

# **Text Translation of Voice over Video Calls**

**A**

**Progress Report of project**

**Submitted in Partial Fulfilment of the Requirements for the Degree of**

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE & ENGINEERING**

**By**

**REETIKA SINGH (1509510065)**

**PRIYA KASHYAP (1509510057)**

**HIMANSHU VASHISHT (1509510037)**



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**MGM's College of Engineering & Technology, Noida**

**April, 2019**

## **DECLARATION**

I hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Signature :

Name : Reetika Singh

Roll no. : 1509510065

Date :

Signature :

Name : Priya Kashyap

Roll no. : 1509510057

Date :

Signature :

Name : Himanshu Vashisht

Roll no. : 1509510037

Date :

## **CERTIFICATE**

This is to certify that Project Report entitled “**Text Translation of Voice over Video Calls**” which is submitted by **Reetika Singh, Priya Kashyap** and **Himanshu Vashisht** in partial fulfilment of the requirement for the award of degree B. Tech. in Department of Computer Science & Engineering of Dr. A.P.J Abdul Kalam Technical University, is a record of the candidate own work carried out by him under my supervision. The matter embodied in this thesis is original and has not been submitted for the award of any other degree.

**Date:**

**(Ms Archana Sar)**

**Supervisor name with Signature**

### **External Viva**

Name of Examiner

Signature with Date

## ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the B. Tech Project undertaken during B. Tech. Final Year. I owe special debt of gratitude to **Ms Archana Sar**, Department of Computer Science & Engineering, MGM College of Engineering & Technology, Noida for her constant support and guidance throughout the course of our work. Her sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only her cognizant efforts that our endeavours have seen light of the day.

I also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project. Last but not the least, I acknowledge our friends for their contribution in the completion of the project.

Signature :  
Name : Reetika Singh  
Roll no. : 1509510065  
Date :

Signature :  
Name : Priya Kashyap  
Roll no. : 1509510057  
Date :

Signature :  
Name : Himanshu Vashisht  
Roll no. : 1509510037  
Date :

## **Abstract**

### **Text Translation of Voice over Video Calls**

A video call is a phone call using an Internet connection, sometimes called VoIP, which utilizes video to transmit a live picture of the person making the call. Video calls are made using a computer's webcam or other electronic device with a video-capable camera, like a smart phone, tablet, or video-capable phone system. On a computer additional software is typically required. One such program is Skype, which allows for video calls, as well as normal voice calls, using an Internet connection. On a smart phone if they have the capacity, smart phones usually have their own built-in application for video calls that vary from brand to brand. However, others may be downloaded and installed via your smart phone's app store.

A high bandwidth communication link is required to transmit and receive high quality images. There is a short time lag between speaking and receiving a response that can disrupt the natural flow of a conversation.

Our objective in is to enhance the communication in the video call by providing a text translation of the speech from user and transmitting it to the receiver's end. This will help in the development of a better communication system and revamp the current system of calling over the video.

## **List of Tables**

<b>Sr. No.</b>	<b>Table Number</b>	<b>Table Name</b>	<b>Page Number</b>
01.	1	Hardware interfaces	7
02.	2	Software interfaces	8
03.	3	Black Box Test Cases	46
04.	4	White Box Test Cases	47
05.	5	People involved during development phase	48

## List of Figures

Sr. No.	Figure Number	Figure name	Page Number
1	3.1	ER Diagram for Text Translation of Voice over Video Call.	9
2	3.2	Use Case Diagram for Text Translation of Voice over Video Call.	11
3	3.3	Sequence diagram for login and video calling.	16
4	3.4	Class diagram for Text Translation of Voice over Video calls	17
5	3.5(a)	Activity Diagram for Login & Sign-up	19
6	3.5(b)	Activity Diagram for Start Video Calling.	20
7	3.6.1	DFD of level 0	22
8	3.6.2	DFD of level 1	23
9	3.6.3(a)	DFD of level 2 for Contacts	23
10	3.6.3(b)	DFD of level 2 for Call logs	24
11	3.6.3(c)	DFD of level 2 for Settings	24
12	3.6.3(d)	DFD of level 2 for Notifications	25
13	3.6.3(e)	DFD of level 2 for Start Video Calling	25
14	4.1	Installing Flask	26
15	4.2	Import flask file and call a html file	27
16	4.3	Flask server	27
17	4.4	Sign-up page	28

18	4.5	Login Page	29
19	4.6	Front-End	29
20	4.7	Coding part of Flask Server	30
21	4.8	Online Speech Recognition using Google API.	31
22	4.9	Offline Speech Recognition using Sphinx engine	32
23	4.10	Speech to text in infinity loop	33
24	4.11	Audio recording	34
25	4.12.1	Server-Side Script	36
26	4.12.2	Client-Side Script	37
27	4.13	Message exchanging of client with server	38
28	4.14	Video Streaming from one client to server.	38
29	7.1	Pert Chart	51
30	7.2	Gantt Chart	52



## **Table of Contents**

## **Page no.**

Declaration	i
Certificate	ii
Acknowledgement	ii
Abstract	iv
List of Tables	v
List of Figures	vi
 CHAPTER 1 INTRODUCTION	
1.1 Problem Definition	1
1.2 Brief Introduction of the Project	1
1.2.1 Preliminary Investigation	2
1.2.2 Identification of need	2
1.3 Proposed Modules	3
1.4 Feasibility Study	4
1.4.1 Technical Feasibility	4
1.4.2 Economical Feasibility	5
1.4.3 Operational Feasibility	5
 CHAPTER 2 SOFTWARE REQUIREMENTS SPECIFICATIONS	
Introduction	6
2.1 Purpose	6

2.2	Intended Audience and Reading Suggestions	6
2.3	Project Scope	6
CHAPTER 3 DESIGNING		
3.1	ER Diagram	9
3.2	Use case Diagram	11
3.3	Sequence Diagram	16
3.4	Class Diagram	17
3.5	Activity Diagram	19
3.6	DFD	22
	DFD LEVEL 0	22
	DFD LEVEL 1	23
	DFD LEVEL 2	23
CHAPTER 4 IMPLEMENTATION MODULE and CODING		
4.1	Flask Program with Python	27
4.2	Setting up Flask Server	27
4.3	Sign-up Page	28
4.4	Login Page	28
4.5	Front-End	29
4.6	Coding	30

4.6.1	Coding for flask server	30
4.7	Google Speech Recognition (Online)	31
4.8	Speech Recognition (Offline)	32
4.9	Speech Recognition in Infinity loop	33
4.10	Coding for importing Audio File	34
4.11	Message Exchange	35
4.11.1	Server-Side Script	35
4.11.2	Client-Side Script	36
4.12	Running State of Client with Server	37
4.13	Code Efficiency	38
4.14	Optimization of Code	40
CHAPTER 5 VALIDATION CHECKS		
5.1	Verification checks	41
5.2	Validation checks	41
CHAPTER 6 IMPLEMENTATION and MAINTENANCE		
6.1	Module Description	42
6.1.1	Video Calling with GUI	42
6.1.2	Speech to Text	43
6.1.3	Network Connectivity	44
6.2	Testing	44

6.2.1 Unit Testing	44
6.2.2 Integration Testing	45
6.2.3 System Testing	45
6.2.4 Accessibility Testing	45
6.2.5 Functional Testing	45
6.3 Test Cases	45
6.3.1 Black Box Test Cases	45
6.3.2 White Box Test Cases	46
6.4 System Security Measures	47
6.5 Cost Estimation	48
CHAPTER 7 PERT and GANTT CHART	
7.1 Pert chart	50
7.2 Gantt chart	51
CHAPTER 8 FUTURE SCOPE	52
REFERENCES	54
GLOSSARY	55

# **Chapter 1**

## **Introduction**

### **1.1 Problem definition**

The development of video calling enhanced the capabilities of communication by adding a live video of the person with whom we are going to communicate. It brings a feeling of joy and pleasure on the faces of people when they see a live video of the other person. Video calls proved to be the boon in communication for the persons who are communicating over a long distance. There are a few irregularities which are very well known to us like the requirement of high frequency bandwidth, good connection speeds of the network, background noises in the channel and many more [1]. These disadvantages prove to be a boundation for video calls and hence prevent this method from becoming one of the best modes of communication.

The Objective with this project is to develop a system which is capable of utilizing the existing technology of video calls fastened with the technologies of Speech recognition and Machine learning so as to provide a better and efficient mode of communication which is free from the limitations of bandwidth and signal loss [1]. The idea is to provide the receiver or the other end user with a text translation of the senders or first users voice or speech. This will require capturing the voice of the sender which can easily be acquired at the time of providing the input. These translations will appear to the receiver as normal subtitles and they will help in increasing the communication efficiency.

### **1.2 Brief Introduction of the Project**

A video call is a phone call using an Internet connection, sometimes called VoIP, which utilizes video to transmit a live picture of the person making the call. Video calls are made using a computer's webcam or other electronic device which possesses camera that captures live video frames, like a smart phone, tablet, or video-capable phone system. On a computer additional software might be required. One such program is Skype, which allows for video calls, as well as normal voice calls, using an Internet connection. Our project aims to enhance the

communication in the video call by providing a text translation of the speech from user and transmitting it to the receiver's end. This will help in the development of a better communication system and revamp the current system of calling over the video.

### **1.2.1 Preliminary Investigation**

The current system of video calls is basically divided into two parts depending upon their implementation. The first is Point to Point also referred to as Video Chatting, that involves one to one interaction between the users. This happens in the case of a normal face time call, take the Skype software for instance. The other one is Multipoint also referred to as Video Conferencing, that involves one to many interactions between the users, take the Google Hangouts for instance [2]. Although these terms can be interchangeably used with each other they combine the idea of video calling. Due to increased time and space complexities modifications were needed in the first proposed model by John Logie Baird and AT&T's Bell Labs. PictureTel Corp came up with a better model with the scoping horizons of the ISDN network that were fast and much efficient [3]. ISDN (Integrated Services Digital Network) phone lines had greater bandwidth than the traditional phone lines, allowing for higher quality video calling systems to be developed.

The Video Chat Revolution Online video visit was blasting by the mid-2000s. By 2003, all the real texting customers support video calling. A variety of webcams showed up from sellers like Logitech, Microsoft, and Apple. Skype started offering video in 2005. Video calling had sneaked in under the noses of a once restless to-be-seen people shockingly. A more youthful age saw video visit as another interchange's worldview.

Skype added new feature in January 2019, Skype Translator that help to accommodate those with deafness or hard of hearing that benefited all users[10]. The technologies behind Skype Translator not only break down the language barrier, they also break down the hearing barrier.

### **1.2.2 Identification of need**

The video calling software was first introduced by A&T and Bell Labs with a motive to provide video telephony service for two remote devices over a network. These early models required extraordinary associations with work over telephone lines and they basically were excessively

costly, the controls were excessively troublesome, making it impossible to work and individuals simply didn't care for being seen via telephone. Since then the changes in the technology of telephone networks have led to the development of digital technologies like ISDN (Integrated Services Digital Network), Greater bandwidth lines allowing for higher quality video calling systems to be developed.

The current system of connectivity is capable to solve any problems of connectivity but it still lacks the advancements. It still requires higher bandwidth of the data connection lines to perform an effective communication, there comes a disruption or disturbance over smaller bandwidths. Also, the current system is not useful for some sections of the society like the people from the deaf sections of the society who cannot listen to the voice transmitted. The introduction of the text translation of voice into video calls will in turn provide them with the way to communicate. Also, the text translated packets will require a less amount of bandwidth to transmit over the network as compared to the video calling packets. These identified problems form the basic ground of the need for a better and effective application to be developed.

### **1.3 Proposed modules**

The project can be divided into a few basic modules which will assist the development of the project. These basic modules partition the project into small fragments. The modules proposed in this project are as follows: -

#### **Video Calling with GUI: -**

The prime module of the project is the module of video calls. This module is responsible for the development of the video calling system and on the top of which the GUI will be running so as to provide user with a better and easy understanding of the software. This will make use of HTML, CSS, VoIP, STUN Server, TURN Server, Python libraries.

**Speech to Text: -**

The module will be responsible for translation of the voice (speech) into the text. This can be easily done with the help of the various speech APIs like Google Speech Recognition, SAPI voice etc. The module plays a significant role in the development of the project.

**Network Connectivity: -**

The module will be responsible for the connection part in the software. The project itself requires a big part of network as it involves sending and receiving of the packets generated from the video calls as well as the translated voice. These packets are to be transferred over the network in order to get desired results. The network connectivity can be achieved by using servers and APIs.

**1.4 System Feasibility Study**

The most important step in our project after a project planning is to determine the viability of the idea. Feasibility study is done to determine whether the project is technically feasible or not, or project is feasible within estimated cost and whether it will be profitable or not.

There are three types of feasibility study:

- Technical feasibility
- Economic feasibility
- Operational feasibility

**1.4.1 Technical Feasibility:**

The project “TEXT TRANSLATION OF VOICE OVER VIDEO CALLS” is a web-based application. The main technologies and tools associated with this project are

1. Python
2. VoIP
3. Speech API



#### 4. Clock synchronization technique

Each of the projected technologies are open source and the technical skills required are manageable. Time limitations of the project development and the ease of implementing using these technologies are synchronized.

Initially the web site will be hosted in a free web hosting space, but for later implementations it will be hosted in a paid web hosting space with a sufficient bandwidth.

From these it's clear that the project "TEXT TRANSLATION OF VOICE OVER VIDEO CALLS" is technically feasible.

##### **1.4.2 Economical feasibility:**

Being a web application, the project will have an associated hosting cost. The system will follow the freeware software standards. No cost will be charged from the potential customers. Bug fixes and maintaining tasks will have an associated cost. Software require are basically all open source e.g. Python, VoIP, speech API.

Beside the associated cost, there will be many benefits for the customers. Especially, to avoid the noise in the background, works on the high/low bandwidth and useful for deaf person also.

From these it's clear that the project" TEXT TRANSLATION OF VOICE OVER VIDEO CALLS" is economical feasibility.

##### **1.4.3 Operational Feasibility:**

With the development of the user-friendly GUI, the system can be easily and effectively operated by the end user. The system will support easy interaction and functionality and the user need not to have a pre understanding of the software. The system will perform tasks in a better manner and in less time by making use of the APIs like WebRTC which are helpful in creating video calling software's.

# **CHAPTER 2**

## **Software Requirements Specifications**

### **2 INTRODUCTIONS**

#### **2.1 PURPOSE**

The purpose of this document is to build a system for end user to provide a new feature in a video calls and is capable for utilizing the existing technology of video calls in a better and efficient mode of communication which is free from the limitations of bandwidth and signal loss[4].

The idea is to provide the receiver with a text translation of the sender's voice or speech.

#### **2.2 INTENDED AUDIENCE AND READING SUGGESTIONS**

This project is a prototype for the "Text Translation of Voice Over Video Calls" and it can be used by anyone and from anywhere. This has been implemented under the guidance of college professors. This project is useful for the persons who are communicating over long distances and who faces many difficulties (like lack of networks, low bandwidth frequency, bad quality resolution and many more) during video calls. The intended audience in this project can be anyone who communicates through video calls.

#### **2.3 PROJECT SCOPE**

The purpose of the making this project is to create a convenient and easy-to-use application for each and every person. The system is based on a client to client interaction, in which the client server as a sender as well as a receiver. Both the sender as well as receiver will establish a peer to peer connection among each other over a network and this will help them to share the data packets on this network. Also, the system makes use of the translations in the audio as well as the in the text format. This translation will be provided to enhance the communication process. By providing the text translation we aim at providing the communication even at lower bandwidths which is not possible in the case of normal video calls. The users from various age

groups can easily use the application due to the simplicity of the User Interface and a proper responsive mechanism that will be able to respond as per the user's command. The software will perform effectively and will deliver the results accurately in a predefined time limit. Above all, we hope to provide a comfortable user experience along with the best pricing available.

