AI Chatbot For Home Appliance Control

DE-42 (MTS)

Hassaan Ahmed Khan

COLLEGE OF
ELECTRICAL AND MECHANICAL ENGINEERING
NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY
RAWALPINDI
2024

COLLEGE OF ELECTRICAL AND MECHANICAL ENGINEERING

DE-42 MTS
PROJECT REPORT

AI ChatBot For Home Appliance Control

Submitted to the Department of Mechatronics Engineering
in partial fulfillment of the requirements
for the degree of
Bachelor of Engineering
in
Mechatronics
2024

Sponsoring DS:
Dr Kunwar Faraz Ahmed Khan
Dr Anjum Naeem Malik

Submitted By:
Hassaan Ahmed Khan

ACKNOWLEDGMENTS

First and foremost, I would like to humbly state praises to Allah Almighty for His blessings and guidance throughout the research process and in general, towards every aspect of life. The faith and belief in His blessings kept me motivated to complete my research.

I would like to express my sincere gratitude to my research supervisor Dr. Kunwar Faraz, Head of Department, Department of Robotics and Artificial Intelligence(R&AI), National University of Sciences and Technology (NUST) for believing in me and giving me the opportunity to work under his supervision. He remained very patient and polite and kept me motivated during the entire research phase.

I would also like to acknowledge the support and guidance of my co-supervisor Dr.Anjum Naeem , Professor, Department of Mechatronics Engineering, National University of Sciences and Technology (NUST) for his assistance and guidance. Special thanks to my friends and family who supported me and kept me motivated throughout the research journey.

I would like express my gratitude to my seniors, colleagues and friends, Miss Saman Khan and Miss Rameesha for providing valuable advices and emotional support during my research program.

ABSTRACT

With the increase in usage of Large Language Models and ChatGPT and advancement in IoT technologies. Their impact on the health industry cannot be negated.

The project aims to utilize the abilities of existing technology of local LLMs and their conversational prowess along with the ability to control multiple devices over the internet through cloud services the combined use of NLP and IoT to assist isolated patients and reduce strain on medical facilities.

The project also aims at app development to help in the regard to allow medical professionals to communicate and monitor more than a single patient at a time.

Table of Contents

ACKNOWLEDGMENTS ..... **iii**
ABSTRACT ..... **iv**
Table of Contents ..... **v**
Table of Figures ..... **vii**
CHAPTER 1: INTRODUCTION **1**
1.1 Problem Statement 1
1.2 Solution 2
1.3 Deliverables 2
1.4 Thesis Outline 2
CHAPTER 2: Literature Review **4**
2.1 Natural Language Processing / Artificial Intelligence 4
2.1.1 Natural Language Understanding 5
2.1.2 Natural Language Generation 7
2.2 Chatbots 8
2.2.1 Chatbots based on Deep Learning: 10
2.2.2 Ensemble Methods: 10
2.2.3 Domain-Specific Chatbot: 10
2.2.4 Chatbot Builders: 11
2.3 Internet of Things 11
2.4 Chatbot Application in Healthcare sector 13
CHAPTER 3: Natural Language Processing **15**
3.1 Overview 15
3.2 Bag of Words 16
3.3 Large Language Models 16
3.3.1 Llama: 17
3.3.2 Prompt engineering: 18
3.3.3 Retrieval Augmented Generation (RAG): 19
3.3.4 Fine Tuning: 20
3.4 Software implementation 20
3.4.1 Python: 20
3.4.2 Libraries: 21
CHAPTER 4: Internet of Things **23**
4.1 Introduction 23
4.2 Hardware Overview 23

Chapter 5 Implementation **28**
5.1 Process starts: 28
5.2 Implementation of Bag of Words: 28
5.3 Implementation of the chatbot with all capabilities and features: 30
5.3.1 Code implementation: 30
5.3.2 Implementation of home appliance control: 32
5.3.3 Firebase Notification System 33
5.3.4 Real-Time Notification System 34
5.3.5 Python Application GUI 37
Chapter 6 - Conclusions and future work **38**
. ..... **38**
References **39**
Annexture **42**

Table of Figures

Figure 1. Complete Program Flow 16
Figure 2. Retrieval Augmented Generation 19
Figure 3 ESP 32 24
Figure 4 8-channel relay module 25
Figure 5.Circuit Diagram 25
Figure 6.Bag of Words Output 29
Figure 7 RAG database development 30
Figure 8 RAG test run 30
Figure 9 Hardware 32
Figure 10 ESP32 Setup 33
Figure 11 Home Screen 34
Figure 12 Signup page 35
Figure 13 Login Page 35
Figure 14 User Interface 36
Figure 15 AI ChatBot Application GUI 37

CHAPTER 1: INTRODUCTION

Artificial intelligence (AI) chatbots utilize large language models (LLMs) to produce responses that resemble those of a human. These LLMs are complex algorithms built to handle enormous volumes of data and use machine learning methods to understand the information. Now limited only by their training parameters and the amount of their datasets, they can generate responses that closely resemble human speech and converse naturally on a broad variety of topics.

The field of home automation is growing because it uses cloud computing to create networks between devices so that they may be controlled remotely without requiring human interaction. With centralized control interfaces that can be accessed by computers, tablets, or smartphones, this technology gives customers the ability to regulate a variety of home features, including lighting, temperature, and security systems.

1.1 Problem Statement

During the peak of the COVID-19 pandemic in 2020, hospitals worldwide faced a critical shortage of personnel to attend to the increasing number of patients, exacerbating the strain on healthcare systems. With the number of COVID-19 cases soaring into the millions globally, hospitals struggled to cope with the overwhelming demand for medical care. For instance, in some regions, hospitals were operating at over 150% of their normal capacity, with medical staff working extended shifts to manage the influx of patients. This underscored the urgent need for a solution that could effectively monitor patients while ensuring their isolation to prevent further transmission of the virus. Concurrently, patients experienced heightened feelings of isolation due to restricted visitation policies, exacerbating their psychological well-being. Additionally, the need for assistance with basic needs, such as controlling home appliances, became more pronounced as patients struggled with limited mobility and resources.

1.2 Solution

The project aims the development of an online assistant to assist the needs of patients and medical professionals. The project focuses to integrate the revolutionary potential of Large Language Models (LLMs) and Internet of Things (IoT) technologies to offer support and companionship to individuals. The envisioned system will mitigate the adverse psychological effects experienced by isolated patients, while the IoT services embedded within the application will grant patients a sense of independence and liberty. Moreover, the project encompasses a monitoring system allowing caregivers to oversee and address the needs of multiple patients simultaneously, thereby alleviating the burden on the healthcare system, particularly evident during the COVID-19 pandemic.

1.3 Deliverables

Following are the deliverables for Ai chatbot for home appliance control:

- Train NLP dataset for greetings and commands for on/off control of appliances.
- Voice to text transcription for processing in chatbot environment in API.
- Using telepresence robot/ standalone microphone as input medium for the patient.
- Using mobile device to send alerts to the caretaker.
- Hardware implementation as a standalone system for a local network.

1.4 Thesis Outline

The structure of the final year project report is explained. Chapter 2 mainly deals with background and literature review including different existing models of versions and available solutions. Chapter-3 includes the methodology we adopted to design the natural language processing and its existing solutions. Chapter-4 deals with the IoT implementation and how it working in our

project. Chapter-5 shows the complete implementation of the project. Chapter-6 consists of concluding the report and exploring future possibilities and directions in which the project can be taken.

CHAPTER 2: Literature Review

This chapter aims to provide a detailed overview about the major domains covered in the scope of this project. The project is a combination of technologies involving Natural Language Processing (NLP) and Internet of things (IOT). This chapter provides a detailed overview about various components involved in the project. The literature review provides a brief review of current progress in these areas of technology.

2.1 Natural Language Processing / Artificial Intelligence

Natural language processing is a well-known field in Artificial Intelligence. Natural language processing is used to derive meaning from human response. It helps machines understand text like people. It has a lot of useful applications in Machine Translation, Question Answering, information retrieval and other related fields.[1]

It emerged to simplify user tasks and fulfil the desire to communicate naturally with computers. It can be categorized into two components: Natural Language Understanding, concerning comprehending language, and Natural Language Generation, aimed at generating text. Linguistics, the study of language, encompasses Phonology (sound), Morphology (word formation), Syntax (sentence structure), Semantics (syntax), and Pragmatics (understanding). Noam Chomsky, an influential linguist, significantly transformed the field of syntax [2]. Additionally, Natural Language Generation (NLG) involves creating meaningful phrases, sentences, and paragraphs from an internal representation. The primary aim of this document is to elucidate key terminologies in NLP and NLG.[3]

Natural Language Processing is distributed into 2 parts i.e Natural Language Understanding and Natural Language Generation. The field of NLP can be classified broadly in this way so first we will discuss NLU.

2.1.1 Natural Language Understanding

NLU empowers machines to comprehend and assess natural language by extracting elements such as concepts, entities, emotions, and keywords. Its application in customer care involves comprehending issues communicated by customers, whether spoken or written. Linguistics, as the scientific study of language, delves into the meaning, context, and diverse language forms. Therefore, understanding crucial NLP terminologies and the various levels of NLP is pivotal. Subsequently, we delve into some frequently used terminologies across the different levels of NLP.

A) Phonology:

A branch of Linguistics deals with the organized structure of sounds within a language. Its name originates from the Ancient Greek words "phono," meaning voice or sound, and "-logy," denoting word or speech. In 1993, Nikolai Trubetzkoy defined Phonology as "the study of sound within the language system," while Lass (1998) [4] described it as a field broadly concerned with the sounds of language, focusing on the behaviour and arrangement of these sounds. Phonology involves the meaningful use of sound to encode language in human communication.

B) Morphology:

Morphology, within the realm of linguistics, is the study of the structure, formation, and relationships of words. It examines how words are formed, their internal structure, and how they are related to other words within a language. Morphology primarily focuses on morphemes, which are the smallest units of meaning within a language. These units can be roots, prefixes, suffixes, or inflections, and understanding their arrangement and combination forms the basis of understanding how words are created and their grammatical function within a language.

C) Lexical:

In Natural Language Processing (NLP), "lexical" refers to matters concerning the vocabulary, individual words, and their attributes within a language. It specifically deals with the vocabulary or lexicon, including word choice, meaning, and the characteristics of individual words. Lexical analysis involves understanding and processing words and their context within a given text or language. It focuses on how individual words or lexemes are stored, accessed, and manipulated within computational systems for linguistic analysis or machine understanding.

