A.B.M.S.P's

Anantrao Pawar College of Engineering & Research, Pune-09

CERTIFICATE

This is to certify that the report entitled

"CHAT GPT VOICE ASSISTANT"

Has been satisfactorily completed by,

Prabhanjan Pravin Bongarde [B191103005]

Krunal Umesh Kosumbkar [B191103018]

Sudhakar Naval Dhangar [B191103010]

Towards fulfillment of Bachelor Degree of Electronics and Telecommunication Engineering In Savitribai Phule Pune University for academic year 2023-24.

Prof. Sharad Jagtap
(Guide)

Anantrao Pawar College of Engineering & Research Pune

Dr. Amar Deshmukh
(HOD)

27-05-24.

Dr. S.B. Thakare
(Principal)

ABSTRACT

Conversational AI has become a key technology in the modern digital age, revolutionizing human-machine interaction. With the use of the ESP32 microcontroller and ChatGPT, an advanced conversational AI model created by OpenAI, this project aims to create a dynamic voice assistant. The proposed system aims to transform user interaction in smart environments by providing voice-activated, user-friendly access to data and services. With its strong connectivity and versatility, the ESP32 microcontroller is the voice assistant's physical core. By utilizing its Wi-Fi capabilities, the ESP32 enhances the assistant's functionality beyond local processing by enabling smooth contact with web platforms. The voice assistant gains the capacity to understand and react to user inquiries in conversational English by incorporating ChatGPT, a state-of-the-art natural language processing and generation model. Configuring the ESP32 for audio input and output, putting advanced speech recognition algorithms into practice to record user commands, integrating ChatGPT for natural language processing capabilities, and creating an easy-to-use interface are all crucial steps in the development process. Moreover, during the development process, strict considerations are made about resource optimization, privacy, and security. The voice assistant that is being suggested has a wide range of features, delivering individualized information, the most recent news headlines, and weather predictions in real time. Controlling a variety of smart home appliances, such as home theater systems, lights, and thermostats, with ease. Assisting with task management by arranging calendar events, alerts, and reminders according to user preferences. Providing thorough web search functionality and quickly obtaining relevant information.

Keywords—*Conversational AI, Privacy, Security, resource optimization, Natural language processing.*

