Akdeniz University

Department of Computer Engineering

Senior Project I

Final Report

Analysis and System Design

Group Members:

Ravan SADIGLI

20160807005

Table of Contents

1. Introduction

The application is not intended to be like any ordinary chat application but rather aims to hold its place in the market with a clean user interface, user-friendly and solid app structure over a secure network. The application will provide a modern approach to chat applications using the new technologies for the community. The UI of the application will provide a familiar structure to the users because the UI mostly will be influenced by the already existing chat application, meaning that users already intuitively know how to use the functionality of the application. The implementation of the application will be built on these technologies: Kotlin, Firebase, SQLite, RxJava, Dagger, MVVM, REST API, Tensorflow and etc. where Kotlin for application structure and user interface tools, Firebase for the central network database like sending, receiving text messages, image, files, and etc., SQLite for the local database, RxJava for the reactive programming, Dagger for the dependency injection, MVVM for software architecture pattern, API for the communication with the database, Tensorflow for applying machine learning and etc. End-to-end encryption algorithms will be used to establish secure communication. Starting any application or service has many issues, but one of the main issues is what tool, language, stack or framework we are going to build the service or application on. Building a real-time application is about slow delayed message delivery, which means latency, the size of data transfer over the network should be as low as possible. Therefore, the main purpose of the application is to provide users with an efficient and smooth product without no latency.

Our applications enable users to communicate with each other and communicate securely by using new technologies. We will provide a great app for people to communicate with each other that they know using their contact lists whether they give permissions. The main functionality of the application gives the opportunity to the user. These are described below:

- send and receive text messages
- sending images and files
- scheduling messages
- sending location
- sending voice recording
- voice and video calling with end-to-end encryption

Another functionality is that it offers features such as taking photos, adding filters, and editing images. We also mentioned earlier that we will be implementing machine learning applications such as image recognition and sentiment analysis. For the profile picture, the application will force to the user set the profile picture as a real picture of themselves. Some machine learning applications will be applied to our applications. Some of them described below:

- *Sentiment Analysis*

  Sometimes we need to know what mood the person we are talking to is in. And when we're texting online, sometimes we don't know that person's feelings. This can lead to some misunderstandings when talking over the internet. We want to avoid these situations by using sentiment analysis. Of course, some people may not like this feature, it may disturb others. This may cause them to stop using our app. Therefore, those who do not like this feature will be able to turn off this feature from settings such as last seen on WhatsApp.

- *Face Recognition*

  A profile picture makes a first impression. The profile picture is an important element of one's online presence. It conveys messages in every situation and can influence the willingness of others to communicate with the person. And the profile picture can let new people see that the person is a friendly person. We want to avoid these situations by using face recognition. And, this picture will also affect sentiment analysis result. For example, if a person is smiling in the picture, it will affect their mood.

- *Spam filtering*

  In some cases, advertising messages are received. Somehow they find their cell phone numbers and this causes people to receive spam and annoying messages. We want to avoid these situations by using spam filtering.

- *Fraud Detection*

  Fraud detection is one of the important ml projects. Sometimes, people can send sensitive card information to someone they don't know while texting over the internet. We want to avoid these situations by using fraud detection.

These are some of the ml project that will be implemented in our application. As we mentioned above, we do not want to be like an ordinary chat application, we want to apply new technologies to our product and launch it in a way that meets today's requirements.

One of the main reasons for using our app is text messaging. It takes the contacts the user interacts with by phone number from the contacts and the user can easily add them when they want to contact them. However, these procedures can sometimes take some time. That's why we're going to give users two options on how to add them to their contact list. The first is by retrieving the phone number from contact lists and the second by sharing the unique barcode known as the QR code. They can easily add a friend's or a family member's contact information to our application using the QR code. However, each option has advantages and disadvantages. By importing the contact list, the user can add the list remotely but slowly. But they can easily add their quick link for QR code, but not remotely.

With the advanced design of our app, users will be able to share files without size restrictions, up to large document files like images, videos and zip, etc. As long as you have a smartphone running Android on it and our application is installed, it will be able to achieve this. The uniqueness of this application is that it works smoothly in the use case and saves you from the stress of uploading and downloading the file as the mobile device, you can download files with the simple help of your smartphone, therefore our application helping the world come closer with ease.

After the application to be developed, we feel that it is time to start testing. We must assure the users that the system should maintain stability between different versions and releases. We also ideally want to automate the build process and automatically publish the app. For this, we need Android testing tools to guarantee that the build is working as expected. Therefore, we will use the following technologies to test our product:

- *JUnit*
- *Mockito*
- *Expresso*

For unit testing and instrumented testing we will use JUnit and Mockito, for UI we will use Expresso.

And we have some deliverables that we hope to achieve. Some major deliverables are:

- *Speed in usage*

  Real-time communication is one of the very important things in online communication. We need to establish a peer-to-peer connection without noticeable delays. Building a real-time application is all about slow delayed message delivery, i.e. latency, size of data transfer over the network should be as low as possible.

- *Smooth and friendly UI*

  Users should have easy access to all the features of our product. Novice users should not have detailed knowledge to use our product. For this reason, our product is aimed to have a simple user interface where the user finds the interface of the application familiar and can intuitively access all the features of the product.

- *Privacy Protection*

  Privacy protection is the most crucial part of the our application. Most products currently on the market create distrust in users of these products. Users have doubts about how their data are protected, and data is one of the most valuable things today. Users need to make sure that their data are secure. Our product aims to eliminate this distrust. One of the most important points is that we don't want to store data in the database and it will only store data while sending and

delete it from the database as soon as the data reaches the receiver. Using this technique, end-to-end encryption and other techniques will ensure product security.

We will also have a web-based application to allow users to access their conversations via a web browser. First we will launch for android mobile app, then we will develop web based desktop app using ReactJS. We will request authentication via a personal QR code. In addition, all conversations will be immediately available in the browser. Even when using the web-based application, communications remain end-to-end encrypted. And when the recipient receives the message, it will be indicated that the user is using the web-based application.

Also, there will be a feature where users can visualize their thoughts using a whiteboard. For example, users may need to visualize something to the person they are talking to while chatting. Using this feature, users can visualize their thoughts. They can simply draw their thoughts on the whiteboard and send them to the person they are talking to.

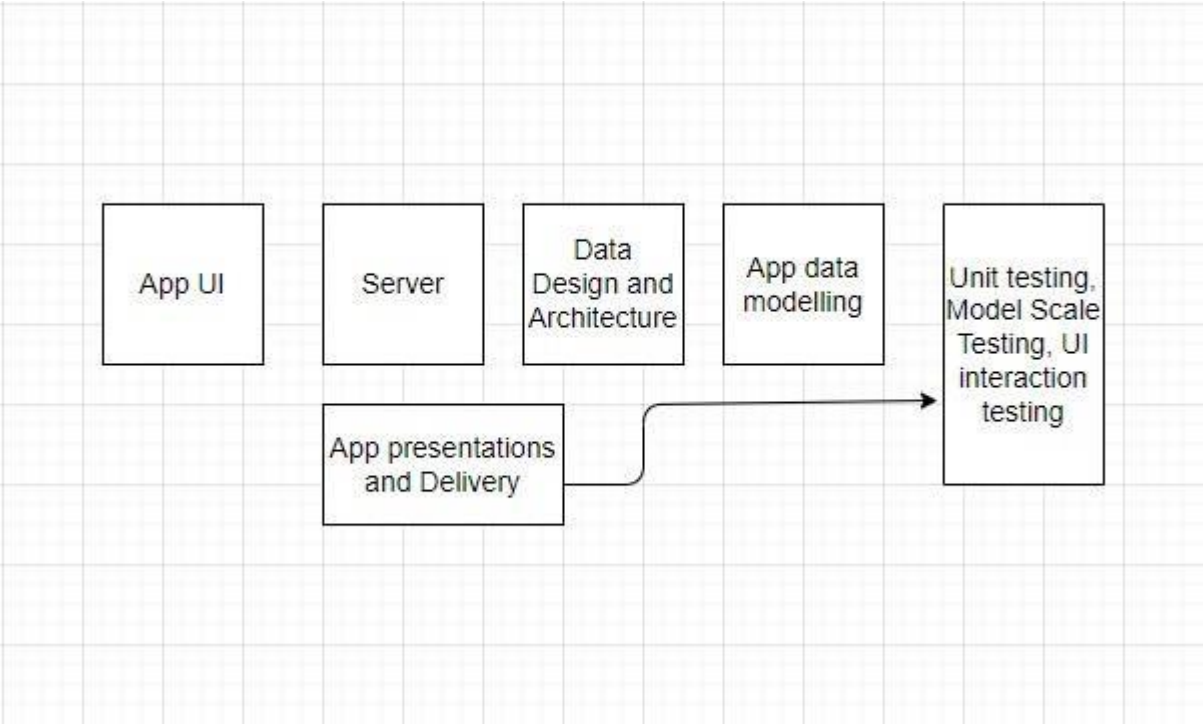


Figure 1.1: *Pre-Development Analysis*

2. Literature Review

Messaging apps now have more global users than social networks. This means that in the future, messaging applications will play an increasingly important role in the distribution of digital information. More than 2.9 billion people use at least one messaging app in 2020. And that means this one-third of the world's total population, and that includes users of various age classes. Today, it's common for offices to use a messaging app for internal communication to coordinate important meetings, share files, and schedule dates. So how did these chat apps become so popular?

Actually, commercial chat apps history date back to the 1980s. CompuServe released CB Simulator in 1980, and 1985 brought the launch of Commodore's Quantum Link. It is an online service allowed multiuser chat, email, file sharing, and games. So, the company changed its name to America Online in 1991. But AOL wouldn't launch its signature product, AOL Instant Messenger until 1997. And, ICQ launched as the first widely-adopted instant messaging platform in 1996. The late 1990s brought dramatic

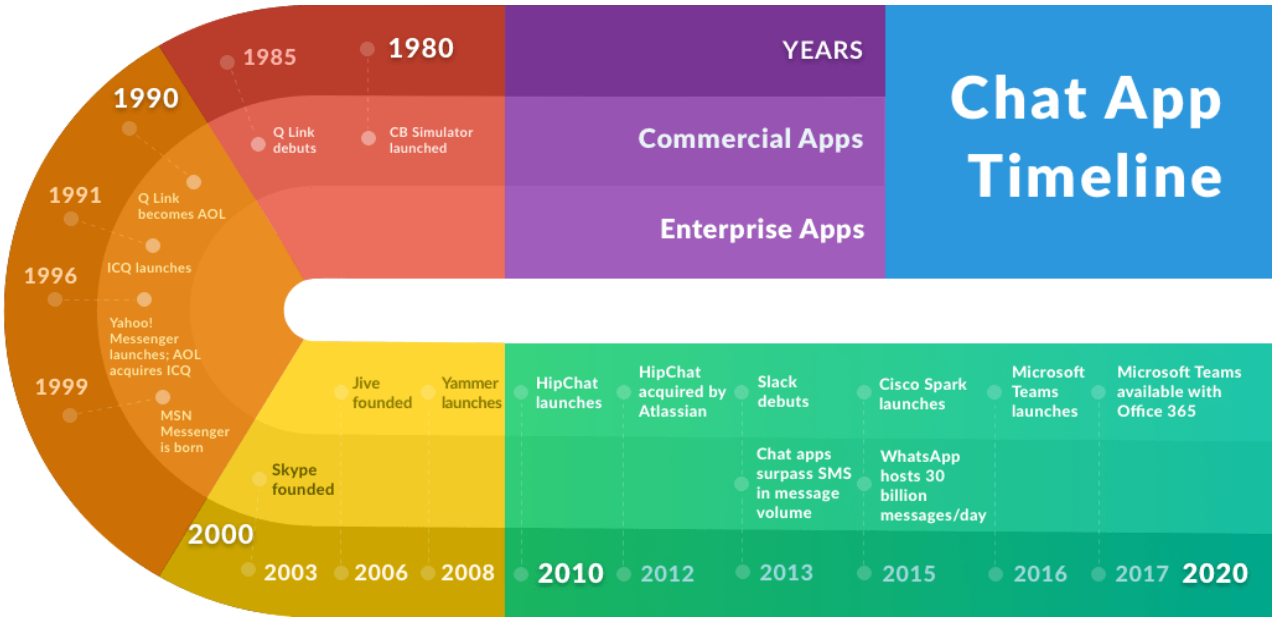


Figure 2.1: *Chat App Timeline*

changes in the chat app market. Both Yahoo! And MSN launched their own instant messengers in 1998 and 1999, respectively, and AOL bought out competitor ICQ's parent company, Mirabilis, for a fee around $612.7 million. As AIM pioneered Chabot, like SmarterChild, it also increased its market share. Its rivalry with MSN Messenger began almost as soon as the competitor launched in 1999. By 2006, AIM controlled

