Project Report

*on*

# “Virtual Assistant Jarvis”

Submitted by

# Anish Bochare Ansh Ghodke Shruti Ghate Arya Deshpande

Under the guidance of Guide name

# Mr. Yogesh Gaikwad

***In partial fulfilment of***

# Diploma in Information Technology

[2022-2023]

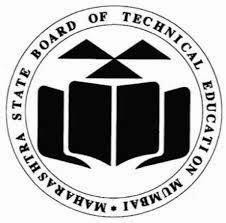
At



DEPARTMENT OF INFORMATION TECHNOLOGY MIT POLYTECHNIC

# PUNE-411038

Affiliated to



MSBTE



# ACKNOWLEDGEMENT

I am profoundly grateful to **Prof. Yogesh Gaikwad** of Information Technology Department for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. I would like to express my deepestappreciation towards **Dr. Prof. R. S. Kale (** Principal), **Prof. Mrunal Fatangare (** HOD of Information Technology Department) MAEER’S MIT POLYTECHNIC, Pune,Whose invaluable guidance supportedme in completing this project.

At last I must express my sincere heartfelt gratitude to all the staff members of Information Technology Department who helped me directly or indirectly during this course of work.

Mr. Anish Bochare Mr. Ansh Ghodke Ms. Shruti Ghate Ms. Arya Deshpande

DIPLOMA IN INFORMATION TECHNOLOGY

# ABSTRACT

The Jarvis Python project aims to create a virtual assistant that can perform various tasks through voice commands. The assistant will use speech recognition and natural language processing to understand user requests and respond with appropriate actions. The project will involve integrating various APIs, such as weather and news, to provide real-time information to the user. The assistant will also be capable of setting reminders, managing to-do lists. The ultimate goal of the Jarvis Python project is to create a reliable and efficient virtual assistant that can simplify daily tasks and enhance the user experience. The project will involve integrating third-party APIs and services to enable the assistant to perform a wide range of tasks, such as playing music, setting alarms, and controlling smart home devices. The assistant will also be customizable, allowing users to add new features and functionality based on their specific needs. The ultimate goal of the Jarvis Python project is to create a powerful and versatile virtual assistant that can improve productivity and enhance the user experience.

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
|  | **CONTENTS** | **PAGE NO.** |
| **1** | **INTRODUCTION OF PROJECT………………………………...** | 1-2 |
| 1.1 Motivation……………………………………………………….. | 1 |
| 1.2 Background …………………………………………………… | 1 |
| 1.3 Need of The Project……………………………………………. | 1 |
| 1.4 Introduction of Project…………………………………………. | 2 |
| **2** | **LITERATURE SURVEY………………………………………….** | 3 |
| **3** | **SCOPE OF PROJECT…………………………………………….** | 4 |
|  | 3.1 Project objectives…………………………………………… | 4 |
| 3.2 Project Features ………………………………………………. | 4 |
| 3.3 Project Cost (Overall Cost) ……………………………………. | 4 |
| **4** | **PROPOSED METHODOLOGY…………………………………** | 5-8 |
|  | 4.1 System Architecture…………………………………………….. | 5-6 |
|  | 4.2 Detailed Description of Modules…………………………………… | 7-8 |
| **5** | **DETAILS DESIGN AND WORKING PROCESSES……………** | 9-36 |
|  | 5.1 Hardware Software  Requirements,…………………………………. | 9 |
|  | 5.2 DFD’s (0,1,2) level ……………………………………………… | 9-10 |
|  | 5.3 Use Case Diagrams ……………………………………………… | 11 |
|  | 5.4 Activity diagram…………………………………………………. | 12 |
|  | 5.5 Sequence Diagram  ……………………………………………. | 13 |
|  | 5.6 Timeline Chart………………………………………………… | 14 |

|  |  |  |
| --- | --- | --- |
|  | 5.7 Implementation ………………………………………………… | 15 |
|  | 5.7.1 Sample Code…………………………………………………  5.7.2 Types of Testing…………………………………………………  5.7.3 Test case………………………………………………………. | 15-32  33-34  35-36 |
| **6** | **RESULTS AND APPLICATION …………………………………** | 37-41 |
| 6.1 Outputs ………………………………………………………. | 37-40 |
| 6.2 Results …………………………………………………………… | 37-40 |
| **7**  **8.** | 6.3 Application………………………………………………………. | 41 |
| **Conclusion and Future Scope…………………………………………….** | 42-43 |
| **References and**  **Bibliography……………………………………………..** | 44 |
| **Paper**  **Published…………………………………………………………..** | 45-52 |

|  |  |  |
| --- | --- | --- |
| **TABLE NO.** | **TITLE** | **PAGE NO.** |
| Table No. 5.1.1 | HARDWARE AND SOFTWARE REQUIREMENT | 9 |
| Table No. 5.7.3.1 | TEST CASE | 35-36 |

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **TITLE** | **PAGE NO.** |
| Fig. 4.1.1 | SYSTEM ARCHITECTURE | 5 |
| Fig. 5.2.1 | DATA FLOW DIAGRAM (DFD LEVEL 0) | 9 |
| Fig. 5.2.2 | DATA FLOW DIAGRAM (DFD LEVEL 1) | 10 |
| Fig. 5.3.1 | USE – CASE DIAGRAM | 11 |
| Fig. 5.4.1 | ACTIVITY DIAGRAM | 12 |
| Fig. 5.5.1 | SEQUENCE DIAGRAM | 13 |
| Fig. 5.6.1 | TIMELINE CHART (GANTT CHART) | 14 |
| Fig. 6.1.1 | INTRODUCTION PAGE | 37 |
| Fig. 6.1.2 | SPEED TEST PAGE | 37 |
| Fig. 6.1.3 | WEATHER UPDATE PAGE | 38 |
| Fig. 6.1.4 | MESSAGE PAGE | 38 |
| Fig. 6.1.5 | NEWS UPDATE PAGE | 39 |
| Fig. 6.1.6 | TIME UPDATE PAGE | 39 |
| Fig 6.1.7 | FACE TRAINING PAGE | 40 |
| Fig 6.1.8 | GREETING AFTER FACE RECOGNITION | 40 |

### MOTIVATION

The actual motivation we got for developing such an application was a voice assistant makes our life easier, saves out time & Jarvis helps user with functions that are the backbone of our day. Someone who talks with you when you are alone or feel lonely someone who manages your daily routines take care of your mental & physical health.

### BACKGROUND

The development of voice assistants has been driven by a desire to create more intuitive and efficient human-computer interfaces. Traditional interactions with computers, such as using a keyboard and mouse, can be time-consuming and require some technical skill. Voice assistants, on the other hand, allow users to interact with computers using natural language, making it easier for people of all ages and technical backgrounds to utilize technology.

Voice assistants have also been fueled by advancements in Artificial Intelligence (AI) and Natural Language Processing (NLP). With these technologies, machines are able to not only recognize speech, but also understand its meaning, context, and intent. This has allowed voice assistants to perform complex tasks and provide personalized responses based on the user's habits, preferences, and past behavior.

Finally, voice assistants have become increasingly popular due to the growing prevalence of smart home devices, which can be controlled via voice commands. As the Internet of Things (IoT) continues to expand, voice assistants will likely play an increasingly important role in managing and controlling these connected devices.

### NEED OF THE PROJECT

There are many reasons why someone might need a virtual assistant. Some reasons include:

* If you are a busy person with a lot of responsibilities, a virtual assistant can help take some tasks off your hands so that you can focus on more important things.
* If you have a disability or illness that makes it difficult to do certain tasks, a virtual assistant can help by doing those tasks for you.
* If you are a student or working professional who needs help with time management and organization, a virtual assistant can be a great asset.
* If you are starting a business or working on a project that requires a lot of research, a virtual assistant can do that research for you.
* There are many reasons why we need to develop virtual assistant. First, virtual assistant can provide us with great convenience. With virtual assistant, we can get things done more quickly and efficiently. For example, if we need to book a hotel room, we can simply ask our virtual assistant to do it for us. Second, virtual assistant can help us save time. By using virtual assistant, we can avoid wasting time on tasks that can be easily done by computers.

### INTRODUCTION OF PROJECT

Jarvis is a fictional character in the Marvel Cinematic Universe, who is the artificial intelligence system created by Tony Stark, also known as Iron Man. Jarvis serves as a loyal assistant and friend to Tony Stark, providing him with information, managing his technology and helping him in various tasks. Jarvis is an acronym for "Just A Rather Very Intelligent System", Although Jarvis is a fictional character, the concept of artificial intelligence systems like him is becoming increasingly prevalent in our daily lives. Virtual assistants such as Siri, Alexa, and Google Assistant, are examples of AI systems that can assist with tasks and provide information. JARVIS, standing for "Just A Really Very Intelligent System," is an acronym. Jarvis is a popular virtual assistant built using Python programming language. It allows users to interact with their computer using voice commands and perform various tasks such as opening applications, browsing the web, sending emails, and more. There are several libraries available in Python that can be used to build Jarvis you need to have a basic understanding of Python and its libraries. Once you have set up the required libraries, you can start building the assistant by defining functions to perform specific actions and integrating them with the speech recognition and text-to-speech conversion modules.

Overall, Jarvis is a fun project that allows you to explore the capabilities of AI and voice recognition technology.

### 2. LITERATURE SURVEY

**Paper 1: Artificial Intelligence-Based Voice Assistant**

Our first paper for literature survey is “Artificial Intelligence-based Voice Assistant” by Subhash, Prajwal N Srivatsa, S Siddesh, A Ullas, B Santhosh. We have added a lot to our project in comparison to other assistance. The fact that it is a hands-free application makes it quite useful in daily life. It's a remarkably simple application. It is also used in business settings, such as laboratories where employees must type while wearing body suits and gloves for protection. Yet, they may access any information by using a voice assistant, which makes their job easier.

### Paper 2: Short Research on Voice Control System Based on Artificial Intelligence Assistant

Our second paper for literature survey is “Short Research on Voice Control System Based on Artificial Intelligence Assistant” by Tae-Kook Kim. This paper describes the design of a voice control system that makes use of embedded devices and open API AIs. The suggested system is a voice command system that uses a user's spoken instructions to activate modules Using the useful speech recognition interface and many modules, this system enables users to construct their own system.

### Paper 3: Next-generation of virtual personal assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home

Our third paper for literature survey is “Next-generation of virtual personal assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home” by Veton Këpuska, Gamal Bohouta. This proposal introduces the structure of the Next-Generation of Virtual Personal Assistants, which is a new VPAs system with a coherent structure designed to converse with a human. In both the input and output channels, this VPAs system communicated using speech, graphics, video, etc. In addition, the VPAs system will be used to improve user-computer interaction by utilising technologies such as image/video recognition, speech recognition, and the Knowledge Base.

### PROJECT OBJECTIVES

The main objective of our project is that it is a software that carries out everyday tasks via voice command. It brings AI and machine learning together to recognize our voice and do what we ask it. Voice assistant software can be found on smart speakers, smartwatches, mobile phones, tablets and other devices.

### PROJECT FEATURES

* + - Can make call to anyone
    - Gives Data About Climate, Temperature, wind speed
    - Use Send WhatsApp Messages
    - Can give News Update
    - Can play any songs we want
    - Can check your Internet speed
    - Control Phone(Call, Answer, Unlock)
    - Face Recognition

### PROJECT COST

Software : All software used are freely available Hardware : Rs.1800

Total : Rs.1800

### SYSTEM ARCHITECTURE

Jarvis is a popular virtual assistant that can perform a variety of tasks such as answering questions, reminders. The architecture of Jarvis can vary depending on how it is implemented, but a common architecture might involve the following components:

Wake word detection: Jarvis needs to be able to listen for a wake word (such as "Hey Jarvis" or "OK Google") that signals that the user wants to interact with the virtual assistant. This component typically involves audio processing and machinelearning algorithms to recognize the wake word. Initially, the condition here is whether or not Jarvis voice assistant is active; if it is active, itrequests user input; otherwise, make Jarvis active (make it on). The user then provides input in the form of speech or text; if the input is in text, it proceeds to the action to be taken or the skill to be executed; if the input is in speech, it uses the speech recognition feature to convert it to text and proceeds to the action. Now, if the skills to be executed are adequate to Jarvis, it responds positively to the user in the form of speech and then executes the commands for operations.

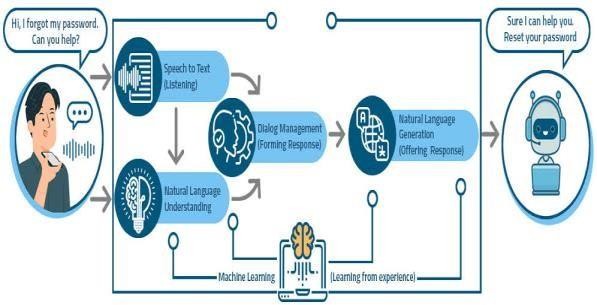


Fig. 4.1.1 SYSTEM ARCHITECTURE

### Speech recognition

After the wake word is detected, Jarvis needs to be able to recognize the user's speech and convert it into text. This component also typically involves audio processing and machine learning algorithms, such as deep neural networks, toperform the speech recognition.

