A

Major Project

On

VOICE CONTROLLED MOUSE AND KEYBOARD

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

B. VINOD KUMAR (187R1A0565)

P. MUKUNDH (187R1A0566)

Under the Guidance of

A. UDAY KIRAN

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)
Recognized Under Section 2(f) & 12(B) of the UGCAct.1956,
Kandlakoya (V), Medchal Road, Hyderabad-501401.

2018-22

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"VOICE CONTROLLED MOUSE AND KEYBOARD"** being submitted by **B. Vinod Kumar (187R1A0565), P.Mukundh (187R1A0566)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by him/her under our guidance and supervision during the year 2021-22.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

INTERNAL GUIDE
Mr. A.UDAY KIRAN
Assistant Professor

DIRECTOR
Dr. A. Raji Reddy

HEAD OF THE DEPARTMENT
Dr. K. Srujan Raju

EXTERNAL EXAMINER

Submitted for viva voice Examination held on_____

ACKNOWLEGDEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **A. UDAY KIRAN,** Assistant Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) **Mr. A. Uday Kiran, Mr. J. Narasimharao, Dr. T. S. Mastan Rao, Mrs. G. Latha, Mr. A. Kiran Kumar,** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

B. VINOD KUMAR (187R1A0565)
P. MUKUNDH (187R1A0566)

ABSTRACT

Human computer interaction is a field which focuses on providing a means of interaction between humans and computers. Controlling the mouse pointer is one of the best ways to provide a meaningful interaction. The Speech Recognition feature helps us in building an application using Python that will accept voice commands from the user and perform certain GUI based actions using the mouse and keyboard. Listening to the input voice from microphone and converting it to text and performs actions as commanded.

LIST OF FIGURES

LIST OF TABLES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1. INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

Voice Controlled Mouse and Keyboard is a system which accepts speech from the users through a microphone. This speech is transcribed to text by the Google's Speech-To-Text API. Then the system will perform actions accordingly. This system can perform actions like moving Mouse-Cursor (up, down, right, left), opening Applications, change some system settings like Volume, Brightness, takes Screenshots, terminates itself or it can also able to shut down or restart the PC.

1.2 PROJECT PURPOSE

There are several reasons why Voice Controlled Mouse and Keyboard is important. The following is a list of various reasons:

1. Convenient human computer interaction.
2. Making more efficient system for people with disabilities.
3. This Voice Controlled Mouse and Keyboard can be embedded in any applications for performing computer operations through our voice.
4. It will be very useful if we combine this Voice Controlled Mouse and keyboard with existing voice assistants.

1.3 PROJECT FEATURES

The main features of this project are that the designer now functions as a problem solver and tries to sort out the difficulties that the drivers face. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied withproposal.

2. SYSTEM ANALYSIS

2.SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. Inanalysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, "what must be done to solve the problem?" The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

Voice controlled mouse and keyboard is a system which will take human voice as input and further this feature can be used to perform computer operations such as moving mouse pointer in the four directions (Left, right, up, down). This system will also allow the user to open various files and to perform operating system's operations for example open notepad, volume up, volume down, shutdown, restart etc. Our project can be embedded in any of the applications.

2.2 EXISTING SYSTEM

Traditional mouse and keyboard are used as input devices. We are having many voice based Assistant Applications for smart actions. These Assistants can perform actions which we assign or order which we want to Perform based on our voice input.

2.2.1 LIMITATIONS OF EXISTINGSYSTEM

- Sometimes users may fail to use traditional mouse and keyboard in some situations.
- Physically disabled Persons May face this difficulty while using this traditional mouse and keyboard.
- These voice-based assistant fails to perform GUI based action.
- This may cause Voice Based Assistant to perform limited actions.
- Physically disabled Persons May face this difficulty while using this Assistant.

2.3 PROPOSED SYSTEM

In this project, the system accepts user commands through the system's microphone and saves the user audio in an audio file. This audio file is sent to the Google cloud. Google's Speech-Recognition API (or Speech-To-Text API) helps us to convert the audio(speech) to the Text. This text is considered as commands and each command is fetched in the program. If the command exists then the appropriate action is performed by the system.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

Following are benefits of Voice Controlled Mouse and Keyboard.

1. Actions can be done without using physical mouse and keyboard.
2. Our project makes more efficient human computer interaction for general users.
3. Our project will also helpful for the people with disabilities.
4. We can handle sessions where we cannot use mouse or keyboard physically.

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is done. This is to ensure that the proposed system is not a burden to the company. Three key considerations involved in the feasibility analysis are

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also, all the resources are already available, it gives an indication of the system is economically possible for development.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 SOCIAL FEASIBILITY

This includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

- System : I5 processor
- Hard Disk : 2 GB
- Input Devices : Microphone
- Output Devices : Basic Monitor
- Ram : 4GB

2.5.2 SOFTWARE REQUIREMENTS

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements.