52% of the IM market. Its dominance, however, was short lived. Struggling to monetize and facing increasing competition from new apps like Skype and Google Talk, AIM fell out of favour, eventually eliminating its entire development team in 2012. With the inception of smart phones, in 2013, chat apps finally surpassed SMS in message volume. By 2015, WhatsApp alone hosted 30 billion messages per day. And in 2016, Facebook Messenger reached one billion users.

So far, we have talked about the history of chat applications and how communication over the internet has developed. In this section, we will talk about today's chat application. Let's look at Figure 2.2, we can see that there are 5 best chat apps from the figure. If we divide chat apps into two types, local and global, we can exclude WeChat and QQ because they are mostly popular in China. This means that their target audience is focusing on the local area. In fact, we are not interested in apps that serve the local area, because they have already taken over the local market places, and they are not in our area of interest. Thus, there're 3 best chat apps in the world like Whatsapp, Facebook Messenger and Telegram. As you know, Facebook Messenger is mostly a social network-based chat application. Telegram is a chat-based social networking application. Whatsapp has mostly taken its place in the market as a chat application. We already exclude Telegram and Facebook Messenger as they are mostly social network based apps. And we only have Whatsapp left. Whatsapp is mostly used for more general topics, daily conversation and

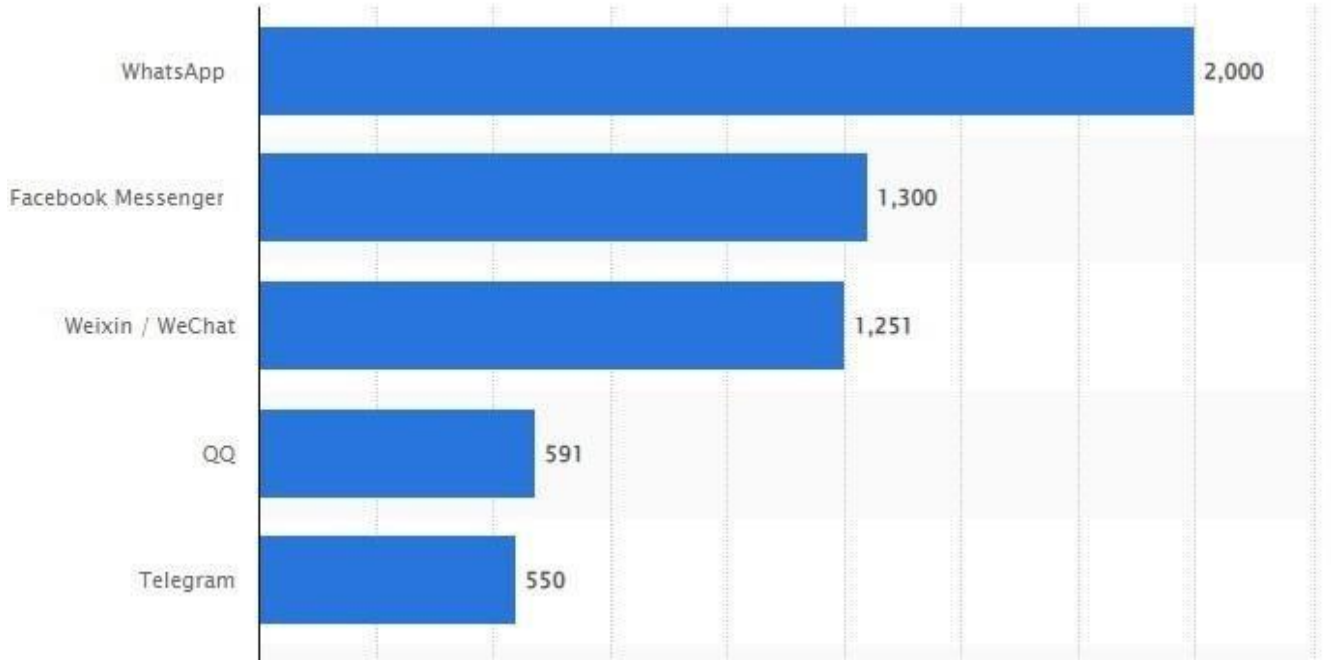


Figure 2.2: *Most popular global mobile messenger apps as of October 2021, based on number of monthly active users*

also for formal formal conversations. And that's where Whatsapp often fails. Using the same app for informal and formal conversation can be distracting. Because sometimes when we have a formal conversation, it can interfere in informal conversations, which can be a distraction from the conversation of formal topic. And, our app mostly focuses on formal communication, it is not intended to be used for daily communication. Roughly speaking, if we consider a social network-based application like Linkedin for more professional topics, we can think of our application as a chat application for more professional topics. We can also call our application a business network based chat application.

3. Proposed System

The proposed system would be able to do the following on each individual user's screen:

- Accept user: The users can accept request to connect from anyone within their contact list.
- Block User: This is used when the user does not want to communicate with the selected user on the network.
- Add user: The users can add anyone who has installed our application and registered.
- Attach: This can be used to send files like images, voice records, location or etc.
- User Authentication/Registration.
- Video and voice call: The users can call each other over the network if they have stable internet connections.
- Settings:
- Settings: The users can edit profile information and application settings.
- Images: The users can edit their pictures through our application.
- Notes: The users can take their important notes through our application.
- Schedule Messages: The users can choose the date and time to send messages in the future
- etc.,

The representation of the various parts of the application that make up the design architecture can be easily done using Unified Modeling Language modeling. The use case diagram and activity diagram will be used to model the design features of the proposed Chat application . UML is a general-purpose modeling language in software engineering designed to provide a standardized way to visualize the design of a system.

A use case is a sequence of transactions performed by a system that yields an outwardly visible, measurable result of value for a particular actor. Our use case will typically take the large piece of system functionality and represent it as smaller pieces. First, we'll look at the use case diagram of the authentication service.

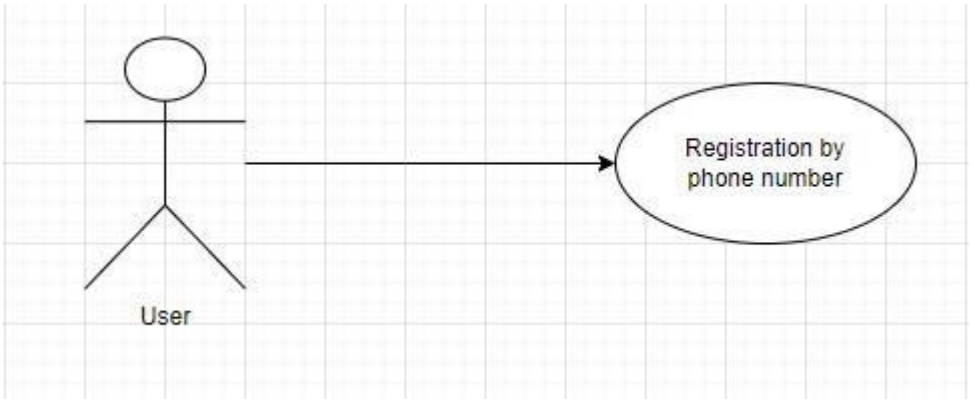


Figure 3.1: *Use Case Diagram of the Authentication Service*

Let’s look at Figure 3.1. Basically, we need the user's mobile number for authentication. Then we will send a message via SMS to verify the phone number. After the user has verified the phone number, the user registration process is considered finished.

From Figure 3.2, we can our use case diagram of the contact form.

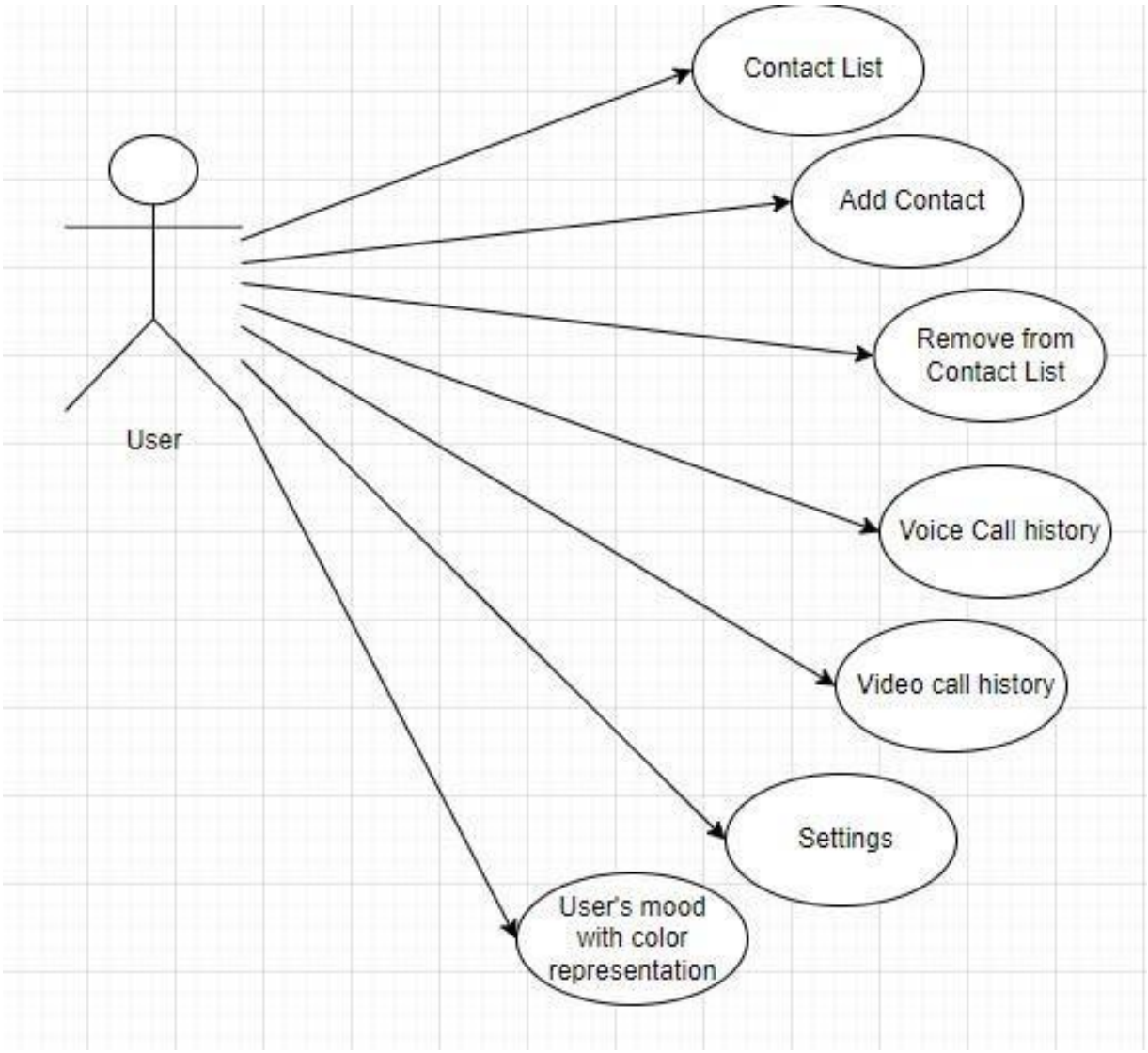


Figure 3.2: *Use Case Diagram of the Contact Form*

For the contact form, users can see their contact lists, add users from contact lists, remove and block users from contact lists, view video and voice call history, edit profile information, and get the user's mood with color representation using our sentimental analysis algorithm. In psychology, the mood of a person can be represented by colors, and in color psychology, each color has a meaning. By using this representation, we will indicate user moods with colors. These colors illustrated in Figure 3.3.

