***Team number and team member names are written below(end of the doc)

PROJECT TOPIC : Distributed messaging app with features like gesture, hate speech recognition

ABSTRACT

The main aim of the project is to build a messaging app with basic features like sending and receiving text and images. Multiple clients on different systems will be connected to a central server hosted on another system.

Here in this project we have added some additional modules (listed below) to our messaging application so as to make it more user friendly .

We use Python to implement core deep learning concepts like CNN, Multilayer Perceptron (MLP), RNN.

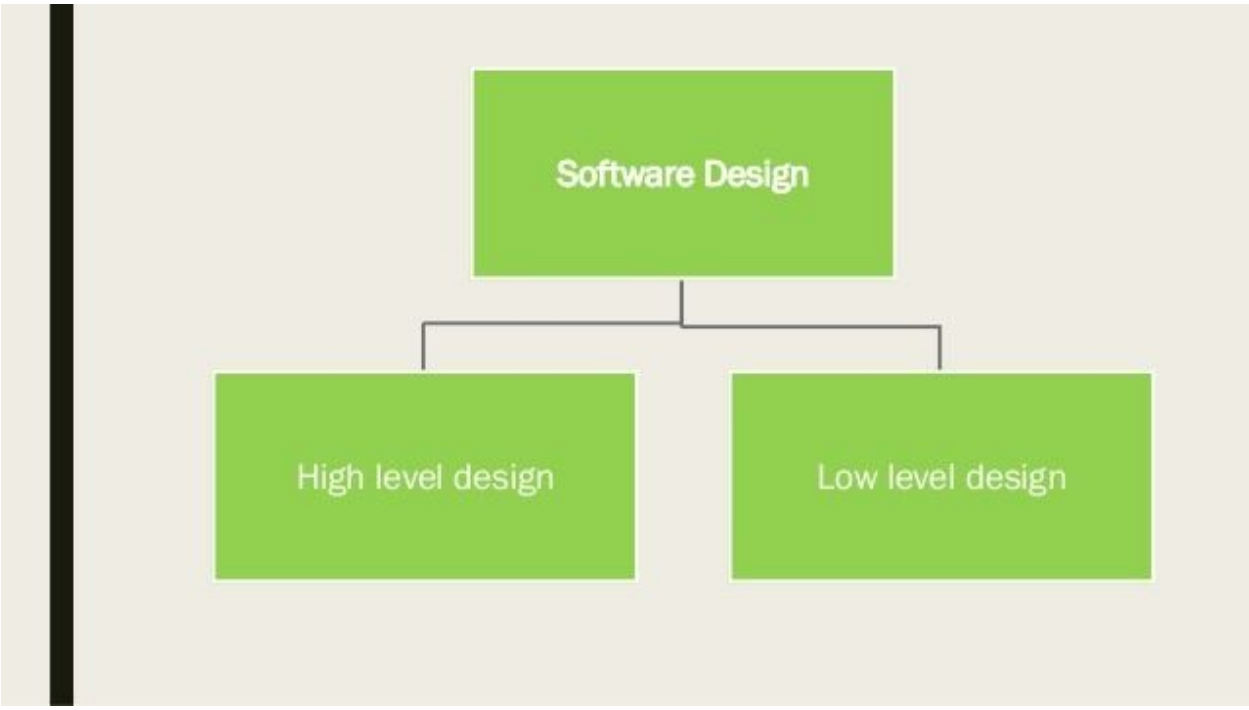
We will use concepts like autoencoders (dimensionality-reduction), Adam (optimizer), Adagrad and RMS prop, Momentum. We use ReLu , sigmoid, tanh as activation functions as required.

We intend to embed the following features in the application :
Gesture recognition.
Emojinator
Face recognition
Hate Speech and offensive content recognition
Digit Recognizer Chatbot (client-server model)

We train all our modules with standard datasets .We mostly use libraries like keras, tensorflow , matplotlib, cv2, dlib, h5py, scipy for standard implementation.