- Operating system : Windows 7,8,10
- Programming Language : Python
- Tool : Visual Studio Code

2.6 MODULES DESCRIPTION

The Different modules in this project are:

1. Voice Detection
2. Performing GUI Actions

2.6.1 Voice Detection

This module is responsible for recording the audio of user's speech or commands. And then converting this audio (analog signals) to the machine understandable text. This conversion is performed using Google's Speech to Text API.

2.6.2 Performing GUI Actions

This module is responsible for responding to the user commands and performing desired actions. Performing GUI actions are performed by using the PyAutoGui Python package.

3.ARCHITECTURE

3. ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows how the user's voice(speech) is converted to text and how this text(command) is used to perform the actions

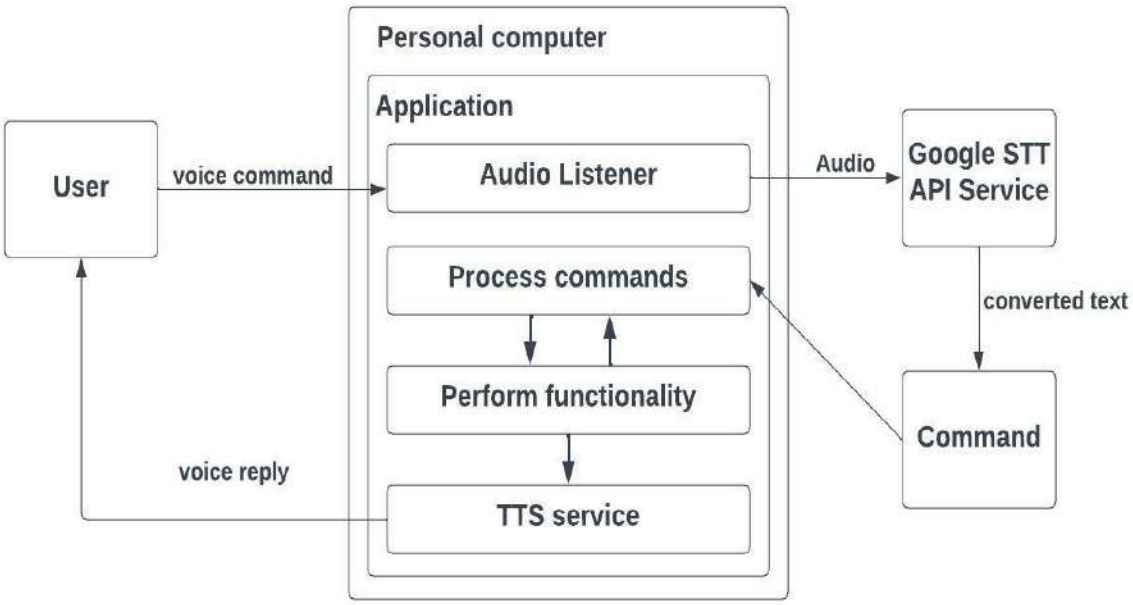

Figure 3.1: Project Architecture of Voice Controlled Mouse and Keyboard

3.2 DESCRIPTION OF THE ARCHITECTURE

User:

User orders the system for his/her requirements through his/her voice.

Audio Listener:

System's microphone is used to record the user's voice(speech).

Audio-file:

The recorded audio is saved in an audio file.

Google Cloud-Speech To Text API:

The saved audio file is sent to the Google's Speech-To-Text API through Google Cloud. This API transcribes the audio to the text. This transcribed text is sent back to the system.

Process Commands:

Here the system tries to understand the commands i.e., system fetches the command(text) in the program.

Perform Functionality:

If the system finds command i.e., the command exists in the program then the pre-programmed actions are performed.

TTS Service:

In this project PyTTS is used for Voice Feedback, for Human computer interaction.

3.3 USE CASE DIAGRAM

In UML, use-case diagrams model the behavior of a system and help to capture the requirements of the system. Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors.