Figure 3.3: *The Psychology of Color*

Let's look at Figure 3.4, we can see the use case diagram of the chat form. User can send text messages, voice messages, locations, edit and send images, send documents, make video and voice calls, set favorite messages, set user's online and last seen status.

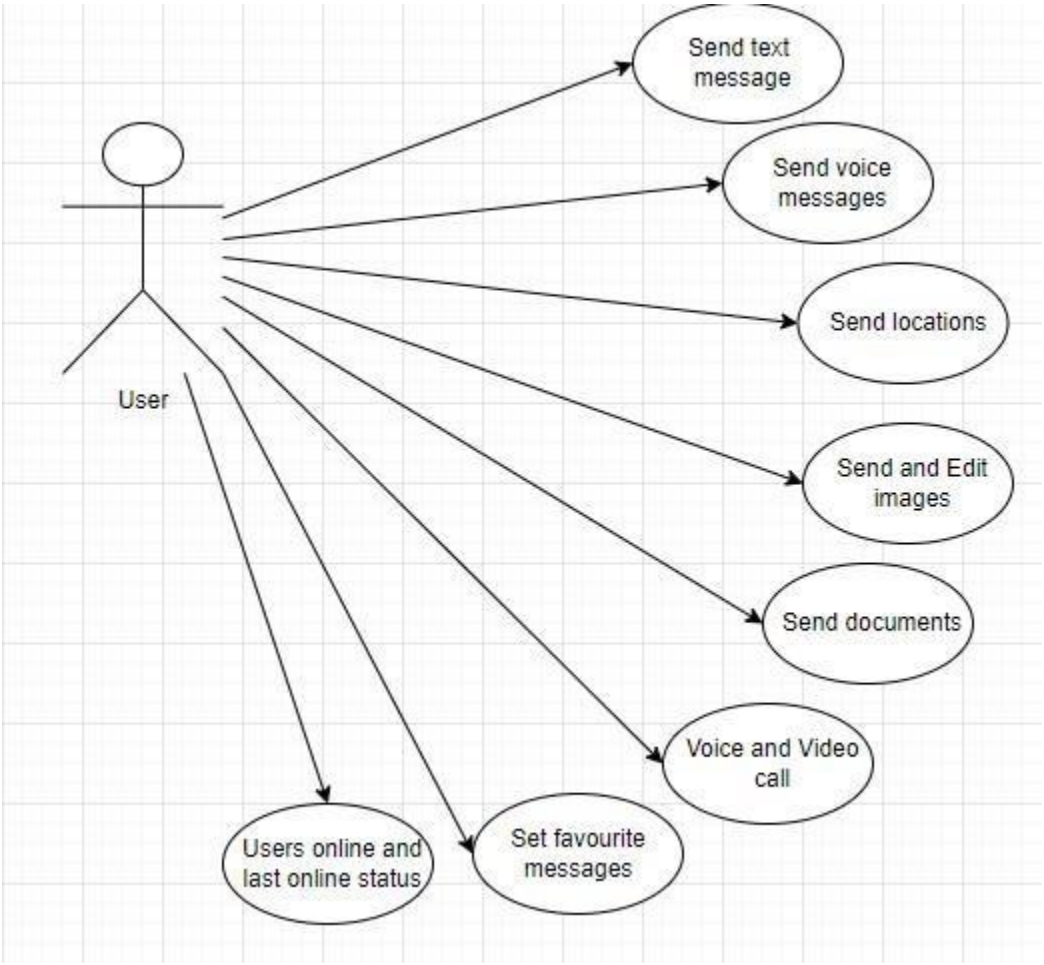


Figure 3.4: *Use Case Diagram of the Chat Form*

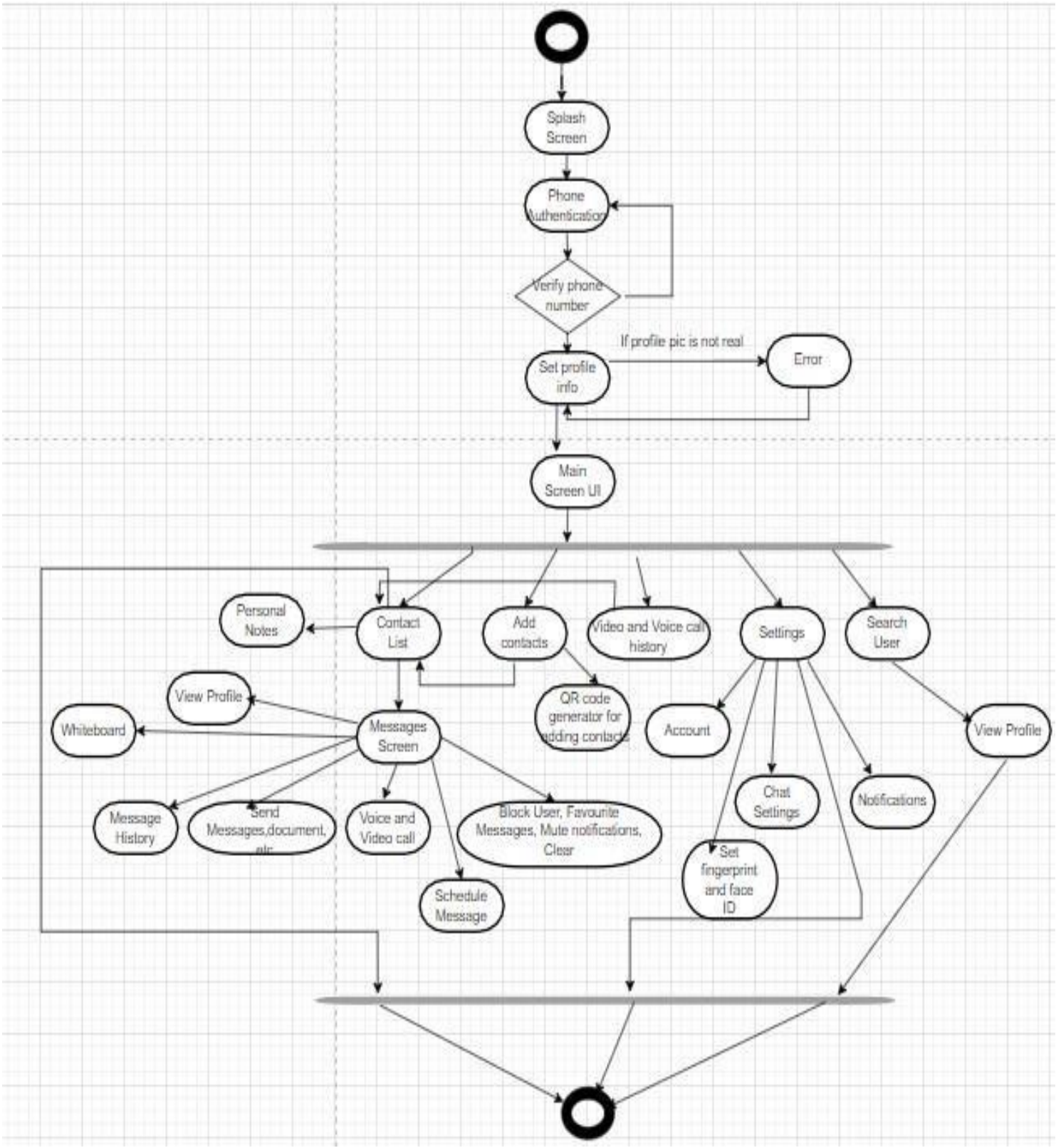


Figure 3.5: *Activity Diagram of the Application*

3.1 User Interface

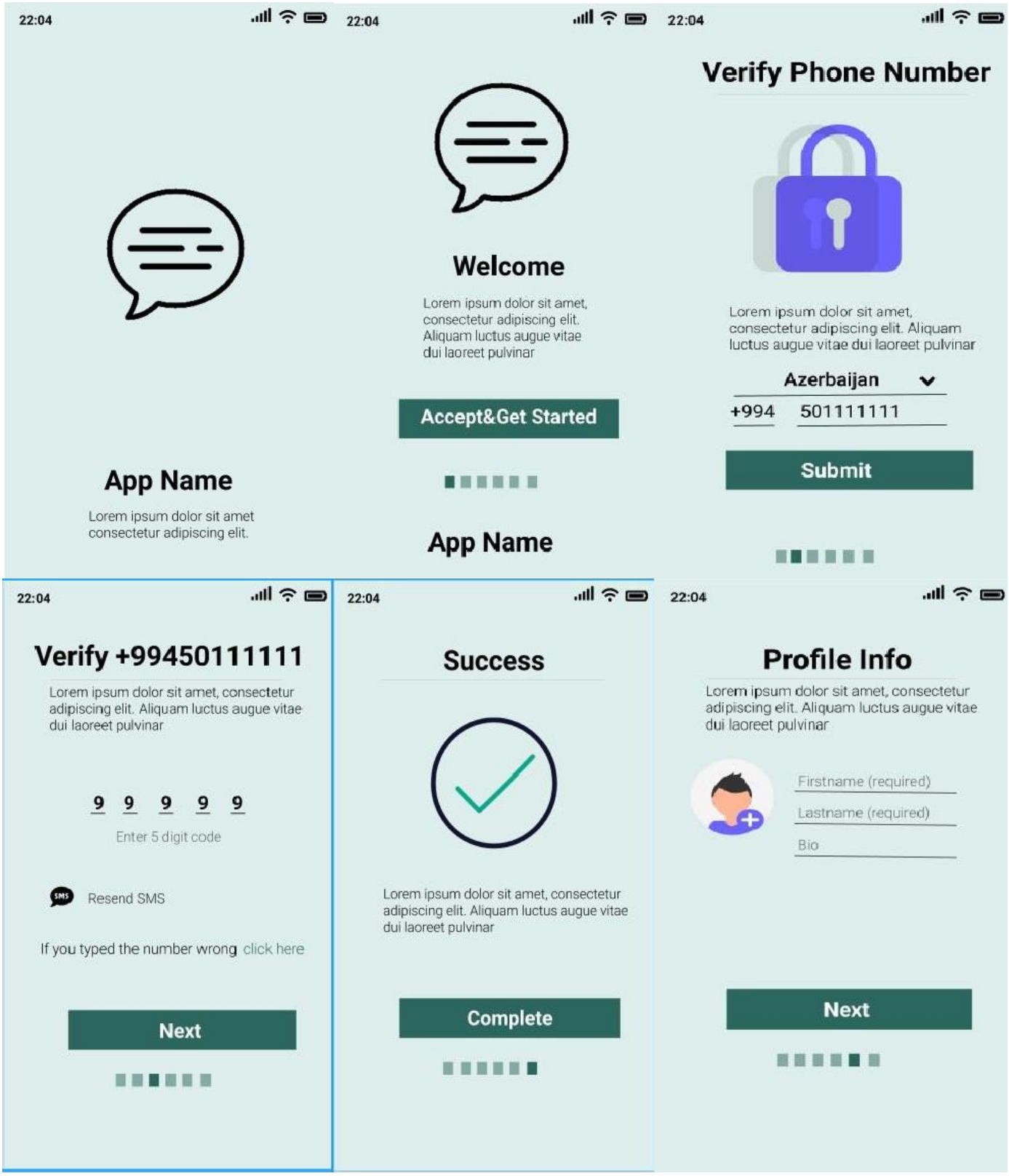


Figure 3.6: *Steps to complete the registration process*

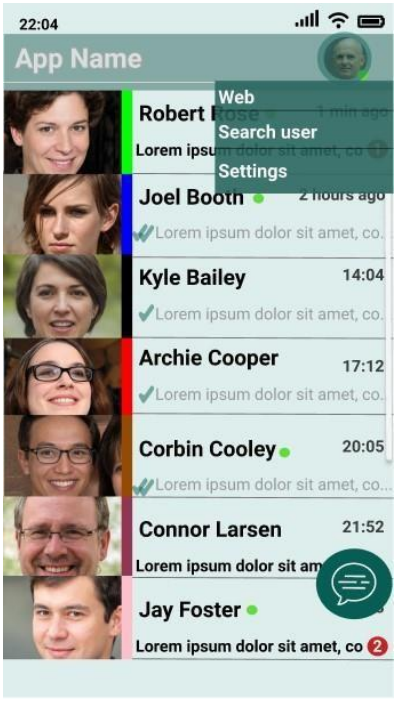


Figure 3.7: *After the user registration process is completed, the user will reach the main page*