### Natural language understanding

Once the user's speech is converted into text, Jarvis needs to be able to understand the user's intent and extract relevant information from the user's request. This component typically involves natural language processing techniques such as named entity recognition and sentiment analysis.

### Dialog management

After understanding the user's intent, Jarvis needs to be able to manage a dialog with the user to gather any additional information needed to fulfill the user's request. This component typically involves a rule-based or machine learning-based approach to dialog management.

### Task fulfillment

Once Jarvis has gathered all the necessary information from the user, it needs to be able to perform the requested task. This could involve calling APIs for external services, controlling smarthome devices, or simply providing information to the user.

### Response generation

Finally, Jarvis needs to be able to generate a response to the user, either as speech or as text. This component typically involves natural language generation techniques such as text-to-speech synthesis or template-based text generation.

Overall, the architecture of Jarvis is a complex system that requires expertise in several areas, including audio processing, natural languageprocessing, and software engineering

### DETAILED DESCRIPTION OF MODULES

* Tkinter - tkinter is a standard Python GUI (Graphical User Interface) package. It provides a set of tools for building graphical applications, including windows, widgets, and other graphical elements. With tkinter, you can create desktop applications that run on Windows, macOS, and Linux.
* Pvporcupine - Pvporcupine is a Python package for creating wake word detection (also known as keyword spotting) systems. A wake word detection system listens to an audio stream in real-time and detects when a specific word or phrase (the "wake word") is spoken, triggering some action or alert.
* Pyaudio - Pyaudio is a Python package that provides bindings for the Port Audio library, which is a cross-platform audio I/O (input/output) library. Pyaudio allows you to play and record audio streams in real-time using Python.
* SpeechRecognition - SpeechRecognition is a Python package for performing speech recognition using various APIs and engines. It provides a simple and easy-to-use interface for transcribing audio files or live speech in real-time.
* Pyttsx3 - pyttsx3 is a Python package that provides a cross-platform text-to-speech (TTS) engine. With pyttsx3, you can convert any written text into spoken words in real-time.
* NLTK - NLTK (Natural Language Toolkit) is a Python package for working with human language data. It provides various tools and resources for natural language processing (NLP), including:
  + Tokenization: splitting text into individual words or sentences Part-of-speech tagging: identifying the grammatical parts of speech of each word in a sentence. Named entity recognition: identifying entities such as people, places, and organizations in text.
  + Sentiment analysis: determining the sentiment (positive, negative, or neutral) of a piece of text.
  + Machine learning algorithms for classifying text documents.
* Torch - Torch is a Python package that provides a scientific computing framework for building and training machine learning models. It is primarily used for deep learning, which is a subset of machine learning that focuses on training neural networks with many hidden layers.
* Pyautogui - pyautogui is a Python package that provides cross-platform control of the mouse and keyboard, as well as other GUI automation tasks. It allows you to automate repetitive tasks such as clicking, scrolling, typing, and taking screenshots.
* ADB - adb stands for Android Debug Bridge. It is a command-line tool that is part of the Android SDK (Software Development Kit). adb provides a bridge between an Android device and a computer, allowing developers to debug and test their Android applications.
* Selenium - Selenium is a Python package for automating web browsers. It provides a simple and easy-to-use interface for controlling web browsers, allowing you to automate repetitive tasks such as clicking, scrolling, typing, and form submission.

### HARDWARE AND SOFTWARE REQURIMENTS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.No. | Name of Resource/material | Specification | Qty | Remarks |
| 1 | PyCharm | Python(3.9.5) |  |  |
| 2 | Operating System | Windows 10,11 |  |  |
| 3 | Processor | Intel i5 11th Gen |  |  |
| 4 | RAM | 8GB |  |  |
| 5 | GPU | GTX 1050 |  |  |
| 6 | Hardware | Webcam |  |  |
| 7 | MS Office | Editor |  |  |

**TABLE NO. 5.1.1**

### DATA FLOW DIAGRAM (DFD’S)

A data-flow diagram (DFD) is a way of representing a flow of a data of a [process](https://en.wikipedia.org/wiki/Process) or a system (usually an [information system](https://en.wikipedia.org/wiki/Information_system)). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no controlflow, there are no decision rules and no loops. Specific operations based on the data can berepresented by a [flowchart.](https://en.wikipedia.org/wiki/Flowchart) For each data flow, at least one of the endpoints (source and / or destination) must exist in aprocess. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes. The data-flow diagram is part of the structured-analysis modelling tools. DFD consists ofseries of bubble joined by lines.DFD may be further partitioned into different levels to show detailed information flow. Forexample: level 0, level 1, level 2.

### DFD LEVEL 0

A level 0 data flow diagram (DFD), also known as a context diagram, shows a data systemas a whole and emphasizes the way it interacts with external entities. This DFD level 0 example shows how such a system might function within a typical retail business.

JARVIS

USER

Fig. 5.2.1 DATA FLOW DIAGRAM (DFD LEVEL 0)

### DFD LEVEL 1

A level 1 data flow diagram (DFD) is more detailed than a level 0 DFD but not as detailedas a level 2 DFD. It breaks down the main processes into subprocesses that can then be analysed and improved on a more intimate level.

### DFD LEVEL 2

A level 2 data flow diagram (DFD) offers a more detailed look at the processes that makeup an information system than a level 1 DFD does. It can be used to plan or record the specific makeup of a system.

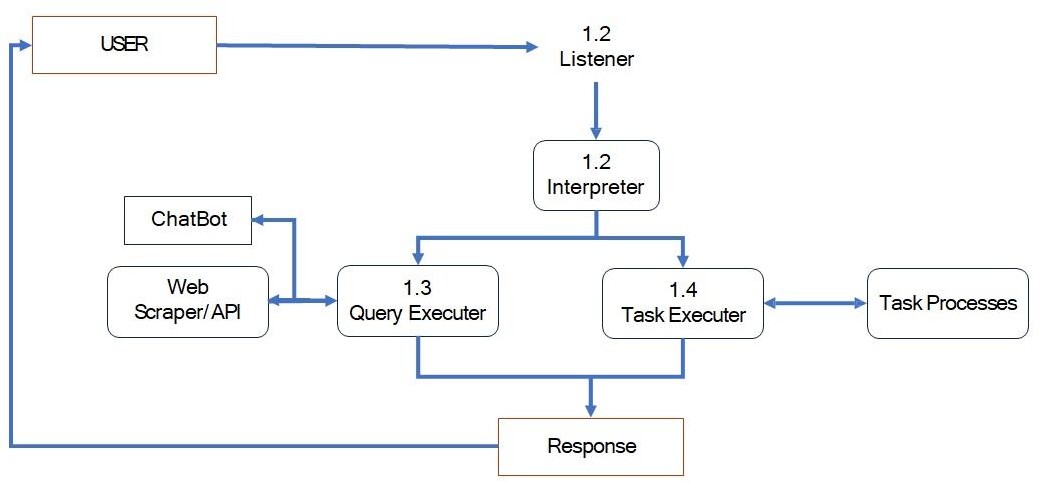


Fig. 5.2.2 DATA FLOW DIAGRAM (DFD LEVEL 1)