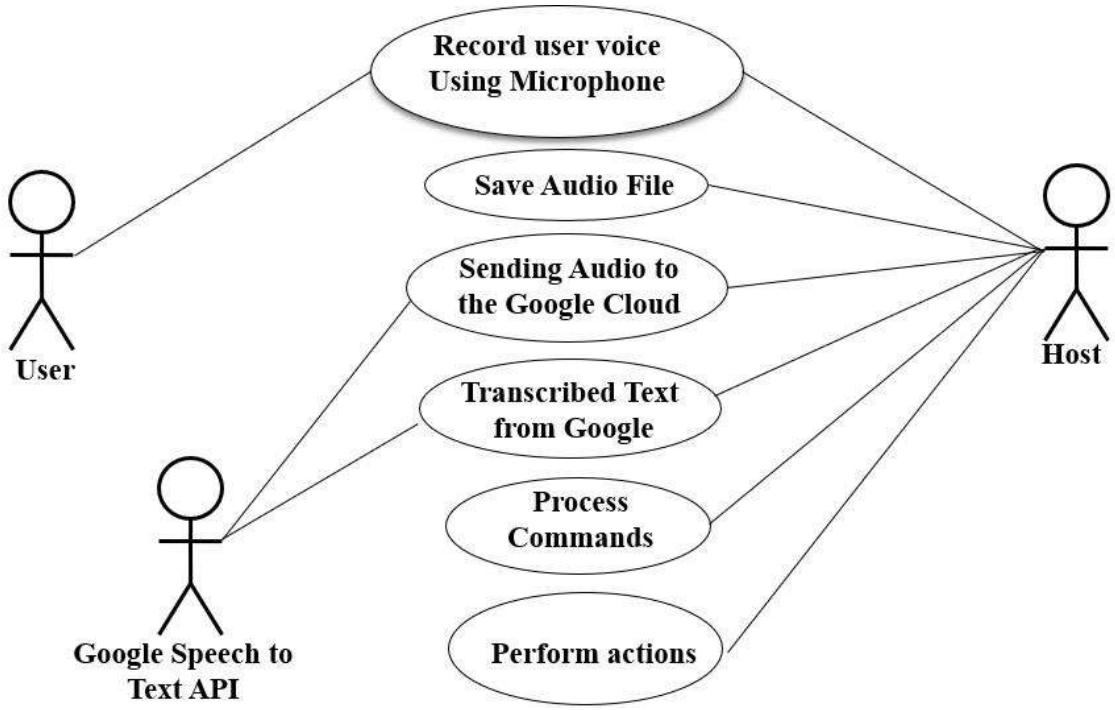

Figure 3.2: Use Case Diagram of Voice Controlled Mouse and Keyboard

3.4 SEQUENCE DIAGRAM

A sequence diagram or system sequence diagram (SSD) shows object interactions arranged in time sequence in the field of software engineering. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of scenario.

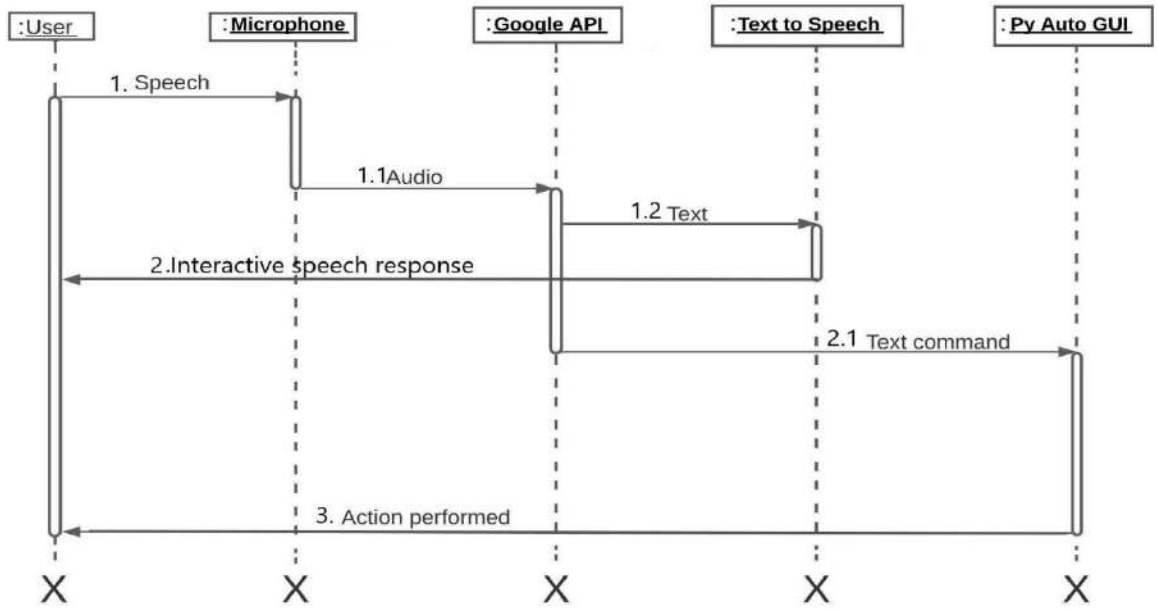

Figure 3.3: Sequence Diagram of Voice Controlled Mouse and Keyboard

3.5 ACTIVITY DIAGRAM

It describes about flow of activity states. Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