Figure 3.8: *Chat screen and edit images screen*

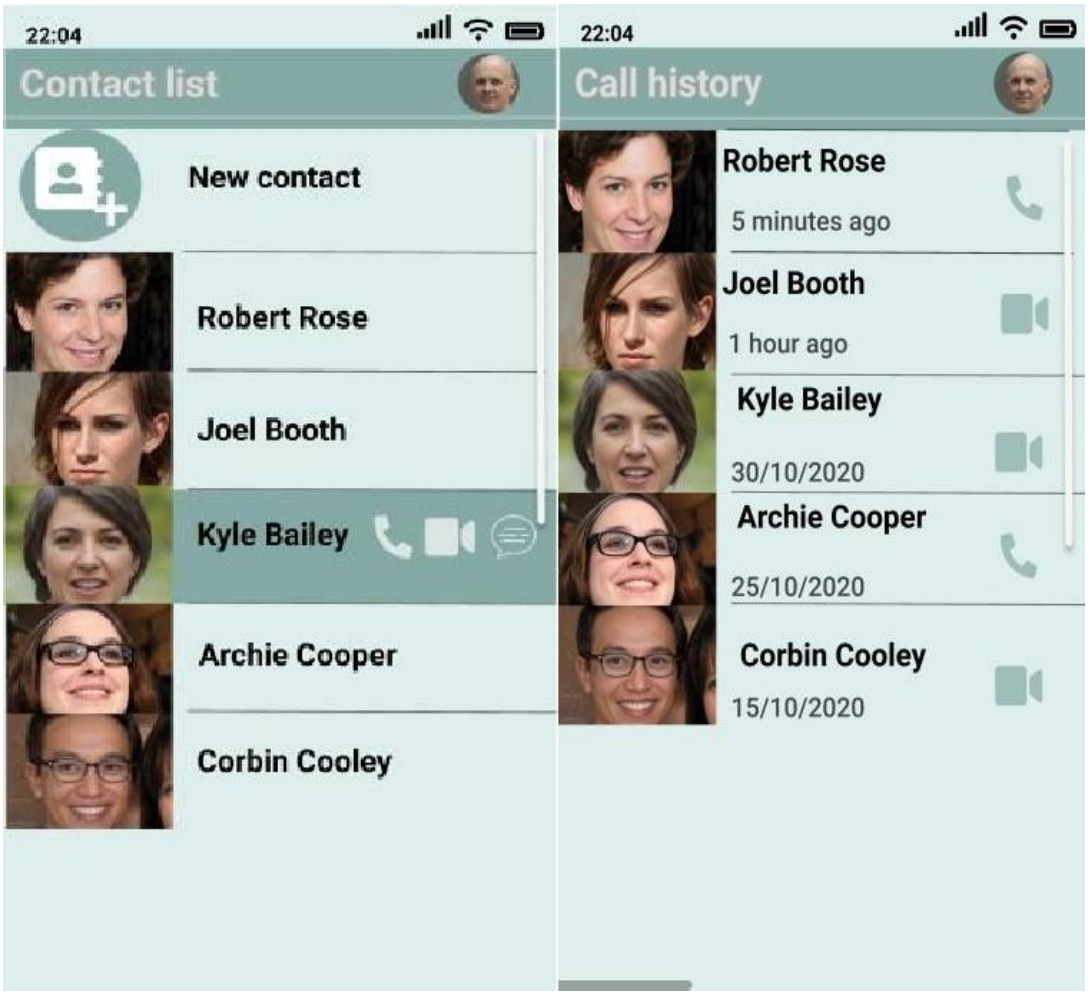


Figure 3.9: *Contact list and Video and Voice call history Screen*

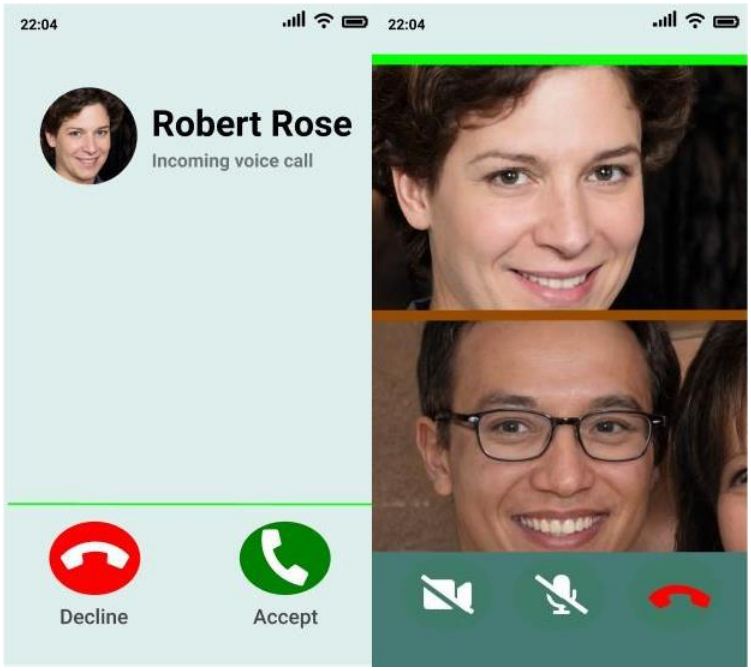


Figure 3.10: *Voice and Video call Screen*

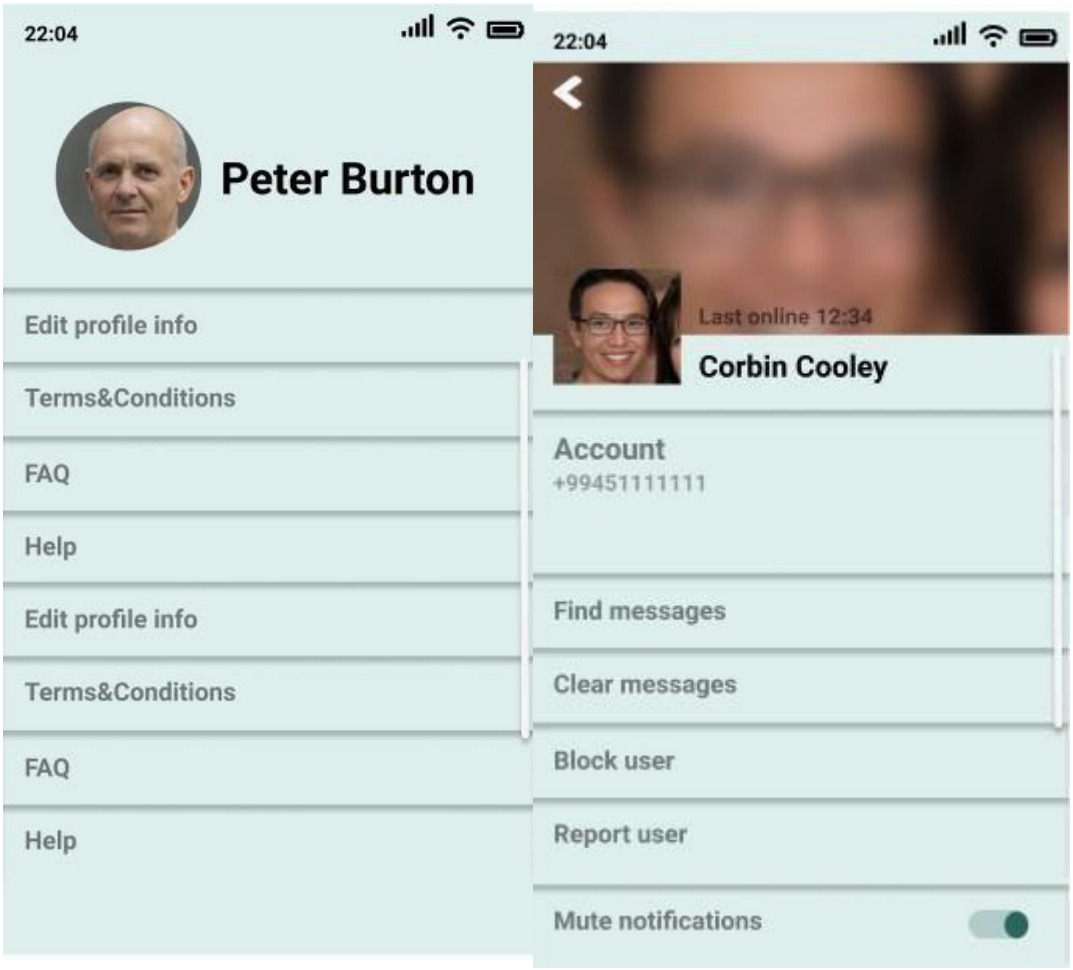


Figure 3.11: *Settings and View Profile Screen*

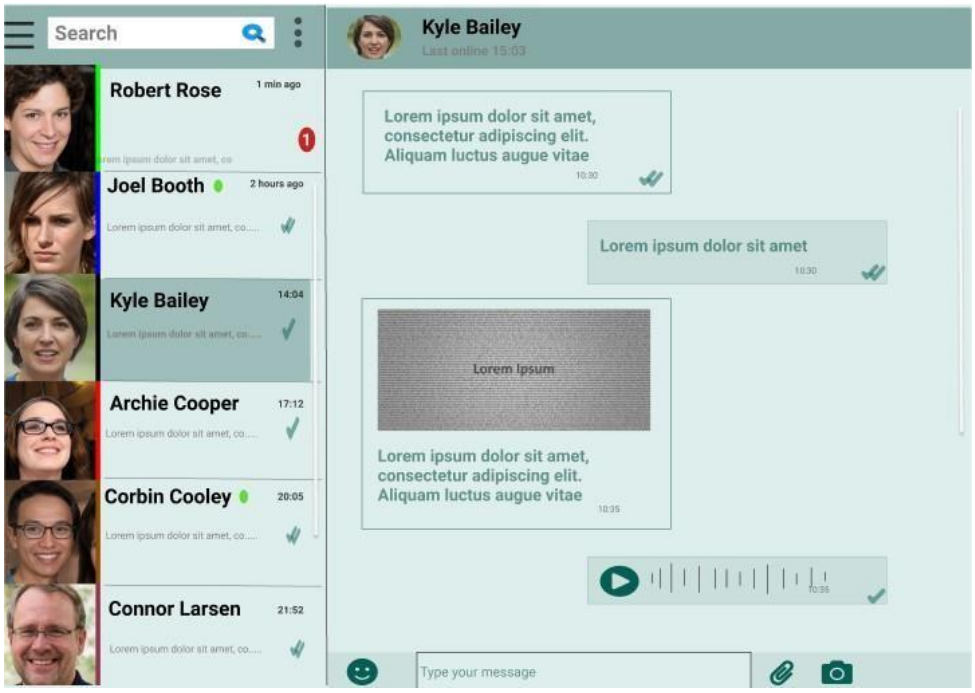


Figure 3.12: *Web based Desktop Application User Interface*

Any application always has room for improvement. Of course, our app can improve in terms of user interface and user experience. With its user-friendly interface, novice user need not hesitate to use our app, because the functionality of the app will be familiar to them. We also mentioned earlier that we will have applied machine learning projects. All of them are planning to run in the background and users do not need to be involve these.

3.2 Backend/Data storage and manipulation

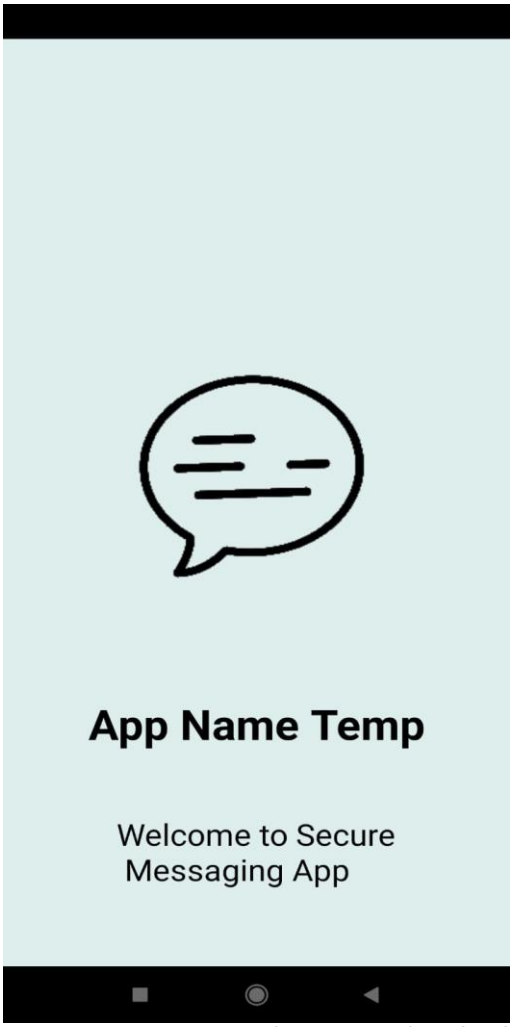
As mentioned earlier, we will use Firebase and SQLite for the database. We will fetch data from Firebase using REST API and save it in local database. Firebase is a NoSQL database with lots of optimizations and features compared to most relational databases. It contains flexible rules that define how data should be structured to provide security and flexibility. In addition, services such as email authentication and sending SMS OTP verification that we will use will be provided by Firebase free of charge, because nowadays, Firebase is also widely used for such services. Also, as we mentioned earlier, we are planning to save the data in local database like SQLite. SQLite is an open source SQL database that stores data in a text file on a device. By doing this, it will provide a more secure way as every user data is saved locally.

In addition, this database will allow users to use our application when the device is not connected to the internet. As long as the device is offline, the user can see the chat history, video and audio call history. Also, users can send offline messages to other users, but through asynchronous programming, their message will be automatically sent as soon as the device is online. Also, if we make sure that the users' data is saved locally, we don't need to save the data to Firebase. If the data is already saved locally, we delete the data in Firebase. By