### USE CASE DIAGRAM

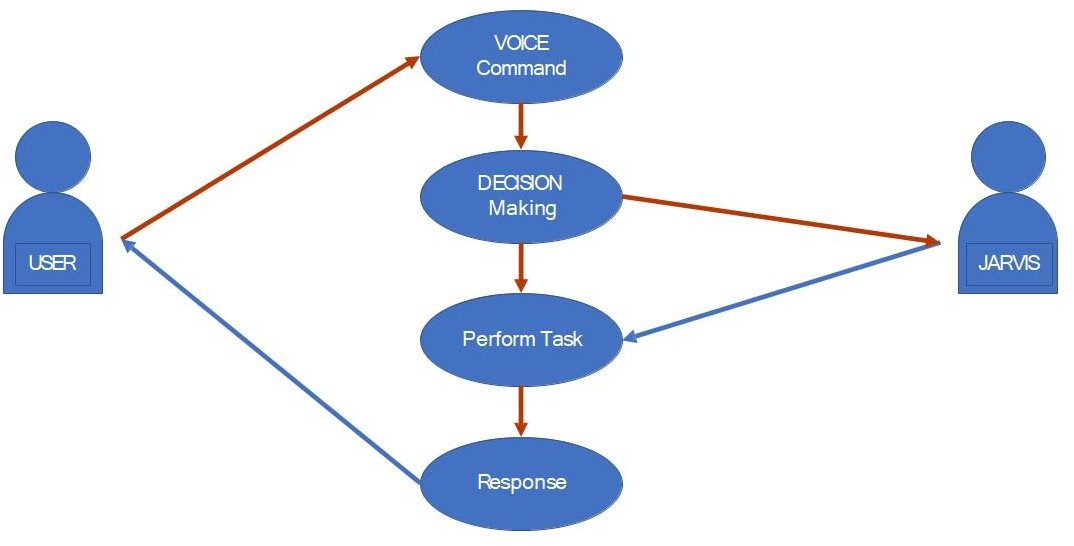


Fig 5.3.1 USE – CASE DIAGRAM

### ACTIVITY DIAGRAM

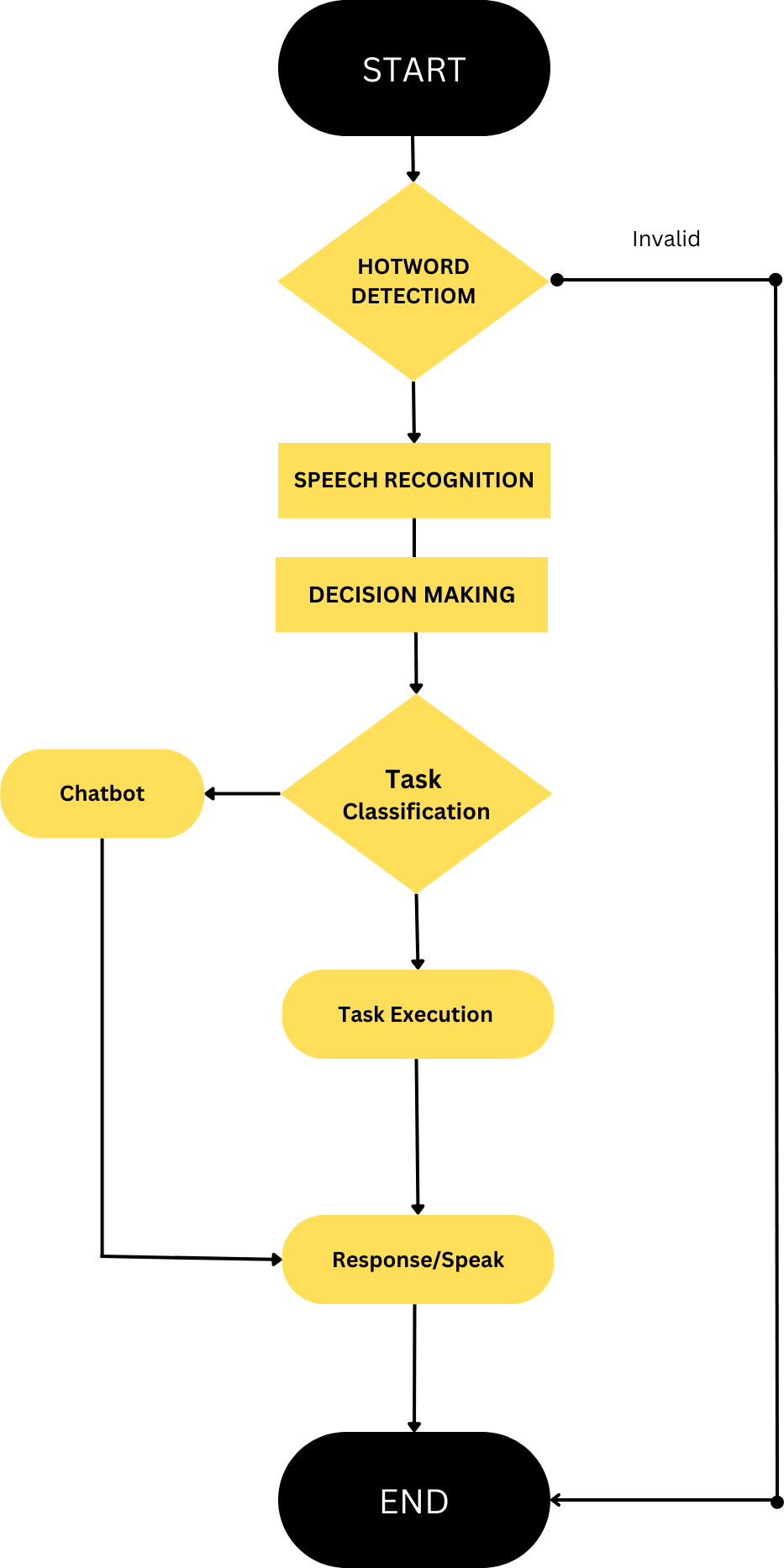


Fig. 5.4.1 ACTIVITY DIAGRAM

### SEQUENCE DIAGRAM

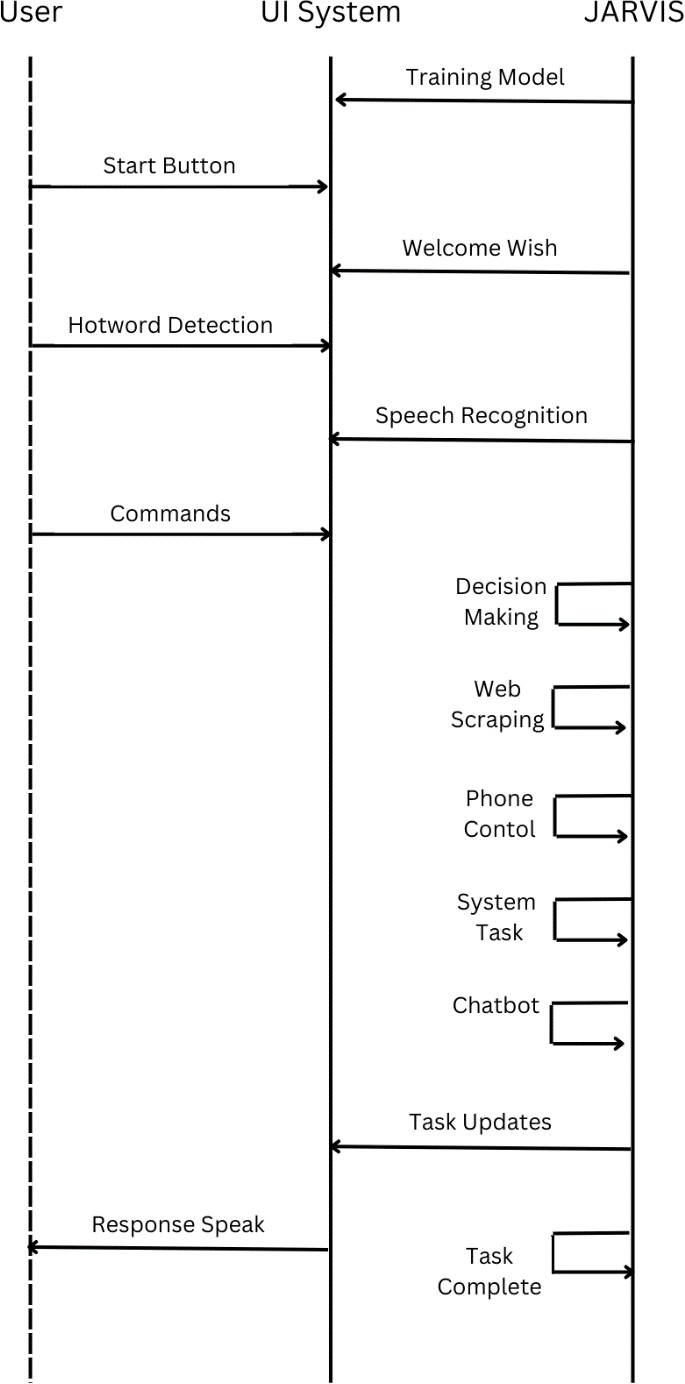


Fig. 5.5.1 SEQUENCE DIAGRAM

### TIMELINE CHART

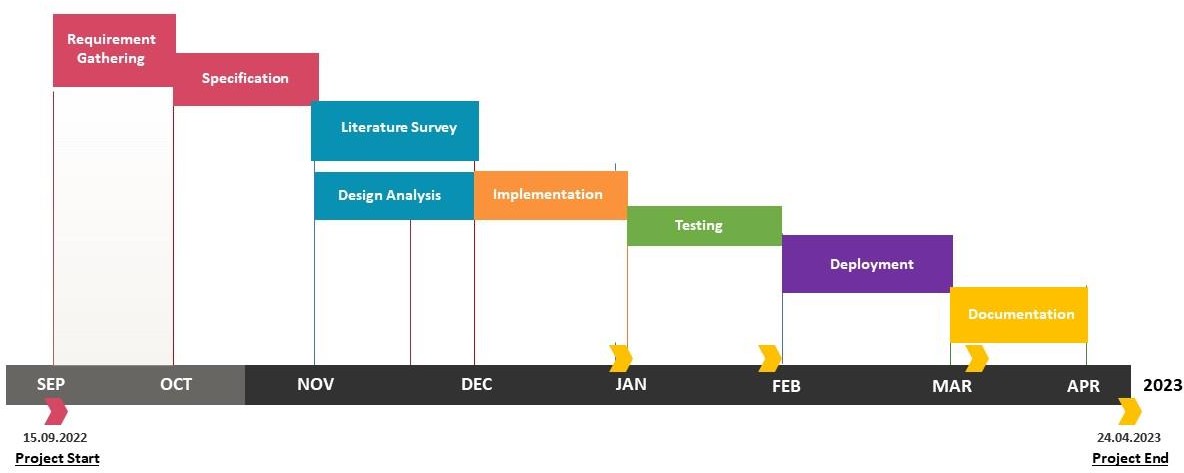


Fig 5.6.1 TIMELINE CHART (GANTT CHART)

### IMPLEMENTATION

* + 1. **JARVIS PYTHON CODE:**

import os import random import threading import tkinter import sys

from tkinter.constants import BOTH, YES, NW from Body.Microphone import MicExecution from Brain.Trainer import TasksExecutor

from Features.Task import Functions from Initilazation.Init import init

def jarvis(): init()

while True:

sentences = MicExecution()

word = TasksExecutor(sentences) Functions(word, sentences)

def Color\_change():

color = ['red', 'blue', 'white', 'gold', 'yellow'] actual\_color = random.choice(color) label.configure(foreground=actual\_color) label.after(1000, Color\_change)

def run(): threading.Thread(target=jarvis).start()

def flush(self): pass

def stop():

os.system("TASKKILL /F /im python.exe") and os.system("TASKKILL /F /im Jarvis.py") and os.system(

"TASKKILL /F /im Jarvis.exe") screen\_main = tkinter.Tk() screen\_main.title('Jarvis')

screen\_main.configure(background='black', highlightbackground='blue', highlightthickness=5) screen\_main.attributes('-fullscreen', True)

screen\_main.iconbitmap('GUI\\Icon\\jarvis.ico')

canvas = tkinter.Canvas(width=200, height=200, bg='black', highlightbackground='cyan', highlightthickness=5)

canvas.pack(expand=YES, fill=BOTH)

photo = tkinter.PhotoImage(file='GUI\\bg\_jarvis.gif') canvas.create\_image(1010, 0, image=photo, anchor=NW)

start\_button = tkinter.Button(screen\_main, background='black', fg='red', font=('IRON MAN OF WAR 2 NCV', 40),

text='Start', command=run)

start\_button.place(x=1350, y=700)

stop\_button = tkinter.Button(screen\_main, background='black', fg='red', font=('IRON MAN OF WAR 2 NCV', 40), text='Stop',

command=stop) stop\_button.place(x=1600, y=700)

label = tkinter.Label(screen\_main, font=('IRON MAN OF WAR 2 NCV', 70), text="JARVIS", background='black')

Color\_change() label.place(x=1480, y=900) Color\_change()

terminal = tkinter.Text(screen\_main)

terminal.configure(background='black', fg='white', highlightbackground='red', highlightthickness=5)

terminal.configure(width=45, height=41.499) terminal.configure(font=('Bruce Forever', 15)) terminal.place(x=13.5, y=13.48)

class Redirect:

def init (self, widget, autoscroll=True): self.widget = widget

self.autoscroll = autoscroll def write(self, text):

self.widget.insert('end', text) if self.autoscroll:

self.widget.see('end') old\_stdout = sys.stdout sys.stdout = Redirect(terminal) screen\_main.mainloop()

### Speaker.py:

import pyttsx3 def Speak(audio):

engine = pyttsx3.init("sapi5")

voices = engine.getProperty("voices") engine.setProperty("voice", voices[1].id) engine.setProperty("rate", 140) engine.say(audio)

print(f">> Jarvis : {audio}")

print(" ")

engine.runAndWait()

### Hotword.py :

import struct import pyaudio import pvporcupine

from Body.Speaker import Speak

access\_key = "RYHozjPJuklGvd/1+6i8aIc2Se48CoS0QdJLBI3CscGnY72CjRqZ7g=="

def jarvis\_dect(): porcupine = None paud = None audio\_stream = None try:

porcupine = pvporcupine.create(keywords=["jarvis"], access\_key=access\_key) paud = pyaudio.PyAudio()

audio\_stream = paud.open(rate=porcupine.sample\_rate, channels=1,

format=pyaudio.paInt16, input=True,

frames\_per\_buffer=porcupine.frame\_length)

while True:

keyword = audio\_stream.read(porcupine.frame\_length)

keyword = struct.unpack\_from("h" \* porcupine.frame\_length, keyword) keyword\_index = porcupine.process(keyword)

if keyword\_index >= 0:

# Speak("Jarvis Detected") break

finally:

if porcupine is not None: porcupine.delete()

if audio\_stream is not None: audio\_stream.close()

if paud is not None: paud.terminate()

### Weather.py :

from time import sleep

import selenium.common.exceptions from selenium import webdriver

from selenium.webdriver.chrome.service import Service from selenium.webdriver.chrome.options import Options from selenium.webdriver.common.by import By

from Body.Speaker import Speak def get\_Weather(sentences):

chrome\_options = Options() chrome\_options.add\_argument('--log-level=3') chrome\_options.add\_argument('--headless')

s = Service('Drivers\\chromedriver.exe')

driver = webdriver.Chrome(service=s, options=chrome\_options) driver.maximize\_window()

Website = f'https:/[/www.google.com/se](http://www.google.com/search?q)a[rch?q=](http://www.google.com/search?q){sentences}' driver.get(Website)

sleep(5) def City():

try:

City\_ = driver.find\_element(By.XPATH, '//\*[@id="oFNiHe"]/div/div/div/div[1]').text return City\_

except selenium.common.exceptions.ElementNotInteractableException:

City\_Xpath = driver.find\_element(By.XPATH, '/html/body/div[7]/div/div[10]/div/div[1]/div['

'1]/div/div/div/div[1]').text

return City\_Xpath def Temp():

try:

Temp\_ = driver.find\_element(By.XPATH, '//\*[@id="wob\_wc"]/div[1]/div[1]/div').text return Temp\_

except selenium.common.exceptions.ElementNotInteractableException: Temp\_XPath = driver.find\_element(By.XPATH,

'/html/body/div[7]/div/div[10]/div/div[2]/div[2]/div/div/div[' '1]/div/div/div/div/div[1]/div[1]/div').text

return Temp\_XPath Temperature = Temp()

Temperature = Temperature.replace("°F", "")

def weather(): try:

Weather\_ = driver.find\_element(By.XPATH, '//\*[@id="wob\_dc"]').text eturn Weather\_

except selenium.common.exceptions.ElementNotInteractableException: Weather\_XPath = driver.find\_element(By.XPATH,

'/html/body/div[7]/div/div[11]/div/div[2]/div[2]/div/div/div[' '1]/div/div/div/div/div[2]/span/div[3]/span').text

return Weather\_XPath Weather = weather()

def Precipitation(): try:

Precipitation\_ = driver.find\_element(By.XPATH, '//\*[@id="wob\_wc"]/div[1]/div[2]/div[1]').text

return Precipitation\_

except selenium.common.exceptions.ElementNotInteractableException: Precipitation\_XPath = driver.find\_element(By.XPATH,

'/html/body/div[7]/div/div[10]/div/div[2]/div[2]/div/div/div[' '1]/div/div/div/div/div[1]/div[2]/div[1]').text

return Precipitation\_XPath def Humidity():

try:

Humidity\_ = driver.find\_element(By.XPATH, '//\*[@id="wob\_wc"]/div[1]/div[2]/div[2]').text

return Humidity\_

except selenium.common.exceptions.ElementNotInteractableException: Humidity\_XPath = driver.find\_element(By.XPATH,

'/html/body/div[7]/div/div[10]/div/div[2]/div[2]/div/div/div[' '1]/div/div/div/div/div[1]/div[2]/div[2]').text

return Humidity\_XPath def Wind():

try:

Wind\_ = driver.find\_element(By.XPATH, '//\*[@id="wob\_wc"]/div[1]/div[2]/div[3]').text return Wind\_

except selenium.common.exceptions.ElementNotInteractableException: Wind\_XPath = driver.find\_element(By.XPATH,

'/html/body/div[7]/div/div[10]/div/div[2]/div[2]/div/div/div[' '1]/div/div/div/div/div[1]/div[2]/div[3]').text

return Wind\_XPath Precipitation = Precipitation() Humidity = Humidity()

Wind = Wind()

City = City().replace("Results for ","")

Result = f''' Sir !! The Temperature in {City} is {Temperature} The Overall Condition : {Weather}.

