Senior Design Project

Project short-name: Project Facera

High Level Design Report

Taha Khurram, Umer Shamaan, Zeynep Berfin Gökalp, Emil Alizada, Verdiyev Zulfugar

Supervisor: Dr. Ayşegül Dündar

Jury Members: Dr. Can Alkan and Dr. Cigdem Gunduz-Demir

High Level Design Report December 27, 2020

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Table of Contents

1 Introduction

AR (short for augmented reality) is a technique for broadening interactive experience in a natural environment. It gives endless opportunities for manipulating the perception of reality. With today's increasing technological capabilities, AR is becoming more useful and popular. Its potential to give intuitive user experience provides developers with many possibilities to create effective and useful platforms especially in the field of entertainment.

We decided to create a video messaging platform that can enhance the user engagement and create a more memorable experience by utilizing aspects of the AR. The main purpose of our platform is to break through the conventional methods of video messaging and harness the power of AR, to make video messaging much more entertaining. This will also open in a new genre of video messaging (In future implementation video calling as well) which will feature 3D models in an interactive environment through AR instead of the conventional video message which enables users to create more realistic and feature-rich facial animation. The primary goal of the platform is to detect and track face expressions in a video-message in which will then be mimicked by a 3D avatar placed in the frame of the camera using AR.

1.1 Purpose of the system

Facera is a mobile application aimed at creating a more enjoyable and interactive user experience in terms of chatting or video messaging. The main purpose of this application is to implement an AR element for its users where they can use their 3D avatars to create a more interactive environment. By using various high quality AI techniques to detect facial expressions, these expressions will be mapped on the 3D object that is being placed in AR.

Object-oriented design techniques and architecture design will be used in order to create this application to provide ease in maintainability, and implementation of the project.

1.2 Design goals

In order for our application to function as desired, the following goals have been taken into consideration for the designing of our project.

1.2.1 Robustness

As we will be using a database, Firebase [1], to store private credentials, the application will provide a secure environment where the user's credentials are kept securely. Moreover, the database will keep

the data safe incase of any unforeseen circumstances to prevent any data loss, such as in the cases of connection loss or server failure a backup database will be maintained. Similarly, the application should not fail if the server is not responding and the user should be able to view previously sent videos or send new videos which will be processed and sent to the receiver later when the servers are back up.

1.2.2 Usability

As the code involved in analysing facial expressions will run on the mobile device, it will result in no dependency on the servers being online or offline and allow for higher usability. Similarly, the application should also be usable regardless of the connection or the server status. both online and offline features should work in tandem to provide stable usability for the user.

1.2.3 Performance

Our application relies on the accuracy of the facial expressions data collected and mapped onto 3D objects, therefore it is crucial to maintain a high level of accuracy in collecting and mapping this data. The facial expressions will be analysed on the recorder's device and the mapping will be done on the receiver's device. This will allow for stable and high accuracy, as well as reducing the load on the mobile device leading to more stability in the application's performance.

1.2.4 Availability

It is important for the system to be active and running throughout the facial recognition process. The database server should not break down and the mobile application shouldn't completely crash due to experiencing any failures or errors.

1.2.5 Scalability

Since we are expecting progressively enlarging popularity of the application, the system should be able to maintain its functionality with the increasing number of users. This expected increase in demand should also not affect the system's efficiency.

1.3 Definitions, acronyms, and abbreviations

Terms and abbreviations are stated below to facilitate understanding of the content in the sections.

AR: Shortened form of "Augmented Reality". AR is reached by covering computer graphics on live camera footage and altering these graphics with the camera movements in a realistic way.

Client: A user of the application. The client-side of the Facera represents the interface in which a user can make video call editings.

Server: Back-end of the application, where the information about accounts is stored.

1.4 Overview

Facera will be an application that is developed for Android platform made especially for smartphones. The application will reframe old-style video messaging by implementing the aspect of AR in normal day to day video messaging. It will have the ability to text or send video messages between users using AR technology. The 3D model mimicking the recorded facial expressions will give an interesting twist to the conventional video messaging services. The users will be able to select an avatar from a variety of preinstalled 3D avatars.