using this method, we will get a safer way with a limited budget. As long as the application meets expectations, we can create our own services for SMS OTP verification and email verification. And using REST API and MVVM architecture, we can easily achieve this with minor changes on the technical side. And for web-based desktop application, we send data from phone to desktop, such as mirroring mobile phone screen to desktop application. Of course, we will have more limited features for the desktop app than for the mobile device.

3.3 Prototype

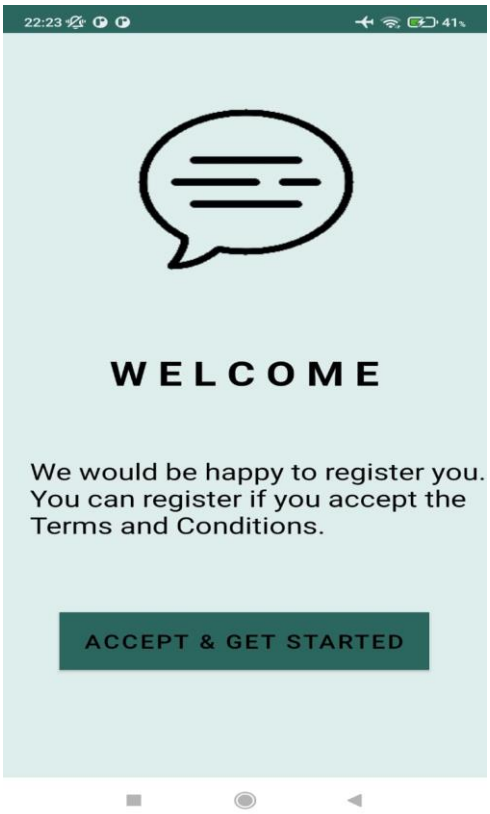
At this stage, We will talk about the progress so far and the implementation of the project configuration as described above step by step. We will also talk about the applications that have been implemented so far. Thus, we added splash screen actvity, registration activity with 1 activity and multiple fragments with OTP verification, adding user information such as first name, last name, status, picture, phone number and uniquely generated user id to database. And finally, we have a small prototype for future sentiment analysis.

3.3.1 Implementation implemented so far



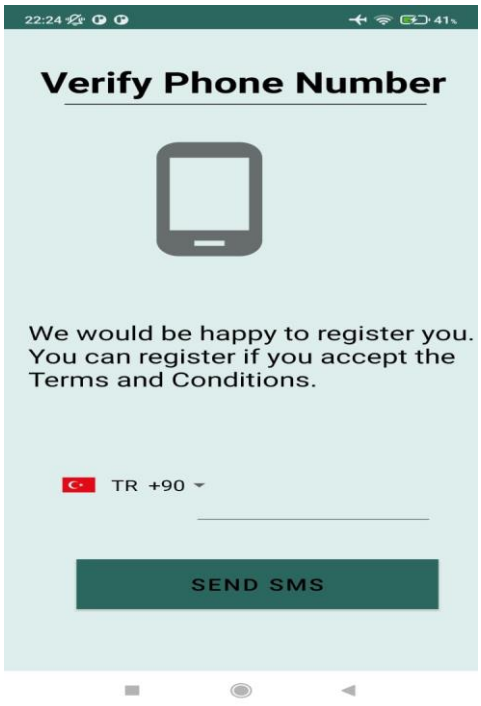
Splash screens are often used, especially by applications, to let the user know that the program is in the process of loading. They provide feedback that a long process is in progress. Sometimes, a progress bar on the splash screen shows the loading progress. For now our app is waiting three seconds to prepare the app. Also, if the user has successfully registered and the phone number has been added to the database, the user phone number is saved in shared preferences (retrieve small amounts of primitive data as key-values to a file in the device storage). And, if the phone number is not null , it will be redirected to home screen, otherwise it will be redirected to registration activity.

Figure 3.13: *Implemented Splash Screen*



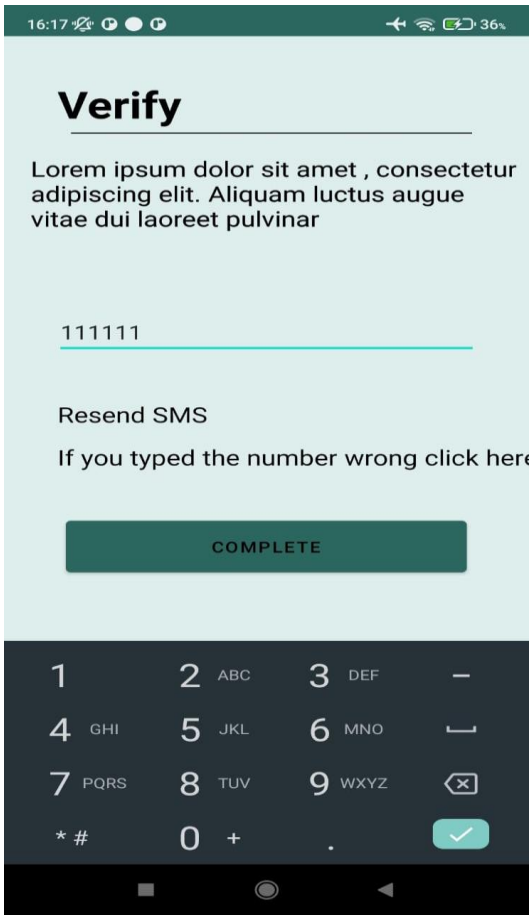
For this screen, we notify the user that the application is ready and says welcome to the user. Here we will have terms & conditions, users will be able to read the terms & conditions documents and if they accept the terms & conditions, they can accept it by clicking the Accept&Start button. This button will also redirect the user to verify the phone number process.