D) Syntactic:

If you're referring to "Syntactic," within the context of language or linguistics, it relates to syntax, which involves the arrangement of words to create well-formed sentences in a language. Syntactic analysis examines the structure, rules, and relationships among words to form meaningful sentences or phrases. In the context of Natural Language Processing (NLP), syntactic analysis is crucial for understanding the grammatical structure of sentences, enabling machines to comprehend language patterns and relationships among words in each text.

E) Semantic:

In the context of language and linguistics, semantics pertains to the meaning of words, phrases, sentences, and how language conveys meaning. It involves the study of the relationships between words, their denotations (literal meanings), and connotations (associations or implied meanings). In Natural Language Processing (NLP), semantic analysis is used to understand and extract the meaning of text or spoken language.

F) Discourse:

Discourse refers to the use of language beyond individual sentences, encompassing larger units of language such as conversations, texts, or dialogues. Discourse analysis examines how language is used to convey meaning in context, including the organization of information, coherence, and the pragmatic aspects of communication.

G) Pragmatic:

Pragmatics deals with the study of language use in context. It focuses on how people use language to achieve specific communicative goals, considering the context, intentions, and implicatures (implied meanings) of communication. In NLP, understanding pragmatics is important for interpreting language in real-world situations, where the meaning of a statement can depend on the context and the speakers' intentions.

2.1.2 Natural Language Generation

Natural Language Generation (NLG) involves creating meaningful phrases, sentences, and paragraphs based on an internal representation. It is a component of Natural Language Processing and occurs in four stages: recognizing objectives, strategizing how to accomplish these goals by assessing the situation and available communication resources, and materializing these plans as coherent text.

A) Speaker and Generator:

To generate text, two primary components are essential: a "speaker" (which could be an application or a person) and a "generator" (which could be a program). The speaker or application expresses intentions, and the generator translates these intentions into coherent and situation-relevant phrases.

B) Components and Levels of Representation:

The process of language generation involves several intertwined tasks. Content Selection: Involves choosing and incorporating information into a set, which may entail the addition or removal of representational units. Textual Organization: Information must be structured according to grammar, involving sequential ordering and linguistic relationships like modifications. Linguistic Resources: Selection of linguistic elements (words, idioms, syntactic constructs) to support information realization. Realization: The chosen and organized linguistic resources must be manifested as actual text or voice output.

C) Application or Speaker:

This is only for maintaining the model of the situation. Here the speaker just initiates the process doesn’t take part in the language generation. It stores the history, structures the content that is potentially relevant and deploys a representation of what it knows. All these forms the situation, while selecting subset of propositions that speaker has. The only requirement is the speaker must make sense of the situation [5].

Chatbots are primary examples of use to NLP technology so that there is natural communication between user and the system[6], [7]. The advance systems not only works on parts of sentences

but also considers other factors such as sentence structure, context or idioms based on pre-programmed databases and knowledge acquired during the working of the systems[8], [9], [10]. Voice assistants also use these technologies but they are not suitable for embedded systems so we need to work on that aspect which is addressed in [11]. Which is Edge intelligence and Green IoT which allows combined use of cloud computing and AI techniques. It also helps so that preprocessing is done locally and avoid the problems in cloud-based architectures.[12].

2.2 Chatbots

A chatbot is a computer program or an artificial intelligence application designed to simulate a conversation with human users, usually via text or speech. Chatbots are often used in various contexts, such as customer service, information retrieval, entertainment, and task automation.

They employ natural language processing (NLP), a branch of artificial intelligence that enables machines to understand, interpret, and respond to human language. Chatbots use predefined rules, machine learning, or a combination of both to comprehend the user's input and generate appropriate responses.

In 1950, Alan Turning known as “The Father of AI” he published a paper named “Computing Machinery and Intelligence” asking the question “Can machines think?”. He proposed the Turing test (originally called imitation game) which acts as a test for checking wheater a machine can display intelligent behaviour which is equivalent to or indistinguishable from a human being its ability to give human like responses. It still remains one of the most known and criticized concept and holds high importance in the field of philosophy of artificial intelligence.[13]

One of the first attempts made at the Turing test were made by ELIZA developed by Joseph Weizenbaum in 1966 at MIT AI laboratory which was made to mimic the behaviour of Rogerian Psychotherapist [16]and PARRY made in 1972 by Kenneth Colby. PARRY was used to simulate a person with paranoid schizophrenia and used as the same approach as ELIZA with more advancement [14]. PARRY passed Turing test with 48% accuracy. However, the first chatbots

used pattern matching and parsing methodology to give illusion of intelligence and natural conversation.

In the current era, chatbots have been classifies the contents of the users input and gives and appropriate response and in addition to this they also try to simulate new conversation by asking new questions [15]. Nowadays chatbots usually have a dialogue database. When users ask queries, the chatbot use natural language understanding and grammatical principles to identify the primary words, verbs, and objects of the sentences. language analysis methods [16], after which the meaning is contrasted and reacts suitably to each term in the database. In a technical sense, chatbots fall approximately into the subsequent three categories.

1. **Scripting Chatbot:**
   Chatbots react by following a predetermined script. In discussions, the decision tree model is frequently employed. The decision tree is utilized to construct the questions and answers, according to the users' interaction. As a result, the system's comprehension of natural language is limited.
2. **Artificial intelligence chatbot:**
   It can solve problems just as well as a human and has strong natural language analysis capabilities. But because artificial intelligence technology is still in its infancy, it is challenging to develop chatbot technology.
3. **Assist agent Chatbot:**
   If the chatbot is unable to resolve the issue, the conversation will be passed to an artificial service, and the artificial response will serve as the chatbot's training data. After a long period of learning, the dependency on AI can be reduced.[17]

Siddhant et. al. [18] have classified chatbot development techniques in the following 5 categories. Rule based: This method is a fundamental technique widely used in many Chatbots. It involves a predefined structure of rules utilized to convert user input into an output. These rules, whether simple or complex, break down the input into a sequence of tokens to identify patterns [18]. Breaking the text into a set of words is crucial to grasp the grammatical structure of the input.[19] A categorized set of keywords or patterns is employed to generate responses. The initial Chatbot,

ELIZA, relied on this approach. Nevertheless, these Chatbots have limitations in responding to patterns that don’t align with the predefined script.

2.2.1 Chatbots based on Deep Learning:

Chatbots created using Deep Learning employ Machine Learning Algorithms. These bots learn from their interactions with humans and the data they're trained on [20] They form their own understanding of text and improve with more data. There are two primary types: Retrieval Based and Generative. Retrieval Based Bots use heuristics and semantic networks to derive the most appropriate responses from predefined repositories. In contrast, Generative models are more advanced and do not rely on pre-existing repositories. Generative Chatbots can generate responses they haven't encountered before, but they can also make errors and are challenging to train. Retrieval-based Chatbots also produce incorrect outputs because they rely entirely on data retrieval [21]

2.2.2 Ensemble Methods:

Contemporary Chatbots like Alexa emulate conversation similar to a virtual family member. These advanced bots are constructed using Ensemble Methods, combining functionalities from Rule-Based, Retrieval-Based, and Generative methods to handle user requests. Chatbots built with this approach can engage in conversations on various topics.

2.2.3 Domain-Specific Chatbot:

The strategy for developing Domain-Specific Chatbots enhances their efficiency. These Chatbots find application in different domains like Education and Healthcare. This approach allows Chatbots to implement various techniques and methods tailored for a specific domain, broadening their coverage in a particular subject area.

2.2.4 Chatbot Builders:

Chatbot Builders are non-coding tools designed to create Chatbots. These platforms offer ready-made solutions and predefined scenarios that enable the development of Chatbots within a short timeframe. They function using a simple drag-and-drop approach. Several AI Bot builders like Mobile Monkey, Botsify, Chatfuel, SnatchBot, among others, are available to swiftly and affordably develop Chatbots.[22]

2.3 Internet of Things

The term IOT was first coined by the Kevin Ashton in 1999. He came up with the idea to track missing items using RFID tags to resolve the problem of supply chain.[23]. He is reffered as the "Father of Internet of Things" as he made it accessible for both technical and non-technical people.[24]. IOT is essentially a system which takes input from nultiple sources and gives output to perform meaningful actions and justify it using the modes of big data and cloud computing.[25] This can be expanded to home and building automation using IP-based protocols.[26].

Automation means essentially a system that works with minimal human intervention and automates a manufacturing process. For Home automation we can apply the same concept using Internet of Things and enabling connectivity between devices and gain control over home appliances.[27]

Since the beginning there have been efforts to make life more comfortable and luxurious for human beings. The idea is to make life ideal while also taking into consideration the power consumption of such models.

The ideal model is the one that can be accessed by the end user from anywhere in the world. The field of home automation has been advancing rapidly and several works have been proposed which will be discussed later on. One such framework is the use of PIC16F887 microcontroller with participation of GSM(Global System of Mobile Communications) which gives a clear robotized home framework with a baud rate of 9600 bps.[[28]The scope of GSM is that it can be utilized worldwide. Home automation using IoT uses Bluetooth module and ethernet port which can be

used to detect devices connected to the internet and an Android application is used to control home appliances from a distant location.[29]

The research in home automation has drastically increased since the rise of IoT applications which has made it possible to have direct communication between two or more devices.[30]. An architecture to regulate home devices remotely and optimize energy consumption according to the users' needs has been proposed.[31] Advances have also been made in door automation and home surveillance for controlling the door remotely using mobile application have been proposed. In the particular work, a DC motor was used along with a raspberry Pi using L293D to control door movement.

Emergence of internet of things has made it so that our life is more and more interconnected and digitization has taken a hold. This has made provided more comfort and control to the people over there everyday activities. This however is yet only available to the people who have considerable understanding of computer systems and are able to use latest tech devices like smart phones, tablets, smart watches etc. It leaves the people who are unable to interact using such devices such a disabled and the elderly a marginalized group and they are unable to take advantage of this. In this regard voice-based systems provide acts as an enabler to this populace and provides them access to IoT applications which are a high percentage of our society.[11]

Current trends indicate that people live longer on average [32] and such as the loss of autonomy and its associated risks can be curtailed by providing them access to IoT tech can improve their independence and a better life quality and reduce additional healthcare related costs associated with admission to residential healthcare systems. The problem with such systems is however that they are quite complex and require a bit technical knowledge. This raises the need for a voice-based interfaces are a convenient method which can be used to communicate with IoT embedded systems.[11]

Sahoo et. al.[33] presented an home automation systems which utilizes Intel Galileo to integrate systems with cloud computing and wireless communication so that the users can interact with the system of various fans, lights and appliances and storing the data on the cloud.

