different flutter open source packages available.

Features includes

* Authentication
  + Email login/ signup
  + Google login/signup
* 1:1 video calling
  + User1 can call User2 while chatting with each other.
  + User2 will receive a pickup screen letting him/her to either decline the call or receive it.
  + Both the users can chat within the video call as well and can continue the chat even after the call ends.
  + Calling history is also saved in the database.
* Group video calling (more than 2 users)
  + User1 can create a room and can share the link/code to other users.
  + Code can be shared either within the app or to the other apps.
* Chatting, sharing images
  + User1 can chat with User2 using the app.
  + Both the users can share images either from gallery or by clicking the picture.
  + Online Dot Indicator- if the user is online a green dot is displayed beside his/her name.

Packages and APIs

* For authenticating the user
  + Used firebase authentication flutter package and implemented email- password sign in and google sign in.
  + Assigns each user a unique uid which is used as key for storing the user’s details in the database.
* For implementing the video calling feature
  + Used Agora.io Engine in flutter for implementing various features such as video rendering, mic on/off, video on/off, switch camera.
* Database
  + Used Firestore database for storing the data such as user’s information, contacts, chats, images shared by the users.
* Local Database
  + Used Hive Database for storing the call history of the user.
* Used various other packages for improving the UI of the application.

## Experience

As this is a communication based application let’s suppose two users User1 and User2 registers in the application either via google sign in or email password sign in. Ensure that the email entered while registering is a correct and valid email id.

One both the users login, they will be displayed Create Team/Join Team screen. The User1 need to open the chat screen and search for the name or email of User2 and text him/her. As soon as the user1 texts, the chat screen of user2 is updated. Now both can chat with each other and share images.

Now User1 calls User2 by tapping on the video calling icon on top right. User2 receives a pickup screen with receive/reject options. According to the user2’s choice, the call history is updated for both the users.

If the User2 accepts the call then both of them is connected via video call with video and mic on/off option. They both can tap on the chat icon at bottom right and can chat within the video call.

Once the call is cut, they both can continue chatting and the messages inside the video call is also retained.

For the group calling feature, User1 can create a room and share the code/link with any on his contacts lets say User2, User3, User4. All these three users will receive a message from User1 with a link to join the group video call. They can simply tap onto the button and join the call.

The user can log out from the app by tapping on profile photo at top left.

## Data flow

Once the user login through the application, the data is fetched from google/email sign in that firebase provides and is saved in our “user” database with unique uid (provided by firebase) as key.

When the user searches for any other user, he/she is displayed all the users in our database according to his search keywords.

Message exchange flow.

* When User1 messages user2, the message is stored in a new database “messages”. Inside the collection "messages” , there are documents for all the user registered in the database, and inside each user document, there is a list of all the contacts of that particular user. Inside each contact there are message objects which includes information related to a message such as time, content, type, sender’s and receiver’s uid.
* Once all the messages are fetched, it is displayed on the screen with 2 layouts, either sender(on right) and receiver(on left).

Video Calling Dataflow

* Once a call is made, a call object is created and is stored in the “calls” database. In the other end, the application receives a live stream of data, and when it is not null and the call object’s receiver id matches with the current user’s id, he/she gets a pick up screen. As soon as the call is disconnected, the call object is deleted from the database.
* According to the call status (received, missed, called), the data is stored in tabular manner in a local database using Hive. To make the local accessible only to the specified user, the key of the table is same as the uid we used above in firestore database.

## Testing (Required)

The app is tested for 1:1 video calls, chatting and group video calls over the internet. Downloading the apk of the application, a user can use the app and its features.

# Timeline

4 weeks timeline is provided. Divided the work in 4 sprints and managed them using Azure DevOps.

<https://dev.azure.com/kandhwaykavya/KavyaKandhway%20Teams%20Clone>

# Appendix

Link to packages and APIs used

* [agora\_rtc\_engine | Flutter Package (pub.dev)](https://pub.dev/packages/agora_rtc_engine)
* [cloud\_firestore | Flutter Package (pub.dev)](https://pub.dev/packages/cloud_firestore)
* [firebase\_auth | Flutter Package (pub.dev)](https://pub.dev/packages/firebase_auth)
* [google\_sign\_in | Flutter Package (pub.dev)](https://pub.dev/packages/google_sign_in)
* [firebase\_storage | Flutter Package (pub.dev)](https://pub.dev/packages/firebase_storage)
* [flutter\_share | Flutter Package (pub.dev)](https://pub.dev/packages/flutter_share)
* [hive | Dart Package (pub.dev)](https://pub.dev/packages/hive)
* [google\_fonts | Flutter Package (pub.dev)](https://pub.dev/packages/google_fonts)
* [image\_picker | Flutter Package (pub.dev)](https://pub.dev/packages/image_picker)