Figure 3.14: *Implemented Welcome Screen*



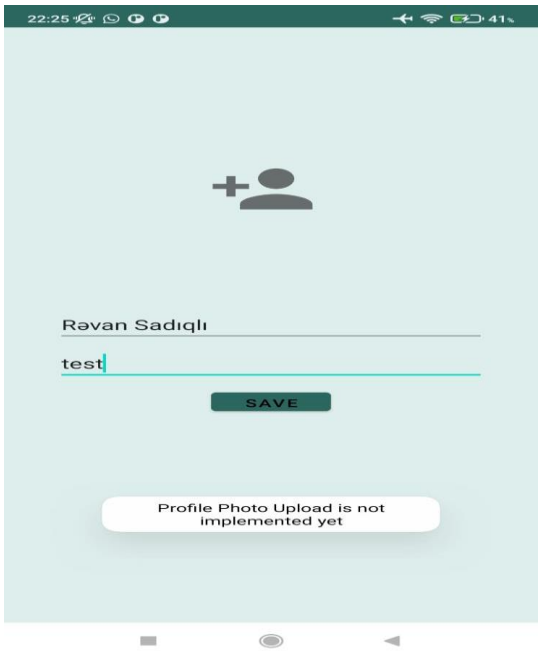
What we want from the user on this screen is to verify the phone number. The user must press the SEND SMS button after entering the user number and selecting the country code. Here we send a one-time password using Firebase Phone Authentication services. A one-time password is a one-time PIN code, a one-time authorization code, a password that is only valid for one login session for our application. Firebase Authentication services must ensure that the user is not a robot. So the user needs to verify using reCaptcha and redirect the browser to authenticate with reCaptcha. For now we are using a browser to authenticate the robot. In the future we will use the SafetyNet reCaptcha method that will be built into the application. So we will send the PIN using the phone number and redirect to the Verify Number fragment

Figure 3.15: *Implemented Verify Phone Number Screen*



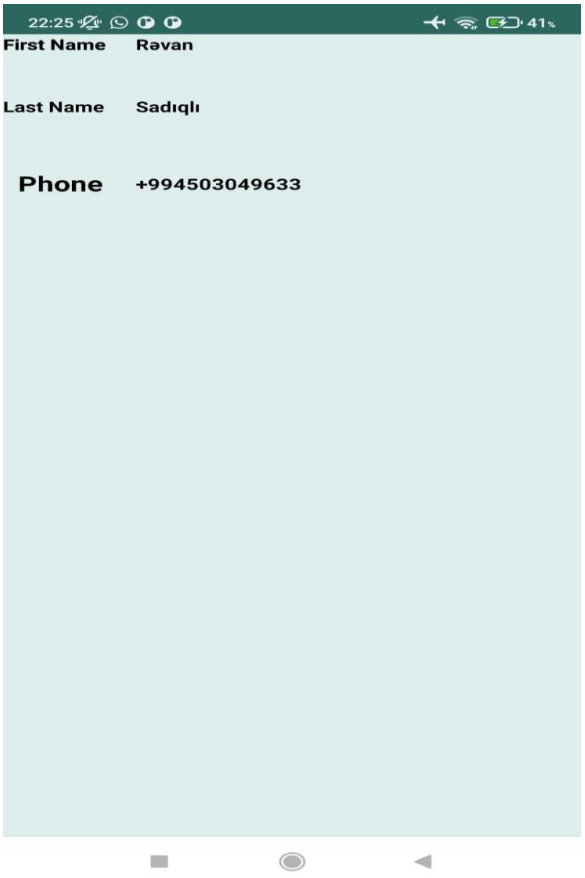
After entering the user number, we directed the user to this screen. Here, the user should type the PIN code that we sent from his/her phone number and press the COMPLETE button. In the future we will have a controller that in case the user is unable to receive sms due to wrong phone number or some circumstances. These are not working for now, but we will have these features in the future.

Figure 3.16: *Implemented Verify Phone Number Screen*



This screen requires the user to define their full name and status. And this information has been added to the database. For now, image upload to database is not implemented yet, but we will have this feature in the future as well. And the user has successfully registered after completing all the registration procedures. And the user will be redirected to the Main Chat Screen.

Figure 3.17: *Implemented Profile Info Screen*



For now, the user will be redirected to their profile information. Here we will show the user's information such as first name, last name and phone number. The advanced UI is not implemented yet, it just shows the user information in text.

Figure 3.18: *Implemented Profile Info Screen*

As described above, user information will store Firebase Real Time Database. Since, the Firebase Realtime Database is a cloud-hosted NoSQL database that lets use to store and sync data between your users in realtime. And, the our JSON type user database illustrated in the figure. In here, we have users' unique id, image, name, status and phone number.

Figure 3.19: *Firebase*

3.3.2 Sentiment Analysis Prototype

As explained above, we mentioned that we will use sentiment analysis. To do this, we have various methods in the literature, such as applying machine learning models on mobile devices. We will use one of these methods, like Tensorflow. TensorFlow is a free and open source software library for machine learning and artificial intelligence. It can be used in a number of tasks, but has a special focus on training and inferring deep neural networks. We will use Google Colab to implement the Python code. Google Colab allows us to write and execute arbitrary python code via the browser and is particularly well suited for machine learning, data analysis and education. As our train data grows, our dataset will grow as users communicate with each other, so in this case we need to update our dataset to improve accuracy, we will use Firebase which allows us to update our dataset. To summarize roughly, we will develop python code for machine learning in Google Colab and after the training we will apply the model for our application with our model download file with .tflite extension. Then, we will update the model using Firebase.

Sentiment analysis is a text mining technique that uses machine learning and natural language processing to automatically analyze text for text sensitivity. And text sensitivity classification is a fundamental subfield in natural language processing. Natural Language Processing, or NLP for short, is defined as the automatic processing of natural language, such as speech and text, by software. Natural language processing studies have been around for over 50 years and have emerged from the field of linguistics with the rise of computers. Linguistics is the scientific study of language, including its grammar, semantics, and phonetics. Classical linguistics is the way of designing and re-evaluating the language of syntax and semantics through formal methods. However, natural language is also another phase of mathematical formalisms. Mathematics is the tool of science. Mathematicians working on natural language may refer to their work only as mathematical linguistics, which focuses on the use of discrete mathematical formalisms (eg, discrete mathematics) and natural language theory (e.g, formal languages and automata theory). Deep learning techniques hold great promise for challenging natural language processing problems.

RNN is one of the advanced algorithms in deep learning. RNNs are often used to predict the next step. The other most important deep learning difference is that they can remember. RNNs make relations between inputs to follow the next step and remember all their relations while training. For sentiment analysis, we will use Long short-term memory, which is an artificial recurrent neural network architecture used in the field of deep learning. Also, I have a very basic model for sentiment analysis as a prototype I developed for the CSE 463 ML Course. I've added this detailed information and model to end of report *(See page 30)*

3.3.3 Age and Gender Estimation

Age and gender classification is one of the important features in our application. It provides ease of use to the user. In fact, we can obtain this information by giving the input to the user. However, such convenience features attract users.

4 Project Requirements

Communication is a tool for people to exchange messages. It has started since the beginning of human creation. Remote communication began in the 1800s with the introduction of televisions, telegraphs, and then the telephone. Interestingly, telephony stands out as the fastest growing technology, from landlines to mobile wireless, from voice calls to data transfer. The advent of computer networking and telecommunications technologies has the same purpose as enabling people to communicate. Chat is a method of using technology to bring people and ideas together despite geographic barriers. The technology has been available for years, but its adoption was fairly recent. This project is an example of a real-time chat application. To start chatting, users must install our app and have a stable internet connection. We also have some requirement specifications for our application as described below. Let's look in detail.

- *Project Deliverables*

  1. launched application for Android Mobile Devices in Play Store Market
  2. Web based Desktop Application
  3. Documentation

- *Hardware Interface*

  Android phone 256 MB minimum
  RAM Required Wireless connection
  Processor with speed of 500MHz
  Android Lollipop 5.0 (API Level 21) or above Gradle 4.1.0 or higher

- *Constraints & Limitations*

  The system must be connected to the internet
  Users can use or install this app on android devices

- *Tools and Technology*

  - Integrated Development Environment
    Android Studio
    Google Colab
  - Software Development Kit
    Firebase SDK
    Google Maps SDK
    PhotoEditor SDK
    Android Media Framework
    QuickBlox SDK, etc.
  - Database
    Firebase

SQLite
- Android UI/UX libraries
- Software Development Methodology
  Agile
  Scrum
- Programming Languages
  Kotlin, Python, Java, JavaScript
- Testing
  JUnit, Mockito, Expresso
- Software Analysis
  Design Patterns, UML tools, Mind Map

Figure 4.1: *Functional Requirements (https://gitmind.com/app/doc/1896989944)*

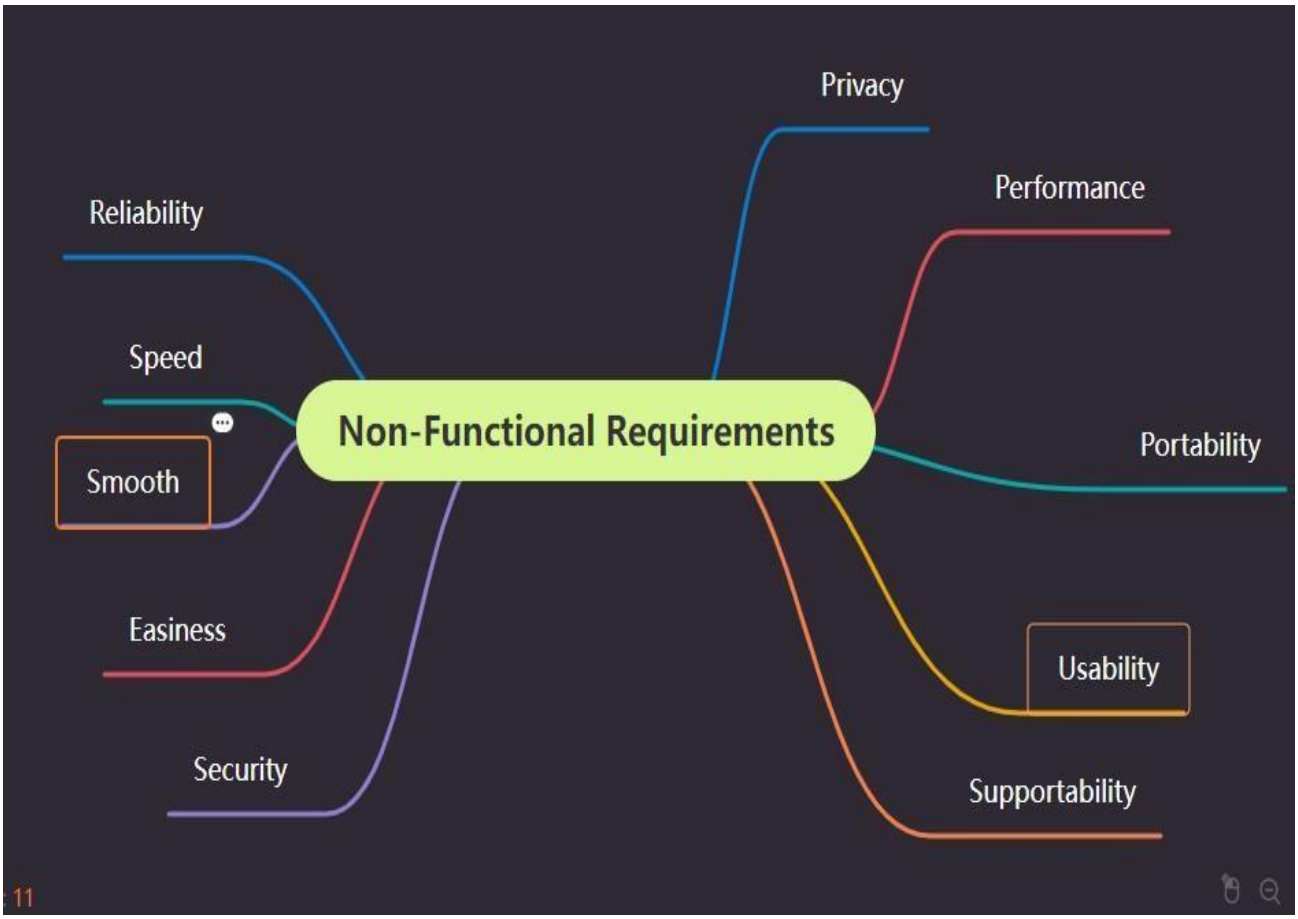


Figure 4.2: Non-*Functional Requirements (https://gitmind.com/app/doc/8bb6990677)*

5 References

[1] SHUYANG ZHU., APPLIED PROJECT - CHAT APPLICATION WITH PREPARED BY: DISTRIBUTED SYSTEM., 4-15-2020., Page 6-17.