Chapter 6

6.1 Circuit Diagram

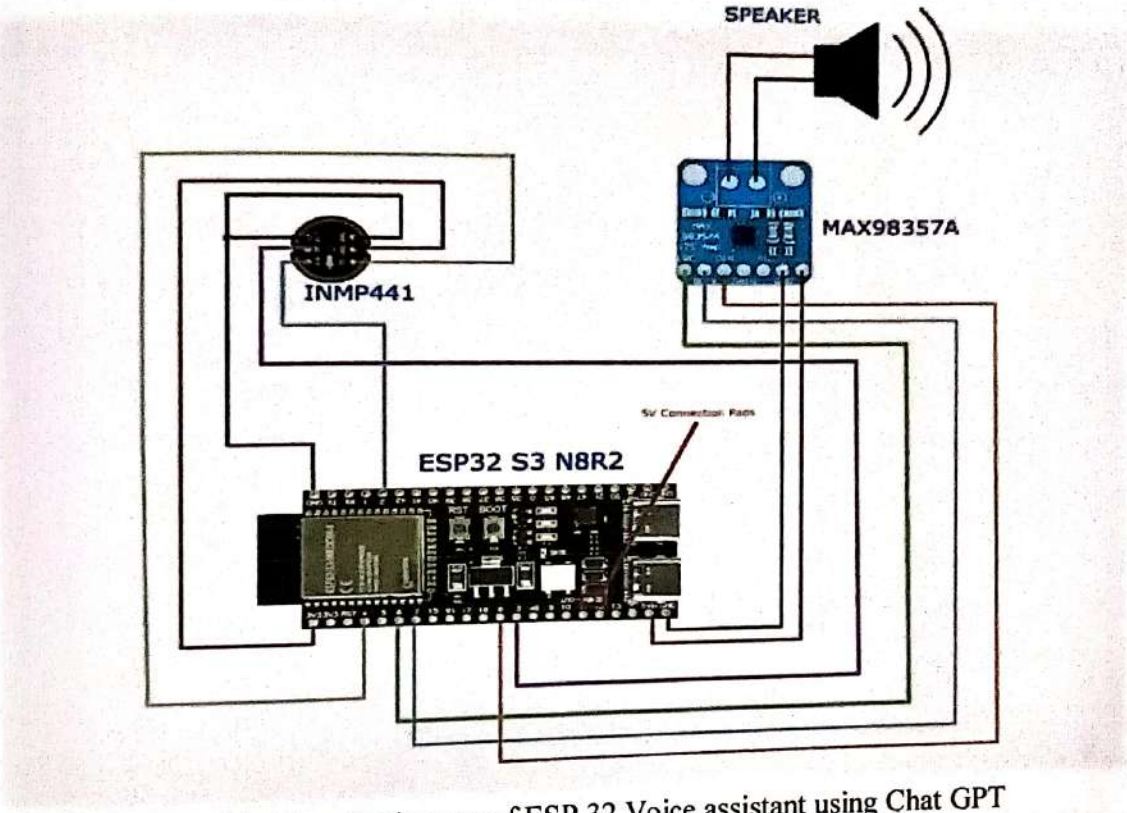

Figure 6.1: Circuit Diagram of ESP 32 Voice assistant using Chat GPT

Table 6.1: Whisper pin mapping with respect to ESP32 S3 GPIO'S

ESP 32 S3 N8R8	INMP441 Microphone	Speaker
Vin (5V)		Vin
GPIO 6		LRC
GPIO 7		BCLK
GPIO 8		Din
GPIO 4	SD	
GPIO 3	WS	
GPIO 2	SCK	
3V3	VDD	
GND	GND L&R	

6.2 PCB Design

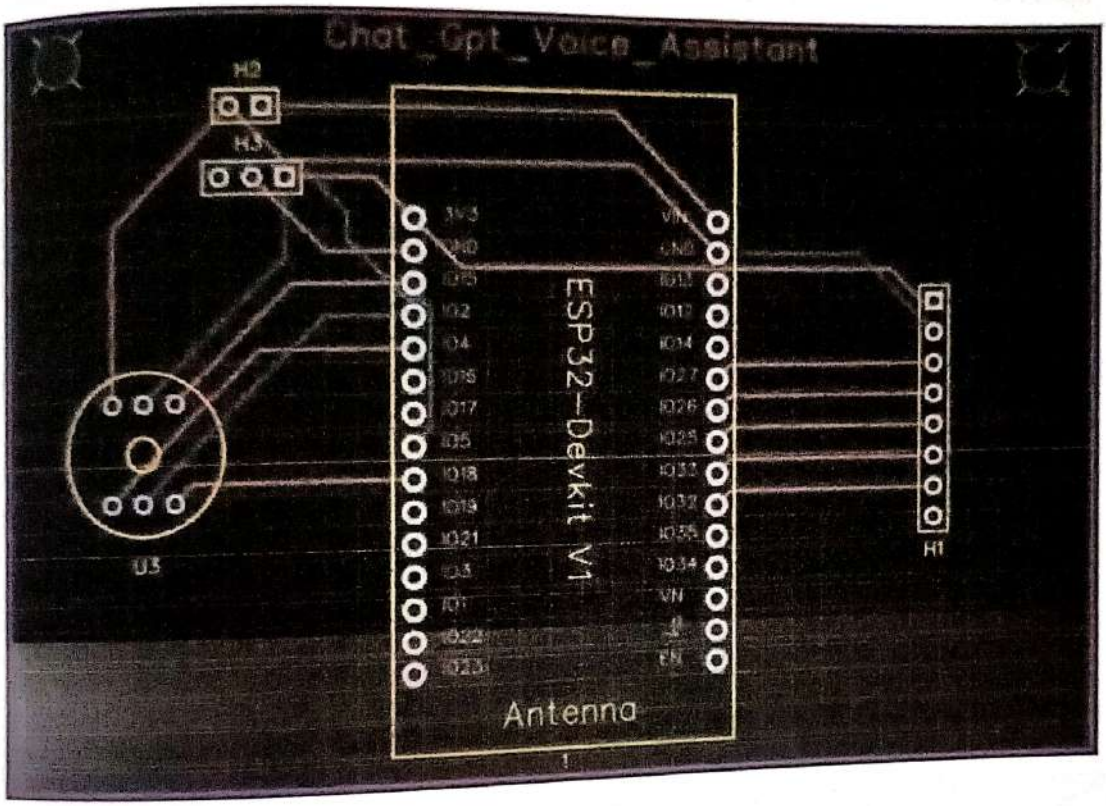

Figure 6.2: PCB Design

A Single Layered PCB is made for the circuit connections mentioned above.