{Precipitation}

{Humidity}

{Wind}

'''

Speak(Result)

### News.py :

import random import requests

from googletrans import Translator def ToEnglish(audio):

if len(audio) > 3:

Translate = Translator()

Sent = Translate.translate(audio) text = Sent.text

return text

elif "None" in audio: print("Could not Understand")

else:

pass

def latest\_news():

country = ['us', 'in', 'cn', 'ru', 'jp', 'fr', 'sg'] cn = random.choice(country) news\_headlines = []

news\_Des = []

res = requests.get( f"https://newsapi.org/v2/top-

headlines?country={cn}&apiKey=587dcad64761431490f655fda8c69018&category=general").jso n()

articles = res["articles"] for article in articles:

news\_headlines.append(article["title"]) news\_Des.append((article["description"])) news\_Des = ToEnglish(str(news\_Des)) news\_headlines = ToEnglish(str(news\_headlines))

news\_Des = news\_Des.replace("[", "").replace("]", "").replace("'", "")

news\_headlines = news\_headlines.replace("[", "").replace("]", "").replace("'", "")

News = f"Sir! Here are the latest news headlines:\n{news\_headlines} \nUpdate of the News:

{news\_Des}"

return News

### Location.py :

import bs4 import requests

from geopy import Nominatim from Body.Speaker import Speak def My\_Location():

Speak("Getting Your Location Sir")

google\_search = requests.get('https:/[/www.google.com/se](http://www.google.com/search?q=Google%27))a[rch?q=Google')](http://www.google.com/search?q=Google%27)) soup = bs4.BeautifulSoup(google\_search.text, 'html.parser') search\_results = soup.select('.dfB0uf')

result = str(search\_results)

addressNow = result.replace('[<span class="dfB0uf">', '').replace('</span>]', '') loc = Nominatim(user\_agent="GetLoc")

getLoc = loc.geocode(addressNow) latitude = str(getLoc.latitude) longitude = str(getLoc.longitude)

Address = f'''Sir , You Are Now In {getLoc.address} .

Latitude = {latitude} Longitude = {longitude}'''

return Address

### Whatsapp.py :

import datetime import pywhatkit

from Body.Microphone import writingCommand from Body.Speaker import Speak

def Whatsapp\_msg(number, name):

Speak(f"Sir ! Tell me what message you want to send to {name}") Message = writingCommand()

time = datetime.datetime.now() hr = time.hour

minute = time.minute minute = minute + 1

pywhatkit.sendwhatmsg(f"+91{number}", Message, hr, minute)

### Find Phone.py :

from twilio.rest import Client from Body.Speaker import Speak def find\_phone():

account\_sid = 'AC5691066a967bb3dbd4a0b920d537ef53' auth\_token = '292df4acaf38ff4252b29e8eb722bb41' client = Client(account\_sid, auth\_token)

message = client.calls.create(

twiml='<Response><Say>Found Your Phone Sir!!, If this Phone does not belong to You then call on '

'9011022124..</Say></Response>',

call\_reason='Found Your Phone Sir!!, If this Phone does not belong to You then call on 9011022124..',

from\_='+1 218 395 5886',

to='+91 9604304414',

url="https://demo.twilio.com/docs/voice.xml")

Speak("Finding your phone Sir , it will take four to five seconds !!!")

### SpeedTest.py :

import speedtest

from Body.Speaker import Speak def Speed\_test():

try:

test = speedtest.Speedtest() Speak("Loading Servers. ")

test.get\_best\_server()

best = test.get\_best\_server()

Speak(f"Found: {best['host']} located in {best['country']}. It would take few second for result !!") download\_result = test.download()

upload\_result = test.upload() ping\_result = test.results.ping

Speed\_Result = f'''Sir , Here are the following result for Speed Test.

Download speed is {download\_result / 1024 / 1024 :.2f} Mb per second. Upload speed is {upload\_result / 1024 / 1024 :.2f} Mb per second.

Sir and the ping is {ping\_result:.2f} ms.''' return Speed\_Result

except speedtest.ConfigRetrievalError or speedtest.SpeedtestException: Speed\_test\_error = "Sir there is a problem with your Internet Connection" return Speed\_test\_error

### Task.py :

from time import sleep import pyautogui

import pywhatkit as pywhatkit import requests

from Body.Speaker import Speak

from Features.OS.Current\_dur import Curr\_time, Curr\_date, Curr\_day from Features.OS.Notepad import create\_note

from Features.OS.Speedtest import Speed\_test

from Features.OS.Volume\_control import change\_volume

from Features.OS.Windows import Shutdown\_pc, Lock\_pc, battery\_per, Screen\_shot from Features.Phone.Activity import Call\_number, Ans\_Call, End\_Call, Unlock from Features.Phone.find\_phone import find\_phone

from Features.Social\_media.Instagram import profile\_info from Features.Social\_media.Whatsapp import Whatsapp\_msg from Features.Web\_Scraping.IP\_address import Ip\_address from Features.Web\_Scraping.Location import My\_Location from Features.Web\_Scraping.News import latest\_news

from Features.Web\_Scraping.Thoughts import thought\_day from Features.Web\_Scraping.Weather import get\_Weather Contact = {

"harshal": "9172691064",

"ajay": "9823513202",

"goku": "9021144118",

"my": "9604304414"

}

def random\_stuff(sentences):

url = "https://random-stuff-api.p.rapidapi.com/ai/response" text = sentences

querystring = {"message": f"{text}", "bot\_name": "Jarvis", "bot\_gender": "male ", "bot\_master": "White\_Shadows", "bot\_age": "18", "bot\_company": "MIT WPU", "bot\_location": "India (OPTIONAL)", "bot\_email": "[anishbochare@gmail.com",](mailto:anishbochare@gmail.com) "bot\_build": "Public (OPTIONAL)", "bot\_birth\_year": "2021", "bot\_birth\_date": "3

March 2021",

"bot\_birth\_place": "India (OPTIONAL)", "bot\_favorite\_color": "Red", "bot\_favorite\_book": "Tony Stark",

"bot\_favorite\_band": "Imagine Doggos (OPTIONAL)", "bot\_favorite\_artist": "Eminem (OPTIONAL)",

"bot\_favorite\_actress": "Emma Watson (OPTIONAL)", "bot\_favorite\_actor": "Jim Carrey (OPTIONAL)",

"user\_id": "420"} headers = {

"Authorization": "HCjo5Y4B4RDD",

"X-RapidAPI-Key": "0477f85d5cmsh94bbbb1f4cc64a5p109fbbjsncdd6553ee50c", "X-RapidAPI-Host": "random-stuff-api.p.rapidapi.com"

}

response = requests.request("GET", url, headers=headers, params=querystring) r = response.text

r = r.split('"message":"') r = r[1]

r = r.replace('"}', "").replace('"', '').replace('warning', '').replace(':No', '').replace(',', '') Speak(r)

def Functions(tag, sentences=""): if 'Exit' in tag:

Speak("Good Bye, Sir. have a Nice Day :)") exit()

elif 'Cur\_Time' in tag: Time = Curr\_time() Speak(Time)

elif "Today\_date" in tag: Date = Curr\_date() Speak(Date)

elif "Today\_day" in tag: Day = Curr\_day() Speak(Day)

elif "Notepad" in tag:

Speak("Okay, I am Ready to Write a Note") create\_note()

elif "Speedtest" in tag: Speed\_Result = Speed\_test() Speak(Speed\_Result)

elif "increase\_vol" in tag: Volume = change\_volume(2) Speak(Volume)

elif "decrease\_vol" in tag: Volume = change\_volume(-2) Speak(Volume)

elif "Shutdown\_Device" in tag: Shut\_status = Shutdown\_pc() Speak(Shut\_status)

elif "Lock" in tag: Lock\_status = Lock\_pc() Speak(Lock\_status)

elif "Battery\_Per" in tag: Percentage = battery\_per() Speak(Percentage)

elif "Screen\_shot" in tag: Screenshot = Screen\_shot() Speak(Screenshot)

elif "My\_PC\_IP" in tag: ip\_address = Ip\_address() Speak(ip\_address)

elif "My\_Location" in tag: address = My\_Location() Speak(address)

elif "News" in tag: News = latest\_news() Speak(News)

elif "Weather" in tag: get\_Weather(sentences)

elif "Music" in tag:

song = str(sentences).replace("play", "").replace("music", "").replace("youtube", "") Speak("Sir! Playing the Song on YouTube")

pywhatkit.playonyt(song) sleep(3) pyautogui.press('f')

elif "Thoughts" in tag: thought = thought\_day() Speak(thought)

elif "Whatsapp\_msg" in tag: contact\_name = str(sentences).lower() names = list(Contact.keys())

for person in names: try:

if person in contact\_name: num = f"{Contact[person]}"

Whatsapp\_msg(num, person)

break except:

Speak("Sir! There is No Number For the Given Name to make voice call") elif "Insta\_Profile" in tag:

profile\_info(sentences) elif "Phone\_Call" in tag:

Call\_number(sentences) elif "Ans\_call" in tag:

Speak("Sir! I am Picking up the Phone call.") Ans\_Call()

elif "End\_Call" in tag:

Speak("Sir! I am Ending the Phone call.") End\_Call()

elif "Phone\_Unlock" in tag: Unlock()

Speak("Sir! Your Phone has been Unlocked.") elif "Self\_Call" in tag:

Speak("Sir ! I am Calling on Your Phone") find\_phone()

else:

random\_stuff(sentences)

### face recognition.py :

import datetime import socket

from Features.Web\_Scraping.Thoughts import thought\_day from Body.Speaker import Speak

global name

def isConnect(): try:

s = socket.create\_connection( [("w](http://www.google.com/)ww[.google.com"](http://www.google.com/), 80))

if s is not None: s.close()

return True except OSError:

pass return False

def wishMe(user):

hour = int(datetime.datetime.now().hour) if 0 <= hour < 12:

Speak(f"Good Morning {user} !") elif 12 <= hour < 18:

Speak(f"Good Afternoon {user} !") else:

Speak(f"Good Evening Sir {user} !") def face\_detection():

global name, id import cv2

recognizer = cv2.face.LBPHFaceRecognizer\_create() # Local Binary Patterns Histograms recognizer.read('Brain\\trainer\\trainer.yml') # load trained model

cascadePath = "Brain\\haarcascade\_frontalface\_default.xml"

faceCascade = cv2.CascadeClassifier(cascadePath) # initializing haar cascade for object detection approach

font = cv2.FONT\_HERSHEY\_SIMPLEX # denotes the font type id = 2 # number of persons you want to Recognize

names = ['', 'Ansh', 'Anish', 'Atharav', 'Tanvi'] # names, leave first empty bcz counter starts from

0

cam = cv2.VideoCapture(0, cv2.CAP\_DSHOW) # cv2.CAP\_DSHOW to remove warning cam.set(3, 640) # set video FrameWidht

cam.set(4, 480) # set video FrameHeight

# Define min window size to be recognized as a face minW = 0.1 \* cam.get(3)

minH = 0.1 \* cam.get(4) # flag = True

while True:

ret, img = cam.read() # read the frames using the above created object converted\_image = cv2.cvtColor(img,

cv2.COLOR\_BGR2GRAY) # The function converts an input image from one color space to another

faces = faceCascade.detectMultiScale( converted\_image,

scaleFactor=1.2, minNeighbors=5, minSize=(int(minW), int(minH)),

)

for (x, y, w, h) in faces:

cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2) # used to draw a rectangle on any

image

id, accuracy = recognizer.predict(converted\_image[y:y + h, x:x + w]) # to predict on every

single image

# Check if accuracy is less them 100 ==> "0" is perfect match if (accuracy <= 70):

detected\_name = names[id - 1] name = detected\_name

accuracy = " {0}%".format(round(100 - accuracy)) else:

detected\_name = "unknown" name = detected\_name

accuracy = " {0}%".format(round(100 - accuracy))

cv2.putText(img, str(detected\_name), (x + 5, y - 5), font, 1, (255, 255, 255), 2)

cv2.putText(img, str(accuracy), (x + 5, y + h - 5), font, 1, (255, 255, 0), 1) cv2.imshow('camera', img)

k = cv2.waitKey(10) & 0xff # Press 'ESC' for exiting video if k == 27:

break

# Do a bit of cleanup cam.release() cv2.destroyAllWindows() if name == "unknown":

user\_name = "Sir/Mam" wishMe(user\_name)

Speak("Sir/Mam, You Don't have access to the Program") exit()

elif name != "unknown": wishMe(name)

def Check\_Net(): Net = isConnect() if True is Net:

print("J.A.R.V.I.S is Online") else:

print("Please Connect to Internet") def thoughts():

thought = thought\_day() Speak(thought)

def init(): Check\_Net() face\_detection()

### TYPES OF TESTING

Software testing is an investigation conducted to provide stakeholders with information about the [quality](https://en.wikipedia.org/wiki/Software_quality) of the [software](https://en.wikipedia.org/wiki/Software) product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding [software bugs](https://en.wikipedia.org/wiki/Software_bug) (errors or other defects), and verifying that the software product is fit for use.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest.

### UNIT TESTING

Unit testing is a software development process in which the smallest testable parts of an application called units, are individually and independently inspected for proper operation. Unit testing is the testing od an individual unit or group of relates units. It falls under the classof white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing excepted output given input.

### INTEGRATION TESTING

Integration testing is testing in which a group of components are combined to produce output.Also, the interaction between software and hardware is tested in integration testing if softwareand hardware components have any relation. It may fall under both white box testing black box testing.

### FUNCTION TESTING

Functional testing is the testing to ensure that the species functional-it required in the system requirements works. It falls under the class of black box testing.