## EXTERNAL INTERFACE REQUIREMENTS

### User Interfaces

- Front-end software: HTML and CSS, PYQT
- Back-end software: Python and various API's

### Hardware Interfaces

Requirements	Specifications (Minimum)
• Processor	: Core i3 or higher
• RAM	: 4GB or higher
• Router	: 802.11n or 802.11ac bandwidth (with a range of 2.4 to 5GHz)
• System Type	: x64-based processor
• Webcam	: Connection type: wireless high speed USB 2.0, Image capture (4:3 SD) Video capture (4:3 SD)
• Microphone	: Computer: Windows PC with Intel/AMD x86-64 processor Memory: - 4 GB of RAM minimum Connection/power: - USB port on computer

Table 1: Hardware interfaces

### Software Interfaces

Following are the software used for the “TEXT TRANSLATION OF VOICE OVER VIDEO CALLS.”

Software used	Version	Description
Operating system	64-bit Operating System	We have chosen Windows operating system for its best support and user friendliness.
Tools used	IDE, PyCharm, Spyder	To implement the project, we have chosen these tools for its more interactive support.
Database	Mnesia db.	To provide data backup we have selected an integrated distributed database
Protocols	TCP/IP FTP HTTP IPSec XMPP	Extensible Messaging and Presence Protocol servers can be used to eliminate the need to have multiple client servers for information exchange. XMPP is a standardized version of open-standards based protocol known as Jabber

Table 2: Software interfaces

## Chapter 3

### System Designing

#### 3.1 ER diagram:

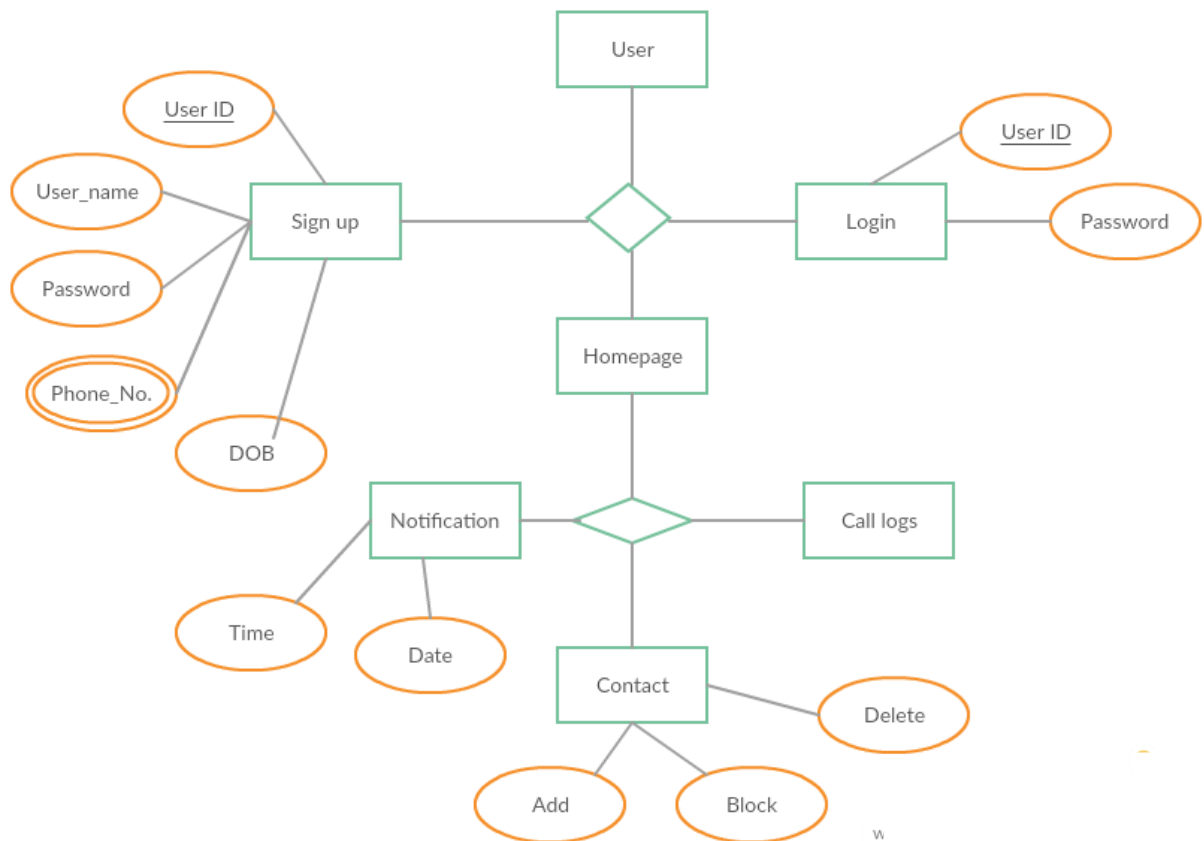


Fig 3.1 ER Diagram for Text Translation of Voice over Video Call [6].

The fig 3.1 show the entity relationship between the user, homepage, login and sign-up. So that user can easily interact with each of them.

In this ER diagram we take seven different entity which is related to one another. Here user have a relationship with home page, login and sign-up whereas homepage have a relation with notifications, contact and call logs.

- 1) Login consists two different attributes like user id and password where,

User id: varchar (primary key)

Password: varchar ()

- 2) Sign-up consists five different attributes that is user id, password, user name, phone no and date of birth(dob).

User id: varchar(primary key)

User name: char()

Password: varchar()

Phone no.: int() which is multivalued

- 3) Notification consists of two different attributes that is date and time.

Date: date()

Time: time()

- 4) Contact consist two different attributes that is user name and phone number. User can easily can add, block or delete contact from list.

Phone no.: int()

User name: varchar()

### 3.2 Use Case Diagram:

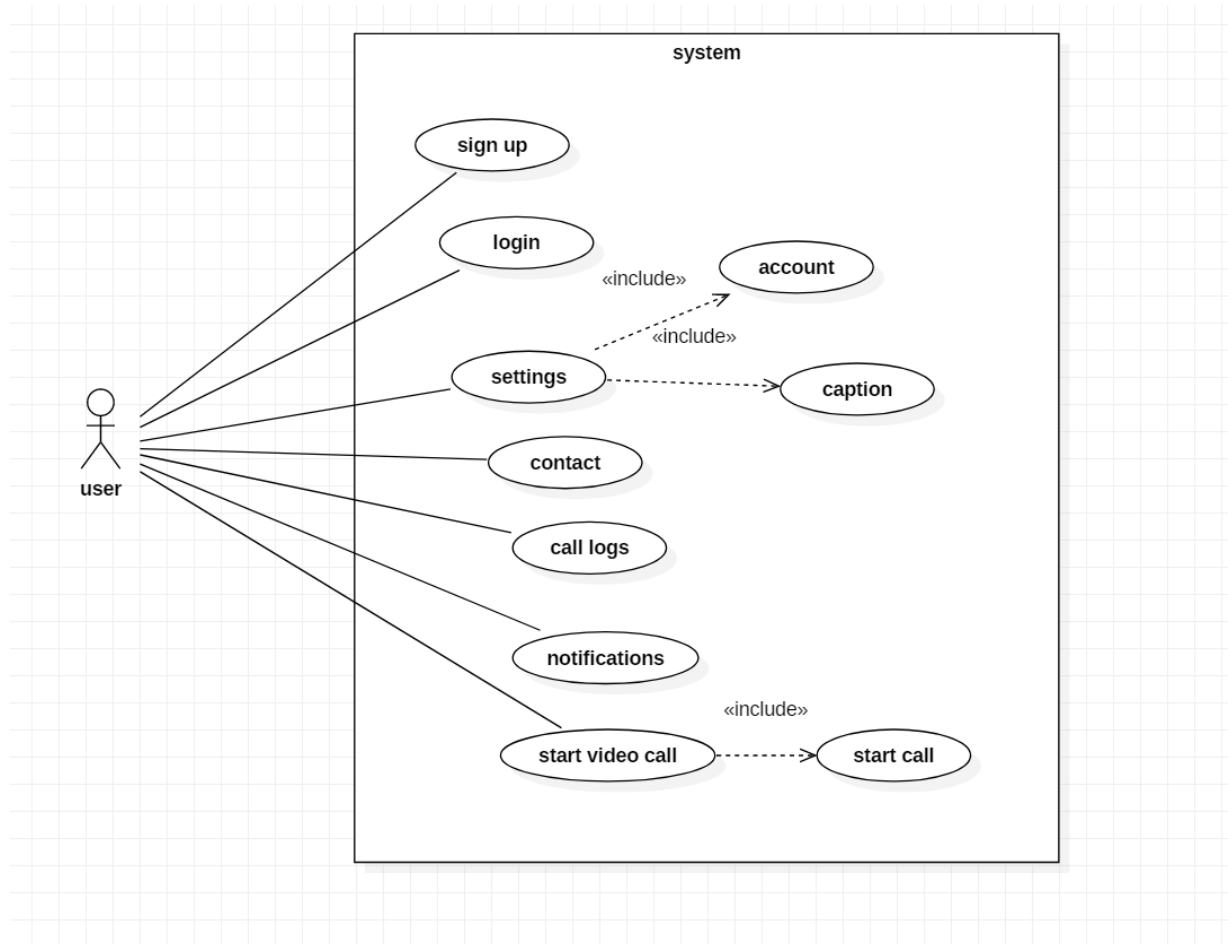


Fig 3.2: Use Case Diagram for Text Translation of Voice over Video Call

#### **Login:**

- a) **Introduction:** This use case describes how the user login into personal manager.
- b) **Actor:** user
- c) **Pre-condition:** user should have user id and password (sign-in).
- d) **Post-condition:** if use case successful, the actor is logged into the system. If not, the system state is unchanged.
- e) **Basic flow:** this use case start when the actor wishes to login the website.
  - i) System request that the actor enter user id and password.
  - ii) The actor enters the id and password.

- iii) System validated id and password, id correct allow the actor to logs into the system.
- f) **Alternate flow:** invalid id and password system display error message then user cancel the login and back to signup process.

### **Contact:**

- a) **Introduction:** allow user to maintain his contact (name, number).
- b) **Actor:** user
- c) **Precondition:** user must be logged onto the system before this use case begin.
- d) **Post condition:** if use case successful, user information is added/updated from the system. Otherwise system state is unchanged.
- e) **Basic flow:** user chooses one of the operations to be performed like add/update/delete. According to the option chosen by the user the sub flow will work.
- i) If user wants to add then add sub flow will work.
- ii) If user wants to update then update sub flow will work.
- iii) If user wants to delete then delete sub flow will work

ADD sub flow:

- System asks to fill the required filled.
- User enters the detail and click on save button.
- The data is added to the database.

UPDATE sub flow:

- User selects the information to be updated and click on update button.
- The new information is saved in database.

DELETE sub flow:

- User select the information to be deleted and click on delete button.
- The information is removed from database.



- f) **Alternate Flow:** For update and add the detail entered by the user can be wrong. Then the system remains same just show the error message the user can cancel the update. The user can cancel the update or delete of the information.

### **Call logs:**

- a) **Introduction:** allow user to maintain the call logs of the contacts (recent calls, outgoing calls, incoming calls, missed calls, search).
- b) **Actor:** user
- c) **Pre-condition:** user must be logged onto the system before this use case begin
- d) **Post-condition:** if use case successful, user information is added/updated from the system. Otherwise system state is unchanged
- e) **Basic flow:** user chooses one of the operations to be performed like add/update/delete.

According to the option chosen by the user the sub flow will work.

- i) If user wants to delete then delete sub flow will work.

DELETE sub flow:

- User select the information to be deleted and click on delete button.
  - The information is removed from database.
- f) **Alternate Flow:** The user can cancel the delete of the information.

### **Start Video call:**

- a) **Introduction:** allow user to call from his contact. This use case include start call.
- b) **Actor:** user
- c) **Pre-condition:** user must be logged onto the system before this use case begin.
- d) **Post-condition:** if the use case successful, user's information is added/updated from the system. Otherwise system is unchanged.

- e) **Basic flow:** if there is the network connectivity then the call will be placed over the network to the specific user otherwise the call operation will be failed as a result of which the call will not be placed.
- f) **Alternate Flow:** It checks the network connectivity and again try to place a video call.

### **Settings:**

- a) **Introduction:** allow user to do changes in the settings. This use case includes account and caption.
- b) **Actor:** user
- c) **Pre-condition:** user must be logged onto the system before this use case begin.
- d) **Post -condition:** if use case successful, the actor is logged into the system. If not, the system state is unchanged.
- e) **Basic flow:** user can do change in the settings and chooses one of the operations to be performed like add/update/delete. According to the option chosen by the user the sub flow will work.
  - i) If user wants to add then add sub flow will work.
  - ii) If user wants to update then update sub flow will work.
  - iii) If user wants to delete then delete sub flow will work.

ADD sub flow:

- User can add their information into the account.

UPDATE sub flow:

- User can update his/her account according to them and click on update.
- The information is then stored in the database.

DELETE sub flow:

- User can also delete the information which he/she does not want to show.
- The information is removed from database.

- f) Alternate Flow:** information entered by user can be wrong the system remain same and user refresh the page and again do changes in the settings.

### 3.3 Sequence Diagram:

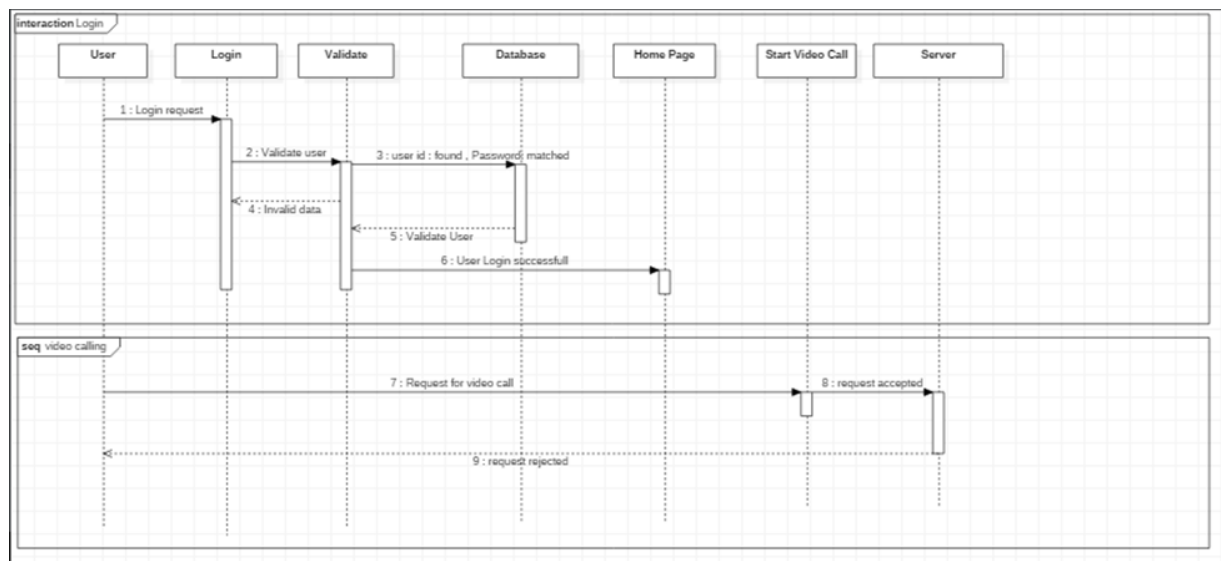


Fig 3.3: Sequence diagram for login and video calling.

#### Main steps:

##### 1. LOGIN

- Users have to provide its user id and password.
- Then the credential goes to database for verification.
- If user-id and password are valid then user is directed to home page otherwise an error message is shown. User needs to retry for login.

##### 2. Video Calling

- In Homepage user invokes start videocall.
- If connection is established user goes to server for sending and receiving data otherwise if connection is rejected user have to retry for calling later.