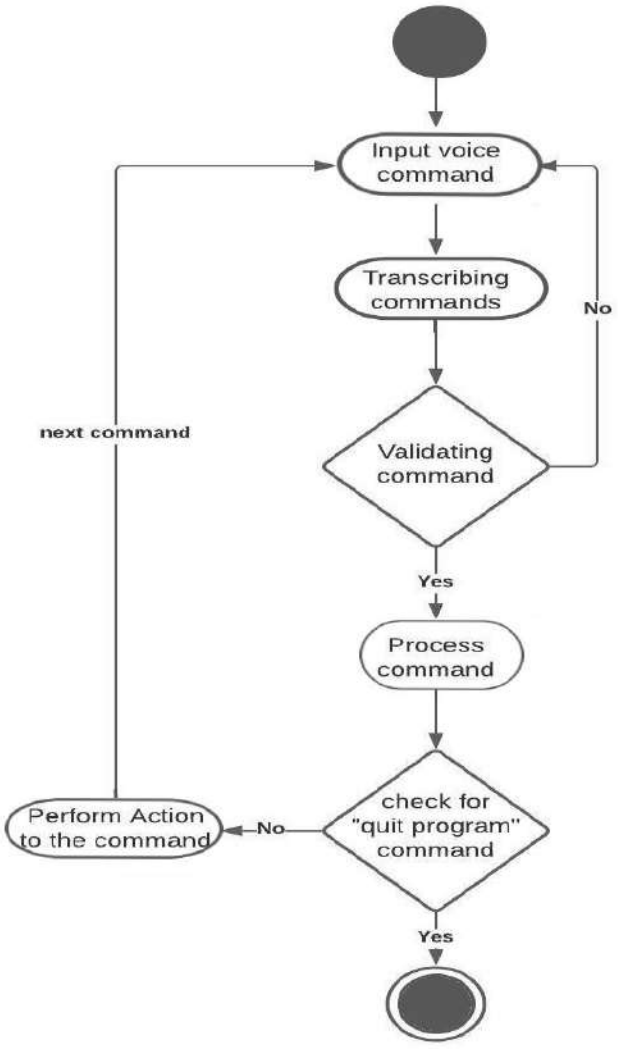

Figure 3.4: Activity Diagram of Voice Controlled Mouse and Keyboard

4. IMPLEMENTATION

4. IMPLEMENTATION

4.1 SAMPLE CODE

4.1.1 VoiceDetection.py

```
import pyautogui
import speech_recognition
import gui_cntrl
import pyttsx3
from playsound import playsound
import time
tts_engine=pyttsx3.init()
# The gui instance will be used to call GUI functions defined by us in 'gui_cntrl.py'
gui = gui_cntrl.gui_control()
recognizer = speech_recognition.Recognizer()
print("\n\nThreshold Value Before calibration:" + str(recognizer.energy_threshold))

with speech_recognition.Microphone() as src:

    while True:
        try:
            audio = recognizer.adjust_for_ambient_noise(src)
            playsound('D:/VS Code Projects/Major_project/2yT.wav')
            print("\n\nThreshold Value After calibration:" +
str(recognizer.energy_threshold))
            print("\nSpeak now:")
            audio = recognizer.listen(src)
            playsound('D:/VS Code Projects/Major_project/3kp.wav')
            speech_to_txt = recognizer.recognize_google(audio,language="en-in").lower()
            #speech_to_txt = recognizer.recognize_google_cloud(audio)
            #speech_recognition.WavFile("file.wav") as src:
```

```
except Exception as ex:
    print("Sorry. I Could not understand.\n\n")
    tts_engine.say("Sorry! I Could not understand.")
    tts_engine.runAndWait()
    time.sleep(0.3)
    continue
print("I heard : " + speech_to_txt)
#-----------------------------------------------------------------
# The following if-else block is for the commands I have chosen and
# call their respective GUI action
#-----------------------------------------------------------------
if (speech_to_txt == "quit program") or (speech_to_txt == "exit program"):
    tts_engine.say("Okay, I'm Quitting now")
    tts_engine.runAndWait()
    break
elif speech_to_txt == "mouse up" or speech_to_txt == "move up":
    tts_engine.say("moving up")
    tts_engine.runAndWait()
elif speech_to_txt == "right click" or speech_to_txt == "right-click":
    tts_engine.say("right clicking")
    tts_engine.runAndWait()
    gui.right_click()
elif speech_to_txt == "double click" or speech_to_txt == "double-click":
    tts_engine.say("double clicking")
    tts_engine.runAndWait()
    gui.double_click()
elif speech_to_txt == "press enter" or speech_to_txt == "enter":
    tts_engine.say("pressing enter key")
    tts_engine.runAndWait()
    gui.enter()
elif speech_to_txt=="press left arrow":
```

```
        tts_engine.say('pressing left arrow key')
        tts_engine.runAndWait()
        gui.left_arrow()
    elif speech_to_txt=="press right arrow":
        tts_engine.say('pressing right arrow key')
        tts_engine.runAndWait()
        gui.right_arrow()
    elif speech_to_txt =="press 1":
        tts_engine.say("pressing 1")
        tts_engine.runAndWait()
        gui.num1()
    elif speech_to_txt =="press 2":
        tts_engine.say("pressing 2")
        tts_engine.runAndWait()
        gui.num2()
    elif speech_to_txt=="plus":
        tts_engine.say("pressing plus key")
        tts_engine.runAndWait()
        gui.plus()
    elif speech_to_txt =="shutdown pc":
        print("\nDo you really want to shutdown your pc? ")
        tts_engine.say("Do you really want to shutdown your pc?")
        tts_engine.runAndWait()
        try:
            audio = recognizer.adjust_for_ambient_noise(src)
            playsound('D:/VS Code Projects/Major_project/2yT.wav')
            audio = recognizer.listen(src)
            playsound('D:/VS Code Projects/Major_project/3kp.wav')
            speech_to_txt = recognizer.recognize_google(audio,language="en-in").lower()
        except:
            print("Sorry! I didn't get you")
```

```
            tts_engine.say("Sorry! I didn't get you")
            tts_engine.runAndWait()
        if speech_to_txt=="yes":
            tts_engine.say("Shutting down your pc")
            tts_engine.runAndWait()
            gui.shutdown()
tts_engine.stop()
```