**“These are not final just indicative we will only include compatible modules only”

DESIGN



We have two types of design:
1)Low level design
2)High level design
Below we have detailed explanation of both of them:

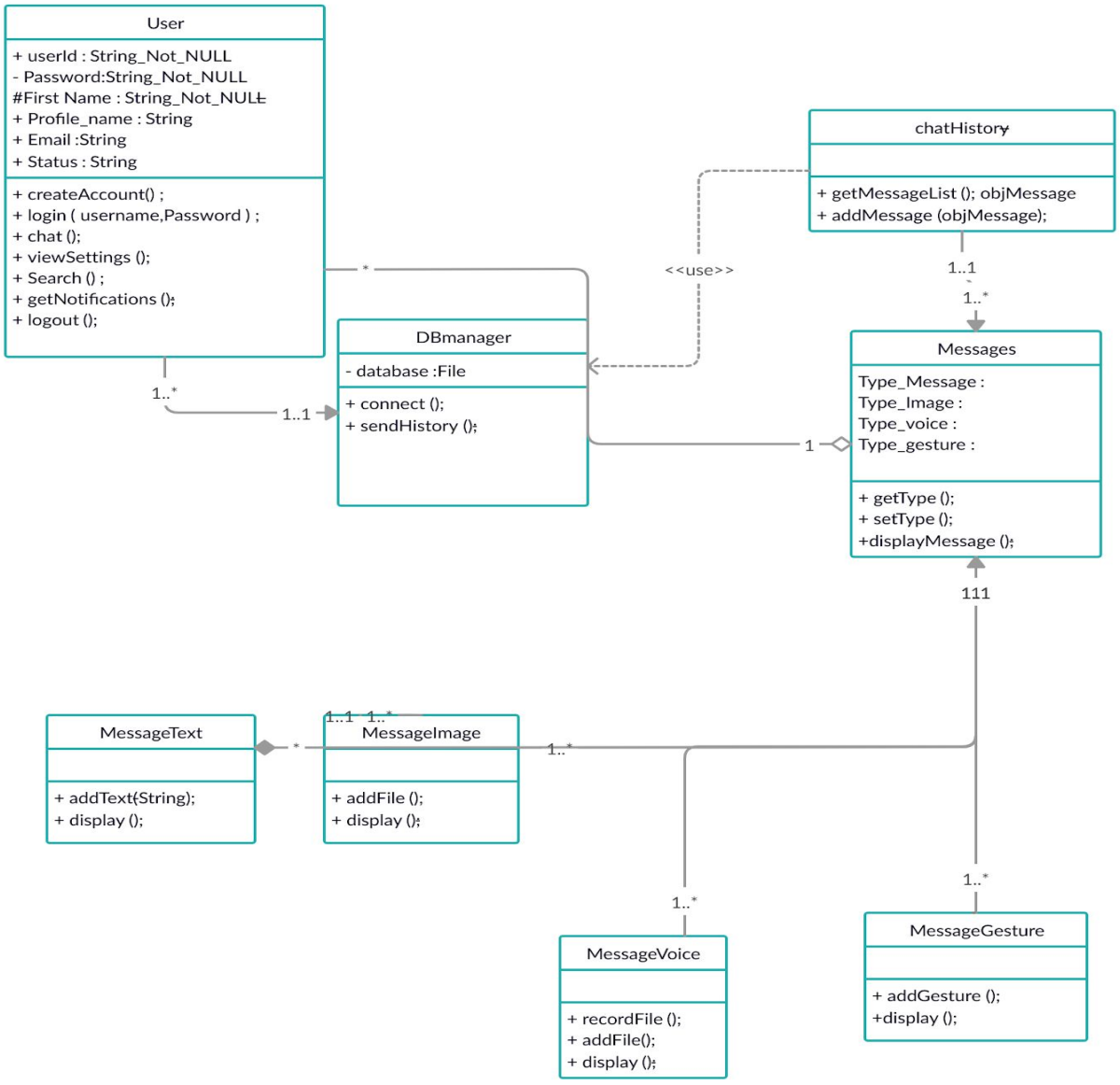
High level design:

High-level design (HLD) explains the architecture that would be used for developing a software product. The architecture diagram provides an overview of an entire system, identifying the main components that would be developed for the product and their interfaces. The HLD uses possibly nontechnical to mildly technical terms that should be understandable to the administrators of the system. In contrast, low-level design further exposes the logical detailed design of each of these elements for programmers.