### SYSTEM TESTING

System testing is the testing to ensure that by putting the software in die rent environments (e.g., Operating Systems) it still works. System testing is done with full system implantation and environment. It falls under the class of black box testing.

### USABILITY TESTING

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user- friendly? How easily can the client learn? After learning how to use, how pro anciently and the client perform? How pleasing is it to use its design.

### STRESS TESTING

Stress testing helps in finding out the product capability to handle a certain n load and whenthe system will break down due to overload of records. Loading of record can be steadily increased to see when the system will fail.

### LOAD TESTING

Load testing is a type of non-functional testing. A load test is type of software testing which is conducted to understand the behaviour of the application under a specific expected load. Load testing is performed to determine a system's behaviour under bothnormal and at peak conditions

### TEST CASE

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr.no | Test Step | Test Data | ExpectedResult | ActualResult | Status(Pass  / Fail) |
| 1 | Start Button | Button = “Start” | Jarvis should be turned on | Jarvis gets on | Pass |
| 2 | Greetings Function | Greetings = Welcome user’s name | Greetings should be displayed | Greetings are displayed | Pass |
| 3 | Hotword Detection | Speak = “Jarvis” | Jarvis should recognize user | Jarvis recognizes user | Pass |
| 4 | Speech Recognition | Speech = “Jarvis” | Jarvis accepts voice command | Jarvis accepts voice command | Pass |
| 5 | Speak Function | Speak = “Jarvis” | Jarvis should listen to user’s command | Jarvis listens to user’s command | Pass |
| 6 | Speed Test | Speak = “Speed Test” | Jarvis should  conduct a speed test | Jarvis conducts a speed test | Pass |
| 7 | Weather Update | Speak = “Weather Update” | Jarvis should give weather update | Jarvis gives weather update | Pass |
| 8 | Location Update | Speak = “Location Update” | Jarvis should display user’s location | Jarvis displays user’s location | Pass |
| 9 | WhatsApp Message | Speak = “Message” | Jarvis should send message to selected contact numbers | Jarvis sends message to selected  contact numbers | Pass |
| 10 | Phone Control | Speak = “Unlock” | Phone should get unlocked | Phone gets unlocked | Pass |
| 11 | News Update | Speak = “News” | Jarvis gives news update | Jarvis gives news update | Pass |
| 12 | Notepad Creation | Speak = “Notepad” | Notepad should gets created | Notepad gets created | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 13 | PC Control | Speak = “Shutdown” | Controls PC | PC gets controlled | Pass |
| 14 | Find Phone | Speak = “Find my Phone” | Jarvis finds user’s phone | User’s phone is found by Jarvis | Pass |
| 15 | Face Recognition Training | Button = “Face train” | Face should get generated and model gets trained | Face gets generated and model gets trained | Pass |
| 16 | Face Recognition | Button = “Start” | Face should get recognized | New customized category is added | Pass |