El-Hajj M. et. al[[34] gives an up to date view of authentication field. It provides a summary of protocols proposed in the literature of IoT. It compares different protocols and compares their positives and negatives.

2.4 Chatbot Application in Healthcare sector

Survey report as stated in [35] directs towards and IoT-based healthcare system. The researches show the realization that there is a need for combining multiple IoT services. This however requires a large amount of data to be handled correctly during monitoring. Cloud computing can be considered as a go to method as the technology is promising in regards to handling huge amount of data as it provides a centralized and efficient solution for information processing in healthcare sector[36].

Progress in technology with time has permitted for refining healthcare services suited to more individualistic needs and patient supervision. The pandemic of COVID-19 has also highlighted the need for taking on several aspects of telemedicine [37]. Still, it is difficult to make such technologies accessible for elderly or disabled individuals as they face numerous challenges in adopting and familiarization with such gadgets as then can only be accessed through latest Graphical User Interface. However, newer systems allow for the users especially elderly individuals to interact in a more natural way by the way of voice communication.,[38], [39], [40] Studies indicate a preference by the elderly towards voice-based assistants as compared to traditional inputs such as touch pads and keyboards. Similarly, most questions asked by the elderly are related to medical field and health questions. [41]

Voice assistants, when coupled with most recent advancements in technology, provide significant improvement to the standard of healthcare services, specifically in remote areas where such services are minimal.[42] These types of applications usually rely on edge computing to deal with critical response time for medical devices and data security. [43].

In terms of growth, voice-based systems have garnered significant ground in healthcare sector.[44]. Alongside telemedicine and advancements in voice processing technologies, voice-based healthcare apps are emerging to deliver high-quality monitoring care and prevent excessive readmission rates [45]. Voice-based applications, when used in conjunction with Internet of Things (IoT) devices, enable smart systems to diagnose problems more quickly, provide more precise treatment, and anticipate possible symptoms through data collection and processing. This is made

possible by the application of machine learning (ML) techniques for diagnostic monitoring, which ultimately improves patient outcomes [46].

CHAPTER 3: Natural Language Processing

This chapter includes a detailed view of how NLP is being utilized in this project. It includes basic principles and techniques as well as the utilization of LLMs and library packages needed for the implementation of the project.

3.1 Overview

Natural Language processing is a branch of artificial intelligence that mainly explores the methods in which computers and human beings communicate. The main object of NLP techniques is to analyze, interpret and respond to human queries in the form of speech or text. It is a branch of computational linguistics. It focuses on the combined use of computer science linguistics and artificial intelligence to study the complexities of human communication.
It can be briefly defined in the following steps:

1. **Data preprocessing:**
   In data preprocessing we make it so that the data is prepared by cleaning it up and giving it readable form so that NLP algorithm can analyze it. Some of these techniques include **tokenization** which breaks data into units such as words, punctuation and phrases. **Lemmatization** reduces the words to their base form. **POS tagging** breaks sentences to in nouns, verbs and other parts. **Parsing** deals with the relation between words
2. **Algorithm development:**
   Algorithm development means that to take useful information from the data so that NLP algorithms can be applied to it. Common natural language processing is sentiment analysis, to determine the emotional context of the input which is labeled as positive, negative or neutral. Named entity recognition identifies people, locations, dates and organizations. Topic modeling is used by pairing similar words and phrase to understand the general theme of the project from a collection of text. Machine translation is utilized to convert one language to another with the application of machine learning. Language modeling is used for autocomplete, auto correct and speech-to-text systems.

Following is the flowchart of the complete program.

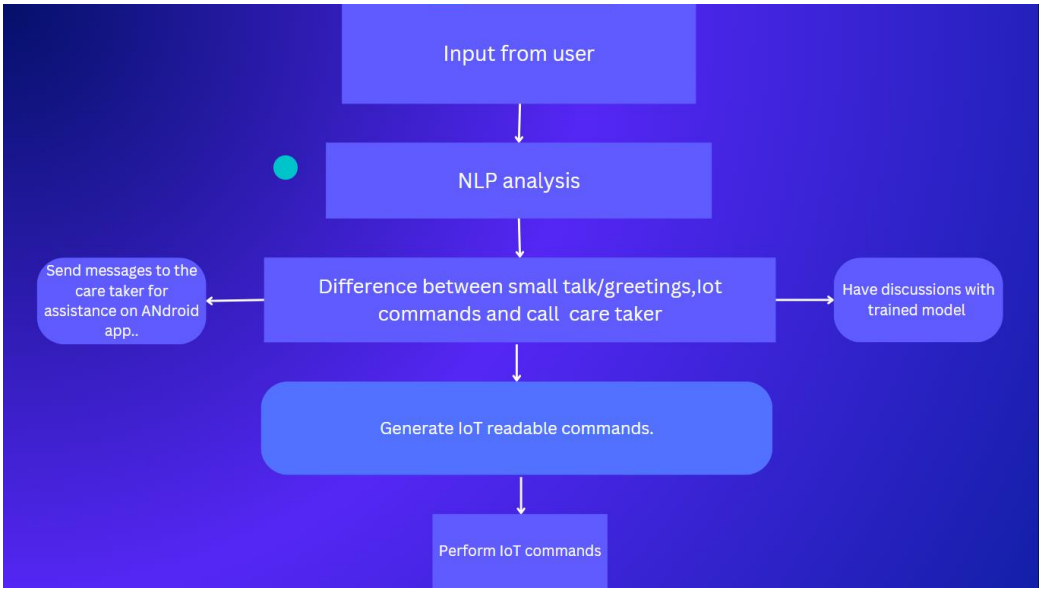

Figure 1. Complete Program Flow

3.2 Bag of Words

Bag of words is the simplest of the text processing algorithms. It is the representation of text in the form of numbers.it describes the occurrence of a word in the document and keeps track of it. It disregards the use of grammar or structure of sentence and all other associated characteristics. This technique is utilized because it enables us to store data in fixed-length vectors which are easier for the machine learning models to process and also, we have numerical data to work with instead of text data.

3.3 Large Language Models

Large language models describe a class of deep learning architectures called transformer networks. Transformers are neural network models that keep track of the context and meaning of sentences by tracking sequence of data in words and phrases in a sentence.

Transformer models were first described in 2017 paper "Attention is All You Need" (Transformer technology is the primary tool utilized in development of NLP.[47]

A transformer consists of a multiple layer. It has self-attention layer, feed forward layers and normalization layers which are used to decode stream of input to anticipate the output. Multiple layers can be utilized to make the LLM more powerful. Innovation in transformer models mainly lies in two features incorporated in LLMs, self-attention and positional encodings.

Positional encoding allows words to be given to the model non-sequentially in the neural networks by embedding the order in which input occurs.

Self-attention enables the model to consider which input is more relevant to the context and the output. This is done by assigning weights to the words. This is a learned behavior which is perfected over time by as the model works on mountain loads of data. This enables the models to take large streams of input which can be processed properly while keeping track of the context and keeping relevant information in the loop.

This non-sequential analysis is usually accompanied by multiple computations which is generally done with GPUs which work on multiple inputs in parallel.

3.3.1 Llama:

Llama 2 is a collection of pretrained and finetuned LLMs which are trained form 7 billion to 70 billon parameters. The fine-tuned models called Llama 2-Chat are optimized for dialogue use cases. The models outperform the open-source chat models on all bench marks tested and based on human evaluation for helpfulness and safety, the models are optimal substitute for closed-source models.

For the use case of this project, I have used "TheBloke/Llama-2-7B-Chat-GGUF/llama-2-7b-chat.Q4_K_M.gguf" which is trained 7B parameters with and has quantization of 4 bits. The model is recommended for most use cases such as ours and is compatible on the operating hardware. It takes 4.08 GB of space and requires maximum running RAM of 6.83 GB.

The Llama2 chat or any other LLM can be utilized in the following manners. LM Studio is a desktop application which is used for accessing local and open source LLMs on PCs. The software allows to download models from hugging face and currently it supports ***GGML*** Llama and StarCoder models available on hugging face. The software has the capability to run LLMs on CPU

completely offline and without the need of internet. It also enables the making of http server of said models in OpenAI compatible server.]

3.3.2 Prompt engineering:

Prompt engineering is a new niche being developed as we move towards a more AI driven world. While the thought process behind it exists in the form of Google search i.e. we can get a variety of results on the same topic by changing a few key words and go for the closest possible result. It can be defined as the practice of designing and refining prompts to guide the AI model towards a specific output.

Few key components play a role in prompt engineering such as

- **Model architecture**
- **Training and tokenization**
- **Model parameters**
- **Temperature and Top-k sampling**
- **Loss function and gradients**

Out of the above understanding of all is important while temperature is the parameter a prompt engineer has to deal with as it influences the randomness of the response and is essential in how does the model work in a typical setting.

Following are the requisites of a good prompt:

Instruction:

It clearly states what is expected from the prompt and can be very decisive in the output.

Context:

It gives the model the ability to focus and provide a clearer response as to what is expected from it. It can be useful while making it work on more specific topics.

Input data:

Along with the context, this is the information which we want the model to work with. This can include paragraphs, numbers, code and words.

Output indicator:

In the form of role-playing scenarios, such as building a bot for assistive measures and communication, it tells the model what output we need from it and what tone is expected from the model.

For my project, I utilized prompt engineering and tested my output against various iterations with multiple temperature variations. The final output prompt is suitable for the task at hand and the input context is continuously appended to keep track of the conversation and the fields are divided into system and user.

3.3.3 Retrieval Augmented Generation (RAG):

AI models are well suited to engage in sentiment analysis and named entity recognition, however as smaller models may not be updated to the latest information and only have an understanding of the data, they have been trained on so these models do not have the capability to give accurate facts and info. What we can do to remedy this is to give them access to external documents which can be utilized as a knowledge base for the model and give more accurate responses.