After downloading and initiating the application, the sign-up or sign-in screen is displayed for the user. The users will be directed to the homepage once they login. There they will have several options. They may view their profile page and change their information as required. Furthermore, they will be able to go to the chat page to start a conversation with their friends. There they can either send a text message, including emojis and stickers, or record a video message to record their face to be used on the avatar seen by the receiver on the other end. Moreover, the users will be able to search for other users to send requests to or view their notifications to see information such as received messages or friend requests from other users. Once the users enter a chat they will have the option to either send/recieve a text message or a recorded video. When they record the video, a video message button will be used to open the camera. The application will be asked for the user's permission before using the camera. The front camera will be used to record the video of one user. The application will start recording the video message, if the user releases the record button this will end the recording. The application pauses recording the video message as the user's face can not be detected. Users' facial expressions will be sent to the back end for analysis and recorded by the application. Furthermore, the user will have the option to save the received video to the gallery.

The users will also be able to view video messages from other users and place AR objects to perform animations. The data of the received video message will then be used by the second user who will use a button to place a 3D model in their camera frame which will be selected from an array of 3D models avatars. The 3D avatar will be resized and moved by dragging it across the screen. The application will open the back camera and try to locate a suitable surface for placing the avatar in AR. The 3D model can be placed on any surface available in the camera frame such as a desk or a bed. The 3D model will mimic the facial expressions received allowing for the implementation of AR in video

messaging. The application will place the facial expressions data on the avatar's face which results in the facial expressions captured in the received video being displayed on the avatar's face along with the audio of the video. If the user presses the quit button or if the video ends, the user will be taken back to the chat menu. The user will also be able to view and search for other users' profiles to send friend requests or accept/decline the received requests. It allows users to call and message each other.

2 Current software architecture (if any)

Apart from normal texting, or normal video messaging apps such as Whatsapp, Skype, Google Duo, Facebook Messenger, we found no app working similar to ours in the sense of using AR in video messaging. Although there are no direct competitors to our project, the similar technologies on the market are defined below.

WhatsApp video calls:

- Very popular nowadays for personal use.
- Does not provide any entertainment like AR masks and filters.
- Implemented only for mobile phone users.
- Can be used for group calls.

Skype:

- Very popular for business communications.
- Provide a lot of technical features like live translation and screen sharing.
- No implementation of an AR.
- Mainly used on desktop computers for business purposes.
- Can be used for big conferences, including lots of people.

Facebook Messenger Video calls:

- Not popular for personal use.
- Can be used for group calls.
- AR masks are implemented.
- Can be used only on mobile phones.

Google Duo:

- Mobile application.
- AR masks are implemented.

- Maximum people in a conversation: 2.
- Not popular.

3 Proposed software architecture

3.1 Overview

In this part of the report we discuss the software architecture of our system. We provide subsystem decomposition of our application as well as their respective services. We discuss each subsystem and its role on our architectural pattern. In our report we present diagrams to visualize and have a clear understanding of our subsystem decomposition. Section 3.3 will illustrate a wider perspective for our hardware components and their roles. Section 3.4 will provide information regarding our storage systems in order to record and send messages as well as to keep user information. Sections 3.5 presents security management for user access.

