**PROPOSED SYSTEM**

The proposed system focuses on integrating all basic features in one place in an application and powering it with an AI chatbot further adds new functionalities like easy navigation, access to the data on availability of doctors, diagnostics information, symptom analysis, precautionary or instant medication suggestions and appointment booking, all these in a single application. Further, considering people who cannot write fluently, those with special needs and those in emergency situations, both voice and text input formats are accepted by the chatbot. We are building the website using Flask, which contains Login and registration page, dashboard of website, appointment booking and viewing pages and also the animated chatbot button at the end of every page of the website.

Speech enabled chatbots provide higher level of interactivity and usability. User can either give their input using text or speech and similarly chatbot is able to give its response by either text or voice. In our project, this process of conversion between text and speech is done by using speech\_recognition and pyttsx3 python modules.

1. Voice Input by User (Speech to Text):

Using systems inbuilt microphone live audio input can be transcribed using Google’s Web Speech API (recognize\_google()).By using adjust for ambient noise function we can set the engine to listen to ambient noise for some time period(here 2 seconds) and adjust energy threshold accordingly. If speech Recognizer unable to detect the speech correctly, respective error messages will be given as response.

1. Voice Output by Bot (Text to Speech):

Pyttsx3 is a Text to Speech Conversion Python Library. Using pyttsx3.init() an engine instance will be created for which we can set various properties like voice rate, volume level and also voices (male or female).We can directly pass the text that need to be converted to voice to this engine and output will be voice saying the text accordingly. User gives a question to interact with the chatbot.

Following that, an LSTM model is used to analyse the user query. LSTMs, shown in Fig. 1, are a type of recurrent neural network that, rather than simply passing its result to the next section of the network, performs a series of math tasks to work on its memory. There are four "gates" in an LSTM. They are forget gate, remember gate, learn gate, and output gate.

Step 1: The three information sources enter the LSTM and are directed to the forget or learn entryways. Long term information is shipped off the forget entryway, where some of it gets lost (the irrelated parts). The learn gate receives the short-term information and "E." This gate determines what information will be gathered.

Step 2: Data that goes through the forget entryway (it isn't neglected; failed to remember data stays at the door) and the learn entryway (it is learned) will be shipped off the remember entryway (which makes new long-term memory) and the utilization entryway (which updates momentary memory is the final result).