“Jarvis -Desktop assistant using python”

A

Project Report

Submitted in partial fulfillment of the

requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

By

Name	Roll No.
Anshika Sharma	R610217003
Divyanshu Singh	R610217007
Kaustavdeep Goswami	R610217010

Under the guidance of

Mr. Sumit Kumar
Assistant Professor, Department of Systemics

Department of Computer Science & Engineering

Centre for Information Technology

University of Petroleum & Energy Studies

Bidholi, Via Prem Nagar, Dehradun, UK

January - May, 2021

CANDIDATE'S DECLARATION

I/We hereby certify that the project work entitled **"Jarvis -Desktop assistant using python"** in partial fulfilment of the requirements for the award of the Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with specialization in **Mainframe Technology** and submitted to the Department of Computer Science & Engineering at Center for Information Technology, University of Petroleum & Energy Studies, Dehradun, is an authentic record of my/ our work carried out during a period from **January, 2021 to May, 2021** under the supervision of **Mr. Sumit Kumar, Assistant Professor, Department of Systemic**

The matter presented in this project has not been submitted by me/ us for the award of any other degree of this or any other University.

(Anshika Sharma)
Roll No. R610217003
(Divyanshu Singh)
Roll No. R610217007
(Kaustavdeep Goswami)
Roll No. R610217010

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: 07/05/2021

(**Mr. Sumit Kumar)**
Project Guide

Dr. Neelu Jyoti Ahuja
HOD, Department of Systemics
Center for Information Technology
University of Petroleum & Energy Studies
Dehradun – 248 001 (Uttarakhand)

ACKNOWLEDGEMENT

We wish to express our deep gratitude to our guide **Mr. Sumit Kumar**, for all advice, encouragement and constant support he has given us throughout our project work. This work would not have been possible without his support and valuable suggestions.

We sincerely thank to our respected Program Head of the Department, **Dr.Neelu Jyoti Ahuja**, for her great support in doing our project in artificial intelligence at **SoCS**.

We are also grateful to **Dr. Priyadarsan Patra, Dean SoCS**, and UPES for giving us the necessary facilities to carry out our project work successfully.

We would like to thank all our **friends** for their help and constructive criticism during our project work. Finally, we have no words to express our sincere gratitude to our **parents** who have shown us this world and for every support they have given us.

Name	**Anshika Sharma**	**Divyanshu Singh**	**Kaustavdeep Goswami**
Roll No.	**R610217003**	**R610217007**	**R610217010**

ABSTRACT

Have you ever dreamed of having your own personal AI assistant? Consider how much easier it will be to send emails without typing a single letter, to conduct Wikipedia searches without opening a window, and to carry out a variety of other everyday activities such as playing music with just a single voice order. Personal digital assistants have recently gotten a lot of coverage. Most commercial websites use chatbots. Training robots to handle day-to-day activities has become the standard, thanks to advances in artificial intelligence. In this age of smart homes and smart devices, voice-based personal assistants have grown in popularity. These personal assistants can be conveniently programmed to perform a variety of routine tasks using only voice commands. Google also popularized voice-based search, which is a blessing for those who are unable to use a keypad or keyboard, such as senior citizens.

Keywords: - Artificial intelligence, pyttsx3, speech recognition.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 History

A desktop assistant is a virtual assistant that can perform the different tasks without any physical intervention with keyboard and mouse. It may use facial expressions and hand gestures for performing different tasks. Different functions can be controlled using similar methods.

The main aim of creating a desktop assistant is to enhance and enlighten the user-machine experience. The no-hands technology is the backbone of this project and idea.

The desktop assistant takes in basic instructions from the user in the form of raw natural language, converts this language into machine language and passes the instruction to the system further. This may be done with the help of gesture recognition, facial recognition, sound recognition etc. The diversity of this project has a wide range, the assistant if trained accordingly can virtually control the entire system. Without receiving any physical interruptions (instructions) from the user.

A desktop assistant is a program developed that basically performs the scheduled tasks in added windows task scheduler in order to launch at various scheduled times.

1.2 System Requirements

1.2.1 Software Requirements

Name of component	Specification
Operating System	Windows 7 and above, Linux
Compiler	Anaconda

Table1.2.1: List of software requirements

1.2.2 Hardware Requirements

Name of component	Specification
Processor	Processor with speed of 500MHz
RAM	128 MB
Hard Disk	512 GB

Table 1.2.2: List of hardware requirements

1.3 Problem Statement

The primary interaction of a user starts with the desktop of a system. The variation and variety have a wide range to discover from. A desktop assistant can be used to perform all the functions which require human involvement by following simple instructions. An automated and fully functional desktop assistant is also acquitted to enhance the user machine interaction.

1.4 Objectives

Objective: To create Desktop assistant

- The objective is to build a desktop assistant that can perform all the basic functions without thephysical involvement of the user.

- Speech to text recognition

2. SYSTEM ANALYSIS

2.1 Motivation

In the current days, People are agitated by typing the commands into the computer. Be it procrastination or a busy schedule. Typing is a big obsolete process. The appropriate solution to this is that we need to switch over to a desktop assistant which can understand us and does the initial work for us. An assistant is now becoming the best replacement for typing commands. To develop such kind of an assistant, python is a suitable language. It provides us with speech recognition API that allows us to convert audio into text for further processing. An end user can perform multiple tasks through audio commands, leaving behind the traditional text command environment to get responses.