This is mainly in knowledge extensive tasks such as a chatbot which gives answer about the documentation of a company's existing technologies. For this, RAG is utilized. The RAG flowchart shown in *Fig* is shows the difference between when ChatGPT is asked a question about a recent phenomenon with and without access to a documentation. This clearly illustrates the contrast between the two use cases.[48]

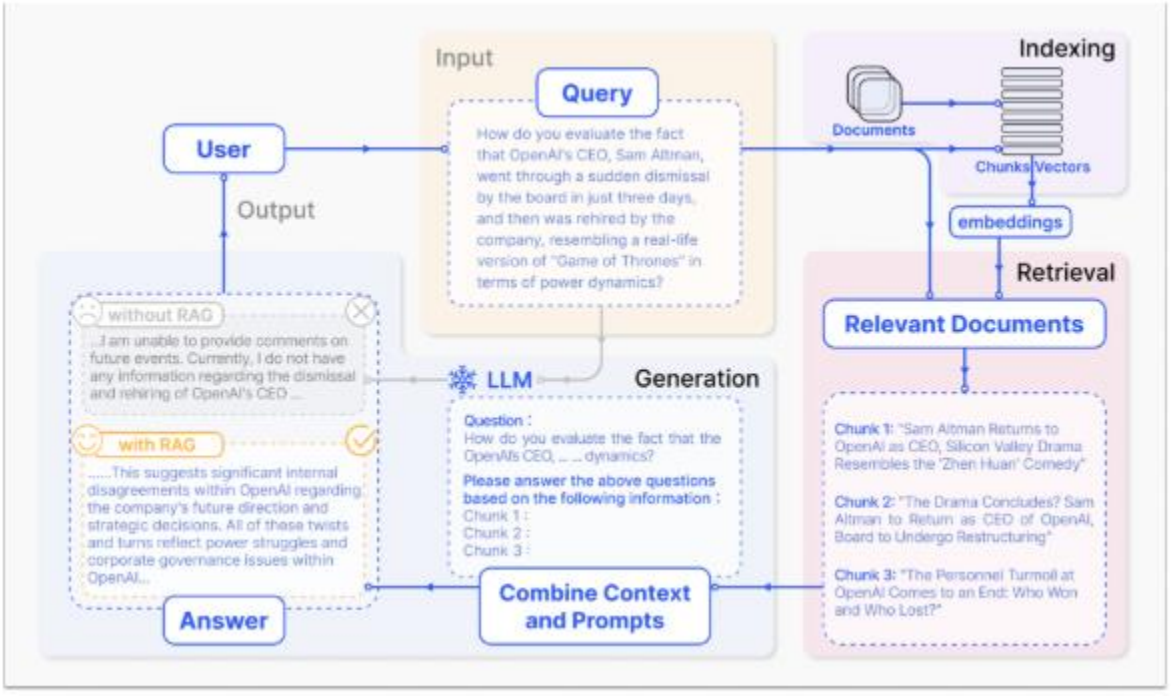

Figure 2. Retrieval Augmented Generation

In my project RAG was initially utilized but it does not enhance the capability of the current scope of the project as it is to have generic discussion with the user and does not need a specialized knowledge base to do so.

3.3.4 Fine Tuning:

Chat model fine-tuning is modifying a pre-trained language model (e.g., GPT-3) to make it more efficient for certain jobs or to conform to certain user needs. In order to accomplish this, the model is usually trained again using a specialized dataset that represents the intended domain, tone, or kind of interaction. Fine-tuning improves performance and relevance by modifying the model's parameters to help it comprehend the subtleties and context unique to the new data. For example, a dataset of customer service chats can be used to refine a general-purpose chatbot model and improve its ability to handle support inquiries. By ensuring the chatbot can respond with precision and awareness of context, this technique improves user experience and increases the model's usefulness for specific applications.

In this project fine tuning is implemented to make it so that the responses are more tailored towards the user.

3.4 Software implementation

For the implementation of this project, we mainly used python for its programming and utilized anaconda navigator for the smooth setup of environments and variables.

3.4.1 Python:

Python is a general-purpose programming language. It is a dynamically typed and garbage-collected language. It is widely used for multiple programming applications as it can support multiple programming paradigms such as structured, object oriented and functional programming.Multiple features of the language are utilized in this project. We have utilized the language for the task of data management, web services, voice processing and AI

integration.Below I have summarized the main features of the language utilized.

The Python and R programming languages are distributed via this Windows-based software platform. It has a rather sizable repository with 1500 data science packages, such as Matplotlib, NumPy, and Pandas. Additionally, it enables the user to establish a virtual environment where packages of various versions can be installed without causing conflicts. Package management, the Jupiter notebook interface, code execution and deployment, and library integration are a few more features. Anaconda was utilized in this project to create and save the virtual environment.

For this project we have utilized VS code as the main IDE. VS code is an open-source and free to use coding editor. It supports multiple languages and is highly efficient in its variety of capabilities such as auto completion and syntax highlighting.

3.4.2 Libraries:

Following libraries provide the main functionality to the program.

VOSK:

Vosk is an open-source speech recognition toolkit with offline streaming capabilities. The Vosk library is easily downloadable and can support more than 20 languages and dialects. For the purpose of this project, we have used the language model of English-In which is English language in Indian accent as it gave the best result when compared against other dialects.[49]

Langchain:

Langchain is an open-source framework for development of applications using LLMs. Its toolkit and APIs ease the process of building chatbots and virtual assistants which harness the capabilities to LLMs.[50]

In this project for the implementation of RAG we utilize RAG which we will be explained in code implementation part.

OpenAi:

OpenAI is a python library which provides convenient access to OpenAI REST API services which is needed to access the capabilities of LLM local server established using LM Studio and also access to all its necessary functions.

Firebase_admin:

This library provides back-end services to python applications such as real time database, authentication, app connection, analytics to name a few. For this project we have utilized the real time database. The detailed implementation will be in the next section.

Eel:

A simple and effective tool for creating contemporary desktop apps with web technologies is the `eel` Python package. With its smooth integration of HTML, CSS, and JavaScript, Python allows programmers to design graphical user interfaces (GUIs) for their Python applications. Using eel`, you may call Python functions from JavaScript and vice versa, allowing for seamless language interoperability. The cross-platform functionality of this library—which runs on Linux, macOS, and Windows—makes it easier to design apps that are compatible with several operating systems. Because eel is lightweight and requires few dependencies, it's simple to install and use. To render the HTML files in a lightweight web browser, you must first build up your frontend using web technologies, initialize {eel} in your Python script, expose Python methods to JavaScript, and launch the program. For example, defining and exposing functions in Python code and developing HTML and JavaScript to interact with these functions can be used to create a simple application. This configuration makes use of the adaptability of web technologies to enable quick prototyping and development of desktop apps. For resource-intensive applications, `eel` might not perform as well as native GUI frameworks, but it is perfect for developing fast prototypes, instructional software, and internal tools.

CHAPTER 4: Internet of Things

4.1 Introduction

The internet of things or IoT is a mesh of interconnected devices to exchange data with each other through cloud services. It can include sensors and software particularly useful in digital machines and consumer electronics. In home automation IoT can be utilized in multiple aspects such as smart lights, security systems, remote temperature control and voice-controlled appliances and voice assistants. All these systems can enhance convenience and efficiency and improve overall quality of life.

The other major aspect of this project is home appliance control. The purpose of incorporating it in our project is to provide autonomy to the patient as they are in isolated environment. We have utilized the IoT to connect the chat application running on device to the microcontroller through the availability of cloud services.

4.2 Hardware Overview

In the system we have a a computer running a python script to take user queries, a cloud database to recognize these changes, a microcontroller module to detect the said changes and change the state of the appliances. The following are the main components used in the project.

ESP32 dev kit:

ESP32 is a prototyping board which is easily programmable in Arduino IDE and it is compatible with bread board. It has built in 2.4 GHz Wifi and a Bluetooth Wireless connection. Furthermore, is has a 512 kb SRAM and a 4 MB flash memory integrated in its development board. The board has 21 pins for interface connection I2C,SPI,UART,DAC and ADC.

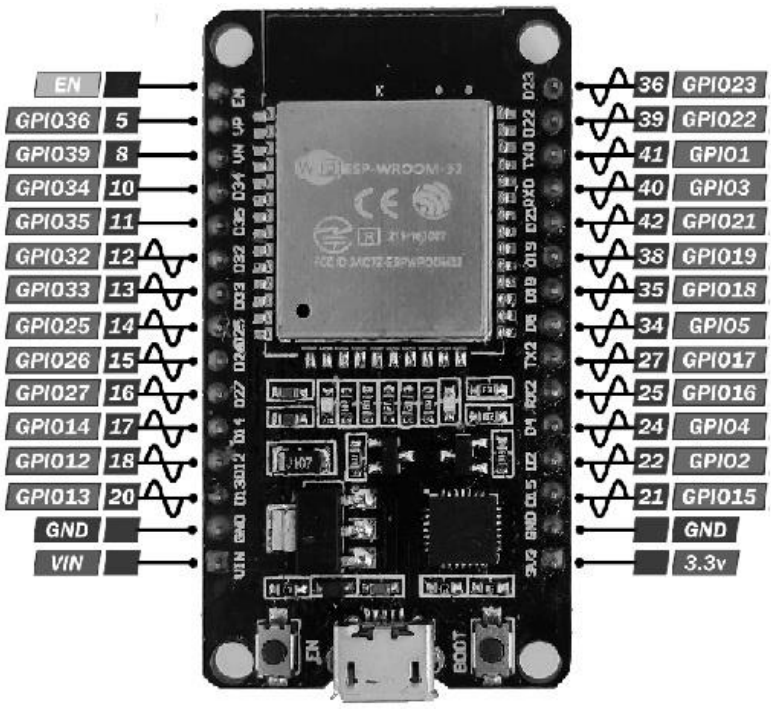

Figure 3 ESP 32

8-channel relay module:

An 8 channel 5 V relay module with LED indicators which is controlled by the microcontroller. It has a wide input range from 3V DC to 5 V DC available on Arduino and ESP boards. The module is rated at 10A@250V AC. It is highly capable in controlling high current devices whose load cannot be handled by digital I/O pins of microcontroller. The module is especially capable in its use for smart home applications.