[2] Vinay Kumar, Kartik Gaur, Raj Singh,Dr. Preeti Bajaj., Android Based Instant Messaging Application Using Firebase., June 2021., Pages. 71-77

[3] Sirisha Potluri., Android based instant messaging application using firebase., January 2019

[4] Rowe-Ann Antenor, Robert Bautista, Francis Paolo Lesaca, Raychelou Valencia., A Message Encrypting Application Utilizing RSA Algorithm for Android-Based Mobile Device., January - June 2018

[5] Team Agora., Real-Time Communication: Tools for Online Messaging., April 2020

[6] Phil Morettini., How Important Is Ease-Of-Use In The Software Business., June 2013

[7] jmu.edu.,ISAT Senior Project Manual: A Guide through the ISAT Senior Project Process.

[8] Team Shadow., Chat Application. Apr. 18, 2017

[9] www.geeksforgeeks.org., How to create a Face Detection Android App using Machine Learning KIT on Firebase., Jun 24, 2021

[10] Ashwini JHA., Human Face & Images Decoded : Face Recognition & Image Classification Android App. ,Jul 28, 2020

[11] www.lexalytics.com.,Sentiment Analysis Explained

[12] Ayusch Jain., Learn Unit Testing in Android by building a sample application., Oct 7 , 2018

[4BY BERNIE ROSEKE, P.ENG.,25-example-project-deliverables., MAY 26, 2014

[13] systeminterview.com., design-a-chat-system.php.

[14] Miriam L. Matteson, Jennifer Salamon and Lindy Brewster., A Systematic Review of Research on Live Chat Service., 72-190 (19 pages)

[15] Vicky, Chau Ka Ki., The Impact Of The Chat: A brief Literature Review

[16] Damjan Jugovic Spajic., Text, Don't Call: Messaging Apps Statistics for 2020., December 11, 2019

[17] www.whoson.com., The historyof live chat software

Bidirectional LSTM network for Sentiment Analysis

1. INTRODUCTION

Bidirectional LSTM is just an extension of LSTM. LSTM is an artificial recurrent neural network mostly used in deep learning. Unlike feedforward neural networks, LSTM also has feedback connections. Actually, LSTM is an extension of RNN with memory expansion. LSTM is used as the building block for the layers of an RNN. LSTMs update weights that help RNNs allow new information and forget information. The downside of RNN is that its train is a huge and tough task, and its gradient has problems such as exploding and vanishing. And if it uses tanh or relu as activation functon it can't handle long sequences data. Also, LSTM RNN networks are effective for sentiment analysis. LSTM uses gates for memorization. To handle the gradient vanishing, LSTM networks use the tanh activation function. It also uses the sigmoid function to forget or remember information as the sigmoid function returns output between 0 and 1. The LSTM RNN network has three steps to train:

1. As mentioned above, LSTM has the feature of forgetting unnecessary information. Sigmoid layer handles for forgetting process. If the sigmoid layer returns 0, it means that the network has forgotten this information. The sigmoid layer takes the current input and the output of the last LSTM unit and decides which part of the sentence is unnecessary and forgets that part of the sentence for sentiment analysis.
2. The next step is to make decisions based on the information from the current input and store them. The sigmoid layer handles to memorizing process. The tanh layer creates a vector of all possible values from the current input. These two are multiplied to update the new cell state. This new memory is then added to the old memory.
3. The final step is what the output will be. The sigmoid layer decides which parts of the state to output. Next, we put the cell state in a tanh that generates all possible values and multiply it by the output of the sigmoid gate, so we only output the parts we decide.

The advantage of bidirectional LSTM over standard LSTM is that it acts in two ways, one of the inputs works from the past to the future and the other from the future to the past. Bidirectional LSTM stores information from the future and combines two hidden states to store information from both the past and the future.

2. DATA PREPROCESSING

We use three different datasets to get more accurate results. Then we will combine these three different datasets into a single dataset. We have tweets from the American airline and Apple,

and also tweets from some random users from twitter. This dataset can be found from the following link:
https://drive.google.com/drive/u/0/folders/13nJLiKdaNTh6WibbYTw7hkDEYM16lSAM

```
%cd /content/drive/MyDrive/CSE463/LSTM/
```

```
/content/drive/MyDrive/CSE463/LSTM
```

We need to import some libraries:

- pandas - for read csv and dataframe
- numpy - working with arrays
- re - for regular expressions
- nltk - for natural language processing

```
!pip install nltk
import pandas as pd
import numpy as np
import re
import nltk
import string
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.7/dist-packages (3.2.5)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from nl
```

First of all, we read the apple.csv file taken from the Apple Twitter official page. The dataset contains information for customer satisfaction. Here we actually need two columns: text and sentiment. Next, we rename the "text" column to "tweet" column. And we will map the sentiment label to integers.

```
df1 = pd.read_csv("apple.csv")
df1 = df1.rename(columns={'text': 'tweet', 'sentiment':'sentiment'})
df1['sentiment'] = df1['sentiment'].map({-1: 'negative', 0: 'neutral', 1:'positive'})

df1.head()
```

	tweet	sentiment
0	Wow. Yall needa step it up @Apple RT @heynyla:...	negative
1	What Happened To Apple Inc? http://t.co/FJEX...	neutral
2	Thank u @apple I can now compile all of the pi...	positive
3	The oddly uplifting story of the Apple co-foun...	neutral
4	@apple can i exchange my iphone for a differen...	neutral

We apply the same procedure to each dataset.

```
df2 = pd.read_csv('america_airline.csv')
df2 = df2.rename(columns={'text': 'tweet', 'airline_sentiment':'sentiment'})
df2['sentiment'] = df2['sentiment'].map({'negative': 'negative', 'neutral': 'neutral', 'po
df2 = df2[['sentiment','tweet']]
df2.head()
```

	sentiment	tweet
0	neutral	@VirginAmerica What @dhepburn said.
1	positive	@VirginAmerica plus you've added commercials t...
2	neutral	@VirginAmerica I didn't today... Must mean I n...
3	negative	@VirginAmerica it's really aggressive to blast...
4	negative	@VirginAmerica and it's a really big bad thing...

```
df3 = pd.read_csv('datafromtwitter.csv')
df3 = df3.rename(columns={'clean_text': 'tweet', 'category':'sentiment'})
df3['sentiment'] = df3['sentiment'].map({-1: 'negative', 0: 'neutral', 1:'positive'})
df3 = df3[['sentiment','tweet']]
df3.head()
```

	sentiment	tweet
0	negative	when modi promised “minimum government maximum...
1	neutral	talk all the nonsense and continue all the dra...
2	positive	what did just say vote for modi welcome bjp t...
3	positive	asking his supporters prefix chowkidar their n...
4	positive	answer who among these the most powerful world...

```
df3.tail()
```

	sentiment	tweet
162975	negative	why these 456 crores paid neerav modi not reco...
162976	negative	dear rss terrorist payal gawar what about modi...
162977	neutral	did you cover her interaction forum where she ...
162978	neutral	there big project came into india modi dream p...
162979	positive	have you ever listen about like gurukul where ...

Here, we combine three datasets into one dataset.

```
df = [df1, df2,df3]
```

```
df = pd.concat(df, ignore_index=True)
df
```

	tweet	sentiment
0	Wow. Yall needa step it up @Apple RT @heynyla:...	negative
1	What Happened To Apple Inc? http://t.co/FJEX...	neutral
2	Thank u @apple I can now compile all of the pi...	positive
3	The oddly uplifting story of the Apple co-foun...	neutral
4	@apple can i exchange my iphone for a differen...	neutral
...	...	...
179245	why these 456 crores paid neerav modi not reco...	negative
179246	dear rss terrorist payal gawar what about modi...	negative
179247	did you cover her interaction forum where she ...	neutral
179248	there big project came into india modi dream p...	neutral
179249	have you ever listen about like gurukul where ...	positive

179250 rows × 2 columns

Our dataset can contain null or NaN. That's why we drop features that contain null.

```
df.isnull().sum()
```

```
tweet        4
sentiment    7
dtype: int64
```

```
df.dropna(axis=0, inplace=True)
```

```
df.isnull().sum()
```

```
tweet        0
sentiment    0
dtype: int64
```

```
df.shape
```

```
(179239, 2)
```

```
df.head()
```

	tweet	sentiment
0	Wow. Yall needa step it up @Apple RT @heynyla:...	negative
1	What Happened To Apple Inc? http://t.co/FJEX...	neutral
2	Thank u @apple I can now compile all of the pi...	positive

```
df.tail()
```

	tweet	sentiment
179245	why these 456 crores paid neerav modi not reco...	negative
179246	dear rss terrorist payal gawar what about modi...	negative
179247	did you cover her interaction forum where she ...	neutral
179248	there big project came into india modi dream p...	neutral
179249	have you ever listen about like gurukul where ...	positive

After combining the datasets, we get the 179239 feature. And we have 1 feature and 1 label. One for tweets from twitter users, the other for sentiment.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 179239 entries, 0 to 179249
Data columns (total 2 columns):
 #   Column     Non-Null Count   Dtype 
---  ------     --------------   ----- 
 0   tweet      179239 non-null  object
 1   sentiment  179239 non-null  object
dtypes: object(2)
memory usage: 4.1+ MB
```

```
def remove_pattern(input_txt, pattern):
    r = re.findall(pattern, input_txt)
    for word in r:
        input_txt = re.sub(word, "", input_txt)
    return input_txt
```

Here, we remove the string that contain @(to mention user in Twitter). We don't need it for sentiment analysis.

```
df['tweet'] = df['tweet'].apply(lambda x:x.lower())
clean_tweet = np.vectorize(remove_pattern)(df['tweet'], "@[\w]*")

print(clean_tweet)
```

```
['wow. yall needa step it up  rt : music and snapchat at the same damn time. thank yo
```

```
'what happened to apple inc?  http://t.co/fjexi3op0u  #aapl #apple  http://t.co/wxl
'thank u  i can now compile all of the pics that i communicate with in one place ht
... 'did you cover her interaction forum where she left '
'there big project came into india modi dream project but not happened reality'
'have you ever listen about like gurukul where discipline are maintained even narenc
```

We don't need special characters, numbers and punctuation in sentiment analysis. That's why we're removing these characters.

```
clean_tweet = df['tweet'].str.replace("[^a-zA-Z#]", " ")
print(clean_tweet)
```

```
0         wow  yall needa step it up  apple rt  heynyla ...
1         what happened to apple inc    http   t co fjex...
2         thank u  apple i can now compile all of the pi...
3         the oddly uplifting story of the apple co foun...
4          apple can i exchange my iphone for a differen...
                              ...
179245    why these     crores paid neerav modi not reco...
179246    dear rss terrorist payal gawar what about modi...
179247    did you cover her interaction forum where she ...
179248    there big project came into india modi dream p...
179249    have you ever listen about like gurukul where ...
Name: tweet, Length: 179239, dtype: object
```

We remove the words that are less than 3 in length. Because it is meaningless for sentiment analysis.