2.2 Proposed System

In this project, we are developing a desktop assistant using python which will help getting the useful information about the system and perform some important tasks such as searching something in the web browser, playing song, opening Youtube, searching something in Wikipedia or sending an email etc. This will act as voice assistant through which a user can give audio commands to get the required information or to get some tasks performed.

System working is mainly divided into following parts:

- User's command.
- Acknowledgement by the system.
- Giving respond to the user.

The command can be given to the assistant by means of sound and if included by means of gesture and action. Once the command is recognized the program processes the natural language using specific python libraries and proceeds with the given command for execution. The assistant will be able to perform all the trivial actions at par minimum necessary. The assistant can also be called using name; no physical action is needed to access the assistant for the first time.

3. DESIGN

3.1 Pert-chart

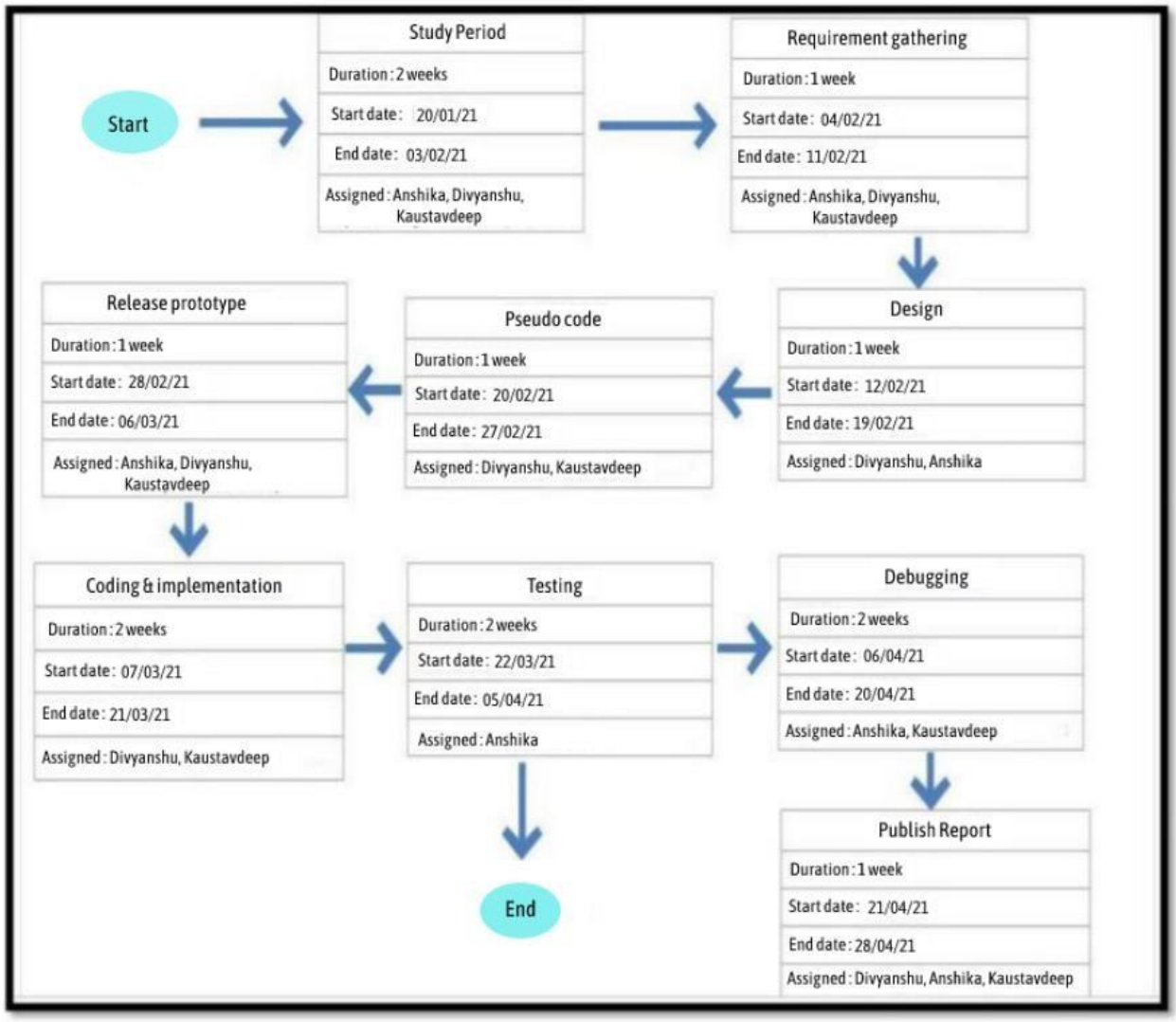

Fig. 3.1: Pert Chart diagram

3.2 Use Case Diagram

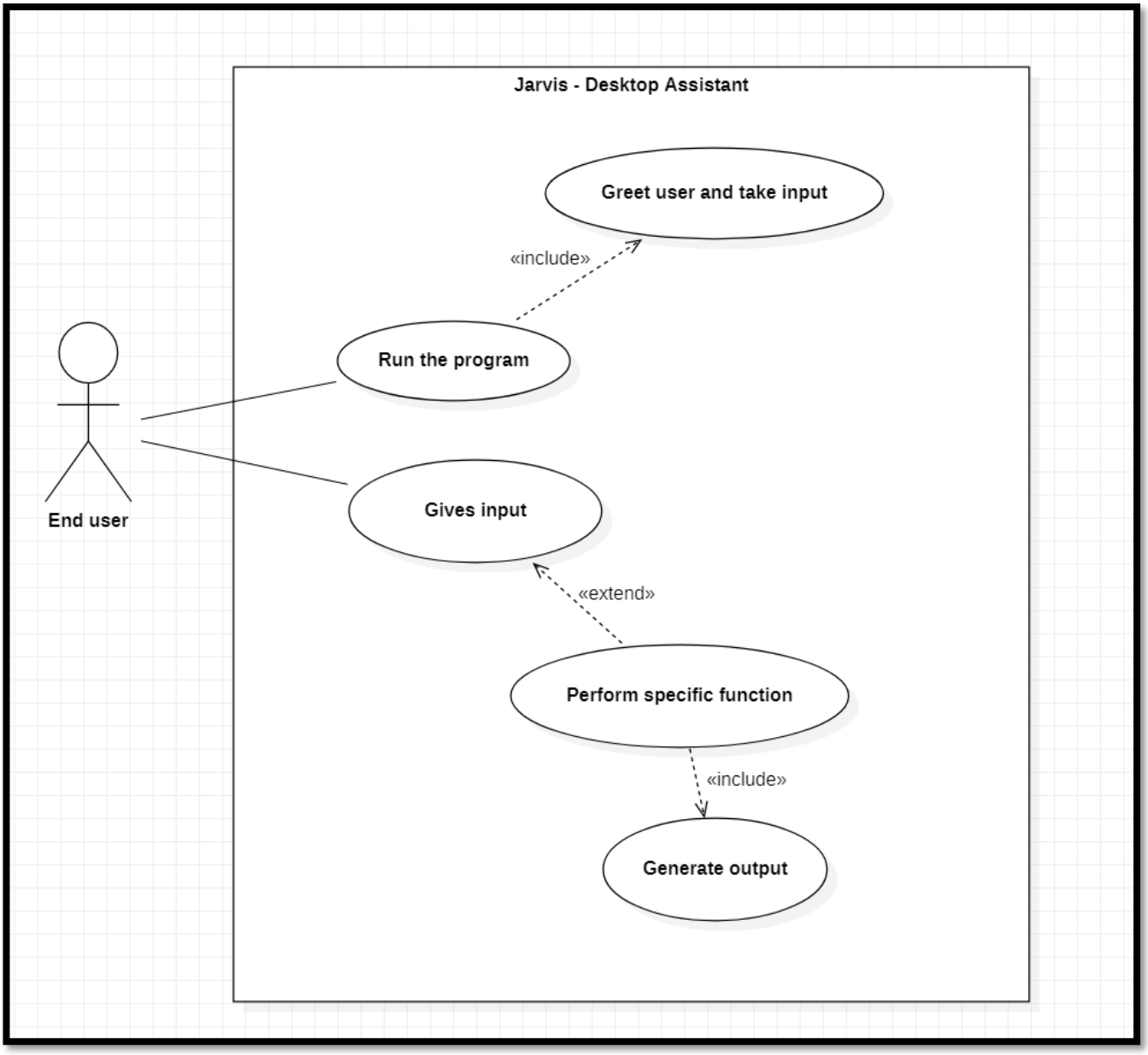

Fig. 3.2: Use case diagram

3.3 Dataflow diagram

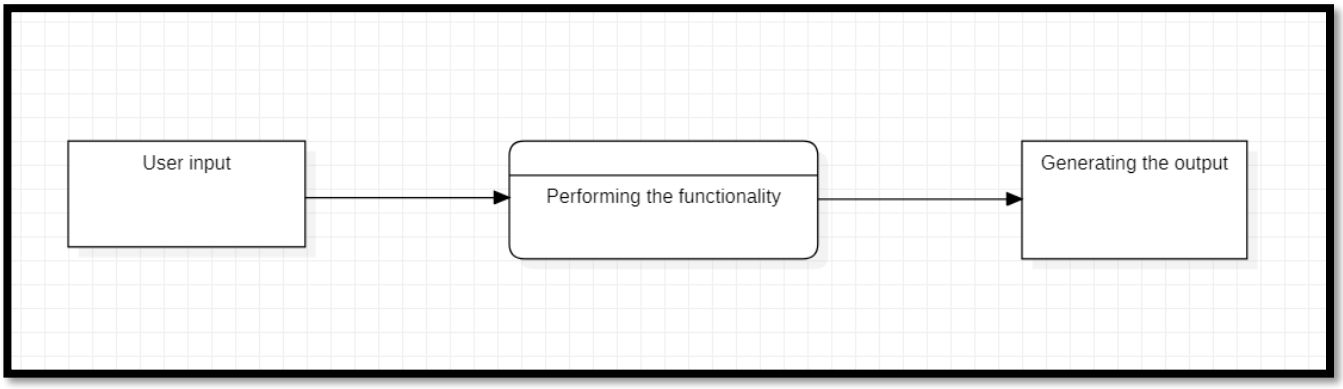

Fig. 3.3.1: Level 0 dataflow diagram

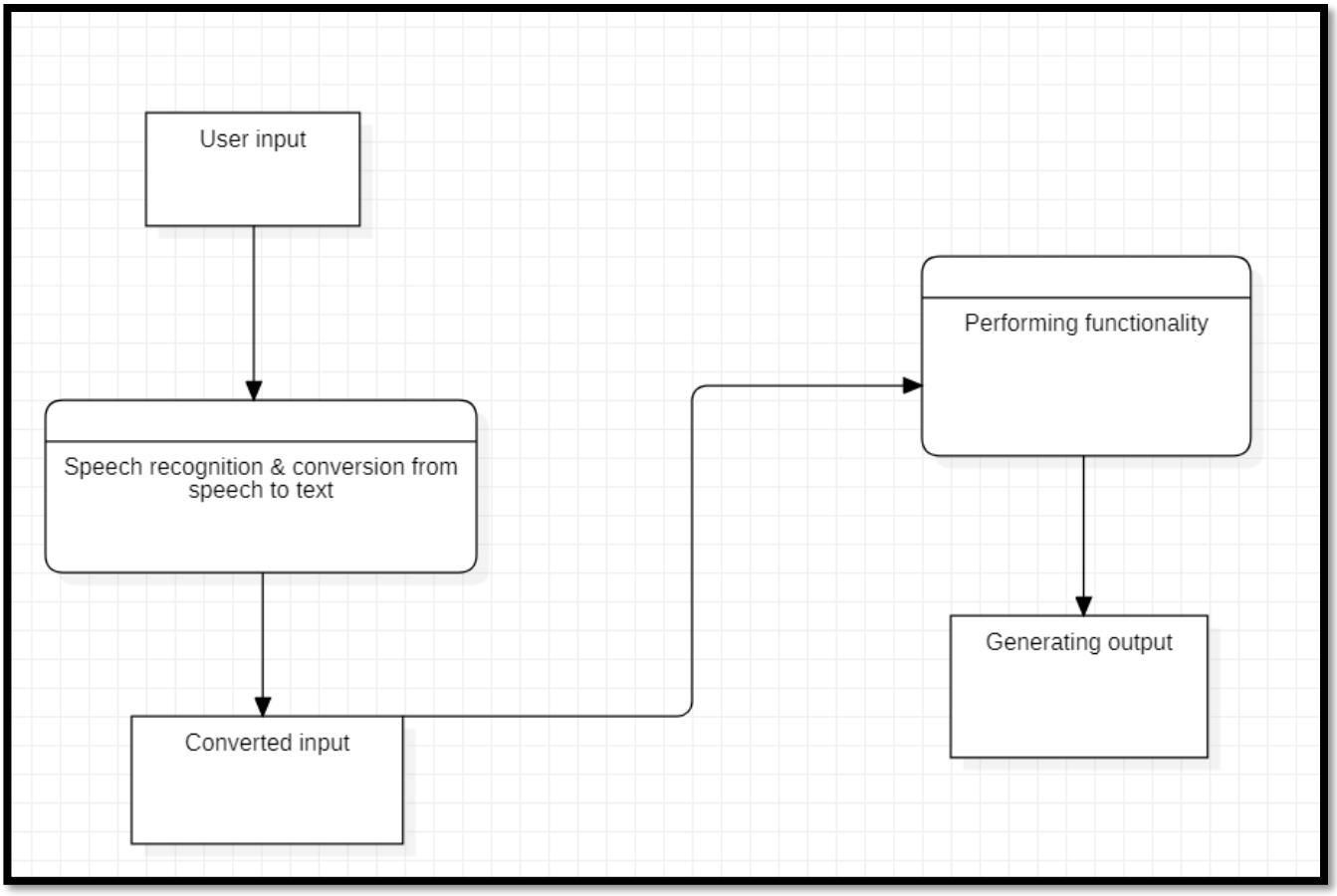

Fig. 3.3.2: Level 1 dataflow diagram

3.4 Activity diagram

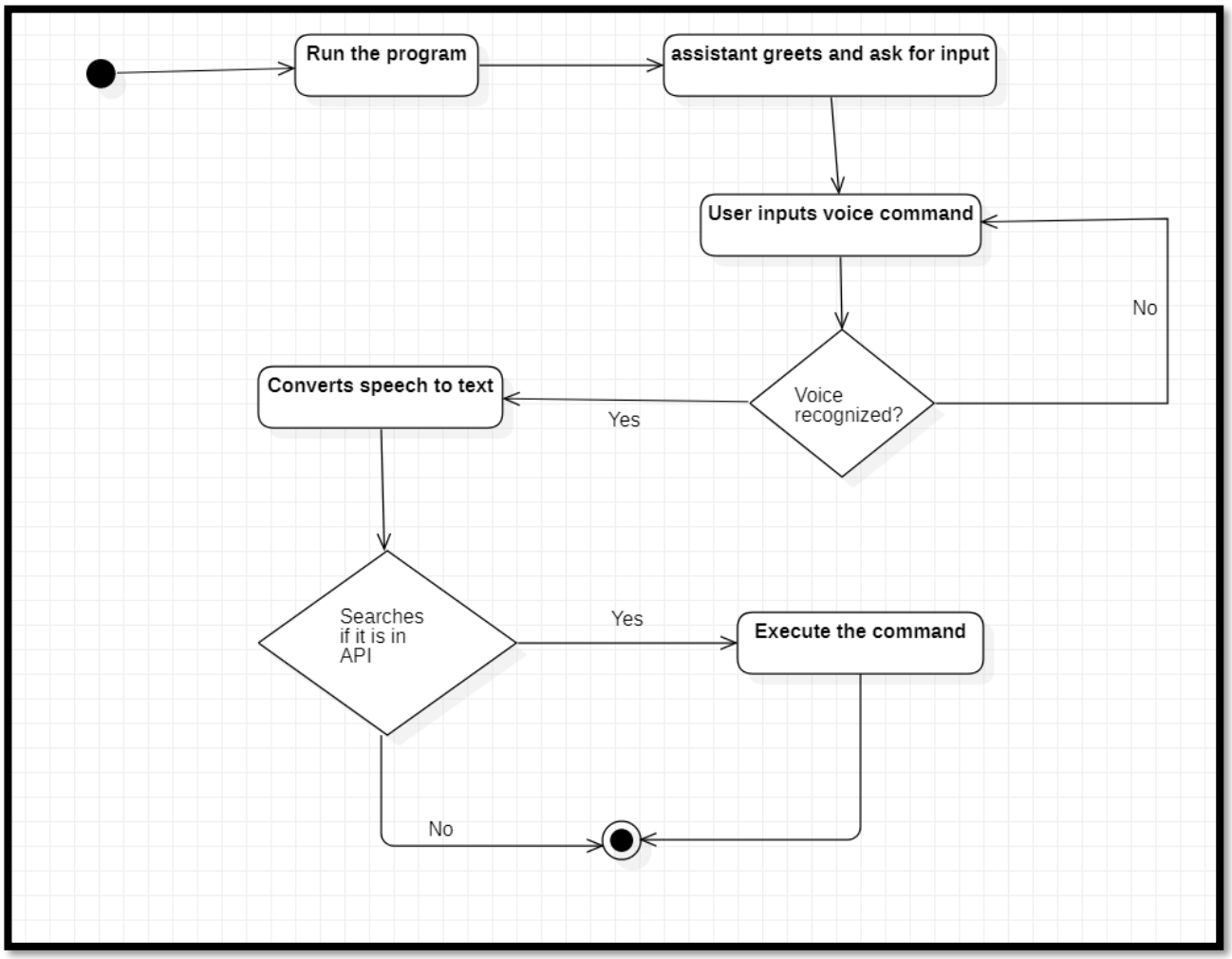

Fig. 3.4: Activity diagram