Chapter 7

Working

The Assist pipelines in Home Assistant are composed of different parts that come together to generate a voice assistant. Various alternatives to select from for each component. Both text-to-speech and speech-to-text are fully local options.

7.1 Whisper

Whisper is the speech-to-text Model by OpenAI model with multilingual support and it is Open-source. We make use of faster-whisper, a forked version. It takes about 8 seconds for a Raspberry Pi 4 to process incoming voice commands. It takes less than a second to complete on an Intel NUC. Several speech processing tasks, such as multilingual speech recognition, speech translation, spoken language identification, and voice activity detection, are used to train a Transformer sequence-to-sequence model. Several steps of a conventional speech-processing pipeline can be replaced by a single model since these tasks are collectively represented as a series of tokens that the decoder must anticipate. A collection of unique tokens that function as task specifiers or classification goals are used in the multitask training format. Each of the five model sizes comes in four English-only versions that compromise accuracy for speed. Below is a list of the accessible models along with an estimate of how much memory they require and how quickly they infer from the large model. Numerous factors, such as the hardware that is available, could affect the actual speed

Table 7.1 Whisper processing models

Parameters	Multilingual model	Required VRAM	Relative Speed
39 M	tiny	~1 Gb	~32x
74 M	base	~1 GB	~16x
244 M	small	~2 GB	~6x
769 M	medium	~5GB	~2x
1550 M	large	~10 GB	~1x

7.2 Piper

Piper is our text-to-speech system. Optimized for the Raspberry Pi 4, Piper is a fast, local neural text-to-speech system with a pleasing voice. Numerous languages are supported. When utilizing models of moderate quality on a Raspberry Pi, it can

generate 1.6s of voice in a second.

The following system is used for naming choices: "name"-"quality>-<language>_<REGION> The <name> section is derived from either the speaker's name, if given, or the dataset used to train the voice.
Four levels of quality exist for a voice:

i. x_low - 16Khz, smallest/fastest
ii. low - 16Khz,
iii. fast medium - 22.05Khz, slower but better sounding
iv. high - 22.05Khz, slowest but best sounding

Only the medium models will operate at a tolerable pace on a Raspberry Pi 4. The low or x-low voices are preferable if audio quality is not important because they will sound much faster than medium.

7.3 Wyoming Protocol

An interprocess event protocol over stdin/stdout for Rhasspy v3. The command line interfaces of many voice programs are comparable. For instance, the majority of text-to-speech applications receive text via standard input and output a WAV file or a file to standard output. example: echo "Input text" | text-to-speech > output.wav Standard input/output protocols would work with any language, operating system, etc. But certain voice applications require the production or consumption of audio or event streams. For instance, if a speech-to-text system can process audio while it's being captured, it might provide a result considerably more quickly.