### 3.4 Class Diagram:

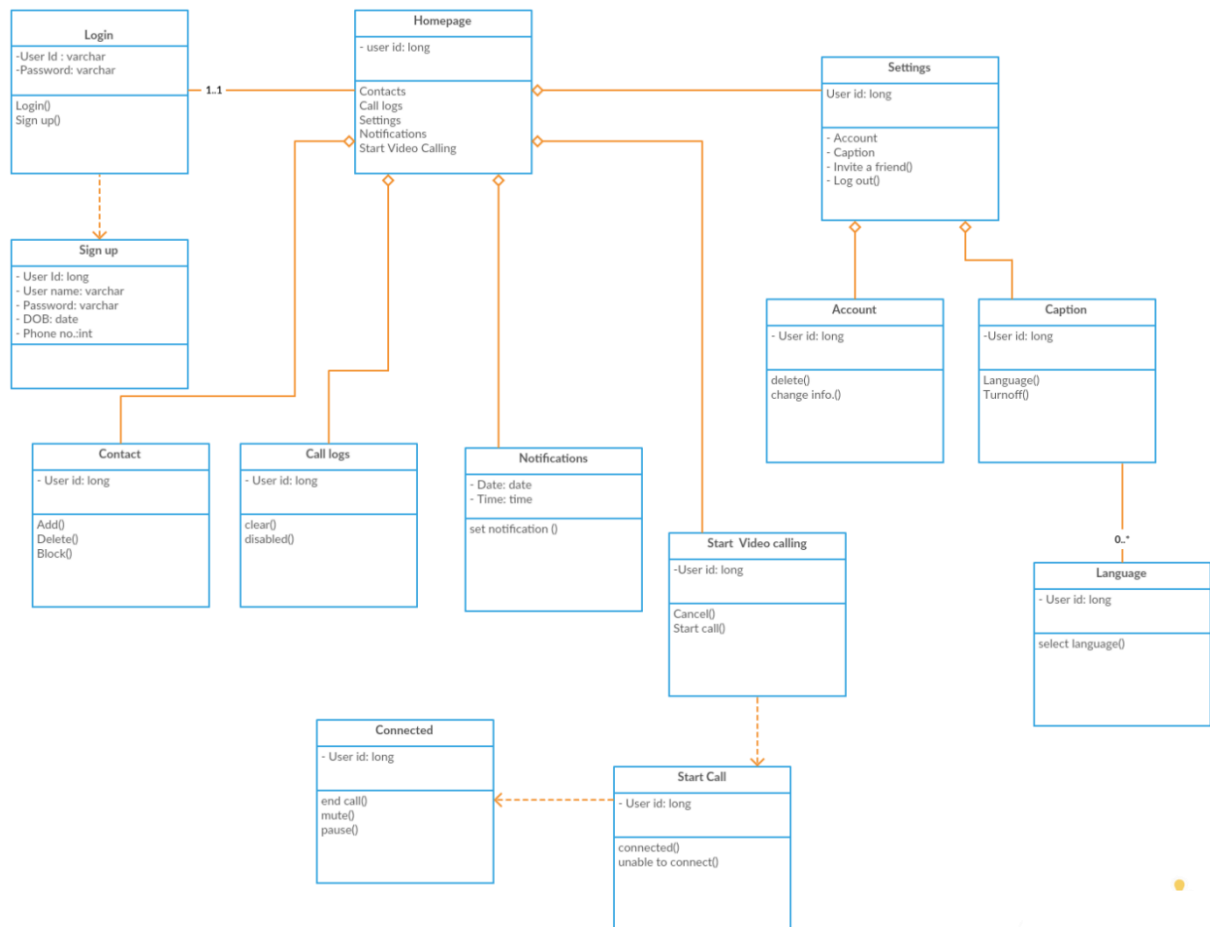


Fig 3.4: Class diagram for Text Translation of Voice over Video calls[6].

There are following classes in this voice to text over video calls:

- 1) **Home page:** here user can select one of the functions. This class contain “has a relationship” with the following classes:
  - i) Contact
  - ii) Call logs
  - iii) Setting
  - iv) Notifications
  - v) Start video calling

- 2) **Login:** Here user can enter its user-id and password and then directed to home page. It has “association” with home page. And has dependency on signup
- 3) **Contact:** using this class different type of contacts are added, updated & deleted etc.
- 4) **Call logs:** In this class logs of all calls are kept which can be cleared and disabled.
- 5) **Settings:** here user can select one of the functions. This class contain “has a relationship” with the following classes:
  - **Account:** here user can update their personal details and can delete existing account
  - **Caption:** here user can change language of captions and disable captions during video calling
- 6) **Notifications:** here various notifications of are displayed with date and time.
- 7) **Start video calling:** here the user can start of cancel a video call .
  - **Start call:** user selects to connect the call and the channel is established hence the calling is invoked otherwise the channel is busy and call is unable to connect
  - **Connected:** call is connected and channel is open. Now user can end call , mute call or pause call.

### 3.5 Activity Diagram:

Activity diagram for login Page:-

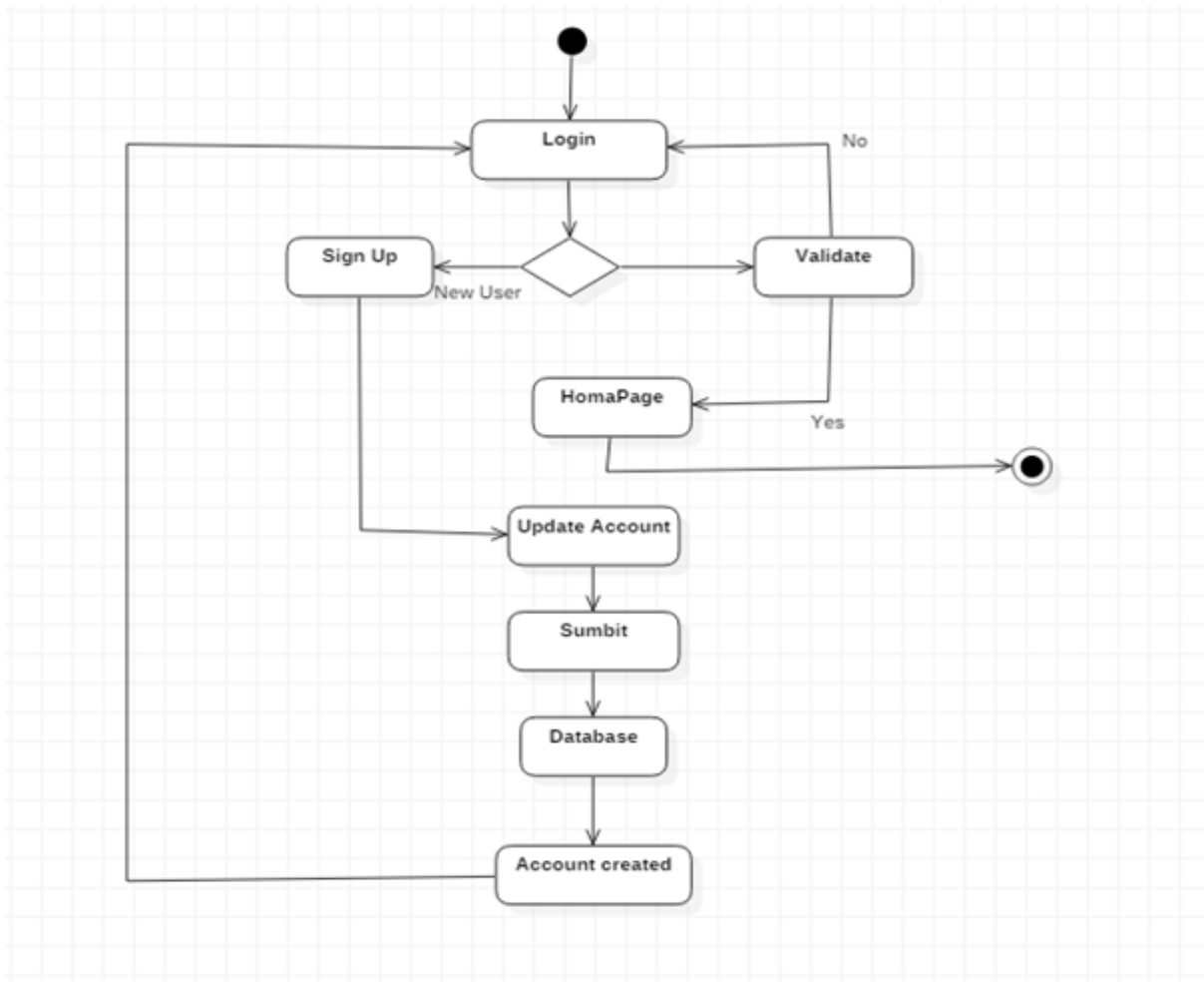


Fig 3.5 (a): Activity Diagram for Login & Sign-up

#### Main Steps:

- 1) The starting point is when the user login to its account and if it does not have the account then user have to sign up for new account.
- 2) For Signing up user is needed to provide personal details and Submit to the Database for Account Creation.
- 3) After creating account, the user is directed to the login page where the userId and password is required to access the homepage.

4) After validation the user is directed to Homepage.

**For video calling:-**

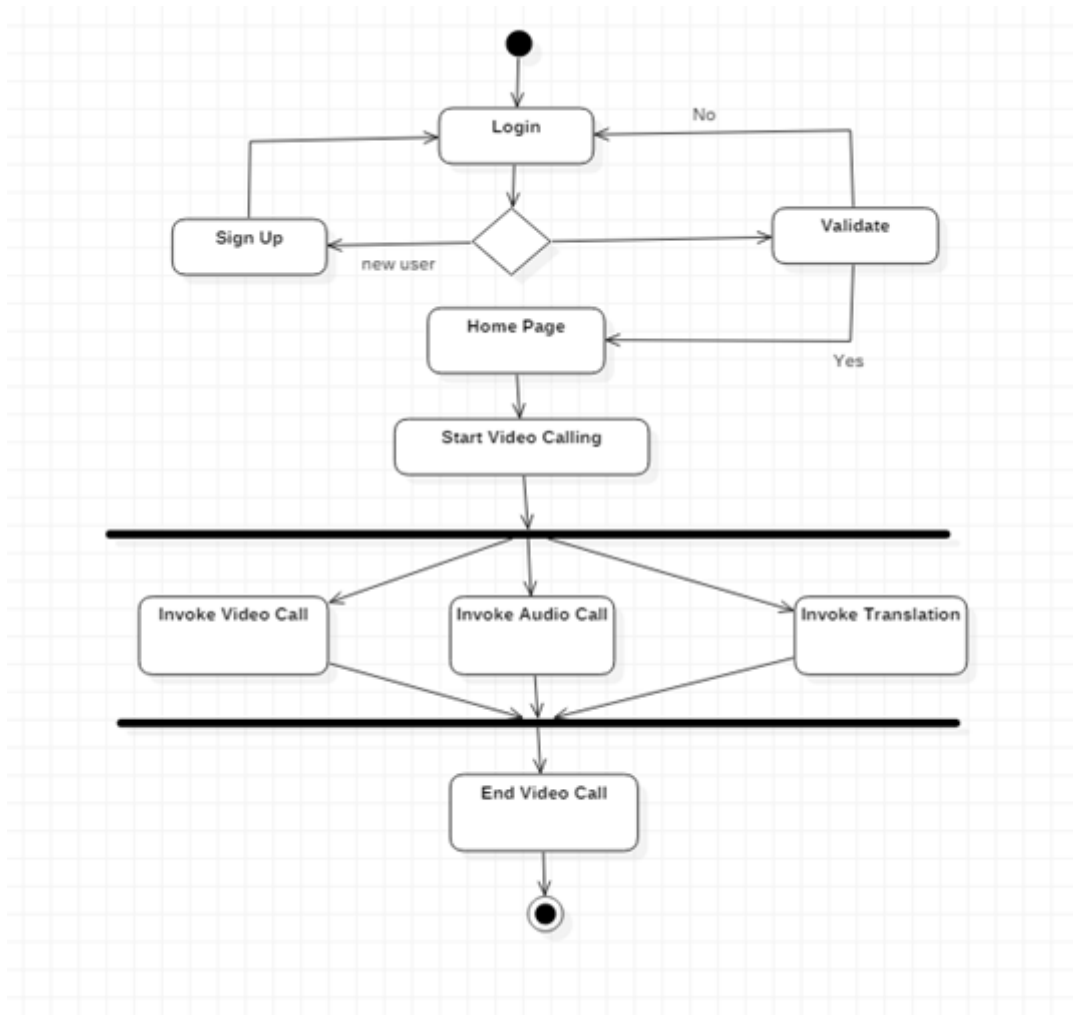


Fig 3.5(b): Activity Diagram for Start Video Calling.

**Main Steps:**

- 1) In login page user is required to give user-id and password to access homepage.
- 2) User-id and password are validated with the database and the access to the homepage is granted to the user.
- 3) Then after login user is directed to home page where they get option for video calling mention above.



- 4) After accessing the video calling option, the user is connected to the server and invokes three functions which are Invoke Video Call, Invoke Audio Call, and Invoke Translation.
- 5) At last the user Ends the Video Call.

### 3.6 Data Flow Diagram (DFD):

#### a) LEVEL 0

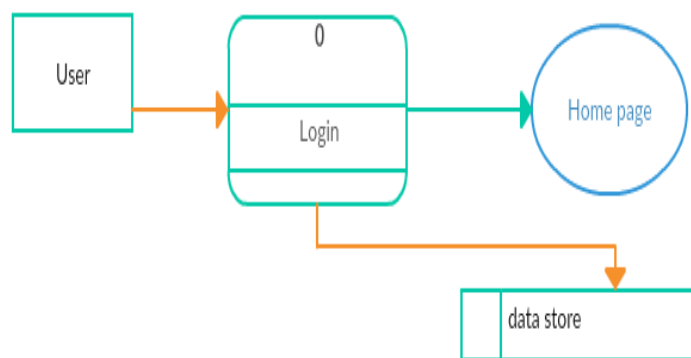


Fig 3.6.1: DFD of level 0 [6].

At this level the Login process is performed and the user is directed to home page and the data is saved in database.

#### b) LEVEL 1

At level 1 when the user is at home page they give five options and the user can select one of them as shown in fig 3.6.2 and these five options are:

- i) Contact
- ii) Call logs
- iii) Settings
- iv) Notifications
- v) Start Call

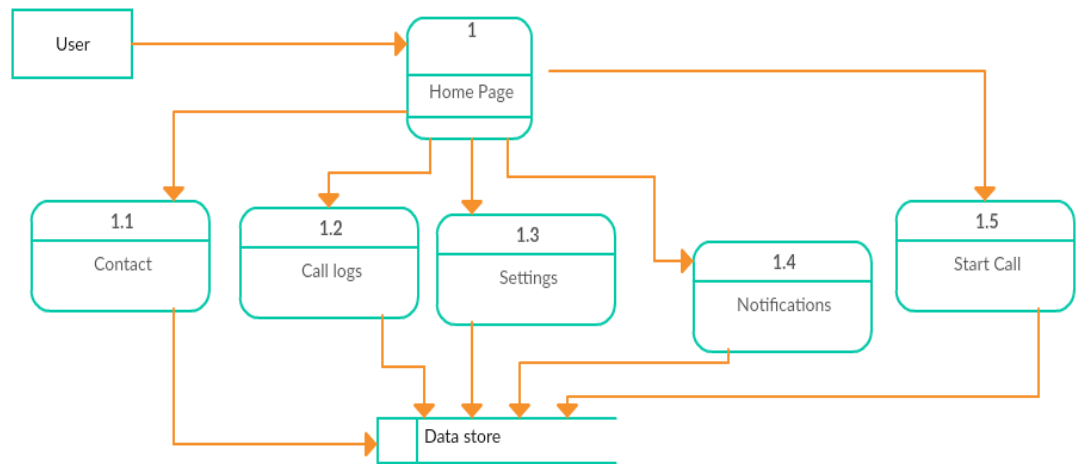


Fig 3.6.2: DFD of level 1[6].