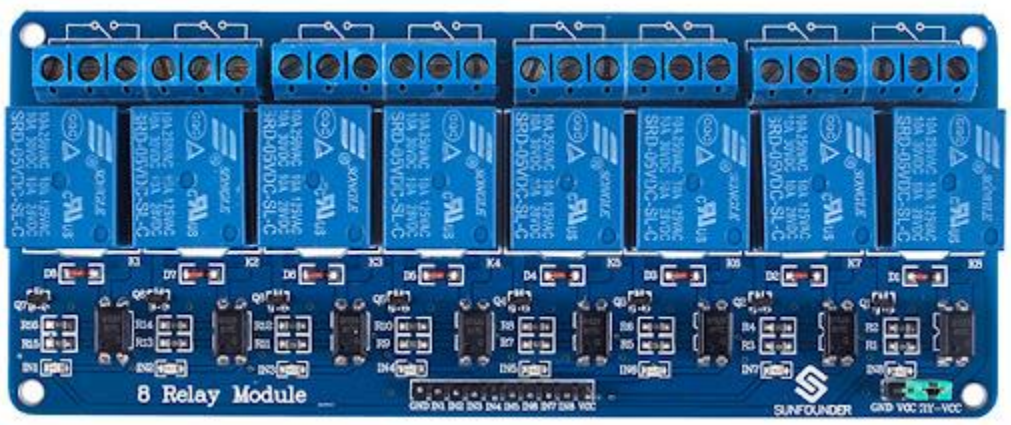

Figure 4 8-channel relay module

Circuit Diagram

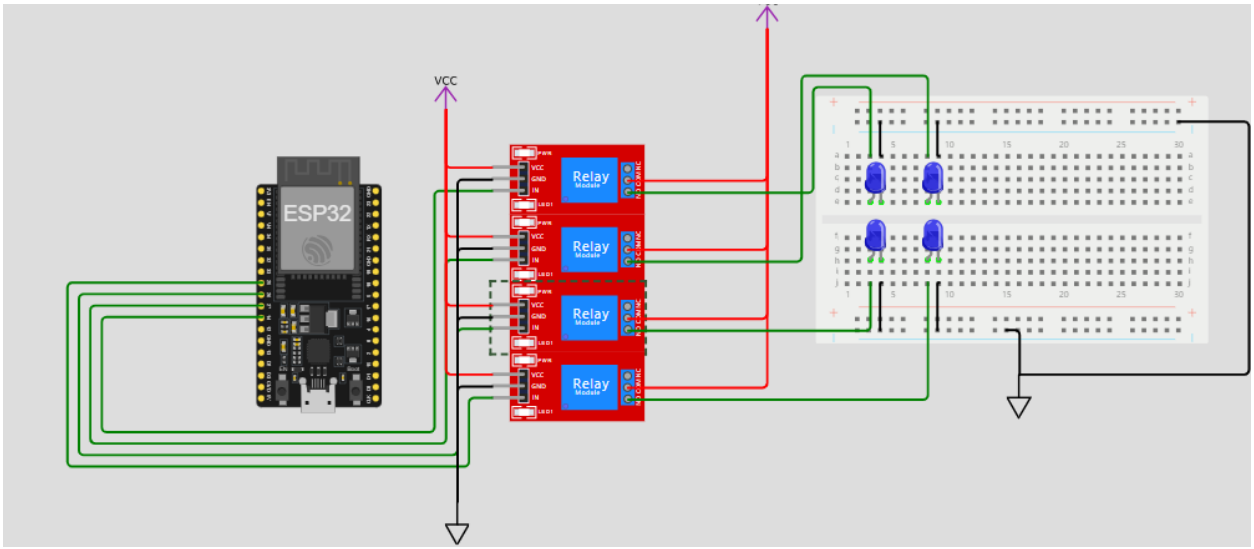

Figure 5.Circuit Diagram

Firebase Database:

A NoSQL database housed in the cloud, the Firebase Realtime Database allows real-time data synchronization and storage across all connected clients. With its dynamic, JSON-based data format, it is made to facilitate server, web, and mobile development. It was developed by Google. Real-time synchronization, which guarantees that any changes to the data are instantly reflected across all clients, is one of its primary benefits. This is especially helpful for apps that need to get

updates instantly, such chat programs, teamwork tools, and Internet of Things systems. Strong offline features are another feature of Firebase, enabling data to be locally saved on the device and synchronized after connectivity is restored. Another crucial component is security, for which Firebase offers a configurable, rules-based security paradigm to manage data validation and access.

Python script for handling:

In python we import necessary libraries for communication with cloud. The library is called Firebase_admin and it is explained in the above section. The script is initialized by giving it firebase access key and JSON variables defined in database. When the program enters the process command stage after hearing the trigger word it identifies each input and test it against the mapped words and phrases in the perform command function. When an IoT command such as turn on is said it generates a variable update is performed on the fire base server indicating change of appliance status.

The script is part of the appendix.

ESP32 initialization:

For the ESP32 we download the ESP32 board in the Arduino IDE. Also, the FirebaseESP32 library is downloaded. The WiFi library is also included from the esp32 package. We give the credentials to he wifi network to bring the esp32 online indicated by the blinking blue led built-in ESP32 board. Once the connection is established it reads the value from the Firebase Real Time Database and indicates the changes in the device status via the pins connected to the relay module.

Communication protocols:

The ESP32 uses Wifi to connect with the internet and get Firebase commands. Internet ensures that commands from anywhere can be sent via a stable internet connection.

The complete system combined with the AI chatbot, Firebase server and ESP32 microcontroller has internet connection to ensure smooth connection between all the devices.

Implementation details.

Data security:

One of the main problems faced in smart home applications is breach of security of the network and compromising its integrity. To ensure this does not occur secure communication channels, data encryption and authentication methods are implemented.

For each individual system there is a pass-key and a user ID associated which is given by using authentication methods available in Firebase. The user id and password are also part of the ESP32 and python script to ensure that network breach is minimized.

Chapter 5 Implementation

This chapter explains the combined working of the entire project. It includes the working of communication protocols and web services used by the system for IoT implementation. The chapter also indicates how each part of the project is connected and the way the service works as a whole for the system. The chapter also explains how all the individual parts are integrated.

5.1 Process starts:

The process starts by running the script on the laptop. The app is downloaded on the mobile phone and the ESP32 module is configured and all the hardware is set up according to the diagram. The initial query is always hey+assistant_name in the program. The voice input taken and is measured against each of the process_command() function use cases. If the chat feature is invoked it enters the chat_with_assistant() feature. The conversation between the LLM and the user is continued and stored until the user indicates to stop the conversation. The system then again starts to listen for its commands. When IoT commands are generated the Firebase database is accessed and the status of each device is updated accordingly on the RTDB. Through the RTDB it is communicated to the ESP32 module and real time result of the command can be observed. If the communication function is invoked, the user is capable to send text notifications to the care taker through voice to text transcription. Multiple messages can be sent by invoking the function repeatedly. These are stored in the firebase RTDB and sent to the android application and are shown as notifications in the patient's profile in the system. New users can be added and patients can be assigned to them.

5.2 Implementation of Bag of Words:

The first implementation of the chatbot was using BOW. The first part of this is to train a neural network for the program. The intents are loaded from the intent file, punctuation marks are ignored. The file is iterated each time and are separated in by making three empty lists named 'words',

‘classes’ and ‘documents’ for storage of processed data. The program tokenizes the data and updates the above lists with the words and also classifies the data according to the said intent. Then training is done in which we lemmatize all the data and keep track of occurrence of each word in the documents. Then a feed forward neural network is initialized using Keras’ sequential API. It consists of an input layer with 128 neurons, a dropout layer to prevent overfitting, a hidden layer with 64 neurons, another dropout layer, and finally an output layer with neurons equal to the number of classes, using SoftMax activation for multi-class classification. The model is compiled using stochastic gradient descent (**SGD**) optimizer with a learning rate of 0.01 and momentum of 0.9. The loss function is set to categorical cross-entropy, and accuracy is used as the evaluation metric.

Then using the training data which contains words and classes are stored.

The chatbot is then initialized to observe the workings. First input text is preprocessed to pass into the model. The training datasets of intents and words and classes. The user input is made into a bag of words and then it predicts the intent of the user and generates the appropriate response.

This technique, however, is limited in its capacity as it cannot function on the larger input and it gives randomized answers. Also, it loses context of the conversation.

- Bag of words was implemented with a few questions stored in its intents file. The program logic is defined in the section. BOW was not suitable for the use case as it of lost track and could not answer beyond its recorded scope.

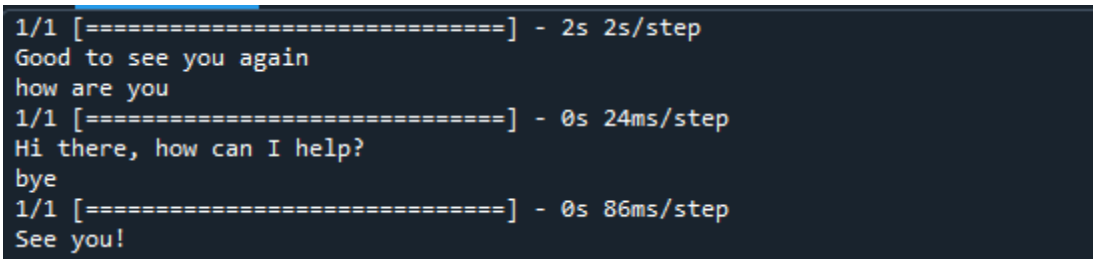

Figure 6.Bag of Words Output

- **Implementation of Retrieval Augmented Generation:**
  RAG was implemented in this project as a means to enhance capability of the system by providing it a greater working knowledge base. The RAG system was implemented using Ollama application as a means to generate vector embeddings and for the database using Langchain framework and for the analysis of the PDF files provided. The RAG system was given the data of board games to read and store in a database as it was easier to verify and work with than some complex medical diagnostics.

Figure 7 RAG database development

The system was provided with test cases to check against the expected response of the system and the response generated by the system. The two test cases were both passed by the system.

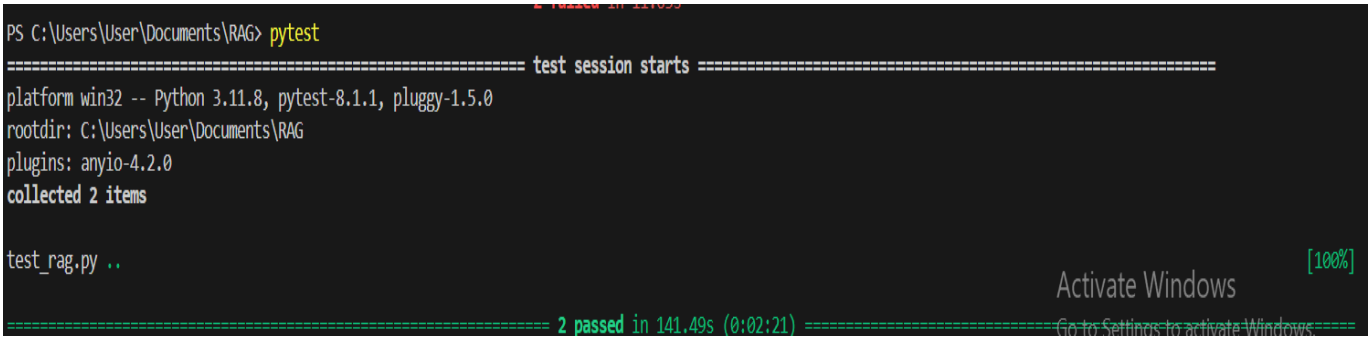

Figure 8 RAG test run

5.3 Implementation of the chatbot with all capabilities and features:

The system implemented with the integration of GUI and all features of IoT, chatting and the application created. The functions are explained in the sections above. When the system is implemented the updates of the data and the hardware are also shown.