4. IMPLEMENTATION

4.1 Basic Concepts

VOICE RECOGNITION SYSTEM

SPEECH-TO-TEXT is a piece of software that allows the user to control computer functions while also dictating text. The system is made up of two parts: the first is for processing acoustic signals captured by a microphone, and the second is for interpreting the processed signals and then mapping them to words. [1]

The methods for speech-to-text and speech-to-speech automatic summarization based on speech unit extraction and concatenation are presented in this project. A two-stage summarization procedure which consists of significant sentence extraction and the word-based sentence compaction is basically studied for the former case. Sentence and word units are extracted from speech recognition results and concatenated to produce summaries that maximize the weighted sum of linguistic likelihood, amount of information, confidence measure, and grammatical likelihood of concatenated units. [2]

The distinction between voiced and unvoiced sounds in speech provides a valuable foundation for subsequent processing. Silence/unvoiced/voiced classification expands the range of tasks that can be processed further, such as stop consonant identification and endpoint detection for isolated utterances. End point detection removes unwanted signals and background noise from speech samples in a noisy environment. The short-term log energy and short-term zero crossing rate are used to find end points. [3]

4.2 Library

1. pyttsx3

pyttsx3 is a Python-based text-to-speech conversion library. It operates offline, unlike other libraries, and is also compatible with Python 2 and 3.

Installation: pip install pyttsx3

An application calls the pyttsx3.init() factory function so that it gets a reference to a pyttsx3

2. speech_recognition

Speech recognition library, that is supported for several engines and APIs, works both online and offline.

Speech recognition engine/API support:

- Google Cloud Speech API
- Google Speech Recognition
- CMU Sphinx (works offline)
- Houndify API
- IBM Speech to Text
- Microsoft Bing Voice Recognition
- Snowboy Hotword Detection (works offline)

3. Datetime

Although a date is not a data type in Python, we can use the datetime module to work with dates as date objects.

Date and time manipulation classes are provided by the datetime module. Although date and time calculations are provided, the implementation's primary focus is on efficient attribute extraction for output formatting and manipulation.

4. Pytz

Pytz combines the Olson tz database with Python, allowing it to support nearly all time zones. This module provides date-to-time transfer functionality and assists users in supporting a global client base. It helps us to calculate timezones in our Python applications and build datetime instances that are aware of timezones.

Installation: Using command line:

pip install pytz

5. webbrowser

The webbrowser module in Python offers a high-level interface that enables users to view Web-based documents.