Low Level design:

Low-level design (LLD) is a component-level design process that follows a step-by-step refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work.
Post-build, each component is specified in detail

LOW LEVEL DESIGN OF MESSAGE APP(CLASS DIAGRAM)



We have multiple components in it

1.Users:

Each user has the following attributes :

UserId,Password,First_name,profile_name,EmailId,Status

Following operations are performed by users:

createAccount(),login(),chat(),viewSettings(),Search(),getNotifications(),logout()

2.DBmanager

Attributes associated: databaseFile
Operations associated: connect(),sendHistory()

3.Messages :
Attributes associated : Type_message,Type_image,Type_voice,Type_gesture
Operations associated : getType(),setType(),displayMessage()

4.Chat History :
Operations associated : getMessageList(),addMessage()

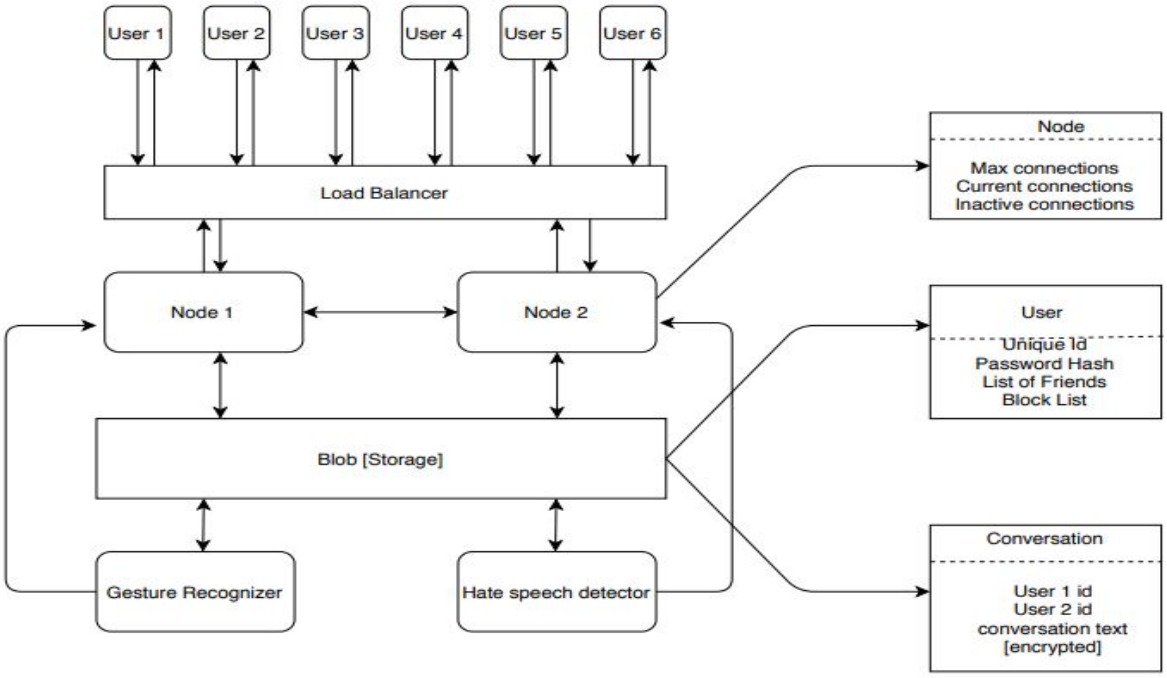
5.Message Text :
Operations associated : addText(),display()

6.Message Image :
Operations associated : addImage(),display()

7.Message Gesture :
Operations associated : addGesture(),display()

8.Message Voice :
Operations associated : recordVoice(),addFile(),display()

HIGH LEVEL DESIGN



Design Diagram for a Messaging App

LOAD BALANCER:

load balancing refers to the process of distributing a set of tasks over a set of resources (computing units), with the aim of making their overall processing more efficient.