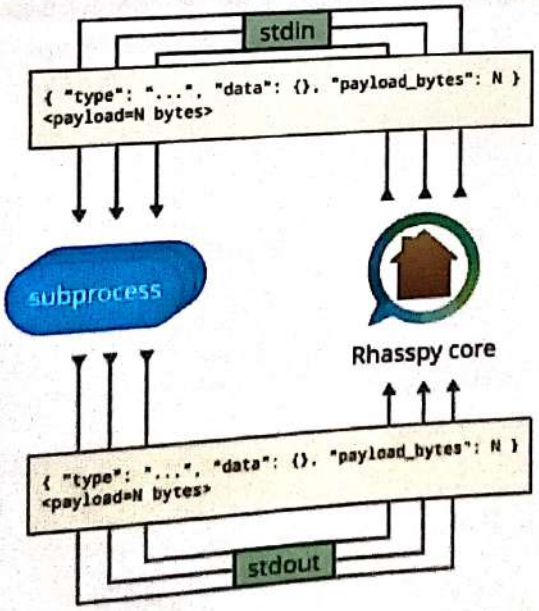

Fig 7.1: Wyoming Protocol For Voice Assistant

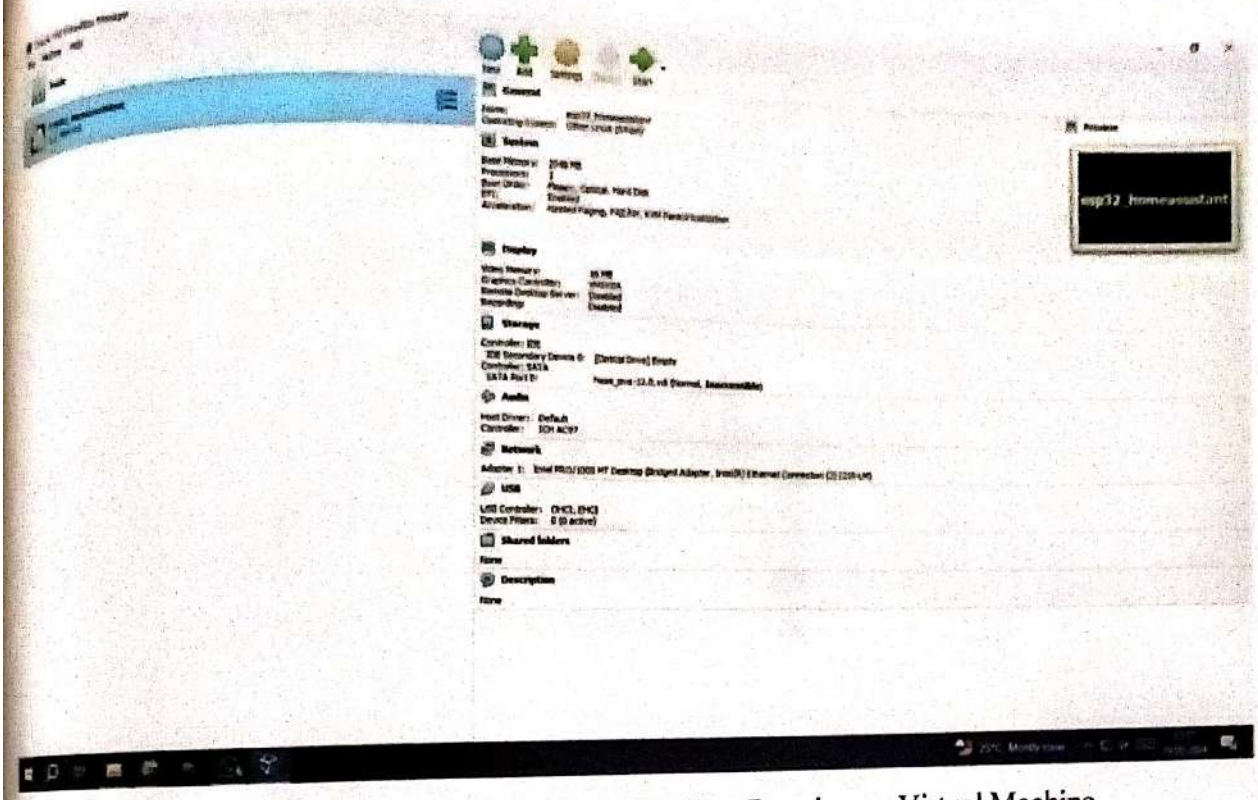

Fig 7.2: Home Assistant OS installed on Linux Running on Virtual Machine.

B.Configuration of Voice assistant Pipeline in Home Assistant OS Using Addons:

a. The Voice assistant pipeline contains Speech to Text , Text to Speech ,Open wake word, Conversation agent.

b. All these modules together form an voice assistant pipeline.

c. All these modules need to be configured via add ons .

d. Whisper is speech to text addon.

e. Piper is text to speech addon.

f. Chat GPT API Key is Pasted in the OpenConverstion Addon to use Chat GPT as voice assistant conversation agent.

C. Installing ESP Home addon to Connect the ESP Firmware To the Home Assistant OS:

a. The ESP home is a platform to connect ESP 32 to Home assistant OS.

b. The ESP home is used to flash the ESP via home assistant and maintaining connection with it.

c. The ESP home connects the home assistant to ESP32 via web or internet through ESP home Web .