c) LEVEL 2

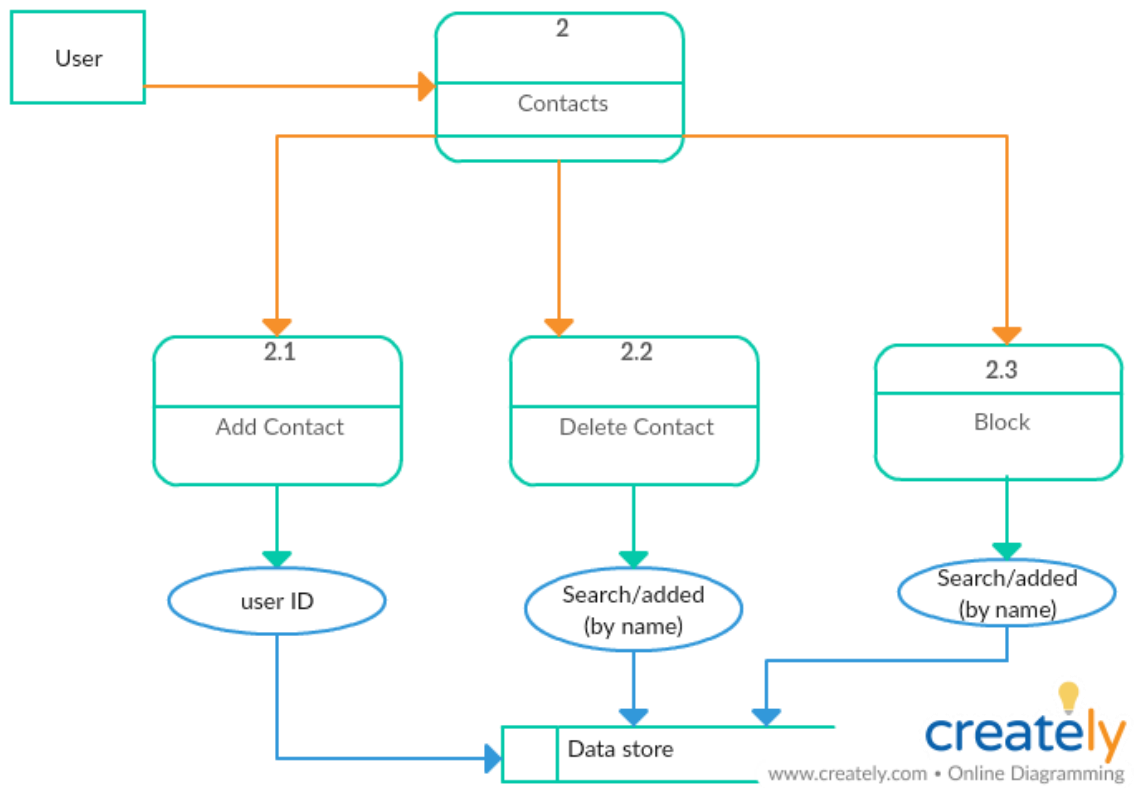


Fig 3.6.3: (a) DFD of level 2 for Contact

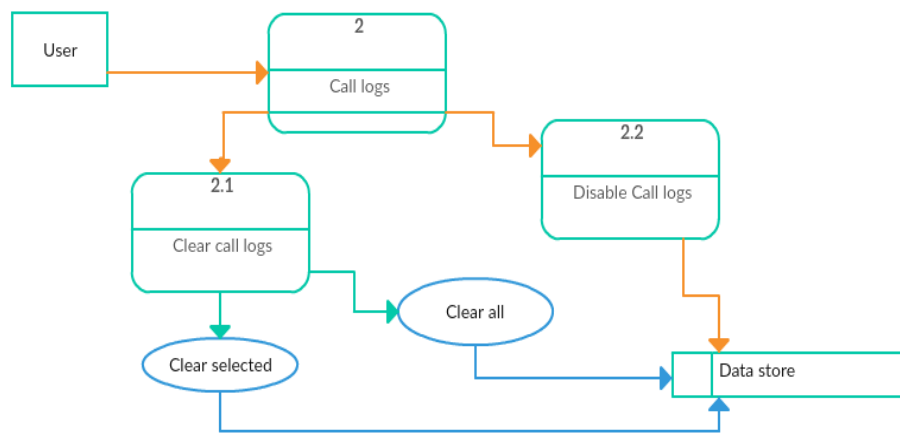


Fig 3.6.3: (b) DFD of level 2 for Call logs[6].

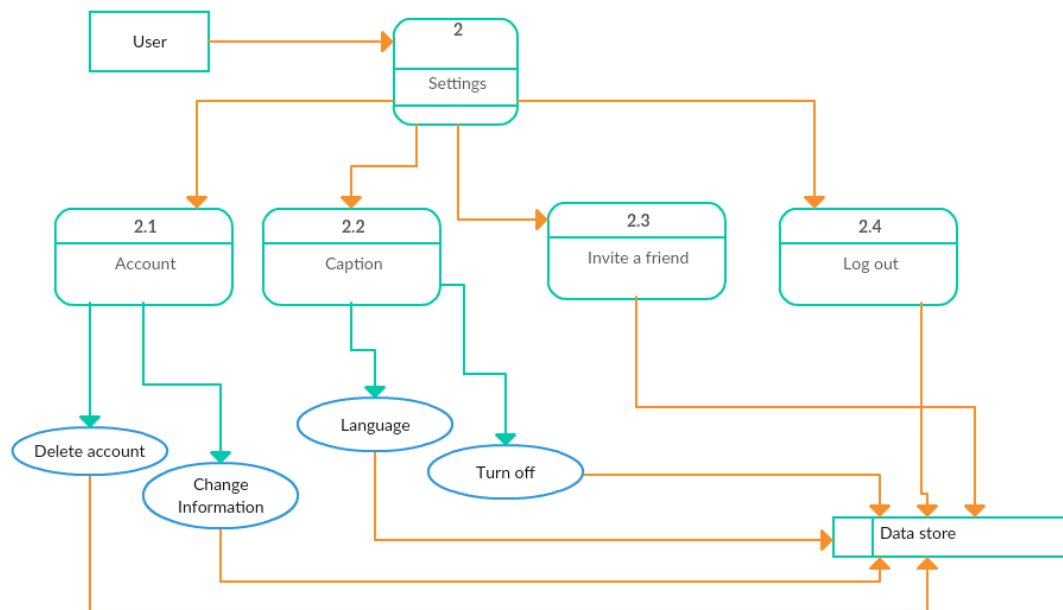


Fig 3.6.3: (c) DFD of level 2 for Settings[6].

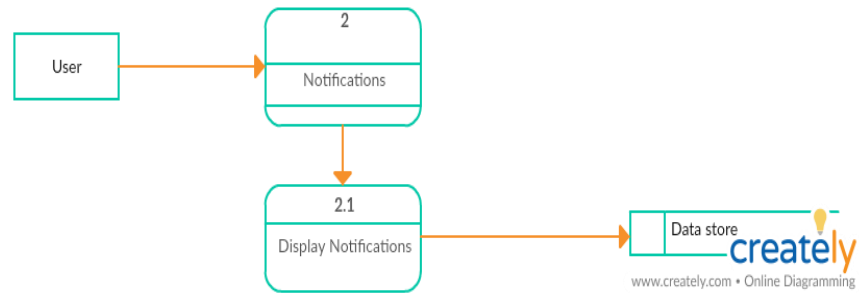


Fig 3.6.3: (d) DFD of level 2 for Notifications[6].

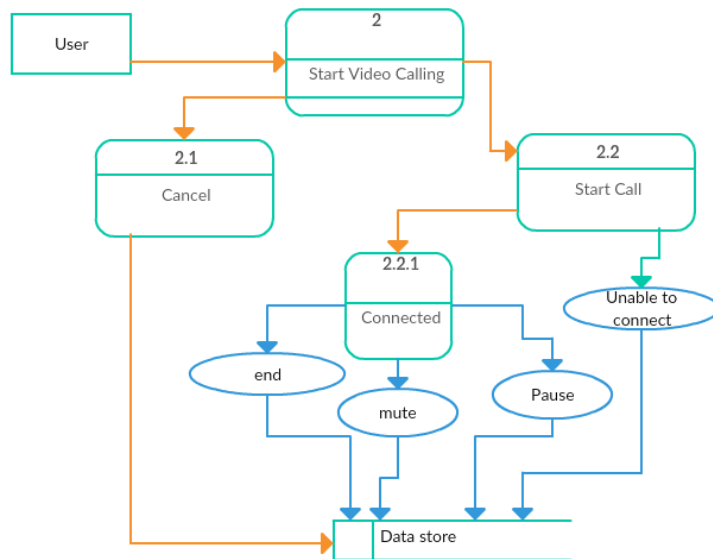


Fig 3.6.3: (e) DFD of level 2 for Start Video Calling[6].

# Chapter 4

## Implementation Module and Coding

Here the modules are described with the help of mock-up screen.

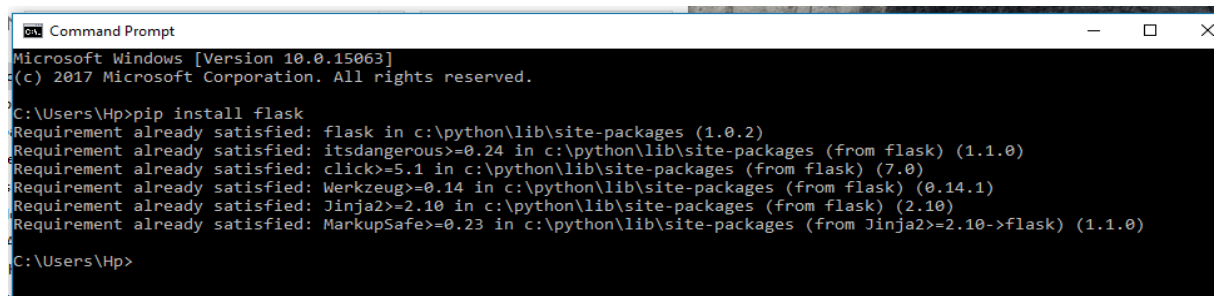
### Module 1:- Video Calling with GUI

The prime module of the project is the module of video calls. This module is responsible for the development of the video calling system and on the top of which the GUI will be running so as to provide user with a better and easy understanding of the software. This will make use of WebRTC, VoIP, STUN Server, TURN Server, Python libraries.

I am using “Flask” in python which is microframework to create a web application with MySQL as the back end[8].

#### Required Installations:

##### 1. Flask



```
Command Prompt
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\Hp>pip install flask
Requirement already satisfied: flask in c:\python\lib\site-packages (1.0.2)
Requirement already satisfied: itsdangerous>=0.24 in c:\python\lib\site-packages (from flask) (1.1.0)
Requirement already satisfied: click>=5.1 in c:\python\lib\site-packages (from flask) (7.0)
Requirement already satisfied: Werkzeug>=0.14 in c:\python\lib\site-packages (from flask) (0.14.1)
Requirement already satisfied: Jinja2>=2.10 in c:\python\lib\site-packages (from flask) (2.10)
Requirement already satisfied: MarkupSafe>=0.23 in c:\python\lib\site-packages (from Jinja2>=2.10->flask) (1.1.0)

C:\Users\Hp>
```

Fig 4.1: Installing Flask

## 4.1 Flask program with python

The Python Flask app will have a new URL route. It helps python to interact with HTML and we can easily design the web frame by using flask.



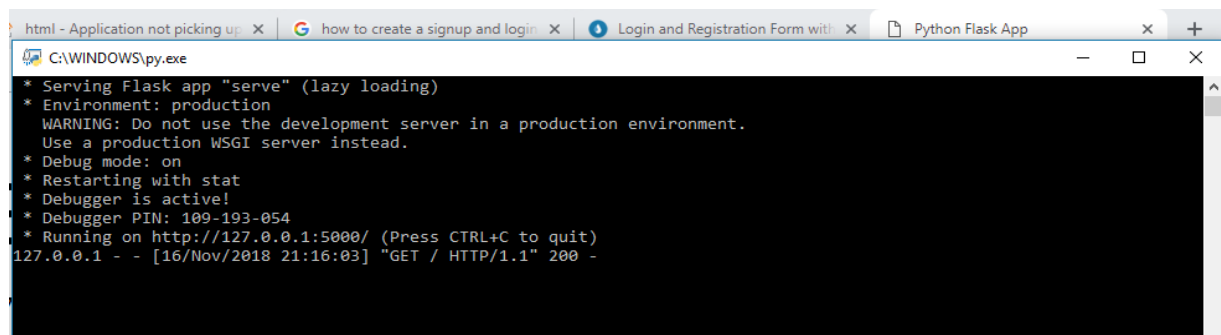
```
flaski.py - C:\Users\Dell\Downloads\total\flaski.py (3.6.5)
File Edit Format Run Options Window Help
from flask import Flask, render_template, Request
import output
import mediapy
app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')
@app.route('/startcall/')
def startcall():
    return render_template('startcall.html',mediapy.audio())

if __name__ == '__main__':
    app.run(debug=True)
```

Fig 4.2: Import flask file and call a html file

## 4.2 Setting up Flask server



```
C:\WINDOWS\py.exe
* Serving Flask app "serve" (lazy loading)
* Environment: production
WARNING: Do not use the development server in a production environment.
Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 109-193-054
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [16/Nov/2018 21:16:03] "GET / HTTP/1.1" 200 -
```

Fig 4.3: Flask server

### 4.3 SIGN-UP Page

Signup is necessary for the login if the users will not signup then it will not login as explained in the login module. Signup is completed by using the following information of the user as Username, User\_Id, Password, DOB and Phone Number.

As the signup is completed successfully and the user can then login and the next activity will be appeared.

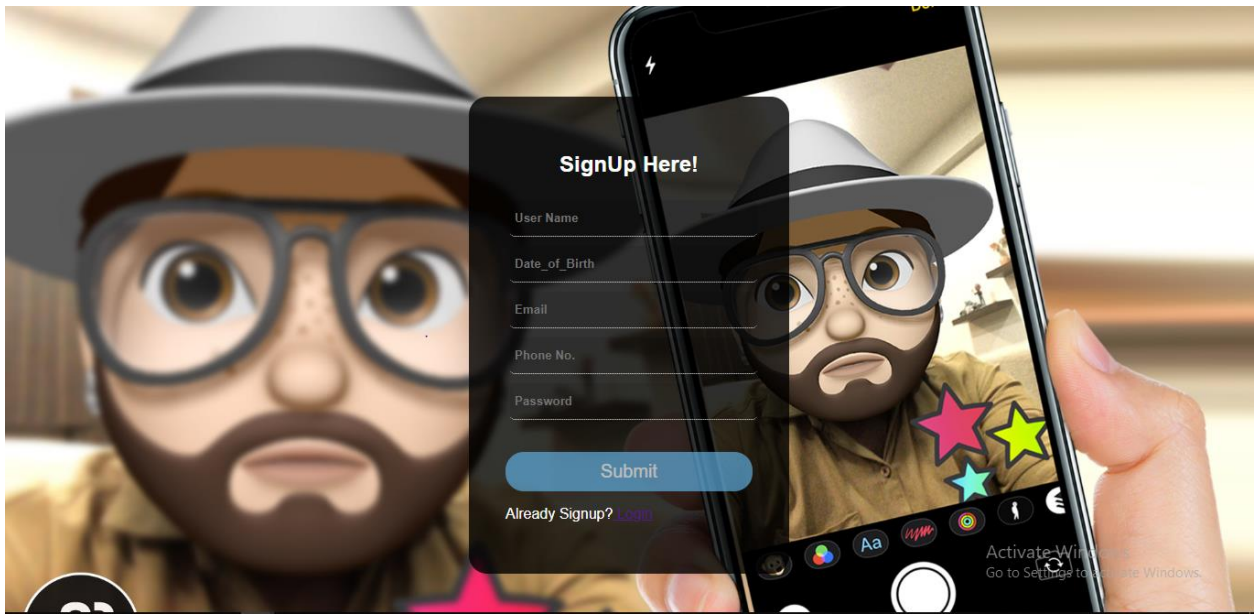


Fig 4.4: Sign-up page

### 4.4 LOGIN Page

This is the first activity (login) of our website. In this activity the user must enter his/her username and password (credentials which are validated at the time of registration). If the user is trying for the first time or is a new to application then he/she should get register first. The login activity will be verifying the credentials entered by the user from the database, if it is found correct then after verifying it the user is redirected to next activity. If the credentials entered are wrong then he/she may try again. Here we provide two button one for SAVE and one for DISCARD.