4.1.2 GUI_Control.py

```
from time import sleep
import pyautogui
import os
from playsound import playsound
# Faster: Moves mouse pointer by 200 pixels
# SLOWER: Moves mouse pointer by 20 pixels
FASTER=200
SLOWER=20

class gui_control:
    def __init__(self):
        pyautogui.PAUSE = 1
        pyautogui.FAILSAFE = True
        pyautogui.size()
    def mouse_up(self,recognizer, src):
        while True:
            speech_to_txt = ""
            pyautogui.moveRel(0, -1*SLOWER, duration=0.25)
            try:
                playsound('D:/VS Code Projects/Major_project/2yT.wav')
                audio = recognizer.listen(src)
```

```
            playsound('D:/VS Code Projects/Major_project/3kp.wav')
            speech_to_txt = recognizer.recognize_google(audio).lower()
        except:
            pass
        print("Inside mouse up :" + speech_to_txt)
        if speech_to_txt == "stop":
            break
        elif speech_to_txt == "faster":
            pyautogui.moveRel(0, -1*FASTER, duration=0.25)
        elif speech_to_txt == "slower":
            pyautogui.moveRel(0, -1*SLOWER, duration=0.25)
    def mouse_right(self,recognizer, src):
        #print("Move mouse right")
        pyautogui.moveRel(100, 0, duration=0.25)
        while True:
            speech_to_txt = ""
            pyautogui.moveRel(1*SLOWER, 0, duration=0.25)
            try:
                playsound('D:/VS Code Projects/Major_project/2yT.wav')
                audio = recognizer.listen(src)
                playsound('D:/VS Code Projects/Major_project/3kp.wav')
                speech_to_txt = recognizer.recognize_google(audio).lower()
            except:
                pass
            print("Inside mouse right :" + speech_to_txt)
            if speech_to_txt == "stop":
                break
            elif speech_to_txt == "faster":
                pyautogui.moveRel(1*FASTER, 0, duration=0.25)
            elif speech_to_txt == "slower":
                pyautogui.moveRel(1*SLOWER, 0, duration=0.25)
```

```
def left_click(self):
    pyautogui.click()
def right_click(self):
    print("Right Clicking")
    pyautogui.click(button='right', clicks=2, interval=0.25)
def double_click(self):
    print("Double Clicking")
    pyautogui.click(button='left', clicks=2, interval=0.25)
def mute_unmute(self):
    print("Pressing Mute/Unmute Key")
    pyautogui.typewrite(['volumemute'])
def play_pause(self):
    print("Pressing SPACE Key")
    pyautogui.typewrite(['space'])
def shutdown(self):
    os.system("shutdown /s /t 1")
def enter(self):
    pyautogui.typewrite(['enter'])
    print("Pressed ENTER key\n")
def delete(self):
    pyautogui.typewrite(['delete'])
    print("Pressed DELETE key\n")
def num1(self):
    pyautogui.press('num1')
def num2(self):
    pyautogui.press('num2')
def num3(self):
    pyautogui.press('num3')
def num4(self):
    pyautogui.press('num4')
def plus(self):
```