As shown below, the webbrowser module can be used to open a browser in a platform-independent manner:

```
import webbrowser
webbrowser.open('http://www.google.com')
```

6. wikipedia

Wikipedia is a Python library that allows users to easily access and parse Wikipedia data.

Check Wikipedia, get article summaries, extract data from a page such as links and images, and more. Wikipedia wraps the MediaWiki API so you can concentrate on using rather than having Wikipedia info.

7. pyautogui

PyAutoGUI is a cross-platform Python module for human GUI automation. The mouse and keyboard can be controlled programmatically.

In the windows of other applications, shift the mouse and click or sort.

8. Requests

Requests is a lightweight HTTP library that is both simple and elegant. Requests makes it very easy to submit HTTP requests. There's no need to add query strings to your URLs by hand.

The response data from an HTTP request is returned as a Response Item (content, encoding, status, etc).

9. pyjokes

Provides with one line jokes for programmers.

10. cv2

The OpenCV-Python library is a compilation of Python bindings for solving computer vision problems. CV2 is a cross-platform library that allows us to build real-time computer vision apps. It focuses primarily on image processing, video recording, and analysis, with features such as face detection and object detection.

11. Wolframalpha

Wolfram Alpha is nothing but a computational search engine which tends to evaluate what the user asks.

How can we use Wolfram Alpha in Python?