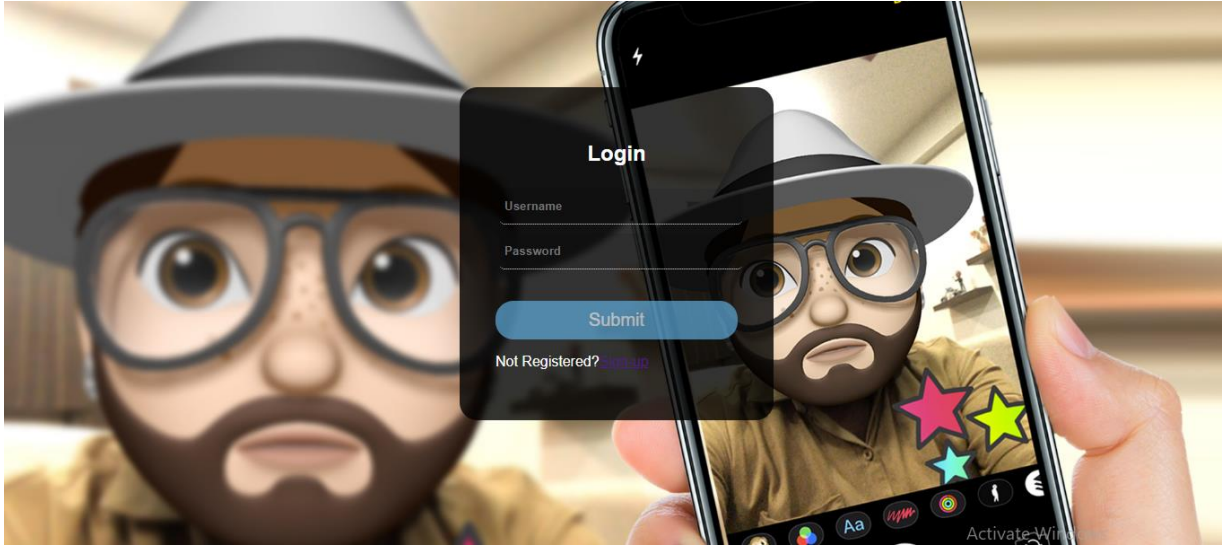


Fig 4.5: Login Page

## 4.5 Front-End

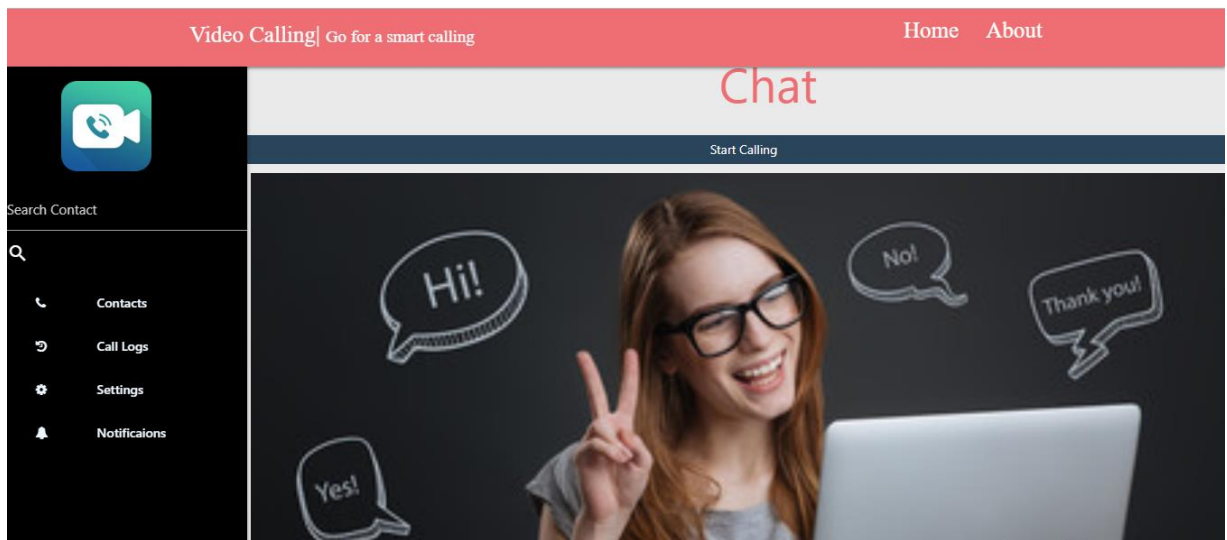
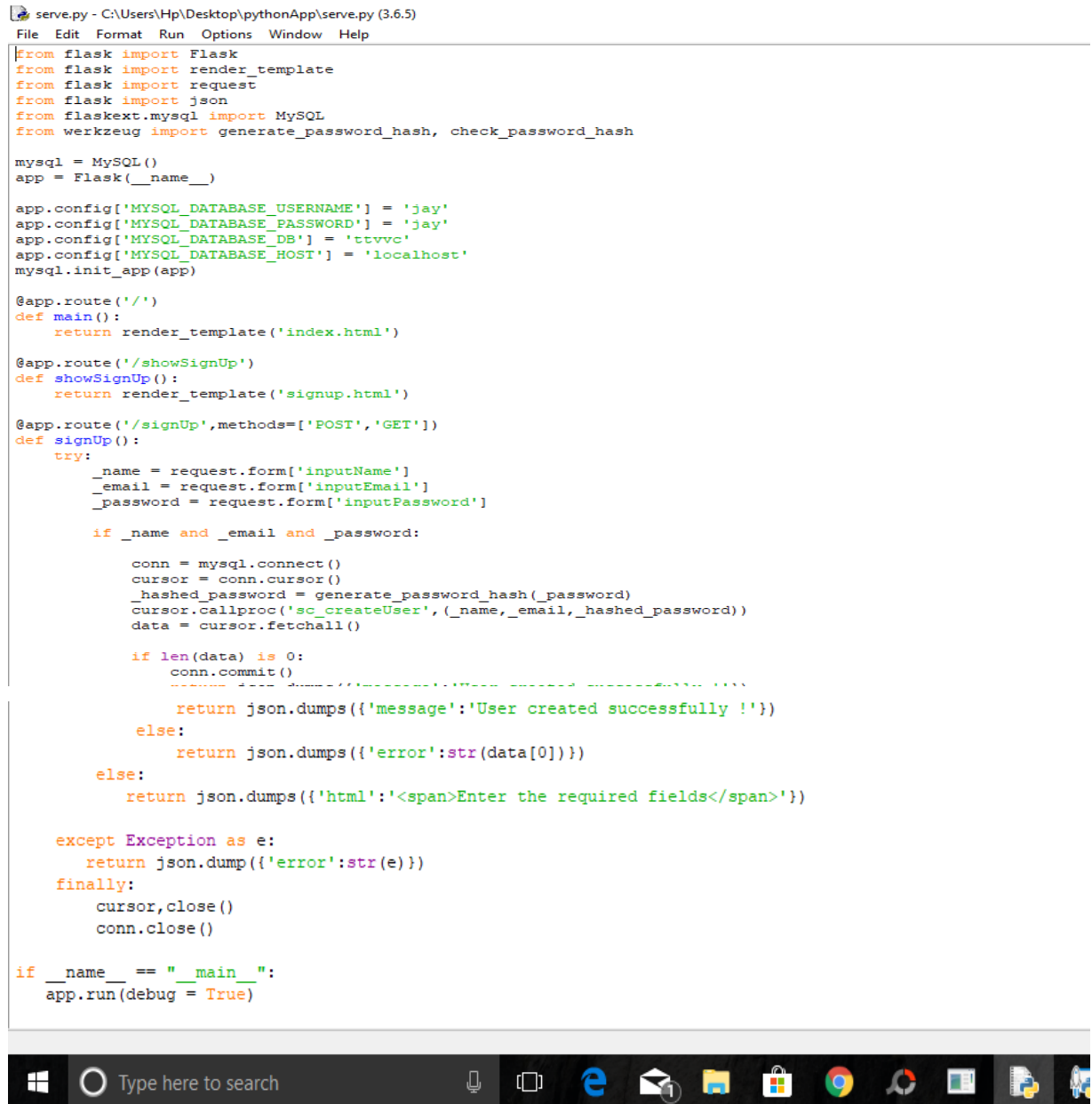


Fig 4.6: Front-End

## 4.6 Coding

### 4.6.1 Coding for flask server:



```
serve.py - C:\Users\Hp\Desktop\pythonApp\serve.py (3.6.5)
File Edit Format Run Options Window Help

from flask import Flask
from flask import render_template
from flask import request
from flask import json
from flaskext.mysql import MySQL
from werkzeug import generate_password_hash, check_password_hash

mysql = MySQL()
app = Flask(__name__)

app.config['MYSQL_DATABASE_USERNAME'] = 'jay'
app.config['MYSQL_DATABASE_PASSWORD'] = 'jay'
app.config['MYSQL_DATABASE_DB'] = 'ttvvc'
app.config['MYSQL_DATABASE_HOST'] = 'localhost'
mysql.init_app(app)

@app.route('/')
def main():
    return render_template('index.html')

@app.route('/showSignUp')
def showSignUp():
    return render_template('signup.html')

@app.route('/signUp', methods=['POST', 'GET'])
def signUp():
    try:
        _name = request.form['inputName']
        _email = request.form['inputEmail']
        _password = request.form['inputPassword']

        if _name and _email and _password:

            conn = mysql.connect()
            cursor = conn.cursor()
            _hashed_password = generate_password_hash(_password)
            cursor.callproc('sc_createUser', (_name, _email, _hashed_password))
            data = cursor.fetchall()

            if len(data) is 0:
                conn.commit()
                return json.dumps({'message': 'User created successfully !'})
            else:
                return json.dumps({'error': str(data[0])})
        else:
            return json.dumps({'html': '<span>Enter the required fields</span>'})

    except Exception as e:
        return json.dumps({'error': str(e)})
    finally:
        cursor.close()
        conn.close()

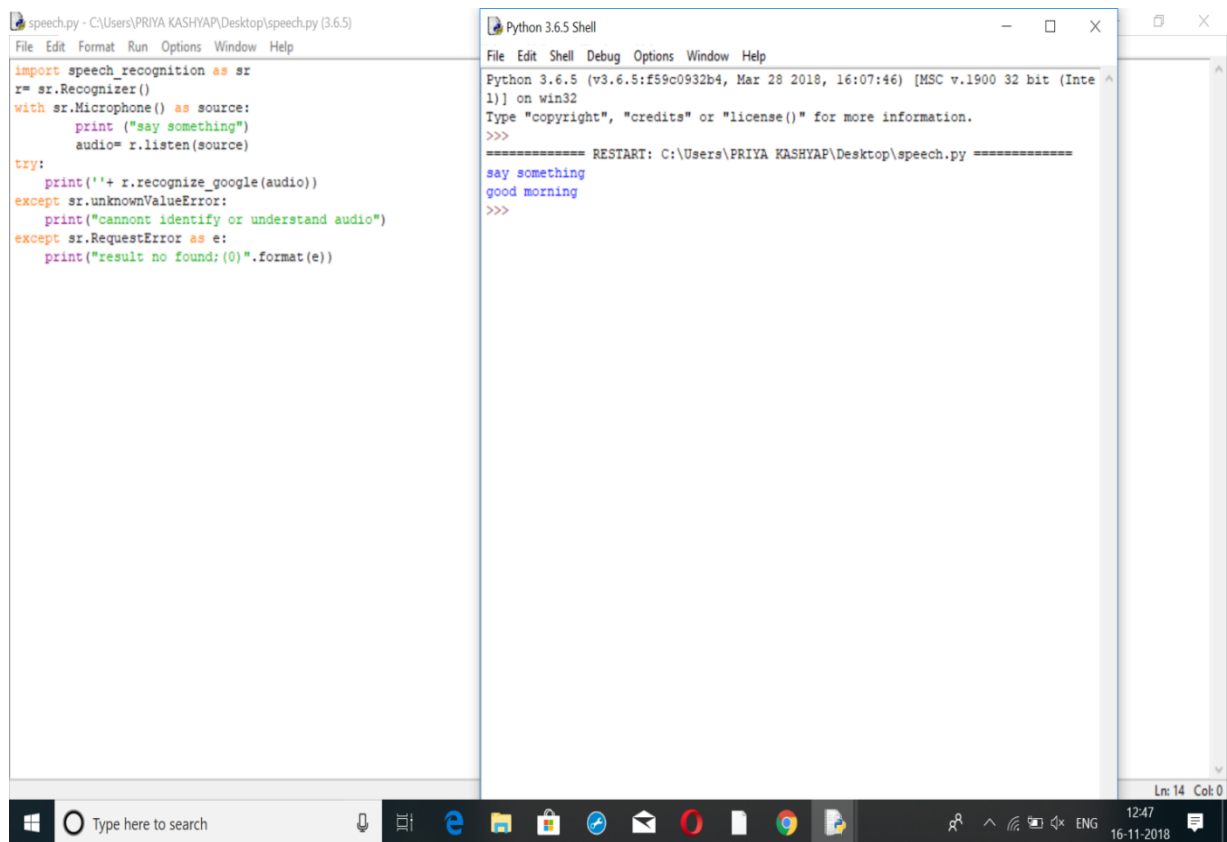
if __name__ == "__main__":
    app.run(debug = True)
```

Fig4.7: Coding part of Flask Server

## Module 2:- Speech to Text

The module will be responsible for translation of the voice (speech) into the text. This can be easily done with the help of the various speech APIs like Google Speech Recognition, SAPI voice etc. The module plays a significant role in the development of the project. API's are online working prewritten programs used for getting the data.

### 4.7 Google Speech Recognition (Online)



The screenshot displays a Python 3.6.5 IDE with two windows. The left window, titled 'speech.py - C:\Users\PRIYA KASHYAP\Desktop\speech.py (3.6.5)', contains the following code:

```
import speech_recognition as sr
r = sr.Recognizer()
with sr.Microphone() as source:
    print("say something")
    audio = r.listen(source)
try:
    print(''+ r.recognize_google(audio))
except sr.UnknownValueError:
    print("cannot identify or understand audio")
except sr.RequestError as e:
    print("result no found;{0}".format(e))
```

The right window, titled 'Python 3.6.5 Shell', shows the execution output:

```
Python 3.6.5 (v3.6.5:ff59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\PRIYA KASHYAP\Desktop\speech.py =====
say something
good morning
>>>
```

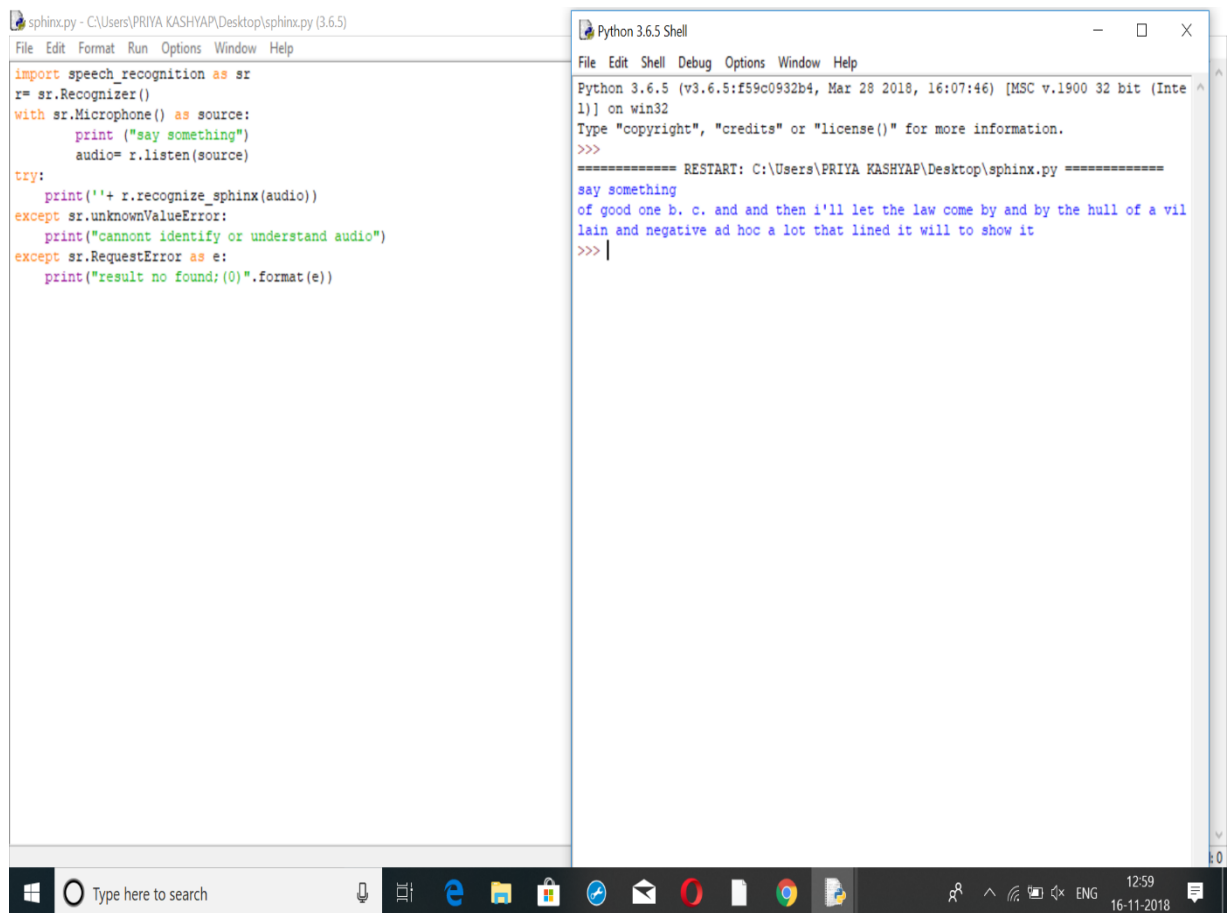
The Windows taskbar at the bottom shows the time as 12:47 on 16-11-2018.