5. SCREENSHOTS

5.Screenshots

5.1 MAIN WINDOW

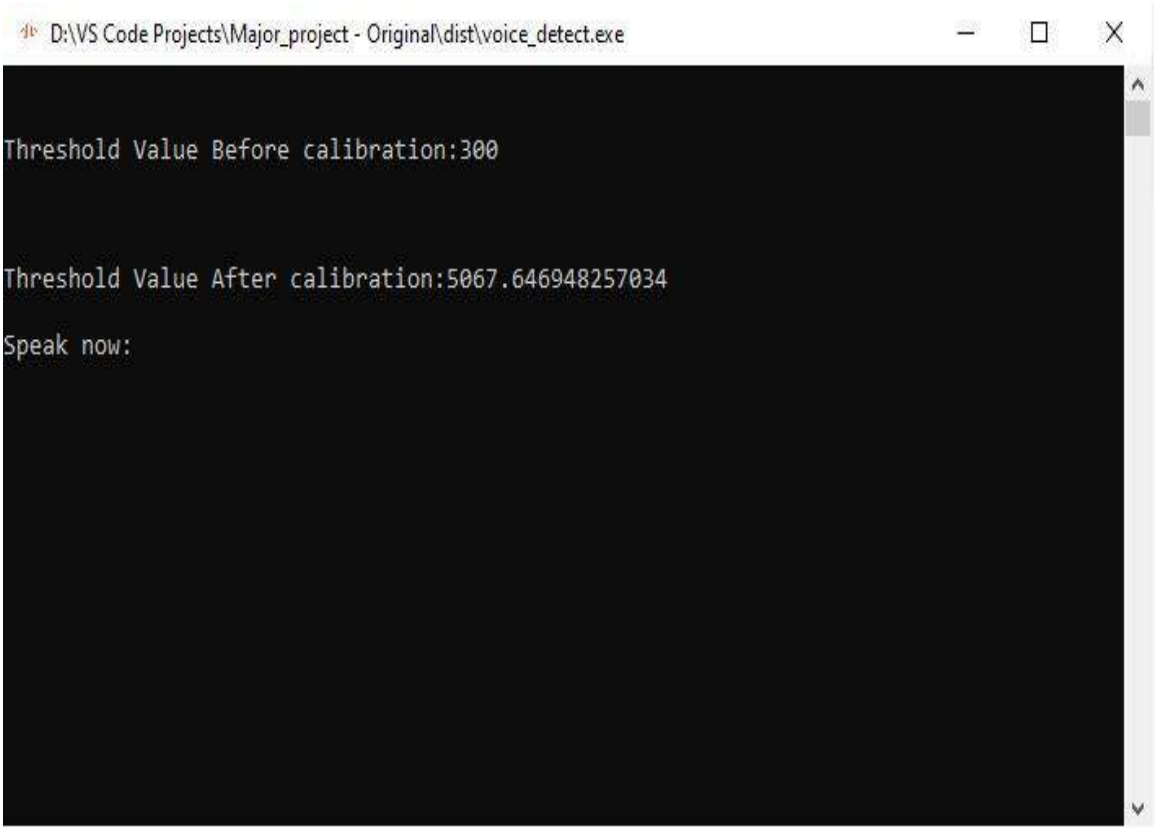

Figure 5.1: Main Window

5.2 MOUSE UP / DOWN / RIGHT / LEFT COMMANDS

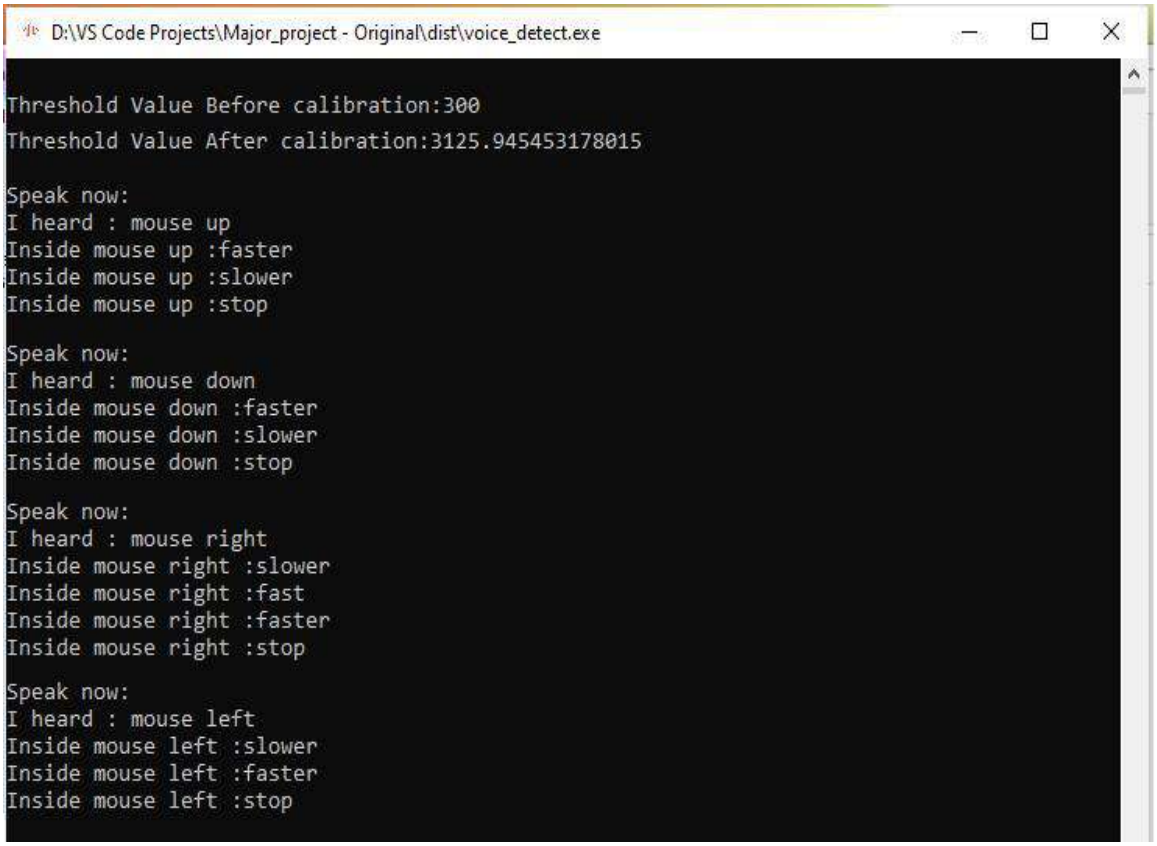

Figure 5.2: Mouse Up/Down/Right/Left Commands Window

5.3 MOUSE RIGHT / LEFT / DOUBLE CLICK COMMANDS

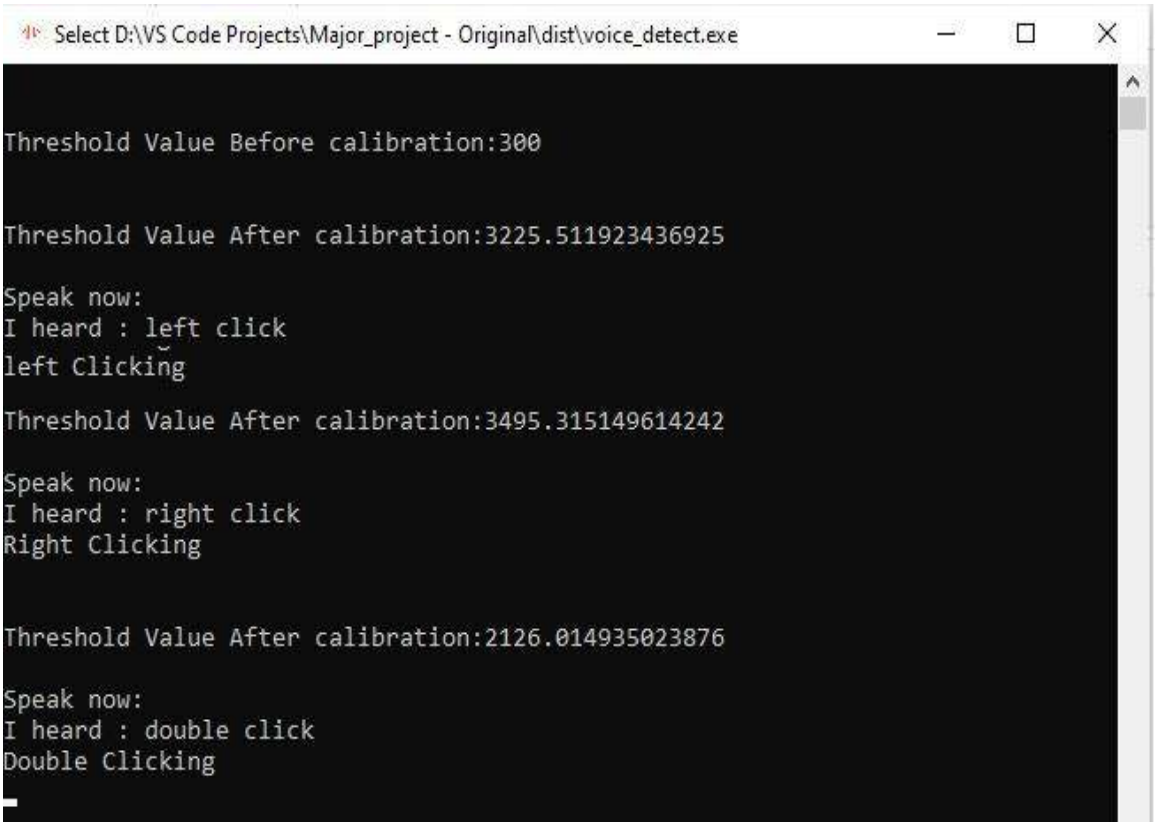

Figure 5.3: Mouse Right/Left/Double Click commands Window

5.4 SOME OF THE KEYBOARD COMMANDS

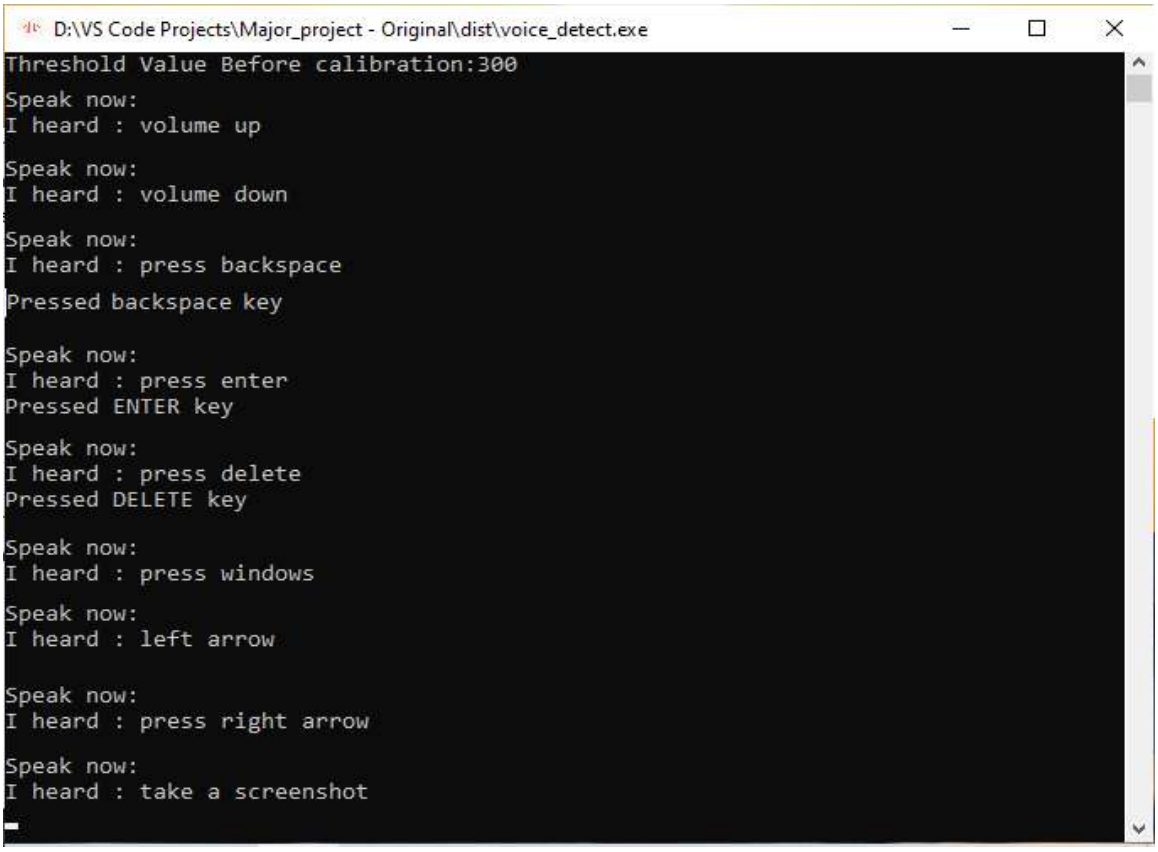

Figure 5.4: Some Keyboard Commands Window

6.TESTING

6. TESTING

6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by

successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes.

6.3 TESTCASES

6.3.1 PROVIDING INPUT AUDIO

Test case ID	Test case name	Purpose	Test Case	Output
1	Microphone recording test 1	Use it for Transcription.	The Microphone records first test audio.	The test input data provided successfully.
2	Microphone recording test 2	Use it for Transcription.	The Microphone records second test audio.	The test input data provided successfully.