3.2 Subsystem decomposition

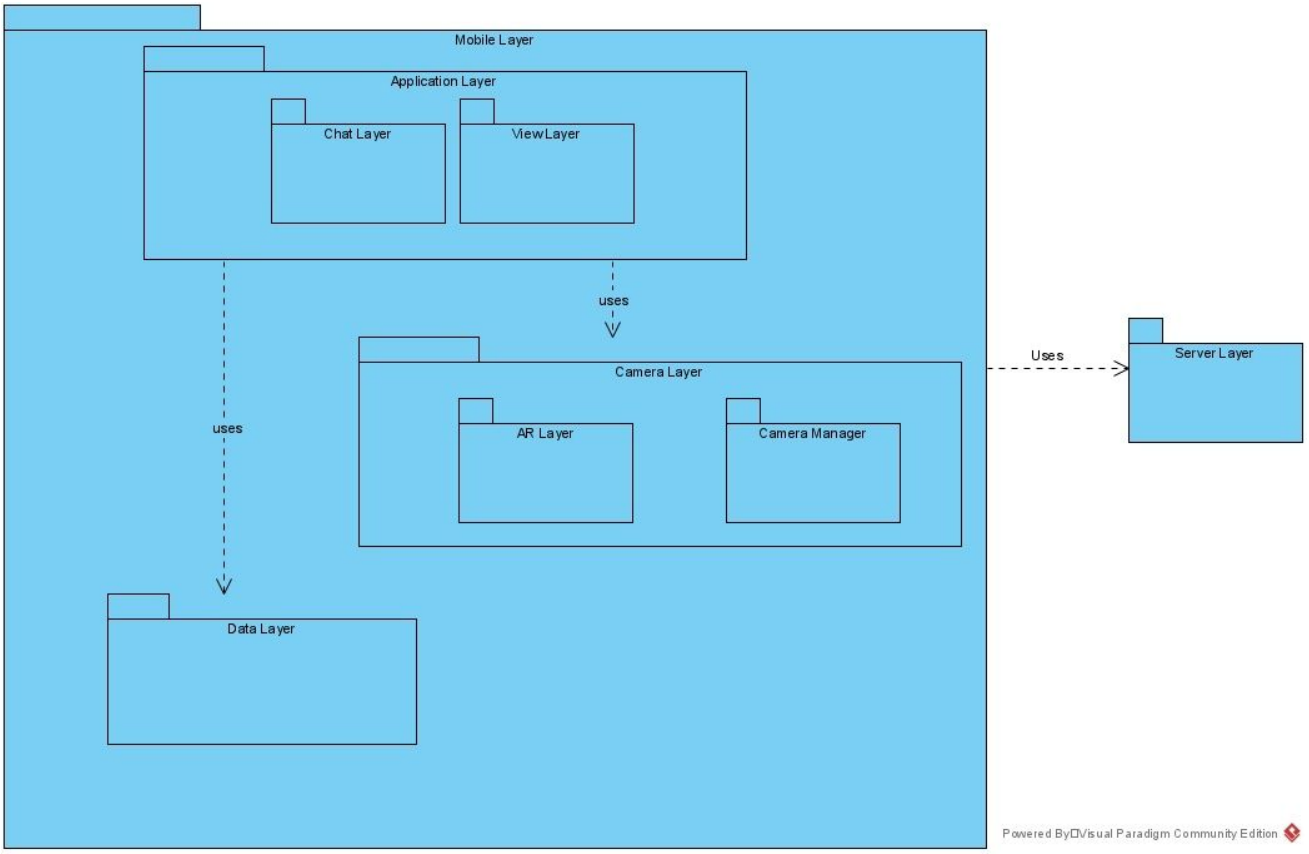
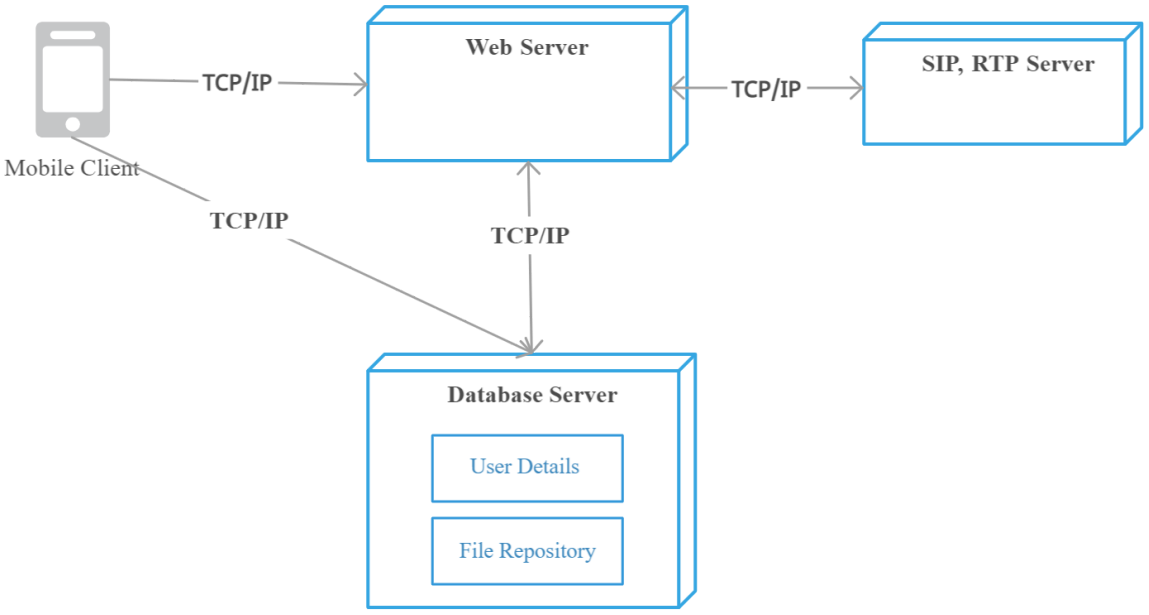


Figure 1 - Subsystem Decomposition

In our implementation, we intend to use a high level of modularity in order to make the development process more efficient. This way it will be simpler to divide the work among the group members. These layers will be maintained as independent modules, and updating or modifying one module will not negatively impact the other. As can be seen from **Figure 1**, the mobile layer is the most comprehensive layer. It connects with the server layer in order to use the Firebase API for authentication and real-time database. Other Google Cloud SDK [2] packages will also be included in the server layer to handle transmission of facial and chat data, account recovery, and (optional) location tracking.