Getting API Id -

- Firstly, create an account at Wolfram alpha.
- After that, sign in using your Wolfram ID.
- Now you will be seeing at the homepage of that website.
- Click on the Get an AppID button so that you can get the id.
- In the next dialog box, provide the app with a suitable name and its description.
- Note down the APPID which appears in the next dialog box

Wolfram Alpha is an API that uses Wolfram's algorithms, knowledgebase, and AI technologies to compute expert-level answers. The Wolfram Language makes this possible.

12. googletrans

Googletrans is a Python library that implements the Google Translate API and is available for free and unlimited use. This makes calls to methods like detect and translate using the Google Translate Ajax API.

13. gTTS

Google Text-to-Speech or gTTS is basically a Python library and command-line tool which is used for communicating with the Google Translate's text-to-speech API.

14. psutil

psutil or ‘process and system utilities’ is also a Python library which actually retrieves information about the running processes and system usage (CPU, disk, memory, network, and sensors). It is primarily used for system control, profiling, and restricting process resources, as well as process management.

15. Os

In Python, the OS module has functions for interacting with the operating system. Python's standard utility modules involve OS. This module allows us to use the OS dependent functionalities on the go. Many functions that need to communicate with the file system are also included in the *os* and *os.path* modules.

16. smtplib

The smtplib module in Python give a description of an SMTP client session object that can be used to send mail to any computer, available on the Internet having an SMTP or ESMTP listener daemon.

17. winshell

The winshell module is nothing but a lightweight wrapper for the Windows shell. It has some support for the hierarchical storage along with convenience functions for accessing the special files and utilizing the shell's file copy, rename and delete features.

4.3 ALGORITHM AND PSEUDOCODE

1. Starting the program
2. Import the required libraries – pyttsx3, speech_recognition, Wikipedia, webbrowser, datetime etc.
3. Define the speak () function to take user's audio as an argument.
4. Create the main () function to call the speak () function.
5. Define a function wishme () that will be used to greet the user.
6. Define take command function. Through this function, the assistant can return a string output by taking voice commands as input from user.
7. Execute the command develop logics for several commands such as playing music, Wikipedia searches, etc.
8. Give commands to the desktop assistant.
9. Execute those commands.
10. End the program