Fig 4.8: Online Speech Recognition using Google API.

This is online speech recognition where Google API converts spoken text (microphone) into written text (Python strings), briefly Speech to Text. Here we simply speak in a microphone and Google API will translate this into written text. And it give much accurate result than the offline speech recognition program.

## 4.8 Speech Recognition (Offline)

In offline speech recognition I decided to start out with the Sphinx engine, as it worked offline and no internet connection is required. But Sphinx is not as accurate as online Google Speech Recognition. To start, first I install PyAudio[9] package as Speech Recognition requires PyAudio in order to interact with the microphone of my computer then secondly, I install the pocket sphinx package.



```
sphinx.py - C:\Users\PRIYA KASHYAP\Desktop\sphinx.py (3.6.5)
File Edit Format Run Options Window Help
import speech_recognition as sr
r= sr.Recognizer()
with sr.Microphone() as source:
    print ("say something")
    audio= r.listen(source)
try:
    print(''+ r.recognize_sphinx(audio))
except sr.unknownValueError:
    print("cannont identify or understand audio")
except sr.RequestError as e:
    print("result no found;(0)".format(e))

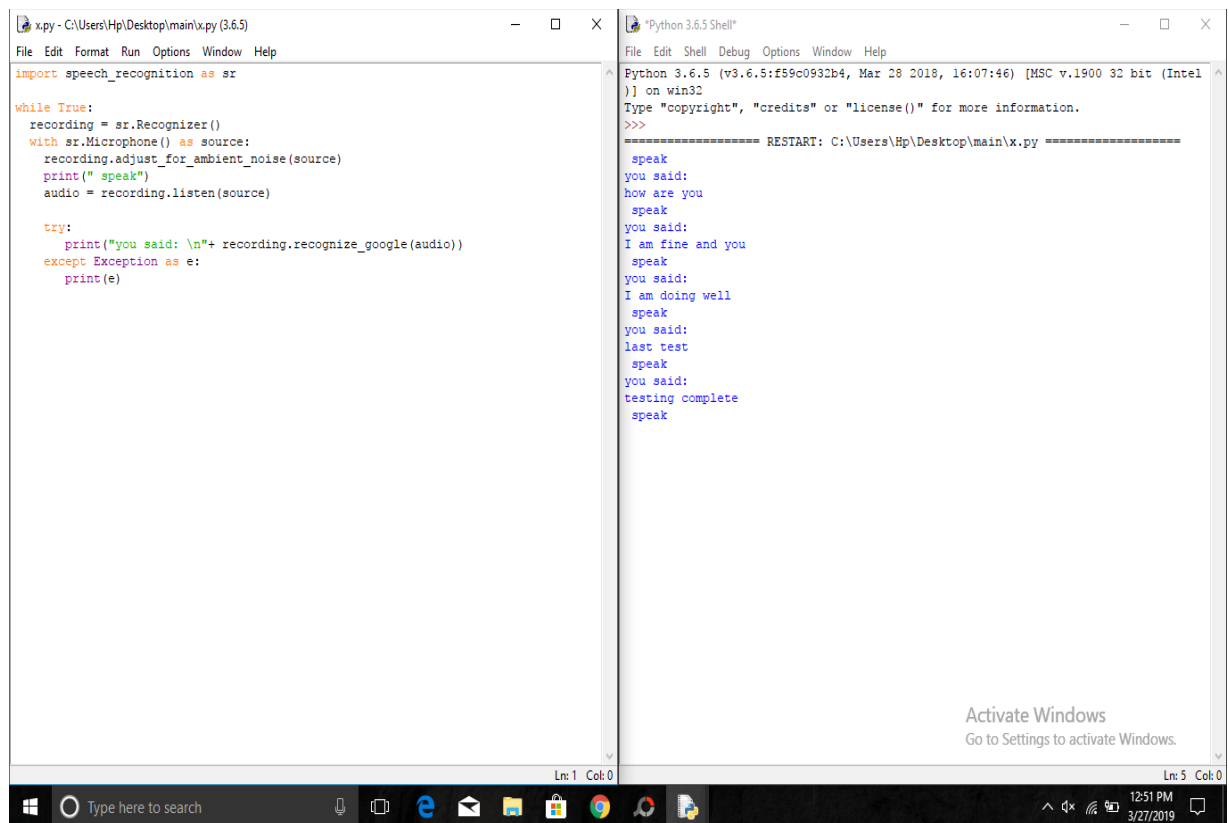
Python 3.6.5 Shell
File Edit Shell Debug Options Window Help
Python 3.6.5 (w3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\PRIYA KASHYAP\Desktop\sphinx.py =====
say something
of good one b. c. and and then i'll let the law come by and by the hull of a vil
lain and negative ad hoc a lot that lined it will to show it
>>>
```

Fig 4.9: Offline Speech Recognition using Sphinx engine

The code will create a Recognizer object, create a Microphone object, listen to the microphone to hear a spoken phrase, and use the appropriate recognizer engine (recognize sphinx here) to recognize the phrase. A microphone will pick up a lot of noise from a background, even though we don't hear it, which will interfere with the speech recognition. Still I am working on it to improve the accuracy but for our project the Google Speech Recognition is best.

## 4.9 Speech recognition in Infinity Loop

This is online speech recognition using Google API in which we are using speech Recognition package file and ambient package of python language where the whole program is running in infinity loop. Microphone will able to remove unwanted noise with help of ambient package and due to this the accuracy has been increased. This code will run continuously and convert speech into text until and unless we will end it.



```
x.py - C:\Users\Hp\Desktop\main\x.py (3.6.5)
File Edit Format Run Options Window Help
import speech_recognition as sr

while True:
    recording = sr.Recognizer()
    with sr.Microphone() as source:
        recording.adjust_for_ambient_noise(source)
        print(" speak")
        audio = recording.listen(source)

    try:
        print("you said: \n"+ recording.recognize_google(audio))
    except Exception as e:
        print(e)

Python 3.6.5 Shell
File Edit Shell Debug Options Window Help
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel
)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Hp\Desktop\main\x.py =====
speak
you said:
how are you
speak
you said:
I am fine and you
speak
you said:
I am doing well
speak
you said:
last test
speak
you said:
testing complete
speak
```

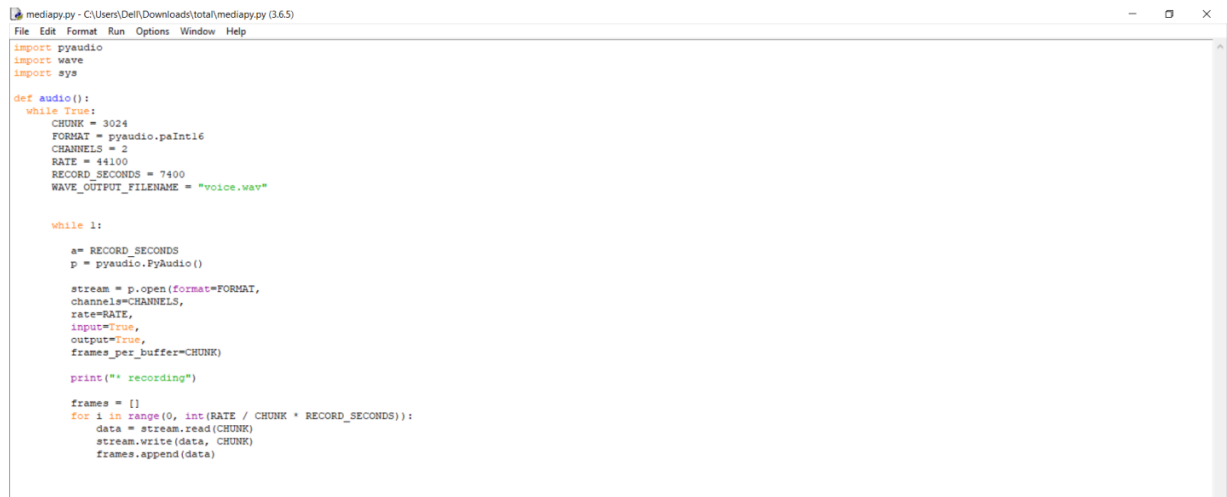
Activate Windows  
Go to Settings to activate Windows.

Ln: 1 Col: 0      Ln: 5 Col: 0

Type here to search      12:51 PM 3/27/2019

Fig 4.10: Speech to text in infinity loop

## 4.10 Coding for importing audio file

A screenshot of a Python IDE window titled 'mediapy.py - C:\Users\ DELL\Downloads\total\mediapy.py (3.6.5)'. The window has a menu bar with 'File', 'Edit', 'Format', 'Run', 'Options', 'Window', and 'Help'. The code is written in Python and implements an audio recording function. It imports 'pyaudio', 'wave', and 'sys'. A function 'def audio():' is defined, which contains a 'while True:' loop for initialization and a 'while 1:' loop for the recording process. The initialization loop sets 'CHUNK = 3024', 'FORMAT = pyaudio.paInt16', 'CHANNELS = 2', 'RATE = 44100', 'RECORD\_SECONDS = 7400', and 'WAVE\_OUTPUT\_FILENAME = "voice.wav"'. The recording loop sets 'a = RECORD\_SECONDS', 'p = pyaudio.PyAudio()', 'stream = p.open(format=FORMAT, channels=CHANNELS, rate=RATE, input=True, output=True, frames\_per\_buffer=CHUNK)', prints '" recording"', and then enters a 'for i in range(0, int(RATE / CHUNK \* RECORD\_SECONDS)):' loop. Inside this loop, it reads 'data = stream.read(CHUNK)', writes 'stream.write(data, CHUNK)', and appends 'frames.append(data)'.

```
mediapy.py - C:\Users\ DELL\Downloads\total\mediapy.py (3.6.5)
File Edit Format Run Options Window Help

import pyaudio
import wave
import sys

def audio():
    while True:
        CHUNK = 3024
        FORMAT = pyaudio.paInt16
        CHANNELS = 2
        RATE = 44100
        RECORD_SECONDS = 7400
        WAVE_OUTPUT_FILENAME = "voice.wav"

    while 1:

        a = RECORD_SECONDS
        p = pyaudio.PyAudio()

        stream = p.open(format=FORMAT,
            channels=CHANNELS,
            rate=RATE,
            input=True,
            output=True,
            frames_per_buffer=CHUNK)

        print('" recording"')

        frames = []
        for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
            data = stream.read(CHUNK)
            stream.write(data, CHUNK)
            frames.append(data)
```

Fig 4.11: Audio recording

### **Module 3:- Network Connectivity**

The module will be responsible for the connection part in the software. The project itself requires a big part of network as it involves sending and receiving of the packets generated from the video calls as well as the translated voice. These packets are to be transferred over the network in order to get desired results. The network connectivity can be achieved by using servers and APIs.

Various Protocols and API'S are available to do this work for the user in VoIP. Connections between two peers are created using and represented by the Peer to Peer Connection interface. Once a connection has been established and opened, media streams (Media Streams) and/or data channels can be added to the connection.

#### **4.11 Message Exchange**

Message exchange server and allow multiple clients to connect to it using a client-side script. The code uses the concept of sockets and threading. Any client that has a socket associated with the same port can communicate with the server socket and a thread is sub process that runs a set of commands individually of any other thread. So, every time a user connects to the server, a separate thread is created for that user and communication from server to client takes place along individual threads based on socket objects created for the sake of identity of each client.

##### **4.11.1 Server-Side Script:**

The server-side script will attempt to establish a socket and bind it to an IP address and port specified by the user (windows users might have to make an exception for the specified port number in their firewall settings, or can rather use a port that is already open). The script will then stay open and receive connection requests, and will append respective socket objects to a list to keep track of active connections. Every time a user connects, a separate thread will be created for that user. In each thread, the server awaits a message, and sends that message to other users currently on the chat. If the server encounters an error while trying to receive a message from a particular thread, it will exit that thread.

```
server-2.py - C:\Users\Hp\Desktop\New folder (2)\server-2.py (3.6.5)
File Edit Format Run Options Window Help

from socketserver import ThreadingTCPServer, BaseRequestHandler
from threading import Thread
import pickle,time,datetime,os

messages = []
temp = []
os.system("cls")
class Echo(BaseRequestHandler):

    def handle(self):
        self.temp = []
        Thread(target=self.send).start()
        self.username = self.request.recv(8192)
        self.username = self.username.decode()
        print("Got connection from {}".format(self.client_address[0],
                                              self.client_address[1]))

        while True:
            msg = self.request.recv(8192)
            msg = "[{} {}]: {}".format(datetime.datetime.now().strftime("%H:%M:%S"),
                                      self.username,
                                      msg.decode())

            messages.append(msg)
            print(msg)

            if not msg:
                break

        def send(self):
            global temp, messages
            while 1:

                if len(self.temp) != len(messages):

                    data_string = pickle.dumps(messages)
                    self.request.send(data_string)
                    self.temp = [item for item in messages]

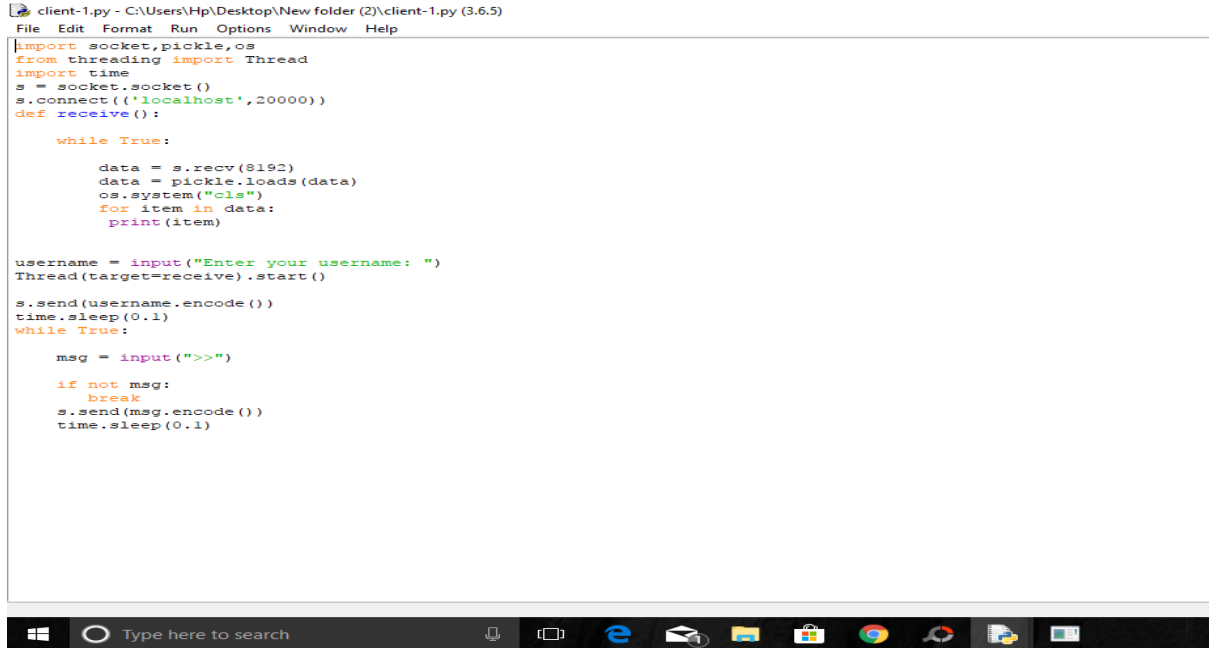
if __name__ == "__main__":
    serv = ThreadingTCPServer(("",20000), Echo)
    serv.serve_forever()
```