The mobile layer contains an application layer which contains two other Chat and View Layers within it. The chat layer will handle the process of sending and receiving text based messages. The View layer will handle the overall graphical user interface of the application. The Camera layer contained within the application layer will handle the overall process of enabling users to record and play video messages. The Camera manager inside the Camera Layer will handle the recording of videos and analysing the facial expressions. The AR layer will handle the mapping of facial expressions to the avatar and avatar animation. The Data layer inside the application layer will handle the basic local storage.

3.3 Hardware/software mapping



The hardware software mapping diagram depicts the overall scheme of how our system will be implemented in terms of hardware and software interaction. Nodes and connections display the dependencies among the system components. The diagram illustrates the hardware that will be implemented and shows dependencies between them. The client's device is dependent on 2 different servers. We are planning to utilize AWS for heavy computations of our app. The database server will be responsible for the user data and chats. SIP/RTP server is reserved for future expansion of the app to the real time calling.

3.4 Persistent data management

One important aspect to be stored is the user accounts. Various information can be related to a user for example user information (eg. usernames, passwords, emails, etc.). Firebase allows the ability to store the user-id, emails and passwords independently from the user defined databases. It automatically builds the database for that information and provides an API to automatically signup and authenticate the users. That way the user instances are independent from the firebase Authentication Instances, so we can have several references to different users within the same context and still can call any of their methods. It also provides the API to signup users directly through their Gmail and Facebook accounts so we will use that feature as well.

Additionally, another important data processed by our app is the user's facial expression data that will be generated locally by the mobile phone's processors (without using any auxiliary servers). Furthermore, this functionality of the application will have strict time constraint in order to provide a better response time. As such, the data will be stored immediately to the local storage after it arrives from the transmitting end temporarily. The recorded videos will also be stored in the local storage. No backup will be provided in case the user deletes the videos from the phone's gallery. Another data that will be stored locally are the chats. When the users log in, they should have immediate access to the previous chats. The total overhead of importing them from the real time database is too high and inconvenient for the users. We will use the sqlite packages provided by the IOS and android OS to store such information locally and securely.

Moreover, to store additional user data such as profile pictures, avatars, settings and backups for conversations, the Firebase real time database system will be used. It is an efficient NoSql database that lets the application store and sync data between the users in real time. It automatically integrates the database with the Firebase Authentication in order to save the developers trouble of adding an additional security layer to separate the users' personal data.

3.5 Access control and security

Multiple users will be able to use this application at the same time. Hence, it is necessary to separate each user's data. As mentioned before, the real time database in Firebase (which will be used for this app) integrates with Firebase authentication to provide simple authentication. It allows user access based on identity.

Another important security concern which the application raises is the secure transmission of the users facial data. Before transmission, the application will encrypt the facial data before transmission using a two layer encryption mechanism. Furthermore, the application will not be given permission to store that data into the phone. It will also not store any statistical data or logs of the user behaviour. During the case of analysis failure during the recording (which might occur due to the users moving their face away from the camera or an application crash), the data will be destroyed immediately. Only after the recording has been completed, the data will be compiled, analyzed, encrypted and transmitted. Furthermore, the chats will also be encrypted before being transmitted or stored in the database.

Another security issue with the usage of the application is the access of the camera. The camera will be only used in recording the facial expressions of the sender and the receiver playing it in order to extrapolate the expressions onto the avatar. The storage of the played videos by the receiver will be maintained with their complete consent.

In case of the users forgetting their passwords, a discrete key will be sent to their personal email in order for them to change it. Only the users will be allowed to restore their passwords. In case of them somehow forgetting their email, they will not be allowed to change it and will have to create a new account. Furthermore, their login locations will be recorded and in case of suspicious login activity, for example their accounts being logged into from a different location, they will be sent an email in order to notify them about it. The location logs will be kept securely in the Firebase database and only the admins will have access to them with the users consent. The users will have the option to turn off location tracking.