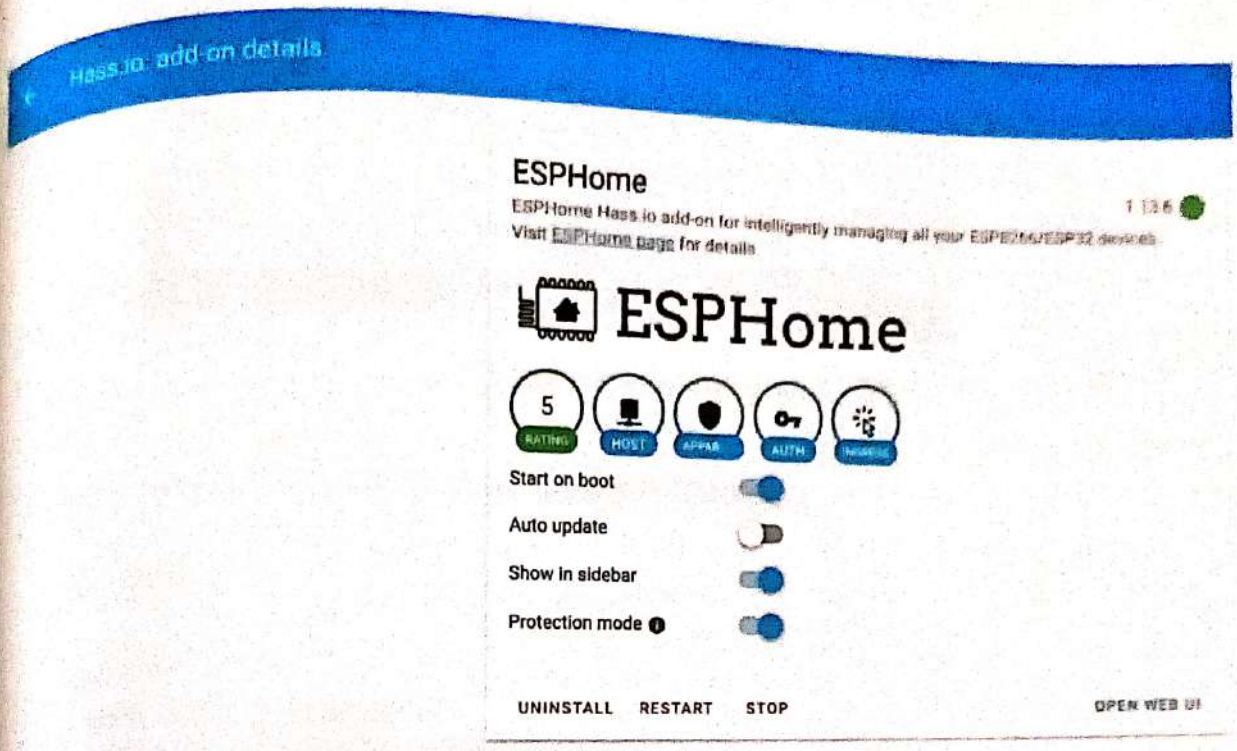

Fig 7.3: ESP Home Add on in home assistsant.

7.5 Flashing ESP 32 S3 N8R8 with ESP Home:

Step 1: Create a new device in the ESPHome dashboard by clicking on "New Device" in the bottom right-hand corner.

Step 2: Give it a name e.g. "Voice Assistant" and click "Next

Step 3: Now select ESP32 S3 from the options

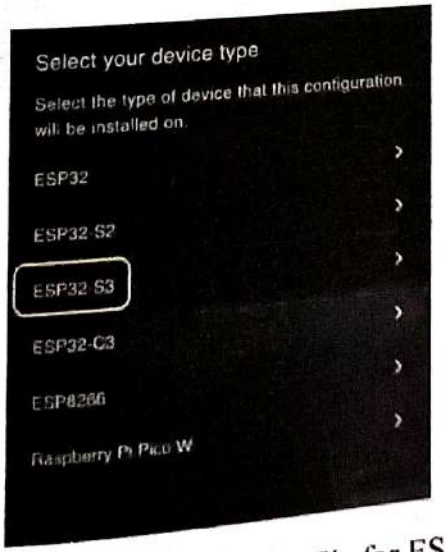

Fig 7.4: Preparing config.yaml file for ESP 32 S3

Step 4: Now, click on "Skip" and open the device card again by clicking on the "Edit" Option

Fig 7.5: Device Configuration Prepared using ESP Home

Step 5: Write Yaml code for Voice assistant configuration:

Yaml Code:

```
esphome:
 name: esp32-s3-wake-word
 friendly_name: ESP32-S3-Wake-word
 platformio_options:
  board_build.flash_mode: dio
 on_boot:
  - light.turn_on:
     id: led_ww
     blue: 100%
     brightness: 60%
     effect: fast pulse

esp32:
 board: esp32-s3-devkitc-1
 framework:
  type: esp-idf

  sdkconfig_options:
   CONFIG_ESP32S3_DEFAULT_CPU_FREQ_240: "y"
   CONFIG_ESP32S3_DATA_CACHE_64KB: "y"
   CONFIG_ESP32S3_DATA_CACHE_LINE_64B: "y"
   CONFIG_AUDIO_BOARD_CUSTOM: "y"

psram:
 mode: quad  # quad for N8R2 and octal for N16R8
 speed: 80MHz

# Enable logging
logger:
```