**TABLE NO. 5.7.3.1**

### 6.1 & 6.2 OUTPUTS / RESULTS

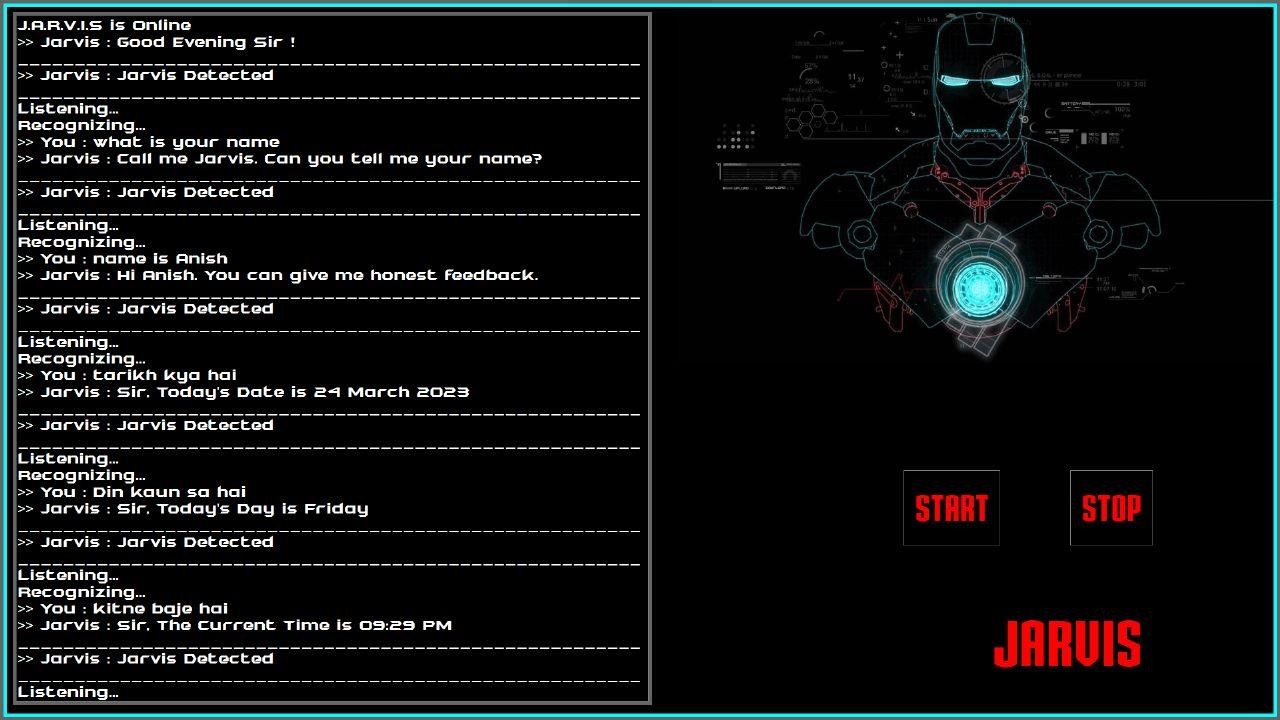


Fig. 6.1.1 Introduction Page

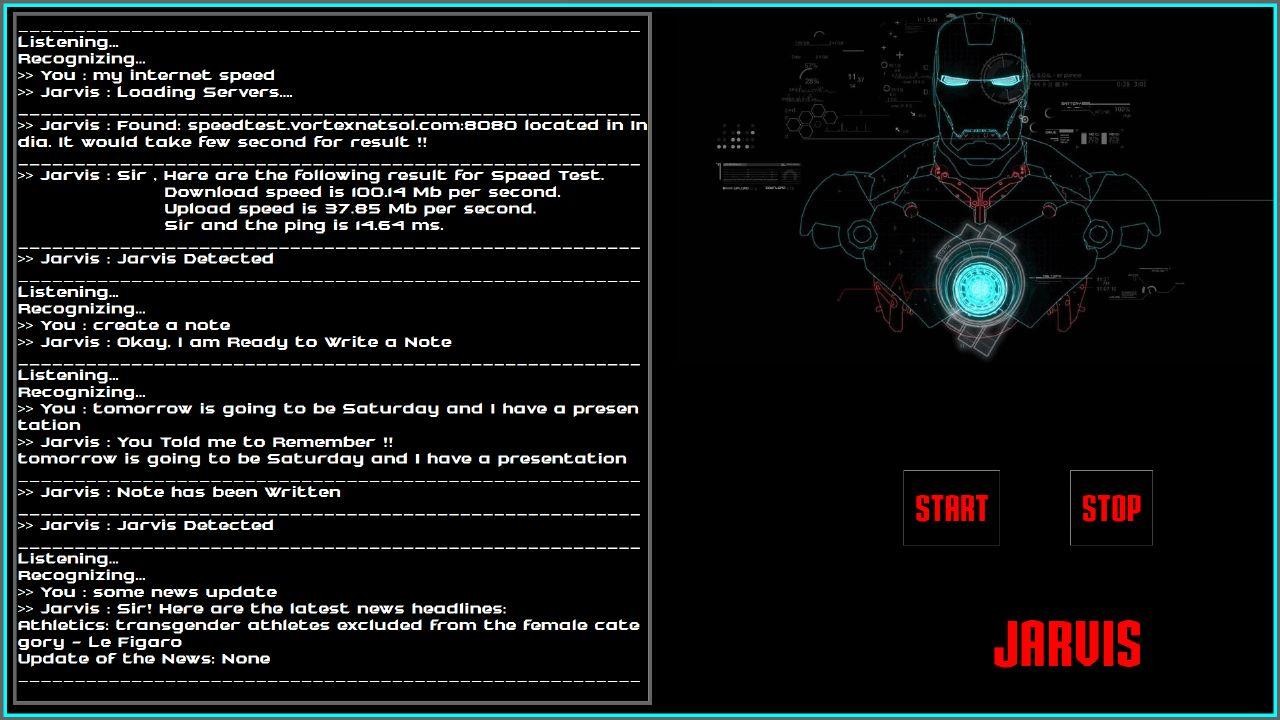


Fig. 6.1.2 Speed Test Page

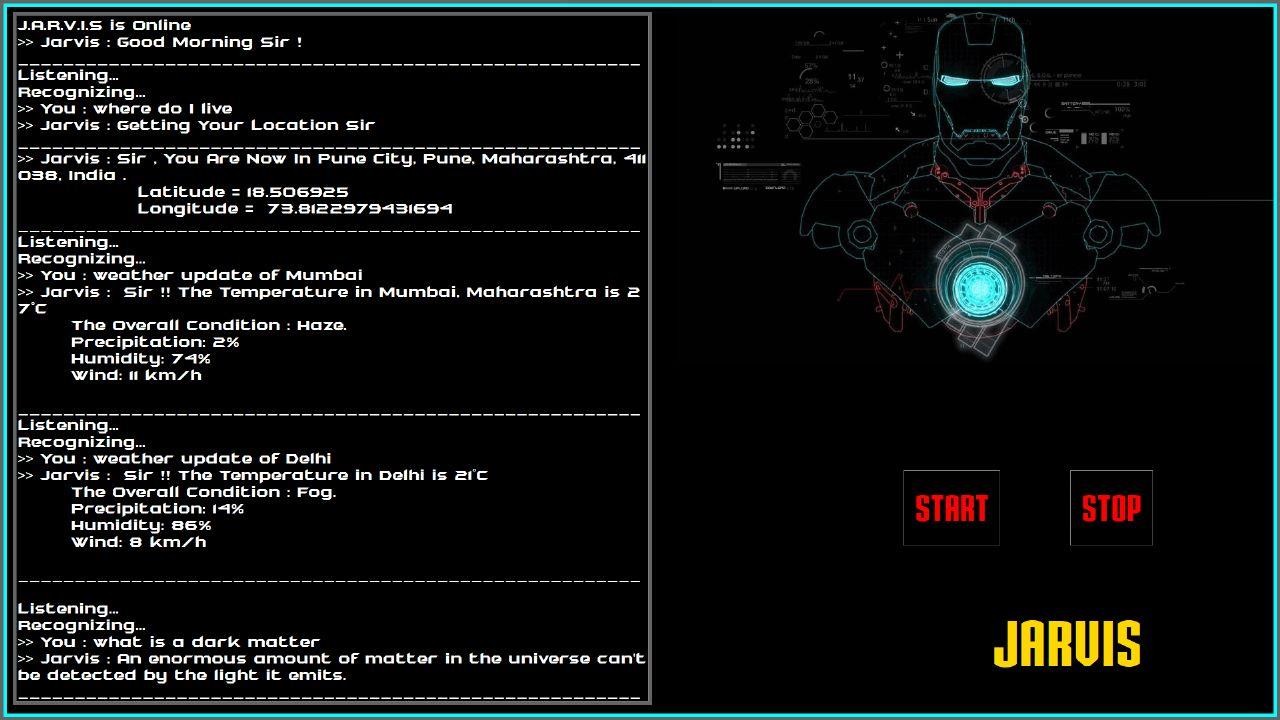


Fig. 6.1.3 Weather Update Page

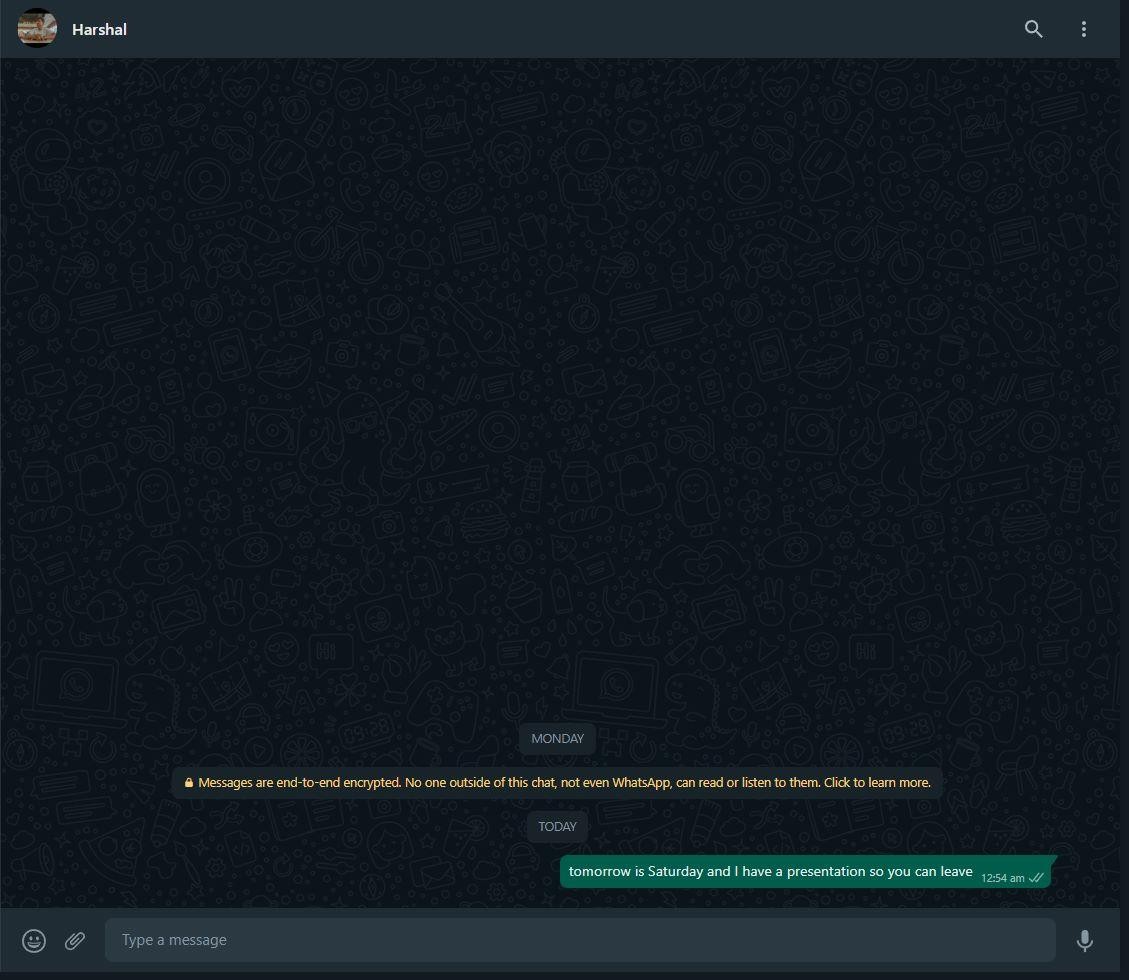


Fig. 6.1.4 Message Page

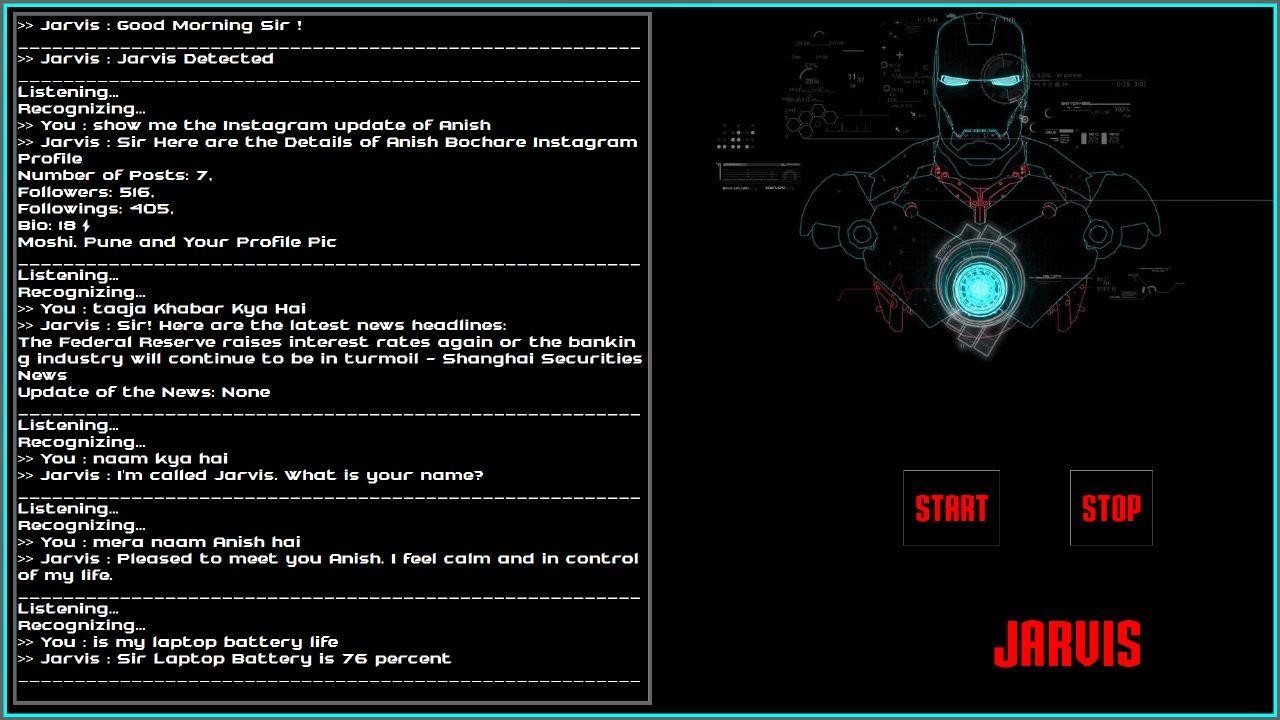


Fig. 6.1.5 News Update Page

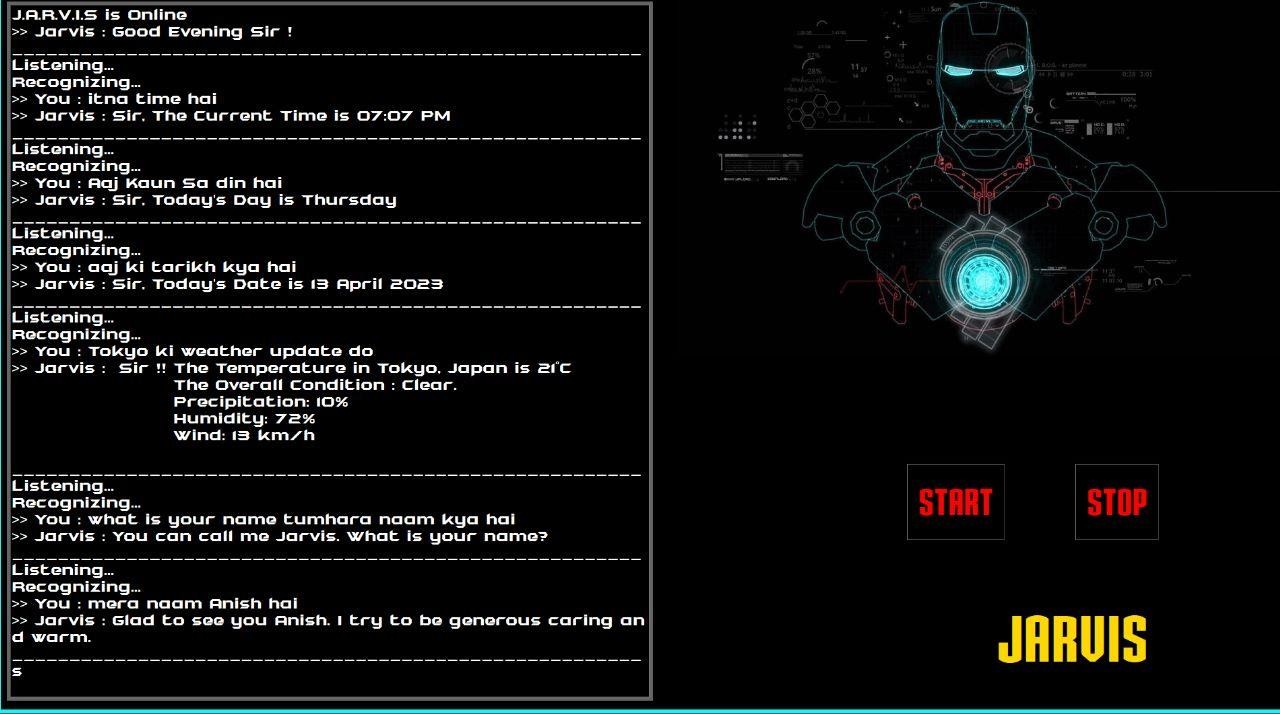


Fig 6.1.6 Time Update Page

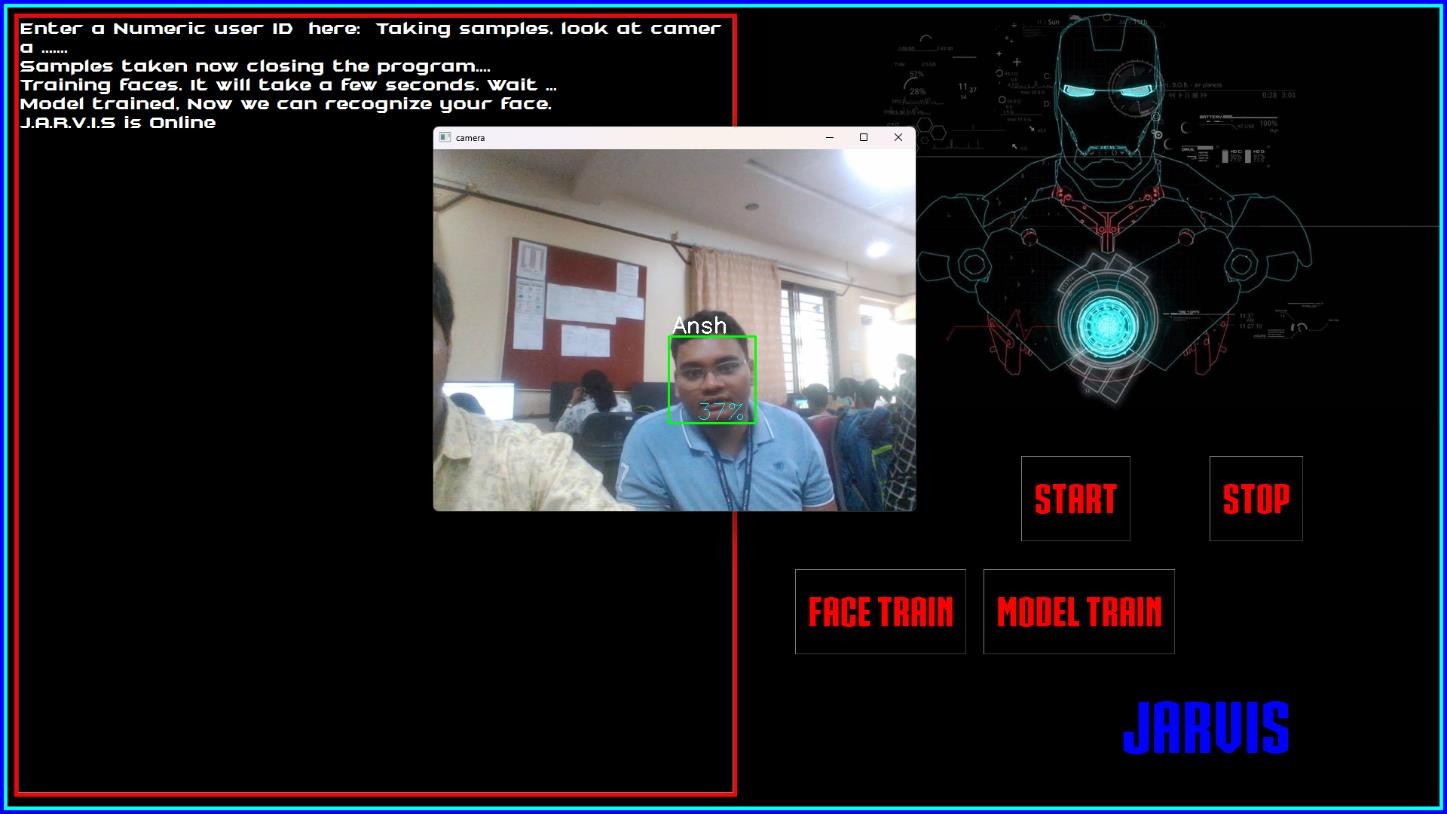


Fig 6.1.7 Face Training Page

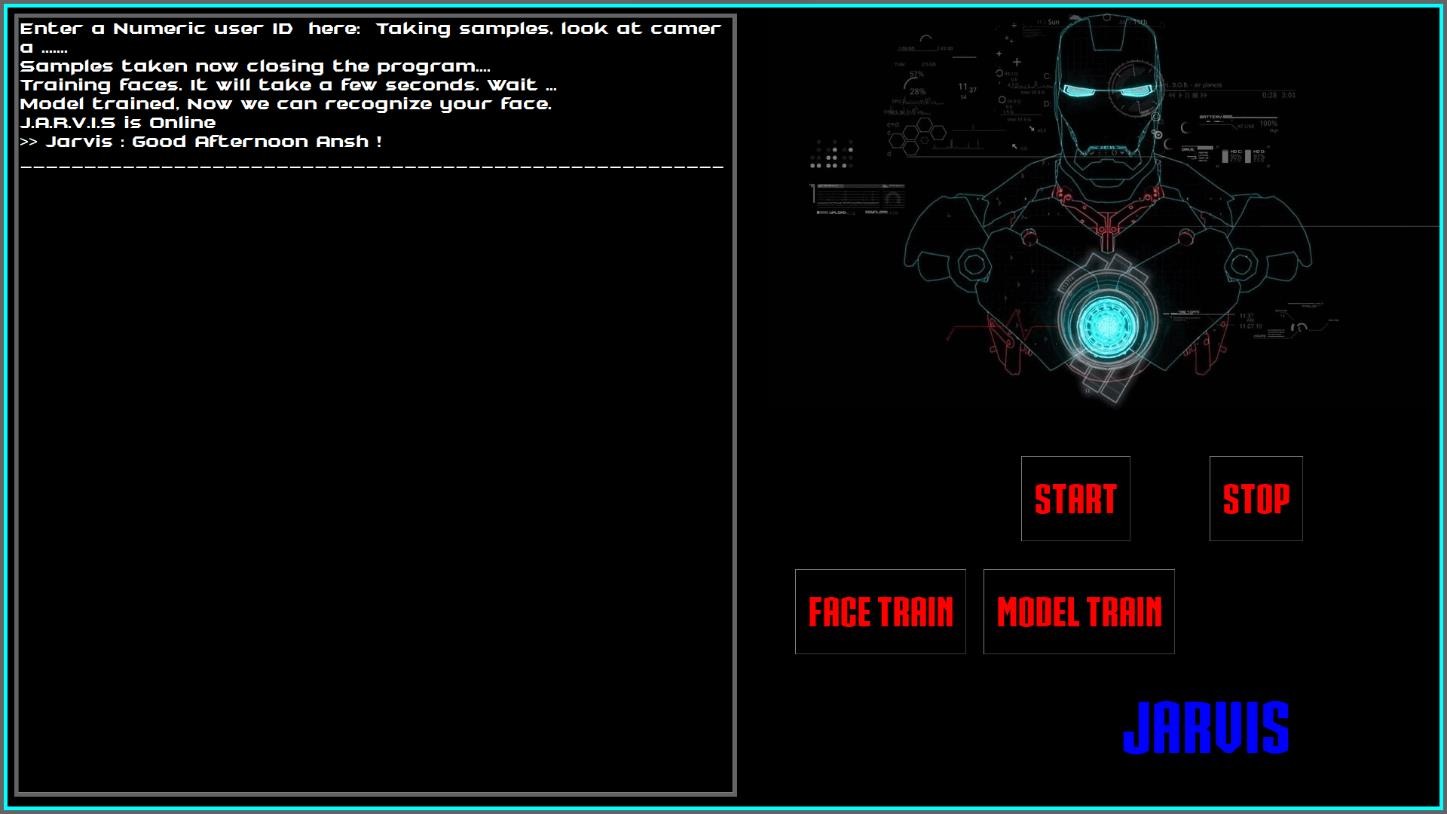


Fig 6.1.8 Greeting After Face Recognition

### 6.3 APPLICATION Customer service

Companies can use voice assistants to automate customer service tasks like answering frequently asked questions, providing information about products or services, and resolving common issues.

### Education

Teachers can use voice assistants to deliver educational content and provide feedback to students, while students can use them to access learning materials and get help with homework.

### Automotive

Voice assistants can control car features like navigation, climate control, and audio systems, making it easier for drivers to keep their hands on the wheel and eyes on the road.

### Business productivity

Voice assistants can help business professionals manage their schedules, send emails, make phone calls and perform other tasks hands free.

### Personal Assistant

Voice assistants can help individuals manage their daily tasks, set reminders, send messages and stay organized.

### 7. CONCLUSION

The project is a great example of how to build a virtual assistant using Python. By integrating various APIs and libraries, it is possible to create a powerful tool that can perform a variety of tasks. In this project, we have seen how to use the speech recognition and text-to-speech conversion libraries to enable the assistant to listen to commands and respond with voice output. We have also explored how to use APIs for various services such as weather, news, and web search. Furthermore, we have discussed how to implement basic natural language processing techniques to understand user input and respond appropriately. We have also shown how to use conditional statements and loops to create a robust decision-making process for the assistant. Overall, the Jarvis Python project is a great starting point for anyone interested in building a virtual assistant. It provides a solid foundation for further customization and expansion, and can be adapted to meet a wide rangeof needs and use cases.

### FUTURE SCOPE

In the future, it is likely that Jarvis will continue to evolve and become even more sophisticated. Here are a few potential scenarios for the future of Jarvis Python: Personalized Recommendations: With access tolarge amounts of data about a user's preferences,Jarvis Python could become better at making personalized recommendations. For example, Jarvis Python could recommend movies or TV shows based on a user's viewing history or suggest recipes based on their dietary preferences. Integration with Business Applications: As more businesses adopt AI and automation technologies, Jarvis Python could be integrated with various business applications to help automate tasks such as scheduling meetings, responding to emails, or even managing inventory. Personalized Recommendations: With access tolarge amounts of data about a user's preferences,Jarvis Python could become better at making personalized recommendations. For example, Jarvis Python could recommend movies or TV shows based on a user's viewing history or suggest recipes based on their dietary preferences. Overall, the possibilities for Jarvis Python are virtually limitless. As technology continues to advance, it will be exciting to see how this virtual assistant evolves and adapts to meet the needs of its use

### REFERENCES

M. L. Mauldin. Chatterbots, tinymuds, and the turing test - entering the loebner prizecompetition In Proceedings of Twelfth National Conference on Artificial Intelligence (AAAI- 94),Seattle, Washington, 1994.

J. Weizenbaum. Eliza - a computer program for the study for the study of natural language communication between man and machine. Commun. ACM, 9(1):36-45, 1966.

R. S. Wallace. Aiml - artificial intelligence markup language, 2006.Accessed on January 05,2019.

J.R.Searle Natural Language Understanding. The Benjamin Cummings Publishing Company, Inc, New York, NY, 1995

1. A. Schegloff. Conversation analysis and communication disorders. In C. Goodwin, editor,Conversation and Brain Damage.Oxford University Press, New York, 2002.

Rabiner Lawrence, Juang Bing-Hwang.Fundamentals of Speech Recognition Prentice Hall, New Jersey, 1993, ISBN 0-13-015157-2

Deller John R., Jr., Hansen John J.L., Proakis John G. ,Discrete-Time Processing of Speech Signals, IEEE Press, ISBN 0-7803-5386-2

Hayes H. Monson,Statistical Digital Signal Processing and Modeling, John Wiley & Sons Inc., Toronto, 1996, ISBN 0-471-59431-8

Proakis John G., Manolakis Dimitris G. ,Digital Signal Processing, principles, algorithms, and applications, Third Edition, Pr entice Hall ,New Jersey, 1996, ISBN 0-13-394338-9

***Virtual assistant Jarvis using python***

Anish Bochare1, Arya Deshpande2, Shruti Ghate3, Ansh Ghodke4, Yogesh J. Gaikwad5 .

Department of Information Technology,

MAEER’s MIT Polytechnic, Pune [anishbochare@gmail.com1](mailto:anishbochare@gmail.com1) [aryadeshpande3004@gmail.com](mailto:aryadeshpande3004@gmail.com) 2,

[shrutighate7@gmail.com](mailto:shrutighate7@gmail.com) 3, [anshghodke29@gmail.com](mailto:anshghodke29@gmail.com) 4, [Yogesh.gaikwad@mitwpu.edu.in](mailto:Yogesh.gaikwad@mitwpu.edu.in)

**Abstract—**

**The Jarvis Python project aims to create a virtual assistant that can perform various tasks through voice commands. The assistant will use speech recognition and natural language processing to understand user requests and respond with appropriate actions. The project will involve integrating various APIs, such as weather and news, to provide real-time information to the user. The assistant will also be capable of setting reminders, managing to-do lists. The ultimate goal of the Jarvis Python project is to create a reliable and efficient virtual assistant that can simplify daily tasks and enhance the user experience.**

**The project will involve integrating third-party APIs and services to enable the assistant to perform a wide range of tasks, such as playing music, setting alarms, and controlling smart home devices. The assistant will also be customizable, allowing users to add new features and functionality based on their specific needs. The ultimate goal of the Jarvis Python project is to create a powerful and versatile virtual assistant that can improve productivity and enhance the user experience.**

* 1. INTRODUCTION