3.6 Global software control

Flow of events in the application starts with user login and sign-up. If the user already has an account he/she tries to login. All the requests that are made to control the login stage are managed through Google's Firebase.

To provide model driven control for chat, we use Google Public DNS. For heavy computations we are going to use AWS Cloud Computing Server , which will be managed by the application layer. The database server will encompass the user data and chats. GDrive APIs will be used to maintain control for video transmission. Our model driven control system has access to both the front and rear camera, which is going to execute data transmissions based on these inputs. Videos will be recorded to the external storage system and processing will be done through the application layer.

3.7 Boundary conditions

The application has three main categories in terms of boundary conditions. The categories have been elaborated below:

3.7.1 Initialization

As Facera is going to be an android and iOS application, the user will have to download the application from the Google play store and the Apple play store, after installing the application the user will have to sign up to use the application which is a one time process.

Next the user has to complete the login procedure which will allow the user access to the main features of the application. From this point onwards, the user will be able to search, add and remove friends from their friends list as well as initiate or continue conversations with one of these friends users.

This is a one time process and the next time the user opens the application (after quitting) they just need to login to gain access to all of these features.

3.7.2 Termination

As most chat based applications work, our application too will run in the background and notify the user incase if they receive a message/ video message or a friend request. In order to exit the application the user will have to log out/quit the application to stop it from running.

If the user opts to stay logged in they can simply open the application and continue off from where they left without the need for relogging as the application will retain their login status.

If the user quits/logs out from the application, the next time they open the application they will have to go through the login procedure again in order to gain access to the application features.

3.7.3 Failure

As the application requires the internet to login, if the user logs out and does not have access to the internet they will not be able to view previously received messages. Moreover, the application will not allow them to send video/ normal messages as well when logged out and offline.

As a result of a failure stemming from being logged out and offline, the user will be unable to use the application for that period of time. This includes being unable to access received messages and/or sending new messages that will be delivered later after connecting to a stable internet connection.

4 Subsystem decomposition

4.1 Application Layer

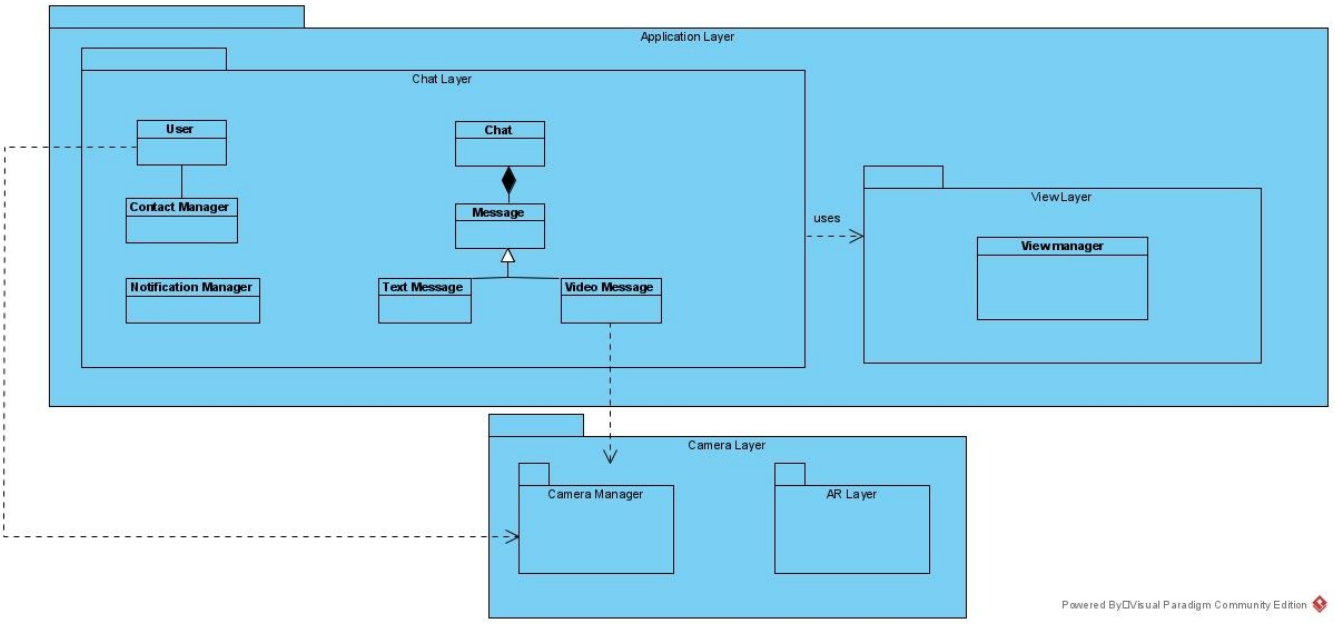


Figure 3 - Application Layer

The application layer is the main layer of our program, this is where the main functionalities of our application are performed.