Fig 4.12.1: Server-side Script

#### 4.11.2 Client-Side Script:

The client-side script will simply attempt to access the server socket created at the specified IP address and port. Once it connects, it will continuously check as to whether the input comes from the server or from the client, and accordingly redirects output. If the input is from the server, it displays the message on the terminal. If the input is from the user, it sends the message that the users enters to the server for it to be broadcasted to other users.





```
client-1.py - C:\Users\Hp\Desktop\New folder (2)\client-1.py (3.6.5)
File Edit Format Run Options Window Help

import socket,pickle,os
from threading import Thread
import time
s = socket.socket()
s.connect(('localhost',20000))
def receive():
    while True:
        data = s.recv(8192)
        data = pickle.loads(data)
        os.system("cls")
        for item in data:
            print(item)

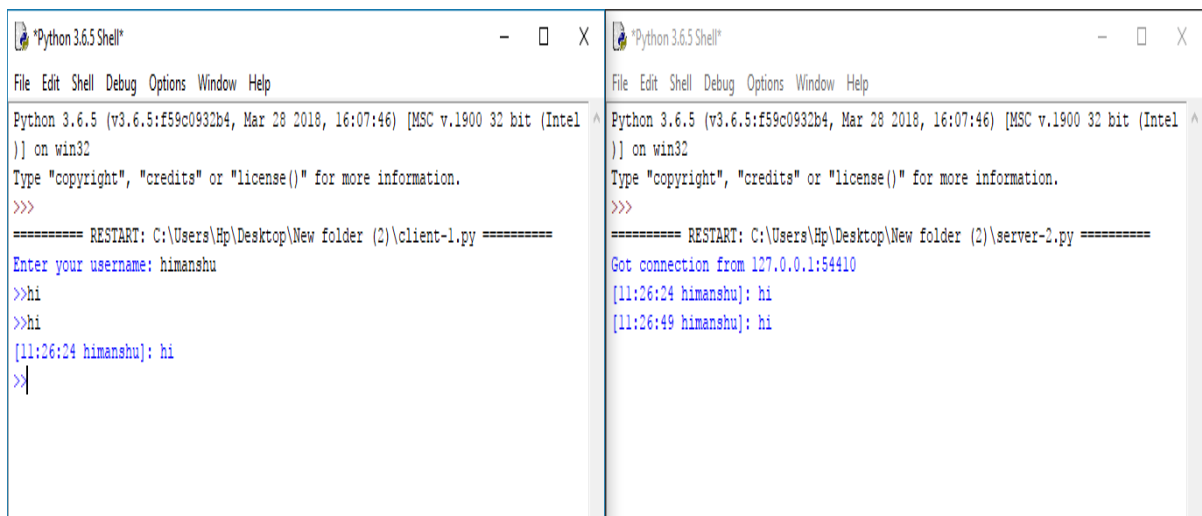
username = input("Enter your username: ")
Thread(target=receive).start()

s.send(username.encode())
time.sleep(0.1)
while True:
    msg = input(">>>")
    if not msg:
        break
    s.send(msg.encode())
    time.sleep(0.1)
```

Fig 4.12.2: Client-side Script

## 4.12 Running State of Client with Server:

First the user is required to create a username for the server to identify the user. Then the Client sends a message, the client access the host and sends the message on server while routing it through Thread and is received by server side as shown in the below figure 4.13.



```
*Python 3.6.5 Shell*
File Edit Shell Debug Options Window Help
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Hp\Desktop\New folder (2)\client-1.py =====
Enter your username: himanshu
>>hi
>>hi
[11:26:24 himanshu]: hi
>>

*Python 3.6.5 Shell*
File Edit Shell Debug Options Window Help
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Hp\Desktop\New folder (2)\server-2.py =====
Got connection from 127.0.0.1:54410
[11:26:24 himanshu]: hi
[11:26:49 himanshu]: hi
```

Fig 4.13: Message exchanging of client with server

In fig 4.14 we are sending video from one client to server by browsing it. When we click start call it invoke the camera and a frame open we video get start streaming.

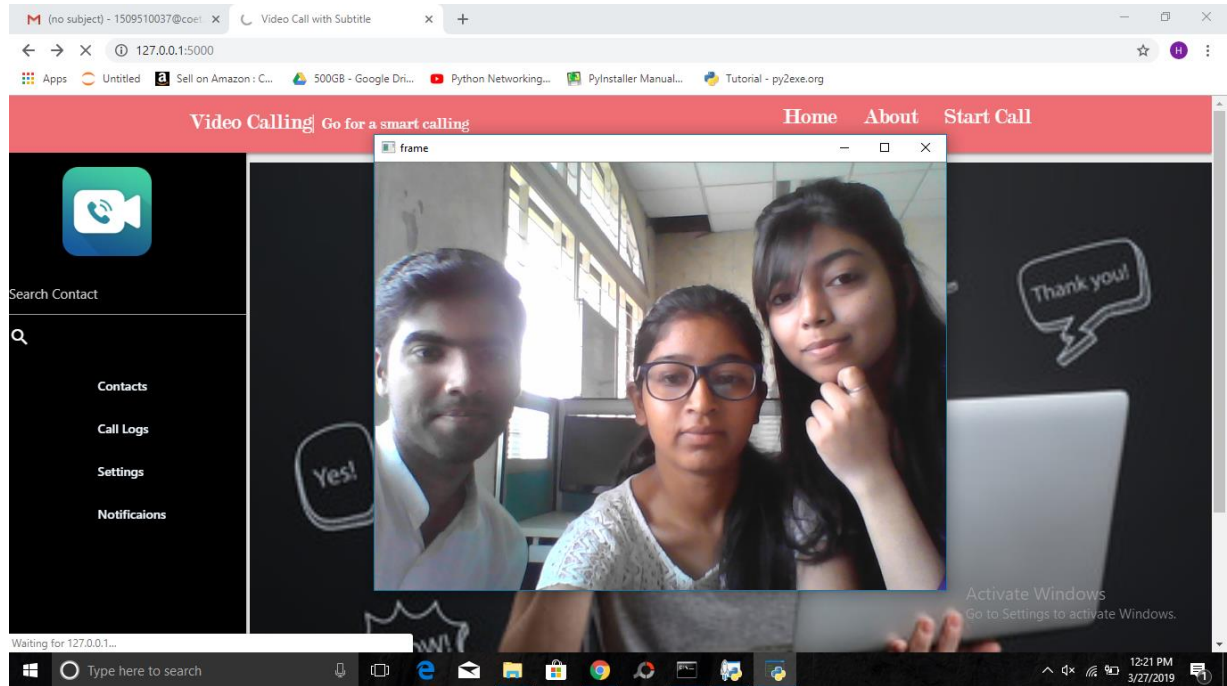


Fig 4.14: Video Streaming from one client to server.

### 4.13 Code Efficiency

In this project we are working on python and for designing HTML, CSS because it is easy to work on python and required less coding comparatively with other languages. Python is fast as well as efficient and support programming paradigms. Because of using Python, we save lots of time in coding.

We choose Python because it has following characteristics [5]:

- **Python is Easy in two ways**
  - a. Easy to code:** Python is very easy to code. Compared to other popular languages like Java and C++, it is easier to code in Python. It is easy to understand and learn python syntax. Hence, it is programmer-friendly.

**b. Easy to read:** Being a high-level language, Python code is quite like English. Also, since it is dynamically-typed, it mandates indentation. This aids readability.

- **It is Free and Open-Source:** Python is freely available and it is open-source. This means that its source code is available to the public.
- **Python is a High- Level Language:** It is a high-level language. This means that as programmers, we don't need to remember the system architecture. Nor do we need to manage the memory. This makes it more programmer-friendly and is 1 of the key python features.
- **Python Is Portable:** Python is portable as we don't need to change the code for different platform. We can run same code on Windows machine as well as run on Mac.
- **It is an Interpreted Language:** In other languages like C++, Java we need to first compile it, and then run it. But, in Python we don't need to compile the code. Internally, its source code is converted into an immediate form called bytecode. By interpreted, we mean the source code is executed line by line, and not all at once. Because of this, it is easier to debug your code.
- **Python Language is Object-Oriented:** Python supports both procedure-oriented and object-oriented programming which is one of the key python features. It also supports multiple inheritance, unlike Java.
- **Python is Extensible:** We can write some of the Python code in other languages like C++. This makes Python an extensible language, meaning that it can be extended to other languages.
- **It is Embeddable:** It is possible to put our Python code in a source code in a different language like C++. This allows us to integrate scripting capabilities into our program of the other language.

- **It had Large Standard Library:** Python downloads with a large library that you can use so you don't have to write our own code for every single thing. There are libraries for regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and a lot of other functionality.
- **Python can be used for GUI Programming:** You can use Tk, PyQt to create basic GUIs.
- **Python is Dynamically Typed:** Python is dynamically-typed. This means that the type for a value is decided at runtime, not in advance. This is why we don't need to specify the type of data while declaring it.

We also use HTML, CSS for designing the FRONT-END and then connect that html file with flask that provide us a web frame.

#### **4.14 Optimization of code**

We worked on Python language which is already known for code optimization. We used some functions and object-oriented feature that makes our code more efficient. As Python required less coding so, the length of our coding is hardly in 30-50 lines which makes our project efficient in terms of coding.

Our code is optimized in terms of space as well as time complexity. And we worked in modules so it helps us to decrease the load on the system and makes our work more optimised.

# **CHAPTER 5**

## **Validation Checks**

### **5.1 Verification checks**

In verification we evaluate the component of our project and confirm that it satisfies all of the specifications and requirements.

Our system does meet all the requirements. The documentation, code and design have been evaluated and approved by our project supervisor and co-ordinator. The design output meets the input design of our project. In this phase of testing execution of code is not necessary. So, we cannot find any bugs during this phase. Every single module is evaluated many times and it perform as per the requirement specification.

The database is also checked and the system is completely as per the requirements given.

### **5.2 Validation checks**

In validation we evaluate the whole system/software during the end of the development and assure that our system meets the need of customers.

In our project, we include all the features and requirements we wanted to add. For the customer's point of view, we have all basic features that present in a video call. The requirements and the final product are validated by our supervisor and co-ordinator. In this phase of testing we evaluate our system as whole. The project fulfils its intended use and it is very easy to handle the web-based application which is user friendly.

# CHAPTER 6

## Implementation and Maintenance

### 6.1 Module Description

#### 6.1.1 Video Calling with GUI:-

The prime module of the project is the module of video calls. This module is responsible for the development of the video calling system and on the top of which the GUI will be running so as to provide user with a better and easy understanding of the software. This will make use of VoIP, STUN Server, TURN Server, Python libraries.

We use python libraries like flask for developing the web frame which connected with html and CSS.

All the interaction with the user is done in this module, it includes various forms which include

- Registration
- Login
- Homepage
- Setting menu
- Notifications

#### **Registration:**

User enter the detail asked by the system when it click on sign up the data is saved in the database and user is directed to login page and then login process continue.

User-Id: Datatype: varchar

User Name: Datatype: char

Password: Datatype: varchar

Phone No. Datatype: Int

**Login:**

When user enters user-name and master password and click on login button then this information is validated. If information is valid then user is directed to home page.

User-Id: Datatype: varchar

Password: Datatype: varchar

**Homepage:**

Consists of various functions and options provided by the system for the user's use (video calling option and adding of various contacts is also present in this form).

**Setting:**

This form consists of various setting options. Here the user can update the account information or can change various settings of the system.

**Notifications:**

Here the data of date and time of missed calls, received calls and dialled calls will be stored and presented to the user.

### **6.1.2 Speech to Text:-**

The module will be responsible for translation of the voice (speech) into the text. This can be easily done with the help of the various speech APIs like Google Speech Recognition, SAPI voice etc. The module plays a significant role in the development of the project. API's are online working prewritten programs used for getting the data.

### 6.1.3 Network Connectivity: -

The module will be responsible for the connection part in the software. The project itself requires a big part of network as it involves sending and receiving of the packets generated from the video calls as well as the translated voice. These packets are to be transferred over the network in order to get desired results. The network connectivity can be achieved by using servers and APIs. Various Protocols and API'S are available to do this work for the user in web and VoIP. Connections between two peers are created using and represented by the Peer to Peer Connection interface. Once a connection has been established and opened, media streams (Media Streams) and/or data channels (Data Channels) can be added to the connection.

- a) **Peer to Peer Connection:** Represents a connection between the local computer and a remote peer. It is used to handle efficient streaming of data between the two peers.
- b) **Data Channel:** Represents a bi-directional data channel between two peers of a connection.
- c) **Data Channel Event:** Represents events that occur while attaching a Data Channel to a Peer Connection. The only event sent with this interface is data channel.
- d) **Session Description:** Represents the parameters of a session. Each Session Description consists of a description type indicating which part of the offer/answer negotiation process it describes and of the SDP descriptor of the session.
- e) **Media Stream:** The Media Stream interface represents a stream of media content. A stream consists of several tracks such as video or audio tracks.

## 6.2 Testing

### 6.2.1 Unit Testing-

We performed unit testing in our system by testing each and every module individually. All modules are working as we expected for all required inputs. In our first module that is video calling with GUI is running perfectly. In our second module that is Speech to Text have some limited test cases as its covert speech to text in few seconds. Because we are using online Google



API it takes few seconds for conversion and depends on internet speed. In our last module that is Network Connectivity (server to client) the client request data/modules for server and server respond to the request by sending the resources/data to client but we are still working for the accuracy and communication between 2 clients.

### **6.2.2 Integration Testing-**

We performed integration testing by integrating different modules together and running them as once. And we get the same result that we expected from this testing and integration testing is work completely fine.

### **6.2.3 System Testing-**

In system testing, all the integrated modules working fine and no harm to the system. The complete system is working completely fine as they supposed to do.

### **6.2.4 Accessibility Testing-**

The system is easily accessible for all type or users. The interface of our project is easy and interactive to any kind of users. There is no need of any special knowledge for handling the interface.

### **6.2.5 Functional Testing-**

In this testing, we checked all the features of our system and checked whether these features are working fine or not.

## **6.3 Test Cases**

### **6.3.1 Black Box Test Cases:**

<b>Sr. No.</b>	<b>Test Cases</b>	<b>Test Purpose</b>	<b>Test Condition</b>	<b>Expected Output</b>	<b>Actual Result</b>
01.	Start Calling	To ensure that camera of the device is invoke and	When a person clicks on a start calling, the camera	Camera invoke and object detected.	<b>True Result.</b>

		camera capture the object/person.	invokes within 1-3 seconds.		Camera invoke and object detected.
02.	Translation Time	To check the speed of real time Speech to Text translation	When the person speaks the speech is translated into text.	To ensure that the translation done within 3 seconds	<b>False Result.</b> Delay in translations by 5-6 seconds
03.	Mic Invoke	To ensure the voice is capture when the person speaks.	The mic is invoked and record the voice.	Record the voice of the person.	<b>True Result.</b> Recorded.
04.	Speaker	To ensure that voice of the person is audible when person start to speak.	Voice of person should be audible without any noise in the background.	Clear voice of the person.	<b>False Result.</b> Some noise detected.