5.3.1 Code implementation:

Text to speech conversion:

An offline speech recognition system can be initialized using the provided Vosk-based code. The first step involves importing the required libraries, which include Vosk's Model, KaldiRecognizer, subprocess, json, and time. The language model (vosk-model-small-en-in-0.4) is then used to initialize a Vosk model, and a KaldiRecognizer is made to analyze audio frames with word-level detail enabled. Audio frames are continuously processed by the speech_recognition function from a queue (recordings). It obtains audio frames, applies rec.AcceptWaveform() to process them, and

then takes the recognized text out of the output. Next, output is used to attach this text to the output.append_stdout(), with the addition of a short sleep interval to control the timing of the loop. Alternatively we can utilize speech_recognition python library which initializes a text-to-speech engine in the program. We initialize Microphone as source and Recognizer as recognizer. Then we select source as **s** and call listen() function. Then we utilize recognize_google to utilize the online transcription service.

This is enclosed in a function called listen_for_command(). When the program is executed, it starts listening for command but does not enter the command loop until it hears the trigger word. Then it prompts the user to state their reservation. The output is then tested against defined use cases in the code which are mapped to specific outputs such as an IoT system to control appliances which is explained in detail in the next section. There is also featured to send custom messages to the caretaker from the patient by giving audio que for the function. This is also explained in detail in the app development section. The main highlight of this section is the communication with LLM running in the background. When the user utters the phrase “lets chat” or “chat” or similar synonym the function enters the server communication function define as chat_with_assistant() and prompts the user to ask about any topic the patient wants to discuss. The queries of the user are then promptly answered by the language model. The model continues to chat with the user until prompted by the user to exit or go to sleep in which it again goes to the state where it goes to listen for the trigger word. The loop continues in perpetuity.

Reply from server:

We define a respond(text) function which is utilized to give replies from the program to the user based on his/her query. All the audio outputs from the system are given by employing this function.

Context memory:

In chat_with_assistant() function there is a functionality which continuously stores responses by the chat application and also the queries of the user. This allows the chatbot to keep in touch with the context of the conversation and give more accurate responses to the user.

Chat_with_assistant():

This function utilizes the components of the OpenAi library and imports functionality direct from it. We define the working parameters of the function such as model, max tokens, temperature, API key and prompt. It also has the append function to store context as mentioned above.

5.3.2 Implementation of home appliance control:

Setting up the ESP32 involves configuring it to connect to the Wi-Fi network and programming it to read commands from the Firebase database and control the relay module. The relay module is connected to the ESP32 and the appliances, and it is configured to switch devices on or off based on the signals received from the ESP32. Firebase is configured to store device states and commands, with the database structure designed to facilitate easy reading and writing of data by the Python script and ESP32. The Python script runs continuously, listening for commands from the AI chatbot, updating the Firebase database, and sending appropriate signals to the ESP32. The script ensures real-time interaction between user commands and the physical devices, providing seamless home automation experience.

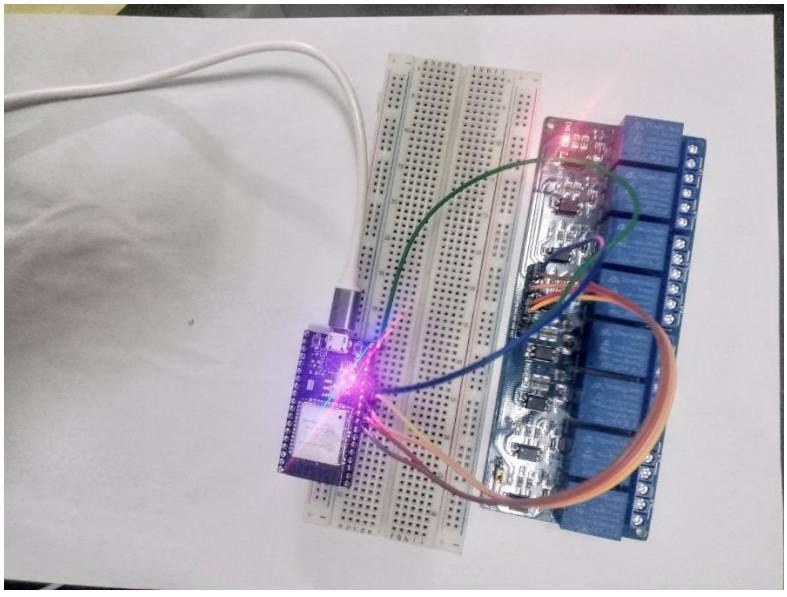

Figure 9 Hardware

Figure 10 ESP32 Setup

5.3.3 Firebase Notification System

AI chatbot for home appliance control aims to provide ease and convenience to users. In this era of modern technology, applications of AI for assistive purposes can bring a revolution in our society. The developed solutions are cost-effective and efficient. One use case is in the assistive health-care categories where AI based applications and hardware systems can be used. Elderly individuals/ patients can use the AI chatbot for home automation control. An extension to this is a notification application. The application is developed as part of an extension of the use-case of this project. The real-time notification application gathers data from firebase real-time database and notifies the health-care assistance about the demand of individual communicating with the chatbot. It also displays the patient's current health status and medications. The application provides a real-time notification interface where the healthcare assistant gets notified about the current feelings of patients along with any needs for scheduling appointments with doctors.

The application is developed using Android Studio. Android studio, the official IDE for Google's Android OS, is a comprehensive tool designed to streamline the app development process. It is a great tool for beginners and experienced developers to get started with application development. The User Interface is user-friendly, clean and intuitive offering features such as syntax highlighting, error and warning notification etc. The UI design is simple due to the presence of drag-drop features and real-time rendering. The IDE also integrates with Firebase, enabling developers to add robust backend services such as authentication, databases, and cloud storage to their apps with ease. Java is the primary language for Android Application development. All this along with the large community support makes it a suitable choice for application development.

5.3.4 Real-Time Notification System

The application is interfaced with Fire-base real-time database. The database contains data of patients. Also, the android application has multiple views/screens. The following is a list of views added.

Home Screen:

A splash screen with the logo of our healthcare assistant “Carebot Companion” app flashes for 3 seconds as soon as the application is launched.

Figure 11 Home Screen

Sign-up Page:

A new user/ healthcare assistant can create an account providing details about name, username, email address and password. Upon this, a new user is added to Firebase real-time database. This page has an option of switching to a Login page if you are already a user.

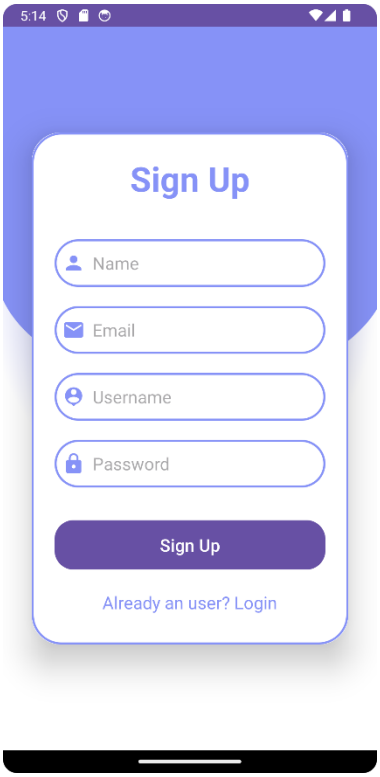

Figure 12 Signup page

Log-in Page:

A user can Log in to his account using log-in credentials including username and password.

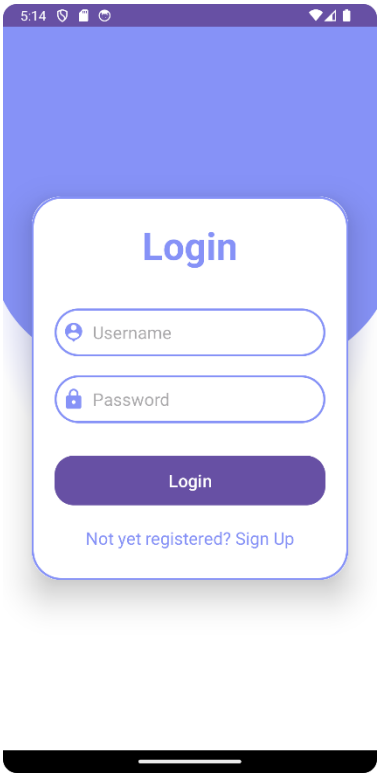

Figure 13 Login Page

User Home Page:

After login , user is directed to its user home page. If the user is new, and no patients have been assigned to him yet, the screen displays a message declaring that particular user has no patients assigned yet. Current use-case have maximum 2 patients. If a particular user has one patient, the user home page will contain details of the patient including name, age, gender, health status and current medication. It also contains a current assistance view that updates as soon as the firebase gets updated. For User with two patients assigned, the home page will have names of the two patients and the user can chose to view notifications of patient by clicking on button specified to view patient's profile.

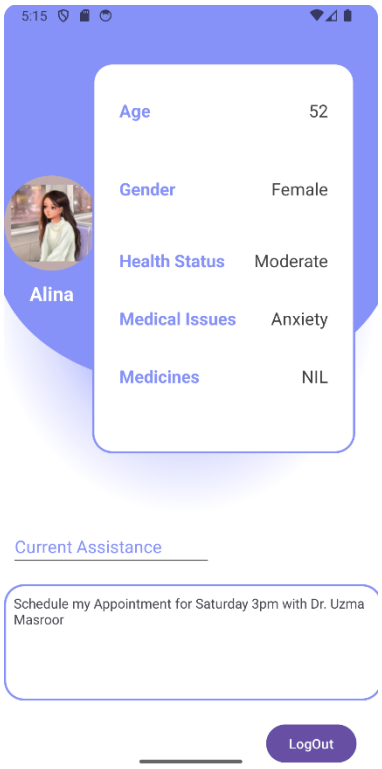

Figure 14 User Interface

5.3.5 Python Application GUI

A graphical user interface is developed using eel. The interface displays if the conversation is active along with the conversation happening between the AI chatbot and the user in a scrollable view. The figure below shows the GUI developed.