4.4 Output Screen

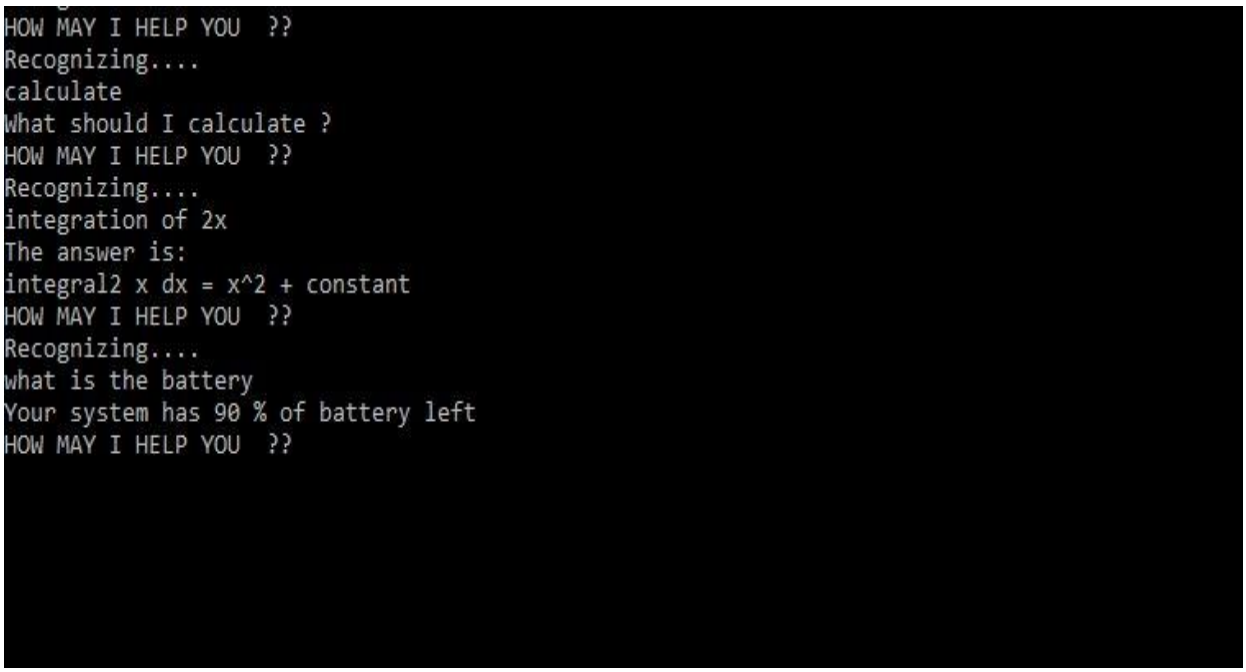

Fig. 4.4.1 – Output screen 1

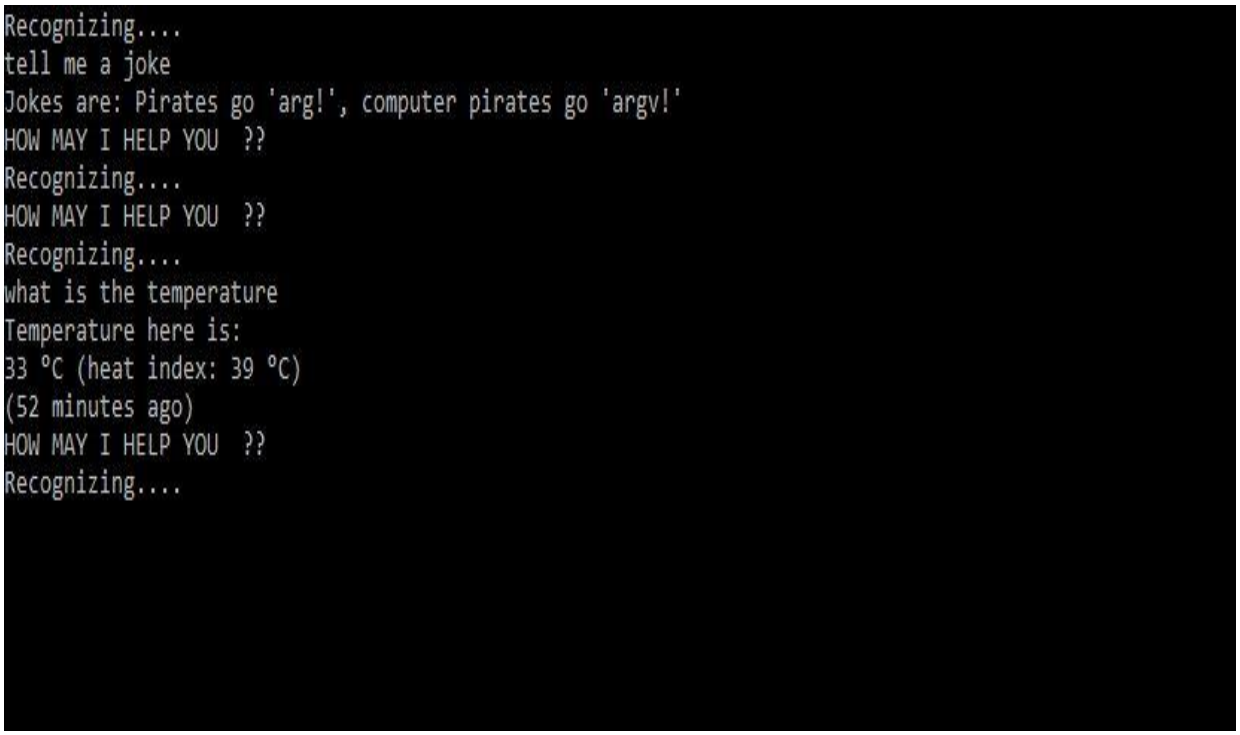

Fig. 4.4.2 – Output screen 2

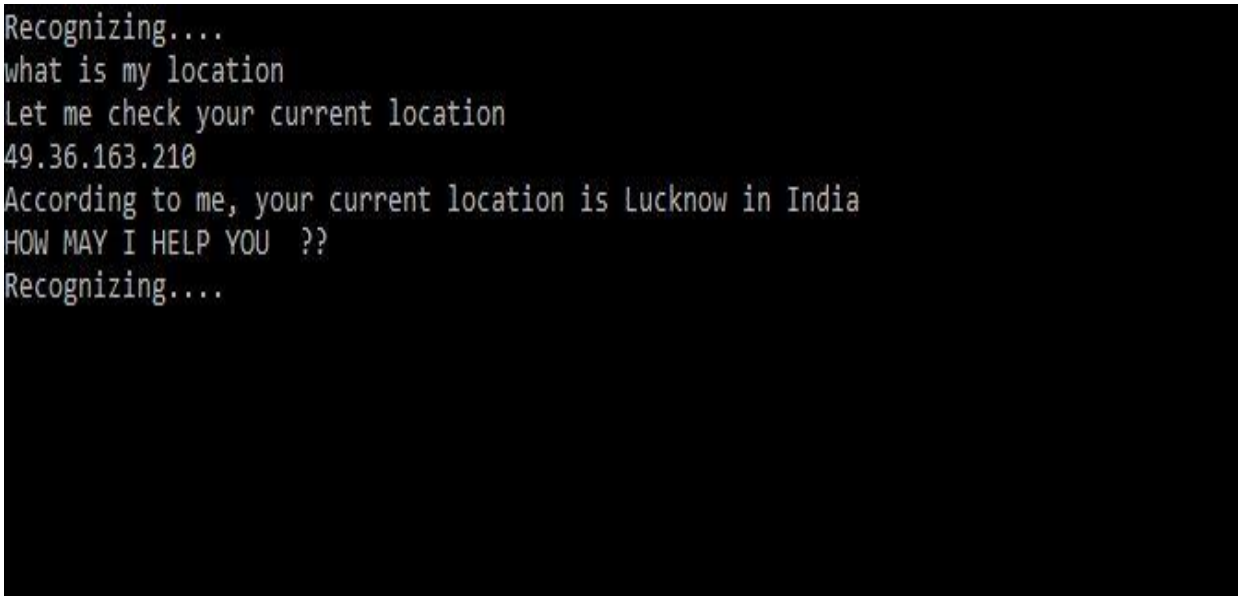

Fig. 4.4.3 – Output screen 3

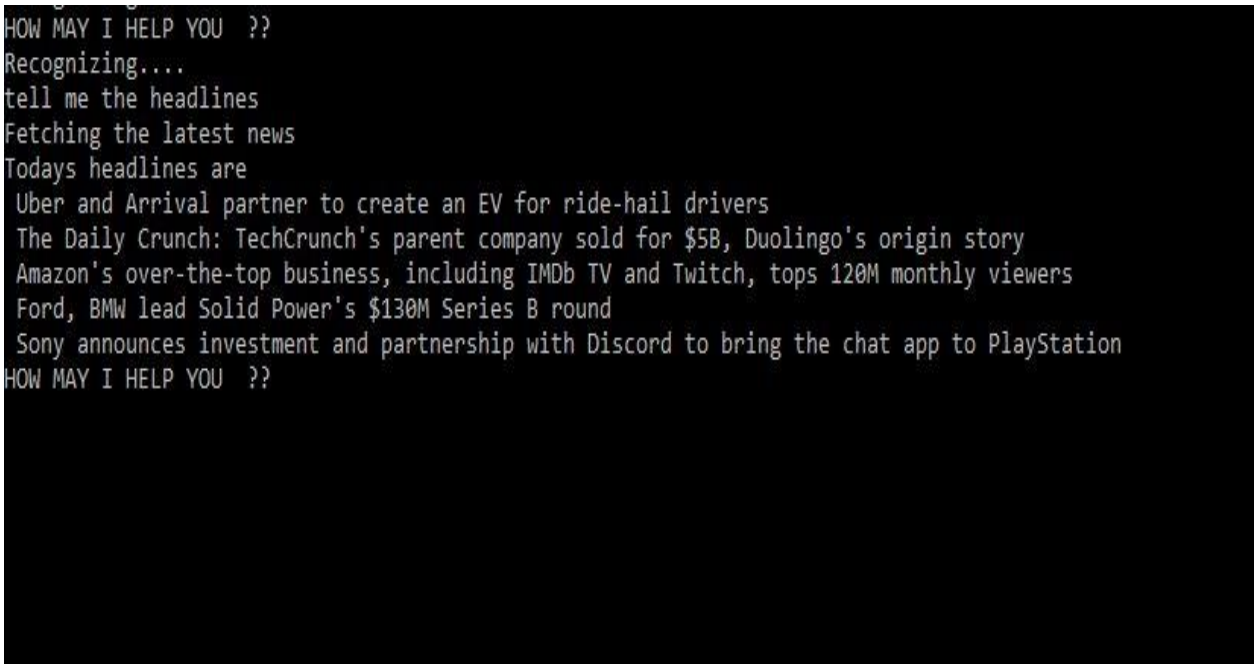

Fig. 4.4.4 – Output screen 4

4.5 Methodology

Phase 1 - Requirement analysis

- Study concepts of Python and its libraries.
- Study of installation and importing of libraries.
- Study of how speech to text library works.

Phase 2 - Designing and development

Figuring out optimized algorithm which will be used to develop the desktop assistant. Designing and development is further divided into various phases. This phase starts with a brainstorming session about all the details of the libraries that will be needed to be implemented to perform a specific task. Decisions over several useful and appropriate libraries are being made out to be implemented. And then we moved towards the coding phase to write codes for each module.

Phase 3 - Coding

On receiving design documents, the work is divided into modules/unit and distributed among the team members and actual coding is started. Since in this phase the code is produced so it is the main focus of the developers. This is going to be the longest phase in this project. The implementation of this project starts in terms of writing program in the python programming language and developing error free executable program efficiently, this phase primarily focuses on coding.

Implementation

- Installing required libraries.
- Importing required libraries.
- Creating a main function.
- Creating ‘Speak’ function which will take audio as its input and then pronounce it accordingly.
- Creating other functions for each task to be performed.
- Calling every function in the main function.

Phase 4 - Testing

- Manual Testing

Phase 5

- Integrating and implementing all the above phases.

5. LIMITATIONS

- Voice assistants use single commands.