Chat Layer

The Chat Layer contains the chat class that has the message feature, the video message and text message classes are its subclasses. These classes are implemented in the application to send and receive messages between users. The video messages are dependent on the Camera manager as the front camera will be used to record the video to be sent and the back camera is used to place the 3D model in the frame of the back camera using AR.

The notification manager is used to send the user notifications regarding a new friend request, or a new text/ video message.

The user class contains the user data such as their name, id, email, and password. The user class calls upon the contact manager class to search, and add other users to their friends list.

View Layer

The view layer consists of the view manager that implements the user interface(UI) for the mobile application and deals with different pages in the mobile application.

4.2 Camera Layer

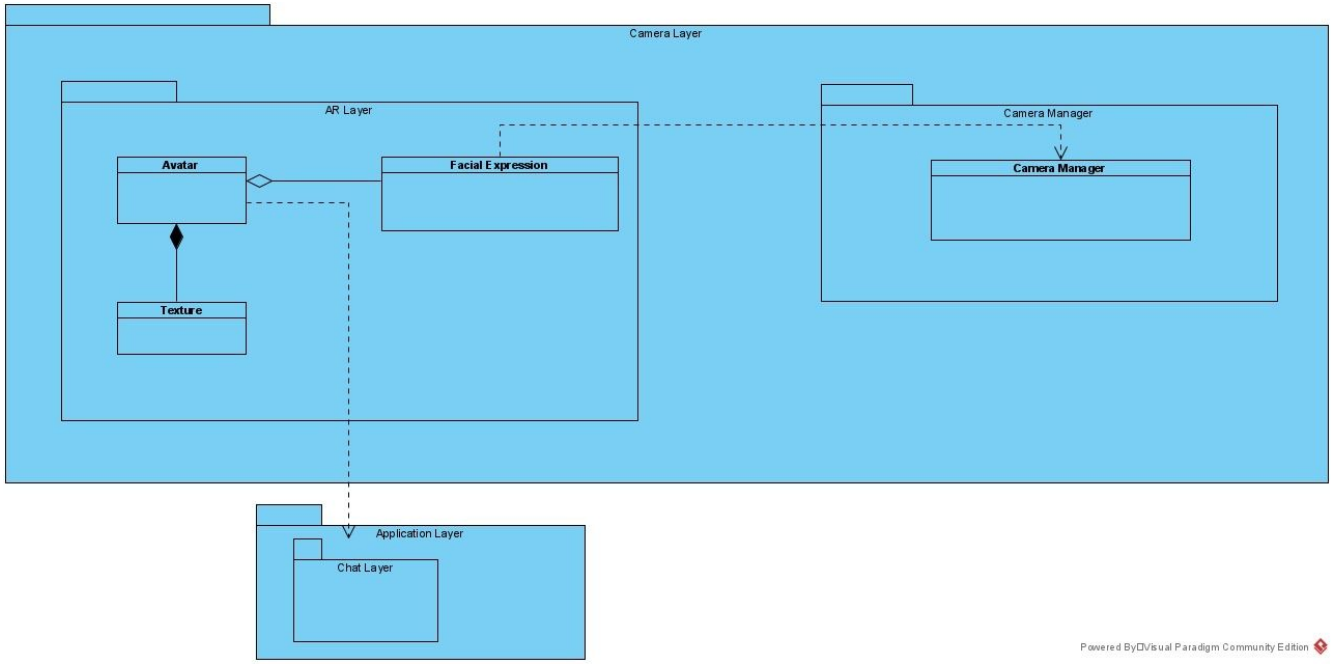


Figure 4 - Camera Layer

The Camera Layer consists of the AR layer and the Camera Manger.

Camera Manager

The camera manager contains the camera manager class, this class is used to access and deal with the front and back cameras of the mobile application that will be used to record and view video messages.