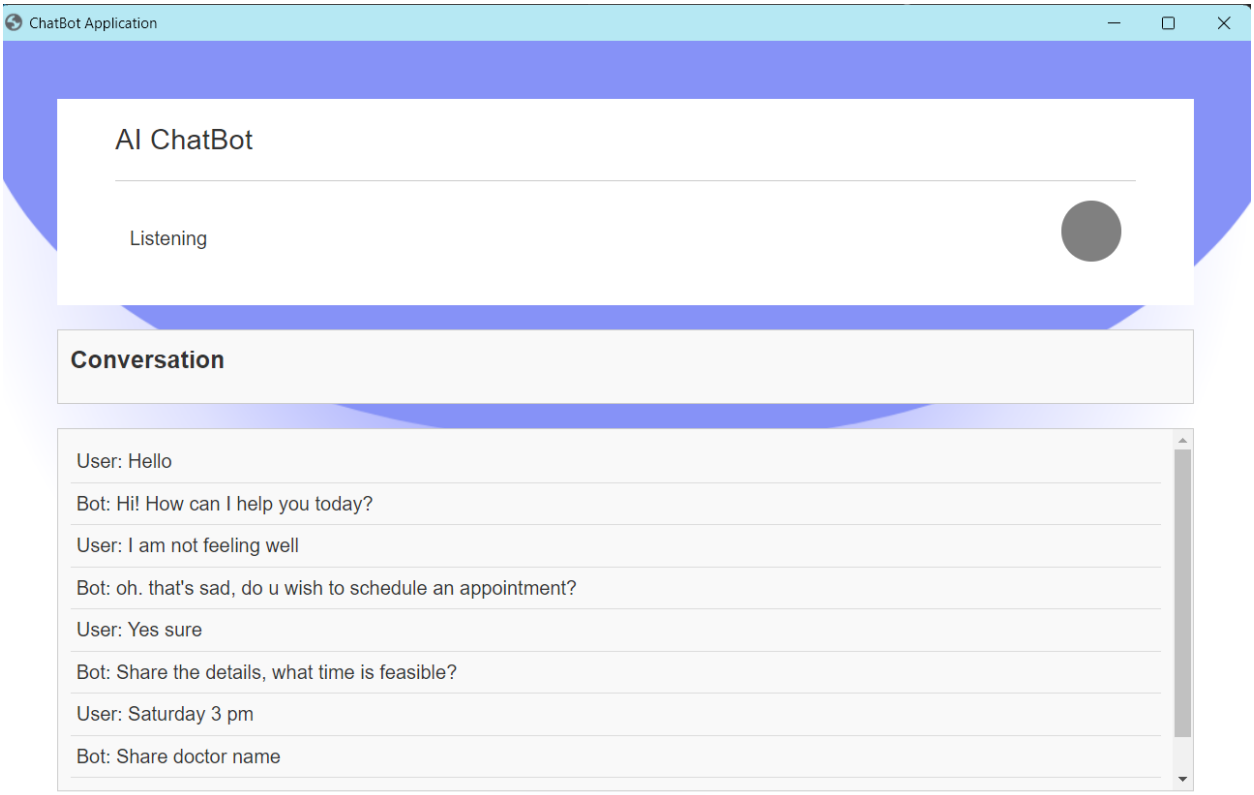

Figure 15 AI ChatBot Application GUI

Chapter 6 - Conclusions and future work

The project has reached its conclusion and is able to do all the said tasks. The project can serve its purpose in dealing with multiple patients and provide monitoring capabilities. The project can be developed on further levels and be more advanced in the following scopes.

Improvements:

- The LLM's output can be improved by training it on bigger datasets of communication between individuals to generate more human like conversation
- The combined system can be brought completely offline by utilizing Bluetooth capabilities of the ESP32 microcontroller and avoid latency issues due to the use to WiFi services.
- The carebot companion can also be trained on medical transcripts and given access to medical data through RAG to give it general diagnostic capabilities.
- The Home appliance control system can be enhanced by providing direct control to device features such as thermostat temperature etc.
- The service can be improved by incorporating system with sensors and providing real time data to the doctors about patient's vitals and in case of emergency where the patient is unable to send a message to the care taker.
- Create features which help patients in their own care such as educational resources, health tips and interactive tools.
- The system can be scaled to allow multiple users with multiple patients while making sure reliability and responsiveness is not affected.
- Provide multilingual support to deal with a diverse population set.
- Integration with active health records of patients with the system to provide comprehensive patient history.
- Develop algorithms to assist in diagnostics providing health care professionals with decision support tools.

References

[1] C. J. Baby, F. A. Khan, and J. N. Swathi, "Home automation using IoT and a chatbot using natural language processing," in *2017 Innovations in Power and Advanced Computing Technologies, i-PACT 2017*, 2017. doi: 10.1109/IPACT.2017.8245185.

[2] N. Chomsky, "Aspects of a theory of syntax," *Cambridge, Mass.: MI'I'. I'rw Language Journal*, vol. 53, 1965.

[3] D. Khurana, A. Koli, K. Khatter, and S. Singh, "Natural language processing: state of the art, current trends and challenges," *Multimed Tools Appl*, vol. 82, no. 3, 2023, doi: 10.1007/s11042-022-13428-4.

[4] R. Lass, "CAMBRIDGE TEXTBOOKS IN LINGUISTICS," 1984.

[5] "Proceedings of the 13th International Conference on Finite State Methods and Natural Language Processing, FSMNLP 2017," *Proceedings of the 13th International Conference on Finite State Methods and Natural Language Processing, FSMNLP 2017*. 2021.

[6] J. Baptista, U. Algarve, G. Fernandes, R. Talhadas, F. Dias, and N. Mamede, "IMPLEMENTING EUROPEAN PORTUGUESE VERBAL IDIOMS IN A NATURAL LANGUAGE PROCESSING SYSTEM."

[7] Y. C. Zhou, Z. Zheng, J. R. Lin, and X. Z. Lu, "Integrating NLP and context-free grammar for complex rule interpretation towards automated compliance checking," *Comput Ind*, vol. 142, 2022, doi: 10.1016/j.compind.2022.103746.

[8] Q. Liu, M. J. Kusner, and P. Blunsom, "A Survey on Contextual Embeddings," Mar. 2020, [Online]. Available: http://arxiv.org/abs/2003.07278

[9] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2019.

[10] Y. Liu *et al.*, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," Jul. 2019, [Online]. Available: http://arxiv.org/abs/1907.11692

[11] I. Froiz-Miguez, P. Fraga-Lamas, and T. M. Fernandez-Carames, "Design, Implementation, and Practical Evaluation of a Voice Recognition Based IoT Home Automation System for Low-Resource Languages and Resource-Constrained Edge IoT Devices: A System for Galician and Mobile Opportunistic Scenarios," *IEEE Access*, vol. 11, 2023, doi: 10.1109/ACCESS.2023.3286391.

[12] W. Sun, J. Liu, and Y. Yue, "AI-enhanced offloading in edge computing: When machine learning meets industrial IoT," *IEEE Netw*, vol. 33, no. 5, 2019, doi: 10.1109/MNET.001.1800510.

[13] A. M. Turing, "Computing Machinery and Intelligence," Oxford University Press on, 1950. [Online]. Available: http://www.jstor.orgStableURL:http://www.jstor.org/stable/2251299Accessed:25/08/200818:56

[14] "Computer-Power-and-Human-Reason".

[15] Y. Yang, "An Innovative Distributed Speech Recognition Platform for Portable, Personalized and Humanized Wireless Devices," 中文計算語言學期刊, vol. 9, no. 2, 2004.

[16] R. Kompella, "Conversational AI chat-bot — Architecture overview," *Medium*, 2018.

[17] J. Hill, W. Randolph Ford, and I. G. Farreras, "Real conversations with artificial intelligence: A comparison between human-human online conversations and human-chatbot conversations," *Comput Human Behav*, vol. 49, 2015, doi: 10.1016/j.chb.2015.02.026.

[18] S. Singh and H. K. Thakur, "Survey of Various AI Chatbots Based on Technology Used," in *ICRITO 2020 - IEEE 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)*, 2020. doi: 10.1109/ICRITO48877.2020.9197943.

[19] "Pattern matching - Wikipedia." Accessed: May 25, 2024. [Online]. Available: https://en.wikipedia.org/wiki/Pattern_matching

[20] "Parsing - Wikipedia." Accessed: May 25, 2024. [Online]. Available: https://en.wikipedia.org/wiki/Parsing

[21] "Deep Learning Chatbots: Everything You Need to Know | HackerNoon." Accessed: May 25, 2024. [Online]. Available: https://hackernoon.com/deep-learning-chatbot-everything-you-need-to-know-r11jm30bc

[22] WildML, "Deep Learning for Chatbots - Part 1," Web.

[23] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of Things (IoT): A Literature Review," *Journal of Computer and Communications*, vol. 03, no. 05, 2015, doi: 10.4236/jcc.2015.35021.