Jarvis is a fictional character in the Marvel Cinematic Universe, who is the artificial

intelligence system created by Tony Stark, also known as Iron Man. Jarvis serves as a loyal assistant and friend to Tony Stark, providing him with information, managing his technology and helping him in various tasks. Jarvis is an acronym for "Just A Rather Very Intelligent System", Although Jarvis is a fictional character, the concept of artificial intelligence systems like him is becoming increasingly prevalent in our daily lives. Virtual assistants such as Siri, Alexa, and Google Assistant, are examples of AI systems that can assist with tasks and provide information. JARVIS, standing for "Just A Really Very Intelligent System," is an acronym. Jarvis is a popular virtual assistant built using Python programming language. It allows users to interact with their computer using voice commands and perform various tasks such as opening applications, browsing the web, sending emails, and more.

There are several libraries available in Python that can be used to build Jarvis you

need to have a basic understanding of Python and its libraries. Once you have set up the required libraries, you can start building the assistant by defining functions to perform specific actions and integrating them with the speech recognition and text- to-speech conversion modules.

Overall, Jarvis is a fun project that allows you to explore the capabilities of AI and voice recognition technology.

* 1. OVERVIEW OF AI

Jarvis is a popular open-source virtual assistant program written in Python that uses AI to provide various services, such as voice-activated commands, scheduling tasks, and more.

In the context of Jarvis, AI is used to enable natural language processing (NLP) and speech recognition capabilities, which allows users to communicate with the program using spoken or written commands. The program uses machine learning algorithms to continuously improve its ability to understand and respond to user inputs, making it more accurate and efficient over time.

* 1. LITERATURE SURVEY

First paper for literature check is “Artificial Intelligence- grounded Voice Assistant” by Subhash, Prajwal N Srivatsa, S Siddesh, A Ullas, B Santhosh. We've added a lot to our design in comparison to other backing. The fact that it's a hands-free operation makes it relatively useful in diurnal life. It's a remarkably simple operation. It is also used in business settings, similar as laboratories where workers must class while wearing body suits and gloves for protection. Yet, they may pierce any information by using a voice adjunct, which makes their job easier.

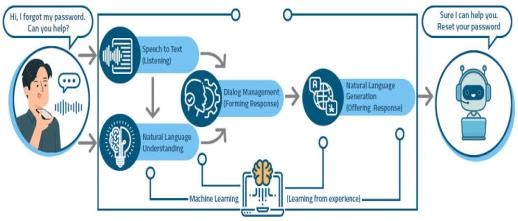
Alternate paper for literature check is “Short exploration on Voice Control System Grounded on Artificial Intelligence Assistant” by Tae- Kook Kim. This paper describes the design of a voice control system that makes use of bedded bias and open API AIs. The suggested system is a voice command system that uses a stoner's spoken instructions to spark modules Using the useful speech recognition interface and numerous modules, this system enables druggies to construct their own system.

Third paper for literature check is “Next- generation of virtual particular sidekicks Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home ” by Veton Këpuska, Gamal Bohouta. This offer introduces the structure of the Coming-

Generation of Virtual particular sidekicks, which is a new VPAs system with a coherent structure designed to discourse with a mortal. In both the input and affair channels, this VPAs system communicated using speech, plates, videotape, etc. In addition, the VPAs system will be used to ameliorate stoner- computer commerce by utilising technologies similar as image/ videotape recognition, speech recognition, and the Knowledge Base.

* 1. ARCHITECTURE

Jarvis is a popular virtual assistant that can perform a variety of tasks such as answering questions, reminders. The architecture of Jarvis can vary depending on how it is implemented, but a common architecture might involve the following components:

Wake word detection: Jarvis needs to be able to listen for a wake word (such as "Hey Jarvis" or "OK Google") that signals that the user wants to interact with the virtual assistant. This component typically involves audio processing and machine learning algorithms to recognize the wake word.

**System Architecture**

Initially, the condition here is whether or not Jarvis voice assistant is active; if it is active, it requests user input; otherwise, make Jarvis active (make it on). The user then provides input in the form of speech or text; if the input is in text, it proceeds to the action to be taken or the skill to be executed; if the input is in speech, it uses the speech recognition feature to convert it to text and proceeds to the action.

Now, if the skills to be executed are adequate to Jarvis, it responds positively to the user in the form of speech and then executes the commands for operations.

1. *Speech recognition*

After the wake word is detected, Jarvis needs to be able to recognize the user's speech and convert it into text. This component also typically involves audio processing and machine learning algorithms, such as deep neural networks, to perform the speech recognition.

1. *Natural language understanding*

Once the user's speech is converted into text, Jarvis needs to be able to understand the user's intent and extract relevant information from the user's request. This component typically involves natural language processing techniques such as named entity recognition and sentiment analysis.

1. *Dialog management*

After understanding the user's intent, Jarvis needs to be able to manage a dialog with the user to gather any additional information needed to fulfill the user's request. This component typically involves a rule-based or machine learning-based approach to dialog management.

1. *Task fulfillment*

Once Jarvis has gathered all the necessary information from the user, it needs to be able to perform the requested task. This could involve calling APIs for external services, controlling smart home devices, or simply providing information to the user.