Techniques used:

Round Robin

Round robin is a simple technique for making sure that a virtual server forwards each client request to a different server based on a rotating list. It is easy for load balancers to implement, but does not take into account the load already on a server. There is a danger that a server may receive a lot of processor-intensive requests and become overloaded.

STORAGE:

We store all message history,user information,block list,list of friends

MULTIPLE USERS:

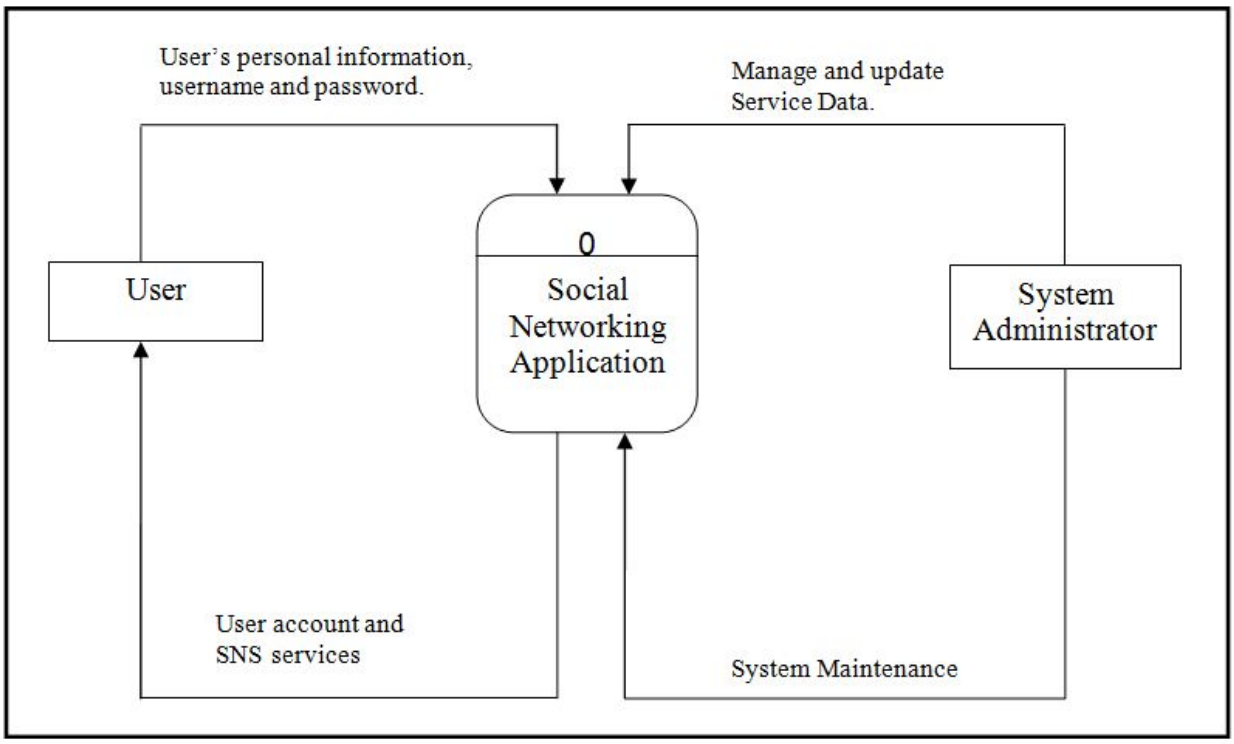
We have many users that uses this model

NODES:

HATE SPEECH DETECTOR:we use deep learning to find hate speech

GESTURE RECOGNIZER: we use deep learning and computer vision (CV) module to find the gesture

DATA FLOW DIAGRAM(DFD)