AR Layer

The AR layer deals with the 3D avatar, its customization using the Texture class and the facial expressions that are to be mapped onto the avatar. The facial expressions class has a link with the

Camera manager as the expressions will be mapped on the avatar while it is placed in the frame of the camera. Moreover, the avatar has a link with the chat layer of the Application Layer as the Avatar data will be used in video message class.

4.3 Data Layer

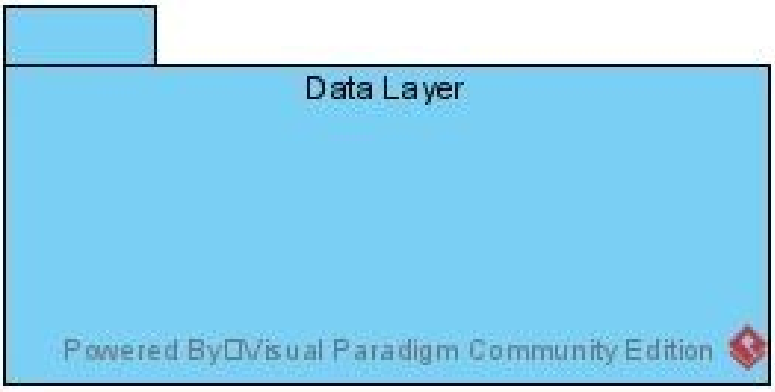


Figure 5 - Data Layer

The data layer will be responsible for the local storage and retrieval of chats and videos in the phone. The SQLi packages provided by IOS and Android Operating Systems will be used for the purpose.

4.4 Server Layer

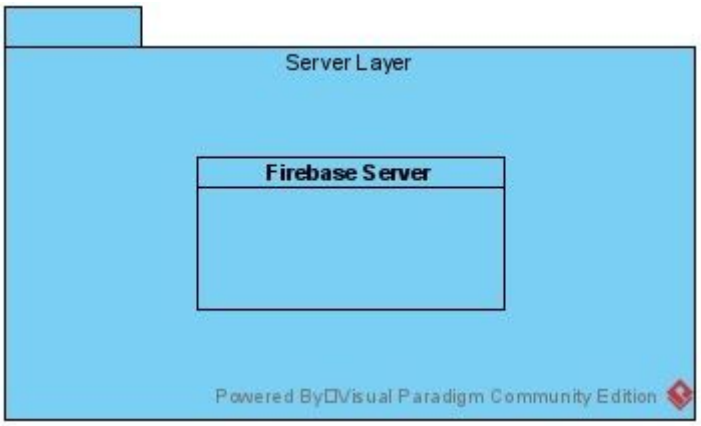


Figure 6 - Server Layer

The Server layer contains the Firebase Server class, this is where the application stores the user data to authenticate and allow for login, signup and chatting. The Firebase server is used as it allows for more security and reliability.

5 Consideration of Various Factors in Engineering Design

Our application is designed to provide entertainment as a means of social service. On a large basis, an application may prove to be useful or harmful the society it is meant to serve. Here we will start discussing various factors that may address certain design concerns which are expected to be challenging from ethical and social perspectives. Main issues encompass data privacy, health and distraction, cultural differences and market competition.

5.1 Data Privacy

Our application due to its communicational features, has a potential to pose an intermediate threat to user privacy. These communicational features include video transmission and text messaging. To eliminate all the risks that would otherwise cause a discontent for the user experience, we will implement safe methods. Video transmission features include video messaging and face recognition. We will use public APIs to add face recognition to our project. We will select these APIs carefully and selectively, to avoid any potential concerns regarding image privacy. To avoid user's data getting into the wrong hands, we will deploy trustable servers (most probably Google Public DNS). Similar technology will be used for text messaging as well. Chat connections will be end-to-end encrypted.

5.2 Effect Of Cultural Differences

Our application will have two main features that concerns one's individuality. Users will only be able to select avatars that are in the catalog. In order to avoid gender bias we will upload two avatars that have the same characteristics, only with different sex. To avoid racial bias we will also multiply avatars with the same characteristics, with different recognized races.

Second main feature is its computer vision design. We will use pretrained models in order to integrate face recognition features to our application. We will double check our models and make sure that face recognition characteristics of our application are fair.

Another ethical concern is undesired messages. Every user will be allowed to block certain individuals who are bothering with calls. We will also implement additional features in order to stop users from messaging sensitive content. Therefore we will have report features later on.