1. *Response generation*

Finally, Jarvis needs to be able to generate a response to the user, either as speech or as text. This component typically involves natural language generation techniques such as text-to-speech synthesis or template-based text generation.

Overall, the architecture of Jarvis is a complex system that requires expertise in several areas, including audio processing, natural language processing, and software engineering

* + PYTHON LIBRARIES

In JARVIS following python libraries were used:

pyttsx3: to convert text to speech this library is used.

Speech Recognition: to convert speech to text this library is used.

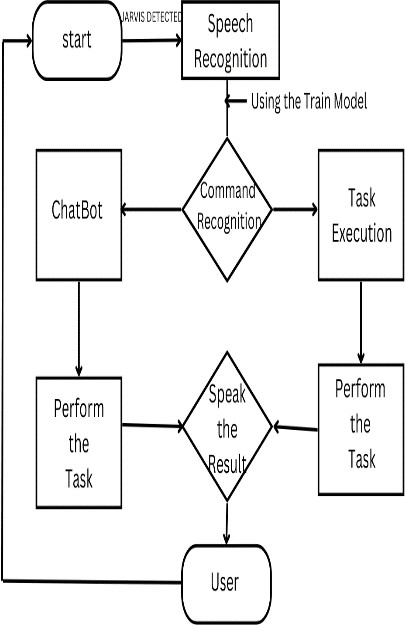
pywhatkit: To launch WhatsApp communication with required features. Datetime: System date and time is used with the help of this library.

Smtplib: SMTP protocol is used send mails and route mails between mail servers. pyPDF2:This module is used for reading, writing of pdf files.

Pyjokes: It’s a python libararies which contains lots of intriguing jokes in it. Webbrowser: It provides interface for displaying web-grounded documents to druggies.

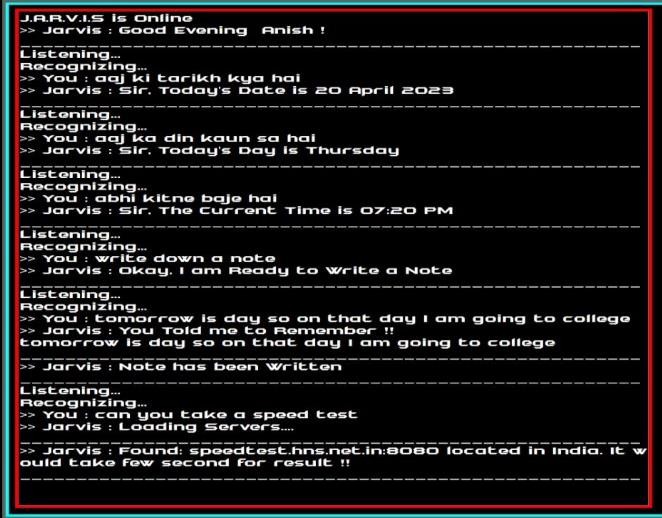
Pyautogui: It’s a python libraries for graphical stoner interface.

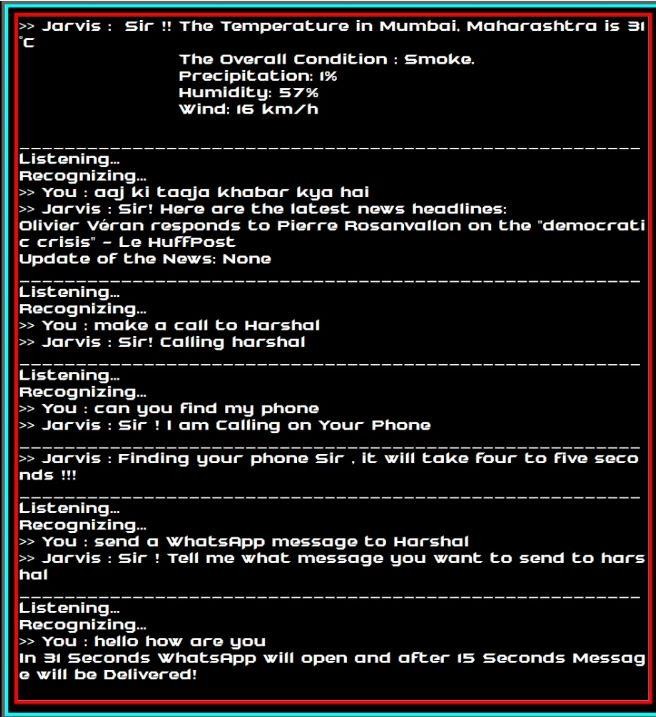
* 1. FLOWCHAT



* 1. RESULT







1. FUTURE SCENARIO

In the future, it is likely that Jarvis will continue to evolve and become even more sophisticated. Here are a few potential scenarios for the future of Jarvis Python: Personalized Recommendations: With access to large amounts of data about a user's preferences, Jarvis Python could become better at making personalized recommendations. For example, Jarvis Python could recommend movies or TV shows based on a user's viewing history or suggest recipes based on their dietary preferences.

Integration with Business Applications: As more businesses adopt AI and automation technologies, Jarvis Python could be integrated with various business applications to help automate tasks such as scheduling meetings, responding to emails, or even managing inventory.

Enhanced Natural Language Processing: With advances in natural language processing (NLP) and machine learning, Jarvis Python could become even better at understanding and responding to natural language queries. This could make it easier for users to communicate with Jarvis Python and get the information they need more quickly and accurately.

Overall, the possibilities for Jarvis Python are virtually limitless. As technology continues to advance, it will be exciting to see how this virtual assistant evolves and adapts to meet the needs of its use

1. CHALLENGES

Despite the potential for growth and development, Jarvis Python also faces several challenges. Here are a few examples:

Privacy Concerns: Jarvis Python will likely need to collect a significant amount of personal data from users in order to function effectively. This could raise privacy concerns among some users, particularly if their personal information is compromised in a security breach.

Language and Cultural Barriers: Jarvis Python will need to be able to understand and respond to queries in multiple languages and dialects in order to be useful on a global scale. Additionally, cultural differences mayrequire adaptations in order to ensure that the technology is culturally appropriate and sensitive.

Ethical Considerations: As with any AI- based technology, there are ethical considerations surrounding the use of Jarvis Python. For example, how will the technology be used in hiring decisions or in determining creditworthiness? These issues will need to be addressed in order to ensure

that Jarvis Python is used fairly and responsibly.

VIII FEATURES OF JARVIS

Clap Detection to Start: Clap detection is a popular method of starting devices such as lights or music players by simply clapping. To implement this feature, you would need to use a microphone to detect the sound of a clap, and then use a program or script to trigger an action based on that sound. You could use a library such as PyAudio in Python to capture the sound, and then use a simple algorithm to detect the sound of a clap. Once the clap is detected, you can use another script or program to perform the desired action, such as turning on a light or playing music.

Jarvis Detection: Jarvis is a popular AI assistant that is used to automate tasks and provide assistance to users. To create a Jarvis detection system, you would need to use a voice recognition library such as SpeechRecognition in Python to capture and analyze the user's speech. You could then use a Natural Language Processing (NLP) library such as NLTK or spaCy to understand the user's intent and determine what action should be taken. Once the user's intent is understood, you can use another script or program to perform the desired action, such as setting a reminder or sending

an email.

Speed Test: A speed test is a useful tool for measuring the speed of an internet connection. To create a speed test, you would need to use a network testing library such as speedtest-cli in Python to measure the download and upload speeds of the internet connection. You could then use a simple algorithm to calculate the average speed, and display the results to the user.

Notepad Creation: Notepad is a simple text editor that is used to create and edit text files. To create a notepad in Python, you could use the built-in tkinter library to create a graphical user interface (GUI) that allows the user to enter and edit text. You could then use another script or program to save the text entered by the user to a file.

Phone Control for Jarvis Python: To create a phone control system for Jarvis, you would need to use a library such as pybluez in Python to communicate with the phone via Bluetooth. You could then use a simple algorithm to detect the user's voice commands and perform the desired action on the phone, such as making a call or sending a text message. You could also use a GUI library such as tkinter to create a

graphical user interface that allows the user to interact with the phone using buttons and other controls.

IX CONCLUSION

The project is a great example of how to build a virtual assistant using Python. By integrating various APIs and libraries, it is possible to create a powerful tool that can perform a variety of tasks.

In this project, we have seen how to use the speech recognition and text-to- speech conversion libraries to enable the assistant to listen to commands and respond with voice output. We have also explored how to use APIs for various services such as weather, news, and web search.

Furthermore, we have discussed how to implement basic natural language processing techniques to understand user input and respond appropriately. We have also shown how to use conditional statements and loops to create a robust decision-making process for the assistant.

Overall, the Jarvis Python project is a great starting point for anyone interested in building a virtual assistant. It provides a solid foundation for further customization and expansion, and can be adapted to meet a wide range of needs and use cases.

X REFERENCES

1.N. H. Tandel, H. B. Prajapati and V. K. Dabhi, "Voice Recognition and Voice Comparison using Machine Learning Techniques: A Survey," 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2020, pp. 459-465, doi: 10.1109/ICACCS48705.2020.9074184.

2.R. Sangpal, T. Gawand, S. Vaykar and N. Madhavi, "JARVIS: An interpretation of AIML with integration of gTTS and Python," 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), Kannur, India,2019,pp.486-489,doi:10.1109/ICICICT46008.2019.8993344.

3.V. Këpuska and G. Bohouta, "Next-generation of virtual personal assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home)," 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2018, pp. 99-103, doi: 10.1109/CCWC.2018.8301638.

4.K. N., R. V., S. S. S. and D. R., "Intelligent Personal Assistant - Implementing Voice Commands enabling Speech Recognition," 2020 International Conference on System, Computation, Automation and Networking (ICSCAN), Pondicherry, India, 2020, pp. 1-5, doi: 10.1109/ICSCAN49426.2020.9262279.

5.S. Subhash, P. N. Srivatsa, S. Siddesh, A. Ullas and B. Santhosh, "Artificial Intelligence-based Voice Assistant," 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), London, UK, 2020, pp. 593-596, doi: 10.1109/WorldS450073.2020.9210344.

1. Harshit Agrawal, Nivedita Singh, Gaurav Kumar, Dr. Diwakar Yagyasen, Mr. Surya Vikram Singh. "Voice Assistant Using Python" An International Open Access-revied, Refereed Journal.Unique Paper ID: 152099, Publication Volume & Issue: Volume 8, Issue 2, Page(s): 419-423
2. Deepak Shende. Ria Umabiya, Monika Raghorte, Aishwarya Bhisikar. Anup Bhange. "Al Based Voice Assistant Using Python", International Journal of Emerging Technologies and Innovative Research (www.jetir.org), ISSN 2349-5162, Vol.6, Issue 2, page no.506-509, February-2019.

8.[4]. Tulshan, Amrita & Dhage, Sudhir. (2019). “Survey on Virtual Assistant: Google Assistant, Siri, Cortana, Alexa”, 4th International Symposium SIRS 2018, Bangalore, India, September 19–22, 2018, Revised Selected Papers. 10.1007/978-981-13-5758-9\_17.[5]. Dr. Ksh

9 T. -K. Kim, "Short Research on Voice Control System Based on Artificial Intelligence Assistant," 2020 International Conference on Electronics, Information, and Communication (ICEIC), Barcelona, Spain, 2020, pp. 1-2, doi: 10.1109/ICEIC49074.2020.9051160.

**Annexure A**

# Certificate

This is to certify that Mr. Anish Atmaram Bochare

From MIT, Polytechnic, Pune Institute having Enrolment No. 2001480142

Has completed project of final year having title Virtual Assistant Jarvis*.* During the

academic year 2022-2023. The project completed by individually in a group

consisting of 4 person under the guidance of the Faculty Guide.

## …………………………...

Name and Signature of Guide **Prof. Y. J. Gaikwad**

**Annexure A**

# Certificate

This is to certify that Mr. Ansh Pradeep Ghodke

From MIT, Polytechnic, Pune Institute having Enrolment No. 2001480253

Has completed project of final year having title Virtual Assistant Jarvis*.* During the

academic year 2022-2023. The project completed by individually in a group

consisting of 4 person under the guidance of the Faculty Guide.

## …………………………...

Name and Signature of Guide **Prof. Y. J. Gaikwad**

**Annexure A**

# Certificate

This is to certify that Ms. Shruti Ravindra Ghate

From MIT, Polytechnic, Pune Institute having Enrolment No. 2001480149

Has completed project of final year having title Virtual Assistant Jarvis*.* During the

academic year 2022-2023. The project completed by individually in a group

consisting of 4 person under the guidance of the Faculty Guide.

## …………………………...

Name and Signature of Guide **Prof. Y. J. Gaikwad**

**Annexure A**

# Certificate

This is to certify that Ms. Arya Prasad Deshpande

From MIT, Polytechnic, Pune Institute having Enrolment No. 2001480145

Has completed project of final year having title Virtual Assistant Jarvis*.* During the

academic year 2022-2023. The project completed by individually in a group

consisting of 4 person under the guidance of the Faculty Guide.

## …………………………...

Name and Signature of Guide **Prof. Y. J. Gaikwad**