[24] K. Maney and Alison Maney, “Kevin Ashton, Father of the Internet of Things & Network Trailblazer,” Cisco Systems, Inc.
[25] “The Evolution of Wireless Sensor Networks Origin and History of Wireless Sensor Networks.”
[26] H. Jarvinen, A. Litvinov, and P. Vuorimaa, “Integration platform for home and building automation systems,” in *2011 IEEE Consumer Communications and Networking Conference, CCNC’2011*, 2011. doi: 10.1109/CCNC.2011.5766475.
[27] T. Parthornratt, D. Kitsawat, P. Putthapipat, and P. Koronjaruwat, “A Smart Home Automation Via Facebook Chatbot and Raspberry Pi,” in *2018 2nd International Conference on Engineering Innovation, ICEI 2018*, 2018. doi: 10.1109/ICEI18.2018.8448761.
[28] R. Teymourzadeh, S. A. Ahmed, K. W. Chan, and M. V. Hoong, “Smart GSM based home automation system,” in *Proceedings - 2013 IEEE Conference on Systems, Process and Control, ICSPC 2013*, 2013. doi: 10.1109/SPC.2013.6735152.
[29] P. Singh, K. Chotalia, S. Pingale, and S. Kadam, “A review paper on smart GSM based home automation system,” *International Research Journal of Engineering and Technology (IRJET)*, vol. 3, no. 04, 2016.
[30] S. A. I. Quadri and P. Sathish, “IoT based home automation and surveillance system,” in *Proceedings of the 2017 International Conference on Intelligent Computing and Control Systems, ICICCS 2017*, 2017. doi: 10.1109/ICCONS.2017.8250586.
[31] F. D. Jivani, M. Malvankar, and R. Shankarmani, “A Voice Controlled Smart Home Solution with a Centralized Management Framework Implemented Using AI and NLP,” in *Proceedings of the 2018 International Conference on Current Trends towards Converging Technologies, ICCTCT 2018*, 2018. doi: 10.1109/ICCTCT.2018.8550972.
[32] United Nations, “World Population Ageing 2020: Highlights (2020),” 2020.
[33] A. Professor, G. Reddi Priya Student, R. Vasanthi Student, and B. Venkatesh Student, “IOT Based Smart Security and Smart Home Automation 1 Sudha Kousalya.” [Online]. Available: www.ijert.org
[34] S. Sahoo, S. Maity, P. Parida, and M. Samal, “IOT BASED HOME AUTOMATION,” 2022.
[35] R. Alekya, N. D. Boddeti, K. Salomi Monica, R. Prabha, and V. Venkatesh, “IoT based Smart Healthcare Monitoring Systems: A Literature Review,” 2020.
[36] P. Franco, J. M. Martinez, Y. C. Kim, and M. A. Ahmed, “A framework for iot based appliance recognition in smart homes,” *IEEE Access*, vol. 9, 2021, doi: 10.1109/ACCESS.2021.3116148.
[37] L. C. Schünke *et al.*, “A rapid review of machine learning approaches for telemedicine in the scope of COVID-19,” *Artificial Intelligence in Medicine*, vol. 129. 2022. doi: 10.1016/j.artmed.2022.102312.
[38] N. Chumuang *et al.*, “Development a home electrical equipment control device via voice commands for elderly assistance,” in *Proceedings - 2020 15th International Joint Symposium on Artificial Intelligence and Natural Language Processing, iSAI-NLP 2020*, 2020. doi: 10.1109/iSAI-NLP51646.2020.9376818.
[39] M. Vacher, B. Lecouteux, and F. Portet, “Multichannel automatic recognition of voice command in a multi-room smart home : An experiment involving seniors and users with visual impairment,” in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2014. doi: 10.21437/interspeech.2014-264.
[40] C. Chen *et al.*, “Understanding Barriers and Design Opportunities to Improve Healthcare and QOL for Older Adults through Voice Assistants,” in *ASSETS 2021 - 23rd International ACM SIGACCESS Conference on Computers and Accessibility*, 2021. doi: 10.1145/3441852.3471218.
[41] A. Pradhan, A. Lazar, and L. Findlater, “Use of intelligent voice assistants by older adults with low technology use,” *ACM Transactions on Computer-Human Interaction*, vol. 27, no. 4, 2020, doi: 10.1145/3373759.
[42] S. Ashwini, N. R. Rajalakshmi, P. Victer Paul, and L. Jayakumar, “Dynamic NLP Enabled Chatbot for Rural Health Care in India,” in *2022 2nd International Conference on Computer Science, Engineering and Applications, ICCSEA 2022*, 2022. doi: 10.1109/ICCSEA54677.2022.9936389.
[43] N. Mani, A. Singh, and S. L. Nimmagadda, “An IoT Guided Healthcare Monitoring System for Managing Real-Time Notifications by Fog Computing Services,” in *Procedia Computer Science*, 2020. doi: 10.1016/j.procs.2020.03.424.
[44] S. Latif, J. Qadir, A. Qayyum, M. Usama, and S. Younis, “Speech Technology for Healthcare: Opportunities, Challenges, and State of the Art,” *IEEE Rev Biomed Eng*, vol. 14, 2021, doi: 10.1109/RBME.2020.3006860.
[45] P. Suter, W. N. Suter, and D. Johnston, “Theory-based telehealth and patient empowerment,” *Popul Health Manag*, vol. 14, no. 2, 2011, doi: 10.1089/pop.2010.0013.

[46] S. U. Amin and M. S. Hossain, “Edge Intelligence and Internet of Things in Healthcare: A Survey,” *IEEE Access*, vol. 9, 2021, doi: 10.1109/ACCESS.2020.3045115.
[47] A. Vaswani *et al.*, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, 2017.
[48] Y. Gao *et al.*, “Retrieval-Augmented Generation for Large Language Models: A Survey”, Accessed: May 25, 2024. [Online]. Available: https://github.com/Tongji-KGLLM/
[49] “GitHub - alphacep/vosk-api: Offline speech recognition API for Android, iOS, Raspberry Pi and servers with Python, Java, C# and Node.” Accessed: May 25, 2024. [Online]. Available: https://github.com/alphacep/vosk-api
[50] “What Is LangChain? | IBM.” Accessed: May 25, 2024. [Online]. Available: https://www.ibm.com/topics/langchain

-

Annexture

```
import firebase_admin
from firebase_admin import credentials, db
import speech_recognition as sr
import sys
import pyttsx3
import time
import pyautogui
import webbrowser
from openai import OpenAI

cred = credentials.Certificate("firepy.json")

firebase_admin.initialize_app(cred, {
   'databaseURL': 'https://firepy-baeed-default-rtdb.asia-southeast1.firebasedatabase.app/'
})

ref = db.reference('test')

assistant_name = "Atif"
listening_for_trigger_word = True
should_run = True

source = sr.Microphone()
recognizer = sr.Recognizer()

if sys.platform != 'darwin':
   engine = pyttsx3.init()

tasks = []
listeningToTask = False
askingAQuestion = False

def listen_for_command():
   with source as s:
      print("Listening for command...")
      recognizer.adjust_for_ambient_noise(s)
      audio = recognizer.listen(s)

   try:
      command = recognizer.recognize_google(audio, language="en-in")
      print(f"User said: {command}")
      return command.lower()
   except sr.UnknownValueError:
      print("Could not understand audio. Please try again.")
      return None
   except sr.RequestError:
      print("Unable to access the Google Speech Recognition API.")
      return None

def respond(text):
   if sys.platform == 'darwin':
      ALLOWED_CHARS =
set("abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789.,?!-_$:+-/ ")
```

```
        clean_text = ''.join(c for c in text if c in ALLOWED_CHARS)
        system(f"say '{clean_text}'")
    else:
        engine.say(text)
        engine.runAndWait()

def chat_with_assistant():
    global should_run

    client = OpenAI(base_url="http://localhost:1234/v1", api_key="lm-studio")

    history = [
        {"role": "system", "content": "You are a companion chatbot. You will make general conversation and answer
any queries the user has. Start by my name is carebot,What would you like to talk about "},
        {"role": "user", "content": "Hello, introduce yourself to someone opening this program for the first time. Be
concise."},
    ]

    while True:
        completion = client.chat.completions.create(
            model="local-model",
            messages=history,
            temperature=0.5,
            max_tokens=50,
            stream=True,
        )

        new_message = {"role": "assistant", "content": ""}
        response_text = ""

        for chunk in completion:
            if chunk.choices[0].delta.content:
                response_text += chunk.choices[0].delta.content

        print(response_text, end="", flush=True)
        respond(response_text)

        print()
        history.append({"role": "assistant", "content": response_text})

        user_input = listen_for_command()
        if user_input is None:
            respond("Sorry, I couldn't understand you. Please try again.")
            continue

        if user_input.lower() == 'exit':
            print("Goodbye!")
            return False
        history.append({"role": "user", "content": user_input})
    return True

def add_task(command):
    global tasks, listeningToTask
    tasks.append(command)
    listeningToTask = False
    respond("Adding " + command + " to your task list. You have " + str(len(tasks)) + " currently in your list.")
```

```
def list_tasks():
    respond("Sure. Your tasks are:")
    for task in tasks:
        respond(task)

def take_screenshot():
    pyautogui.screenshot("screenshot.png")
    respond("I took a screenshot for you.")

def open_chrome():
    respond("Opening Chrome.")
    webbrowser.open("http://www.youtube.com/@JakeEh")

def control_led(ref, led, state):
    ref.update({f'bool{led}': 0 if state == 'on' else 1})

def get_count():
    ref = db.reference('test/int')
    print(ref.get())

def stop_command_processing():
    global should_run
    respond("Stopping command processing.")
    should_run = False

def update_firebase_boolean(ref, device, status):
    ref.update({device: status})

def process_device_command(command, ref):
    device_map = {
        "window": "window",
        "light": "light",
        "ac": "ac",
        "fan": "fan",

    }

    device = None
    status = None

    for key in device_map:
        if key in command:
            device = device_map[key]
            break

    if "turn on" in command or "on" in command:
        status = 1
    elif "turn off" in command or "off" in command:
        status = 0

    if device is not None and status is not None:
        update_firebase_boolean(ref, device, status)
        respond(f"Command processed: Turn {'on' if status == 0 else 'off'} {device}")
    else:
        respond("Sorry, I couldn't process the command.")
```

```
command_functions = {
    "add a task": add_task,
    "new task": add_task,
    "list tasks": list_tasks,
    "show tasks": list_tasks,
    "take a screenshot": take_screenshot,
    "capture screen": take_screenshot,
    "open chrome": open_chrome,
    "launch chrome": open_chrome,
    "stop": stop_command_processing,
    "end": stop_command_processing,
    "chat": chat_with_assistant,
    "let's chat": chat_with_assistant,
    "talk": chat_with_assistant,
}

def perform_command(command, ref):
    global tasks
    global listeningToTask
    global askingAQuestion
    global should_run

    while should_run:
        if command:
            print("Command: ", command)
            if listeningToTask:
                add_task(command)
            else:
                if any(key in command for key in ["window", "light", "ac", "fan", "led", "led 1", "led 2", "led 3"]):
                    process_device_command(command, ref)
                else:
                    for key, func in command_functions.items():
                        if key in command:
                            func()
                            break
                    else:
                        respond("Sorry, I'm not sure how to handle that command.")
        command = listen_for_command()

def main():
    global listening_for_trigger_word
    global assistant_name
    global ref

    while True:
        command = listen_for_command()
        if command and f"hey {assistant_name.lower()}" in command:
            listening_for_trigger_word = False
            respond("Yes, how can I assist you?")
            break

    while should_run:
        command = listen_for_command()
        if listening_for_trigger_word:
            listening_for_trigger_word = False
```

```
        else:
            perform_command(command, ref)
        time.sleep(1)
    respond("Goodbye.")

if __name__ == "__main__":
    main()
```