- As more flexible natural language understanding technology is becoming available, interpretations of speech commands may be become ambiguous.
- It typically does not remember history.
- It is available for a single configured desktop.
- It depends on an Internet connection.

6. FUTURE ENHANCEMENTS

As the project expands we would like to add more enhanced and complex functionalities that will ease the task of an end user, some of the brainstorm idea are like:

- Command to be given by facial expression.
- Command to be given by gestures and actions.

7. CONCLUSION

The project's primary goal was to create a Desktop Assistant that would be used to find answers to questions submitted by users. To provide the user with all of the information they need.

It offers weather, news, and music, as well as the ability to search for topics on Wikipedia, set an alarm, and display the current date and time. This application allows the user to collect data. It saves both time and manpower. Because of the NLP support, users can ask questions in a very formal manner. There is no need to pose questions in a very strict and precise manner. The user should be familiar with the basic principles of English grammar. The aim is to give people a simple and fast way to get answers to their questions.

8. REFERENCES

[1]Prerana Das, Kakali Acharjee, Pranab Dasand Vijay Prasad, https://www.researchgate.net/publication/304651244_VOICE_RECOGNITION_SYSTEM_SPEECH-TO-TEXT

[2] S. Furui, T. Kikuchi, Department of Computer Science, Tokyo Institute of Technology, Tokyo, Japan: Speech-to-text and speech-to-speech summarization of spontaneous speech: https://ieeexplore.ieee.org/document/1306513/authors#authors

[3] Su Myat Mon, Hla Myo Tun: Speech-To-Text Conversion (STT) System Using Hidden Markov Model (HMM): http://www.ijstr.org/final-print/june2015/Speech-to-text-Conversion-stt-System-Using-Hidden-Markov-Model-hmm.pdf

Report Draft verified by

Project Guide	HOD
Mr. Sumit Kumar	**Dr. Neelu Jyoti Ahuja**
(Dept. of Systemics)	**(Dept. of Systemics)**