Table 3: Black Box Test Cases

### 6.3.2 White Box Test Cases:

Sr. No.	Test Cases	Test Purpose	Test Condition	Expected Outcome	Actual Result
01.	Audio Request	To check whether the microphone is invoked and ready to send data	When code runs audio data is recorded and encoded and is ready to send	Microphone is invoked audio data is recorded and is ready to send over the network	Microphone is invoked audio data is recorded and is ready to send over the network

02.	http.request ("server_address/updateStatus.php/ id=x&status=y")	To check if the values in the database are getting updated or not.	When code runs, database gets updated.	Code after the database starts running.	Code after update of database starts running.
03.	Video request ("media.py/audio()")	To check whether the video cam or web cam is invoked and the video is recording	When code runs , the cam is invoked and the video recording is ready to be sent over network	Video data gets recorded and data is to be sent over network	Video data recorded successfully and data started sending
04.	Speech to text	To ensure the translation of audio data to text data when mic is invoked	When code runs the mic is invoked and the audio is sent to the server where the data is converted to text and sent back to the user	Audio data is translated to text is send over the network	Mic is invoked successfully and the audio data is sent to the server for translation to text after which the text data is sent back to the user for display

Table 4: White Box Test Cases

## 6.4 System Security Measures

1. Database is secured with the help of login and signup credentials with further security from double encryption.
2. The access to the terminal is locked by using flask thread creation encryption.

## 6.5 Cost Estimation

In our project for cost estimation we have used Intermediate COCOMO model with Semi-detached mode[7].

<b>Project phase/ Participants</b>	<b>Project Manager</b>	<b>Front-End Developer</b>	<b>Back-End Developer</b>	<b>QA Tester</b>	<b>System Admin</b>
Technical and function Conception	X	X	X	X	
Development	X	X	X	X	
Deployment in staging environment	X	X	X	X	X

Table 5: People involved during development phase[7].

The cost of the project is estimated in the following ways:

Effort: There are 3 members in our team and we worked together to complete our project. We spend almost 5 to 6 months to get it done. Hence, the effort required by each member is high.

Time: The time required to complete the whole system is approximately 6 months.

Cost: As our project is totally based on software and no hardware is used. So, the cost is for hosting the web-application and cost for developer's effort and time.

The page development cost consists of:

- For developing front-end it required 1 month with Rs. 2000
- For developing back-end it required 3 months with Rs. 5000
- For QA testing it required 3-4 hours and charging Rs. 70/hour.
- Hosting cost – Rs. 5000/year

Therefore, the total cost is around Rs. 13000 as all the software we use in our project is open source and no other cost required while developing the web-application.

# CHAPTER 7

## Pert Chart and Gantt Chart

### 7.1 Pert Chart

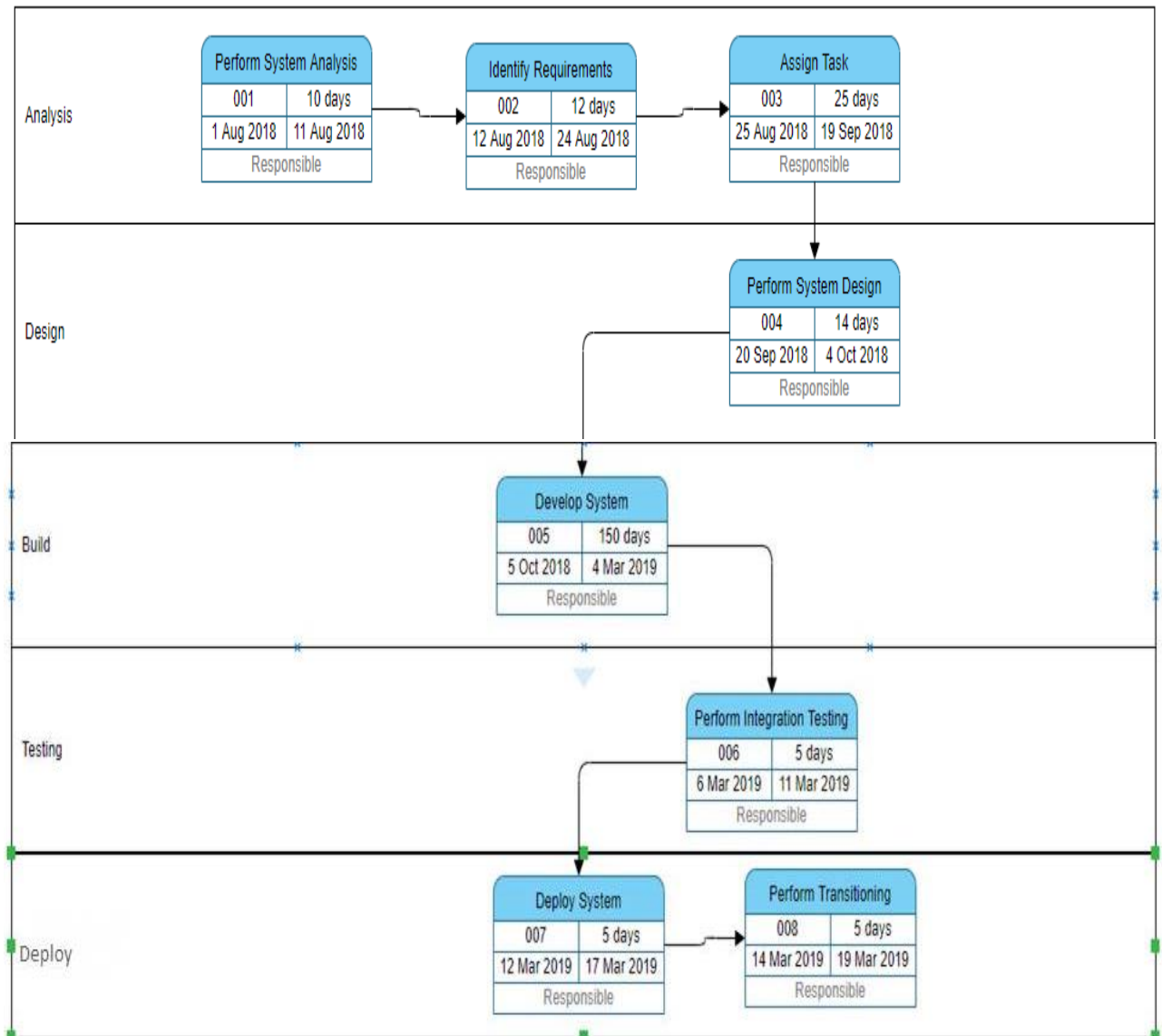


Fig 7.1 Pert chart[11]

## 7.2 Gantt Chart

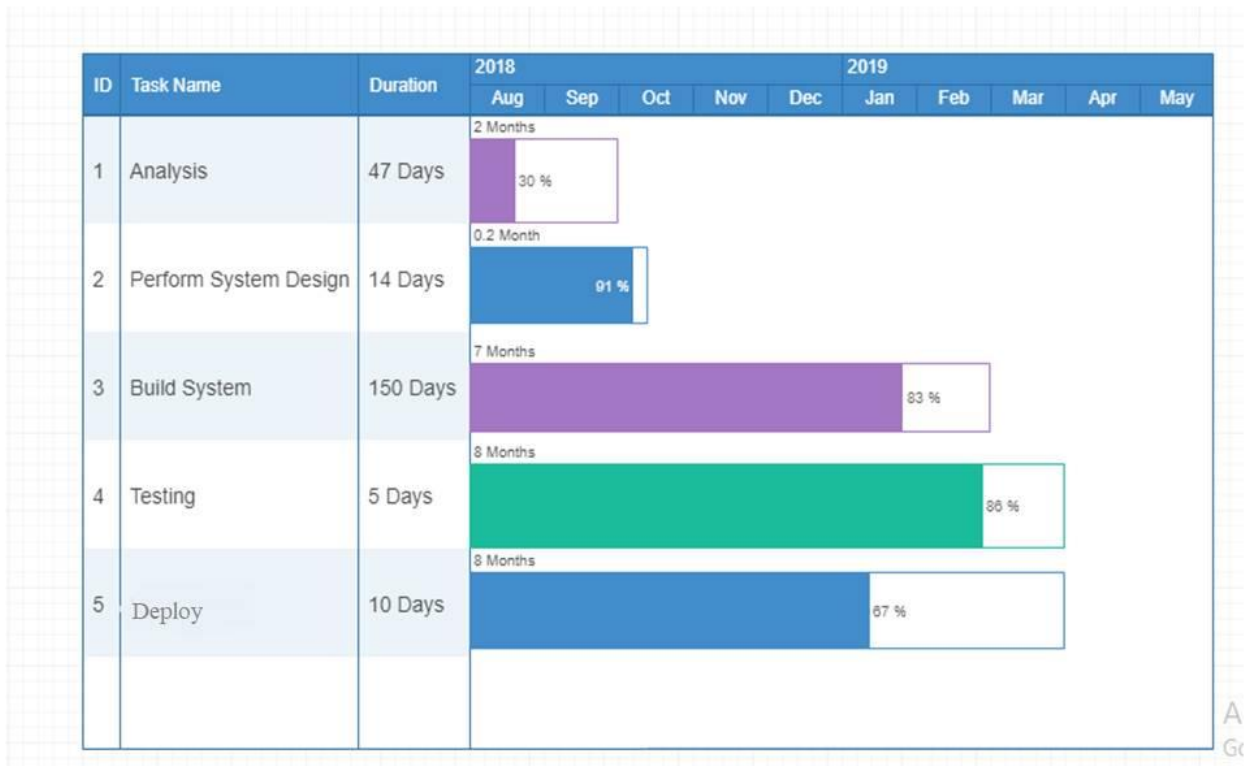


Fig 7.2 Gantt chart[11]

## **CHAPTER 8**

### **Future Scope**

Our project can be considered as a complete package but there are some modifications that can be done in future for making the system more efficient and that will also fulfil the need of the future and for excellent and faster connectivity of different users at their sites to make it more interactive for better user interface. The possible modification for future enhancement of our project can be-

- Introduction of different languages for different users and for good communication and interaction.
- Adding the Option of having video conferencing call with embedded real time text translation for better exchange of information and discussion.
- Installation of text chatting (with stickers embedded) while having video calling. Implanting these two features together for different experience of communication.

### **Limitations**

- 1) Video conferencing feature between more than 3 clients is not available.
- 2) Live streaming feature is only available on Local Host.
- 3) Speech-To-Text is having a time lagging of 5 to 10 seconds depending upon the network speed.
- 4) English (U.K. & U.S.) is the only available language for real time Speech-To-Text translation over video calls.
- 5) Standard Definition is the only type of video quality standard for transmission of video data due to high data transfer cost.

### **Conclusion**

There are a number of software's and applications already present in the market for video calling, but very less of these available software's provide a different experience for communication, our application brings you a new type of experience which is Speech-To-Text



translation of voice over a video call , this feature is also present in various application but is unable to bring out the accuracy required as well as the interest of the user.

Our application brings the accuracy which lacks in its counterparts as well as a better user experience, we look forward to our future work and to make our users satisfied.

## References

- [1] ezTalks, Information of Video Conferencing and Video Calling, Retrieved from <https://www.eztalks.com/videconference/disadvantages-of-video-conferencing.html>
- [2] Webopedia, Information of Video Chat, by Vangie Beal, from [https://www.webopedia.com/TERM/V/video\\_chat.html](https://www.webopedia.com/TERM/V/video_chat.html)
- [3] McGraw-Hill Concise Encyclopedia of Engineering. Videotelephony, McGraw-Hill, 2002. Retrieved from the <https://en.wikipedia.org/wiki/Videotelephony>, January 9, 2010
- [4] Krazytech, Example of Software Requirements Specification Document, by Ravi Bandakkanavar, from <https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database>
- [5] Quora, Features of Python, Retrieved from <https://www.quora.com/What-are-the-features-of-Python>
- [6] Lucid Chart, for making UML Diagram, from <https://www.lucidchart.com>
- [7] Existek (2012), Information for Cost estimation, from <https://existek.com/blog/custom-web-application-development-cost/>
- [8] Flask (2010), information on Flask and Web-frame, Retrieved from <http://flask.pocoo.org/>
- [9] Plos, pyAudioAnalysis, Published on December 11,2015, Retrieved from <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0144610>
- [10] Semantic Scholar, about Skype Translator: Breaking Down Language and Hearing Barriers, published 2015 by William D. Lewis, Retrieved from <https://www.semanticscholar.org/paper/Skype-Translator%3A-Breaking-Down-Language-and-Lewis/5a2b26a9c19906e0c742d5d2e453236a8be91e23>
- [11] Lucid Chart, for making Pert and Gantt Chart, from <https://www.lucidchart.com>

## Glossary

1. **Feasibility-** An analysis and evaluation of a proposed project to determine if it is technically feasible, is feasible within the estimated cost, and will be profitable. A feasibility study should cover all of the salient points on whether a project is reasonable, and should have documentation related to any concerns.
2. **Flask-** Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. Flask is part of the categories of the micro-framework. Micro-framework is normally framework with little to no dependencies to external libraries.
3. **FTP- File Transfer Protocol** is an application layer protocol which moves files between local and remote file systems. It runs on the top of TCP, like HTTP. To transfer a file, 2 TCP connections are used by FTP in parallel: control connection and data connection.
4. **Gantt Chart-** A horizontal bar chart frequently used in project management that provides a graphical illustration of a schedule that helps to plan, coordinate, and track specific tasks in a project.
5. **HTTP-** means **HyperText Transfer Protocol**. HTTP is the underlying protocol used by the World Wide Web and this protocol defines how messages are formatted and transmitted, and what actions Web servers and browsers should take in response to various commands.
6. **IPSec-** also known as the **Internet Protocol Security** or **IP Security** protocol, defines the architecture for security services for IP network traffic. IPsec describes the framework for providing security at the IP layer, as well as the suite of protocols designed to provide that security, through authentication and encryption of IP network packets.

7. **ISDN- Integrated Service Digital Network**, is the original high-speed internet service. ISDN internet service is basically a telephone-based network system that operates by a circuit switch, or dedicated line. It can transmit data and phone conversations digitally over normal telephone wires. This makes it both faster and of higher quality than dial-up internet service.
8. **Machine Learning-** is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that which makes it more similar to humans: The ability to learn.
9. **NTP- Network Time Protocol** is a protocol used to synchronize computer clock times in a network. It belongs to and is one of the oldest parts of the TCP/IP protocol suite. The term NTP applies to both the protocol and the client-server programs that run on computers.
10. **Object Oriented programming-** is a programming paradigm based on the concept of "objects", which can contain data, in the form of fields, and code, in the form of procedures. A feature of objects is an object's procedures that can access and often modify the data fields of the object with which they are associated.
11. **Pert Chart-** A PERT chart is a project management tool used to schedule, organize, and coordinate tasks within a project. PERT stands for **Program Evaluation Review Technique**, a methodology developed by the U.S. Navy in the 1950s to manage the Polaris submarine missile program.
12. **Pocketsphinx-** is a part of the CMU Sphinx Open Source Toolkit for Speech Recognition. This package provides a python interface to CMU Sphinxbase and Pocketsphinx libraries created with SWIG and Setup tools.

13. **SAPI Voice-** (Speech Application Program Interface) is an application program interface (API) provided with the Microsoft Windows operating system that allows programmers to write programs that offer text-to-speech and speech recognition capabilities.
14. **Speech Recognition-** is the ability of a machine or program to identify words and phrases in spoken language and convert them to a machine-readable format. Rudimentary speech recognition software has a limited vocabulary of words and phrases, and it may only identify these if they are spoken very clearly. More sophisticated software has the ability to accept natural speech.
15. **TCP/IP-** or the Transmission Control Protocol/Internet Protocol, is a suite of communication protocols used to interconnect network devices on the internet. TCP/IP can also be used as a communications protocol in a private network (an intranet or an extranet).
16. **VoIP-** (Voice over IP) is the transmission of voice and multimedia content over Internet Protocol (IP) networks. VoIP historically referred to using IP to connect private branch exchanges (PBXs), but the term is now used interchangeably with IP telephony.
17. **XMPP-** Extensible Messaging and Presence Protocol (XMPP) is an open XML technology for real-time communication, which powers a wide range of applications including instant messaging, presence and collaboration.