Table 6.1: Providing input audio

6.3.2 PERFORMING ACTION FOR THE COMMAND

Test case ID	Test case name	Purpose	Input	Output
1	Performance test 1	To check if the model performs its task	A valid command is given.	Recognized the Command and respective action is performed.
2	Performance test 2	To check if the model performs its task	An invalid command is given.	Not recognized and no action is performed.

Table 6.2: Performing action for the command

7. CONCLUSION

7. CONCLUSION & FUTURESCOPE

7.1 PROJECT CONCLUSION

The project titled as “VOICE CONTROLLED MOUSE AND KEYBOARD is a console-based application. This software provides facility of performing the actions a commanded by the user. This software is developed with scalability in mind. The software is developed with modular approach. All modules in the system have been tested with valid data and invalid data and everything work successfully. Thus, the system has fulfilled all the objectives identified and is able to replace the existing system.

Human computer interaction is a field which focuses on providing a means of interaction between humans and computers. Controlling the mouse pointer is one of the best ways to provide a meaningful interaction.

The system is very flexible and versatile. Validation checks induced have greatly reduced errors. Provisions have been made to upgrade the software. The application has been tested with live data and has provided a successful result. Hence the software has proved to work efficiently.

7.2 FUTURE SCOPE

In future we can use offline Speech Recognition by downloading the modules directly into the project files. So that we don’t need internet connection. We simply give the command, then the command will be processed locally and the action will be performed. The software can be developed further to include lot of modules because the proposed system is developed on the view of future.

8. BIBILOGRAPHY

8. BIBILOGRAPHY

8.1 REFERENCES

[1] PyAutoGUI's Documentation by **readthedocs.io**

[2] Python Crash Course by **Eric Matthes**

8.2 GITHUB LINK

https://github.com/VinodKumarBethi/VOICE-CONTROLLED-MOUSE-AND-KEYBOARD.git