```
clean_tweet = clean_tweet.apply(lambda x: " ".join([w for w in x.split() if len(w)>3]))
print(clean_tweet)
```

```
0         yall needa step apple heynyla music snapchat s...
1         what happened apple http fjexi #aapl #apple mo...
2         thank apple compile pics that communicate with...
3         oddly uplifting story apple founder sold stake...
4         apple exchange iphone different color lmao cha...
                              ...
179245    these crores paid neerav modi recovered from c...
179246    dear terrorist payal gawar what about modi kil...
179247                    cover interaction forum where left
179248    there project came into india modi dream proje...
179249    have ever listen about like gurukul where disc...
Name: tweet, Length: 179239, dtype: object
```

This is the step where we divide the text into parts corresponding to individual tokens.

```
tokenized_tweet = clean_tweet.apply(lambda x: x.split()).values
```

Here, we combine words into single sentence.

```
for i in range(len(tokenized_tweet)):
    tokenized_tweet[i] = " ".join(tokenized_tweet[i])
```

This is the step for stemming. Stemming is a process that involves reducing the words to their stems. The stem of a word can be said to be its base form.

```
from nltk.stem.porter import PorterStemmer
stemmer = PorterStemmer()
print(len(tokenized_tweet))
tokenized_tweet = [PorterStemmer().stem(word) for word in tokenized_tweet]
tokenized_tweet
```

```
179239
['yall needa step apple heynyla music snapchat same damn time thank #not',
 'what happened apple http fjexi #aapl #apple moneypress http wxkmmtmarw',
 'thank apple compile pics that communicate with place http',
 'oddly uplifting story apple founder sold stake aapl #aapl http cizbvr',
 'apple exchange iphone different color lmao changed mind',
 'jpdesloges apple acted unfairly suppressing digital music competition paul kedros
 'forget that press effects stock apple defend against ipod antitrust suit http bki
 'apple deleted songs from rival services from ipods http dqwdfpcc #aapl',
 'teamcavuto apple they staging apple store while being their iphon',
 'happy monday camera fancy apple #iphone plus suddenly stopped working this weeker
 'facebook mark zuckerberg criticizes apple aapl #aapl http evrsnvdex',
 'fuck appl',
 'apple need sort your phon',
 'onerepublic apple feet keep fli',
 'markzuckerbergf facebook cook apple sparring talk real connectivity http akfhp',
 'apple youre shit problem',
 '#apple supplier #skyworks still #swks #aapl #brcm #qcom #vti http ycxomnmhqo',
 'theyre naivana gotta kidding apple http geel',
 'what expect from apple aapl #aapl http vbdt',
 'confirmed bose speakers returning #apple stores after removal earlier this year h
 'apple iphone fair think really mean',
 'competition between #wireless carriers benefits #apple motley fool #aapl http',
 'does apple store georgetown sell refurb ipads just refurbs only available onlin',
 'battery already within minutes usage appl',
 'uberfacts think reasons your fuck appl',
 'long does really take phone ship appl',
 'brwnskin beauti cause fuck appl',
 'lanadelreystan kill yourself appl',
 'hiit coach weightsmate apple similar existing called cardio guru interestingly ma
 '#aapl pale hoarse steve jobs defends apple videotaped deposition ipod http snmpsx
 'apple disgrac',
 'apple ipad update absolute disaster nothing works right anymore safari crashes cc
 '#aapl apple trial continues without plaintiff http djdkc hkpz',
 'finally brooklyn getting #apple store #aapl http ipjmgjgqw http bsrlbax',
 'swiftkey excited named apple store best list this year http qlmti #appl',
 'best link usemuzli tech facebook mark zuckerberg goes after cook apple appadvice
 'pcaudiolabs apple emanon records apple premium product average product with premi
 'realised reason apple make huge phones because they safari allowing hide address
 'apple shitti',
 'finally updated this happened weird glitch please apple http lcogg yfez',
 'jopocop thestreet #apple gets price target boost back strong #iphone demand #aapl
 'make #apple special again #aapl http eiap kmelu',
 'swerviinnn apple clearly sign they doing favor',
```

```
'apple #macos #yosemite buggy piece shit makes miss #window',
'surprised there more talk about what looked like flash crash #aapl yesterday bigg
'shockingly imessage desktop fucked again great work apple seriously point with ne
'hell does this mean apple skype login even open http segfvq',
'tryng back snapchat store cooperating appl',
'apple adds flyover locations maps siri movie showtimes more countries blog http #
'apple fucking everyone name group chats their things please thank',
'dougluberts apple vers',
'apple your chargers suck',
'questions every #cio should about apple #iwatch http rsxt cmlz',
'saigeist most offensive thing lack trashcan emoji stop discrimination appl',
'apple fuck',
'jpdesloges higher price targets helping apple shares aapl #aapl http qtliixi',
```

This step involves removing all common words that don't actually convey any information. Examples of these words are “the”, “and” and “it”.

```
print(len(tokenized_tweet))

import nltk
nltk.download("stopwords")
from nltk.corpus import stopwords
tokenized_tweet = [t for t in tokenized_tweet if t not in stopwords.words("english")]
print(len(tokenized_tweet))
df["tweet"] = pd.Series(tokenized_tweet)
df.head()
```

```
179239
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
179195
```

	tweet	sentiment
0	yall needa step apple heynyla music snapchat s...	negative
1	what happened apple http fjexi #aapl #apple mo...	neutral
2	thank apple compile pics that communicate with...	positive
3	oddly uplifting story apple founder sold stake...	neutral
4	apple exchange iphone different color lmao cha...	neutral

```
df.dropna(axis=0, inplace=True)
df.tail()
```

	tweet	sentiment
179191	dear terrorist payal gawar what about modi kil...	neutral

```
X = df["tweet"]
```

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
Y = le.fit_transform(df['sentiment'])
```

Splitting data into testing and training

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X,Y, test_size=0.2, random_state=1)
```

If the model receiving difference length of sentences, that would be a problem. Because the length of all sentences have to be the same length, we will make it the same by padding.

```
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences

max_words = 5000
max_len=50

def tokenize_pad_sequences(text):

    tokenizer = Tokenizer(num_words=max_words, lower=True, split=' ')
    tokenizer.fit_on_texts(text)

    X = tokenizer.texts_to_sequences(text)

    X = pad_sequences(X, padding='post', maxlen=max_len)

    return X, tokenizer

X, tokenizer = tokenize_pad_sequences(df['tweet'])
```

After getting the padding sequences, we now need to get the vector representation of the words.

```
from sklearn.feature_extraction.text import CountVectorizer

count_vector = CountVectorizer(max_features = 5000,
                               preprocessor = lambda x: x,
                              tokenizer = lambda x: x)

X_train = count_vector.fit_transform(X_train).toarray()

X_test = count_vector.transform(X_test).toarray()
```

```
y = pd.get_dummies(df['sentiment'])
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.25, random
print('train:', X_train.shape, y_train.shape)
print('validation:', X_val.shape, y_val.shape)
print('test:', X_test.shape, y_test.shape)
```

```
train: (107510, 50) (107510, 3)
validation: (35837, 50) (35837, 3)
test: (35837, 50) (35837, 3)
```

```
print(y_train.shape)
```

```
(107510, 3)
```

3. MODEL

In our network we have these layers by using the KERAS Sequential API:

Embedding layer

> The embedding layer is a very important and first part of our network and allows us to learn the meanings of words based on the information of other words around them. We have embedding size used for the input sequence into a sequence of dense vectors.

Max Pooling

> We have Max Pooling before the Bidirectional LSTM layer to simplify the computation.

Bidirectional LSTM

> As mentioned above, we have Bidirectional LSTM which can learn from past and future. In bidirectional layers, we can try to learn the meaning of the word in both forward and backward information. By using LSTM network, our network can remember the previous information and also forget irrelevant information.

Dense Layer

> The last layer is a dense layer. Dense layer means that all nodes in this layer are connected to all nodes in the previous layer, so inputs from any LSTM unit can be included. We use the softmax output function and 3 nodes here to generate probabilities for each of the possible outputs for possible sensitivity.

```
from keras.models import Sequential
from keras.layers import Embedding, Conv1D, MaxPooling1D, Bidirectional, LSTM, Dense, Drop
from keras.metrics import Precision, Recall
from keras import losses

vocab_size = 5000
embedding_size = 32
epochs = 20
learning_rate = 0.1
decay_rate = learning_rate / epochs
momentum = 0.8

model= Sequential()
model.add(Embedding(vocab_size, embedding_size, input_length=50))
model.add(Conv1D(filters=32, kernel_size=3, padding='same', activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Bidirectional(LSTM(32)))
model.add(Dropout(0.4))
model.add(Dense(3, activation='softmax'))


print(model.summary())

model.compile(loss='categorical_crossentropy', optimizer="sgd",
              metrics=['accuracy', Precision(), Recall()])

history = model.fit(X_train, y_train,
                    validation_data=(X_val, y_val),
                    batch_size = 64, epochs=epochs, verbose=1)
```

```
Model: "sequential_9"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_9 (Embedding)     (None, 50, 32)            160000

 conv1d_9 (Conv1D)           (None, 50, 32)            3104

 max_pooling1d_8 (MaxPooling  (None, 25, 32)           0
 1D)

 bidirectional_9 (Bidirectio  (None, 64)               16640
 nal)

 dropout_9 (Dropout)         (None, 64)                0

 dense_9 (Dense)             (None, 3)                 195

=================================================================
Total params: 179,939
Trainable params: 179,939
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/20
1680/1680 [==============================] - 39s 21ms/step - loss: 1.0796 - accurac
Epoch 2/20
```

```
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0785 - accurac
Epoch 3/20
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0784 - accurac
Epoch 4/20
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0781 - accurac
Epoch 5/20
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0780 - accurac
Epoch 6/20
1680/1680 [==============================] - 34s 21ms/step - loss: 1.0779 - accurac
Epoch 7/20
1680/1680 [==============================] - 35s 21ms/step - loss: 1.0775 - accurac
Epoch 8/20
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0771 - accurac
Epoch 9/20
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0766 - accurac
Epoch 10/20
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0758 - accurac
Epoch 11/20
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0749 - accurac
Epoch 12/20
1680/1680 [==============================] - 35s 21ms/step - loss: 1.0733 - accurac
Epoch 13/20
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0713 - accurac
Epoch 14/20
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0692 - accurac
Epoch 15/20
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0665 - accurac
Epoch 16/20
1680/1680 [==============================] - 34s 20ms/step - loss: 1.0642 - accurac
Epoch 17/20
```

```
loss, accuracy, precision, recall = model.evaluate(X_test, y_test, verbose=1)
```

```
1120/1120 [==============================] - 10s 9ms/step - loss: 1.0527 - accuracy:
```

```
from keras.models import load_model

def predict_class(text):

    sentiment_classes = ['negative', 'neutral', 'positive']
    max_len=50


    xt = tokenizer.texts_to_sequences(text)

    xt = pad_sequences(xt, padding='post', maxlen=max_len)
    yt = model.predict(xt).argmax(axis=1)

    print('The predicted sentiment is', sentiment_classes[yt[0]])


predict_class(["@VirginAmerica seriously would pay $30 a flight for seats that didn't have
```

```
The predicted sentiment is negative
```

```
predict_class(["@VirginAmerica This is such a great deal! Already thinking about my 2nd tr
```

```
The predicted sentiment is positive
```