5.3 Health and Distraction Concerns

Our application, as well as other social connection applications, has a risk of user obsession. Fortunately, the functionality of our application solves this issue on its own. It has lower probability to cause distraction since the user experience will be temporary. Usage of our application will be preplanned, therefore no risk of distraction. Due to temporary experience, we expect negligible health risks.

5.4 Evaluation of the Constraints

All these factors affecting our design strategy are worth mentioning, but they do not have the same priority levels. In this table we display our priorities when we will manage our design strategy.

	Effect level	Effect
Data Privacy	7	May lead to judicial process and as a result shut down of the application
Cultural Differences	10	May be subjected to criticism by the audience and therefore shrinkage of its popularity
Health and Distraction Concerns	4	May lead to the unsatisfactory user experience.

6 Teamwork Details

6.1 Contributing and functioning effectively on the team

To maximize the team's ability to work together, we have shared the same purpose, finishing the project on time with seperation of the workload. We have discussed the duties to be fulfilled, all team members are provided with collective responsibilities to be held on the project. Thus, all project members had a clear understanding of their responsibility and worked more effectively on the project. In this manner, we have resolved the conflicts that could occur when a team member overcompensates for anyone else or inactive at some point. We have set measurable goals for each task and the necessary tasks will be maintained considering the deadlines to drive success. We have formed small teams for these packages and also we have assigned and reviewed these tasks respectively. Large parts of work packages are separated according to weight of the work, to have a fair workload.

6.2 Helping creating a collaborative and inclusive environment

To create a collaborative environment, the first approach that we have is to be realistic about our strengths and weaknesses so build the project around the individuals who do complement each other. By using this approach, we have easily decided which team members contribute to which part and combine them with the suitable tasks which match their strengths. One team member can be able to compensate the other members to gain more momentum in progress and a proper equal teamwork. We made video conferences through Zoom as a decision-maker and to collaborate better. We have created a judgement-free environment so that expressing new ideas and discussion from team members are welcomed.

6.3 Taking lead role and sharing leadership on the team

To maximize the team effectiveness and better organizational performance, we have considered a shared leadership approach. We have splitted the project into work tasks to let all of the team members be a leader in those project tasks to encourage every team member in leadership. This approach also leads to participation in decision-making and having constructive discussion within all team members.

7 Glossary

API: a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service.

Artificial Intelligence: the theory and development of computer systems able to perform tasks normally requiring human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages.

Augmented reality (AR): is the real-time use of information in the form of text, graphics, audio, and other virtual enhancements integrated with real-world objects.

Expo: is a framework and a platform for universal React applications. It is a set of tools and services built around React Native and native platforms that help you develop, build, deploy, and quickly iterate on iOS, Android, and web apps from the same JavaScript/TypeScript codebase.

Firebase: Firebase is a Backend-as-a-Service — BaaS — that started as a YC11 startup and grew up into a next-generation app-development platform on Google Cloud Platform. Firebase is your server, your API and your datastore, all written so generically that you can modify it to suit most needs.

Mobile app: A mobile application, also referred to as a mobile app or simply an app, is a computer program or software application designed to run on a mobile device such as a phone, tablet, or watch.

React Native: React Native is a JavaScript framework for writing real, natively rendering mobile applications for iOS and Android. It's based on React, Facebook's JavaScript library for building user interfaces, but instead of targeting the browser, it targets mobile platforms.

Software development kit: A software development kit is a collection of software development tools in one installable package. They facilitate the creation of applications by having a compiler, debugger and perhaps a software framework. They are normally specific to a hardware platform and operating system combination.

8 References

[1] *Google*. [Online]. Available: https://firebase.google.com/. [Accessed: 27-Dec-2020].

[2] “Cloud SDK Command Line Tools | Cloud SDK: Command Line Interface,” *Google*. [Online]. Available: https://cloud.google.com/sdk/?utm_source=google. [Accessed: 27-Dec-2020].

[3] *React Native*. [Online]. Available: https://reactnative.dev/. [Accessed: 27-Dec-2020].

[4] *Expo*. [Online]. Available: https://expo.io/. [Accessed: 27-Dec-2020].

[5]F. Vázquez, “3D Face Reconstruction with Position Map Regression Networks,” *Medium*, 05-Feb-2020. [Online]. Available: https://heartbeat.fritz.ai/3d-face-reconstruction-with-position-map-regression-networks-36f0ac2d3ef1. [Accessed: 27-Dec-2020].