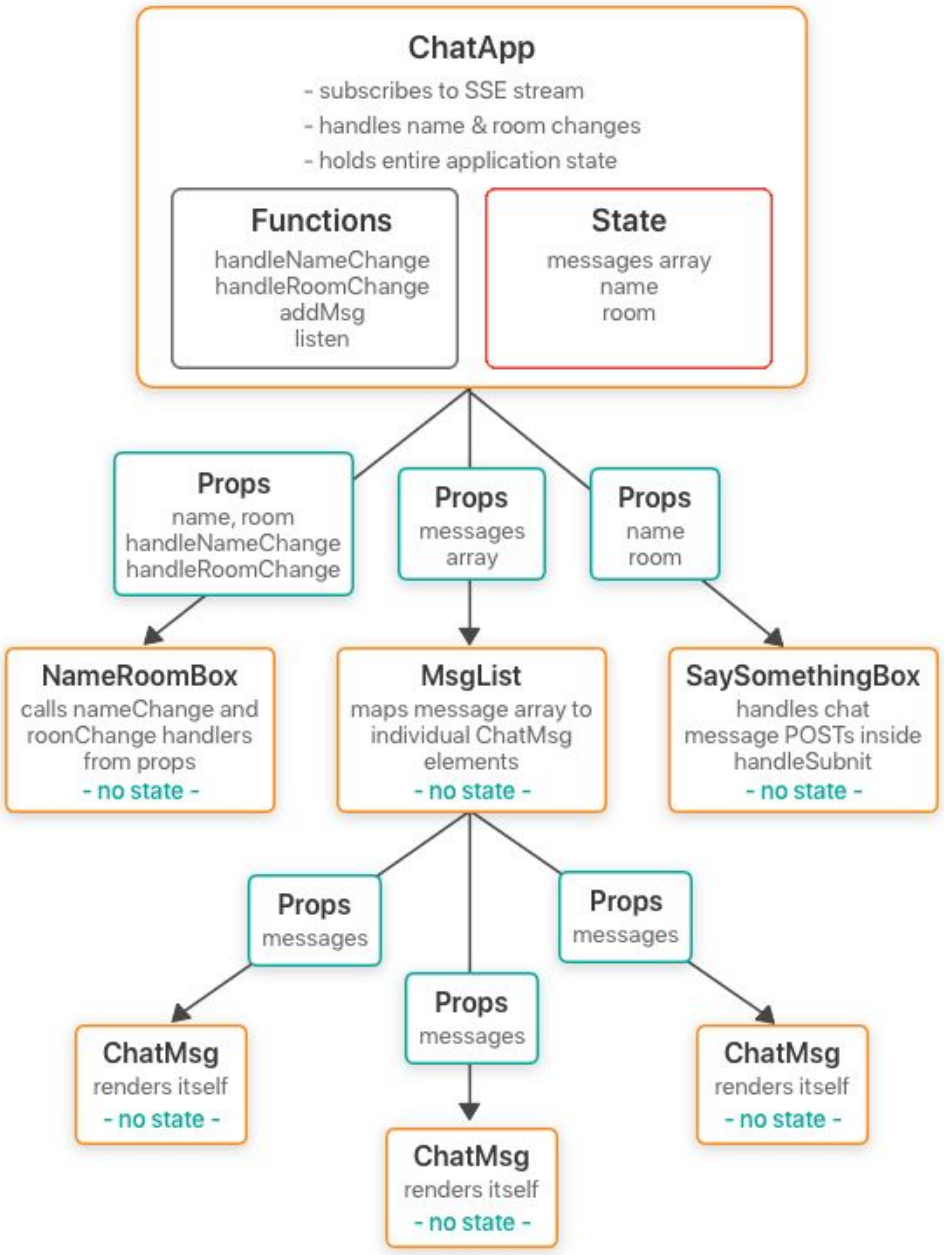
A data flow diagram (DFD) illustrates how data is processed by a system in terms of inputs and outputs. As its name indicates its focus is on the flow of information, where data comes from, where it goes and how it gets stored.

We have 2 components

System administrator:He can change code and has high level authority.

User : who is not allowed to change or modify the code

FLOW CHART



A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task. The flowchart shows the steps as boxes of various kinds, and their order by connecting the

boxes with arrows.Above diagram is just an overview of chat application flow chart after this we embed different modules like gesture and hate recognition

TEAM NUMBER:17
Group Members:
SAI ROHITH ARETI(2019201072)
YALLAMANDA RAO MUNDRU(2019201029)
JAYA KRISHNA KURUVA(2019201076)
SHOVAN SWAIN (20161127)
ARUN G (201564134)
AASHISH SHRIVATSAVA (20161111)