
SRUTHI – A MALAYALAM VOICE ASSISTANT FOR NATIVE LANGUAGE USERS

MINI PROJECT REPORT

by

ALEN GEORGE (VJC19CS018)

JOEL RAJU (VJC19CS079)

ROSHAN ROY (VJC19CS106)

SAN BABY FRANCIS (VJC19CS108)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VISWAJYOTHI COLLEGE OF ENGINEERING AND TECHNOLOGY,
VAZHAKULAM
AUGUST 2022**

SRUTHI – A MALAYALAM VOICE ASSISTANT FOR NATIVE LANGUAGE USERS

MINI PROJECT REPORT

by

ALEN GEORGE (VJC19CS018)

JOEL RAJU (VJC19CS079)

ROSHAN ROY (VJC19CS106)

SAN BABY FRANCIS (VJC19CS108)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

under the guidance

of

Ms. Asha Joseph

Assistant Professor, Dept. of CSE



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VISWAJYOTHI COLLEGE OF ENGINEERING AND TECHNOLOGY,
VAZHAKULAM
AUGUST 2022**

**VISWAJYOTHI COLLEGE OF ENGINEERING AND TECHNOLOGY,
VAZHAKULAM**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vision

Moulding socially responsible and professionally competent Computer Engineers to adapt to the dynamic technological landscape.

Mission

1. Foster the principles and practices of computer science to empower life-long learning and build careers in software and hardware development.
2. Impart value education to elevate students to be successful, ethical and effective problem-solvers to serve the needs of the industry, government, society and the scientific community.
3. Promote industry interaction to pursue new technologies in Computer Science and provide excellent infrastructure to engage faculty and students in scholarly research activities.

Program Outcomes

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design / development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and unread in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Educational Objectives

Our Graduates

1. Shall have creative and critical reasoning skills to solve technical problems ethically and responsibly to serve the society.
2. Shall have competency to collaborate as a team member and team leader to address social, technical and engineering challenges.
3. Shall have ability to contribute to the development of the next generation of information technology either through innovative research or through practice in a corporate setting.
4. Shall have potential to build start-up companies with the foundations, knowledge and experience they acquired from undergraduate education.

Program Specific Outcomes

1. Ability to integrate theory and practice to construct software systems of varying complexity.
2. Able to apply Computer Science skills, tools and mathematical techniques to analyze, design and model complex systems.
3. Ability to design and manage small-scale projects to develop a career in a related industry.

**VISWAJYOTHI COLLEGE OF ENGINEERING AND TECHNOLOGY,
VAZHAKULAM**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



BONAFIDE CERTIFICATE

This is to certify that the Mini Project Report entitled “**SRUTHI – A MALAYALAM VOICE ASSISTANT FOR NATIVE LANGUAGE USERS**” is a bonafide record of the work by **ALEN GEORGE (VJC19CS018), JOEL RAJU (VJC19CS079), ROSHAN ROY (VJC19CS106)** and **SAN BABY FRANCIS (VJC19CS108)** in partial fulfillment of the requirements for the award of the **Degree of Bachelor of Technology in Computer Science and Engineering** of APJ Abdul Kalam Technological University.

Date:

Place: Vazhakulam

Mrs. Rini Simon
Project Coordinator
Assistant Professor
Dept. of CSE, VJCET

Ms. Asha Joseph
Project Guide
Assistant Professor
Dept. of CSE, VJCET

Mr. Amel Austine
Asst. Professor and HOD
Dept. of CSE, VJCET

ACKNOWLEDGEMENT

First of all and whole heartedly, we thank **God Almighty** for giving us this opportunity for successful completion of our mini project report work. We are immensely indebted to our beloved Manager, **Msgr. Dr. Pius Malekandathil** and our beloved Director **Rev. Fr. Paul Nedumpurath**, for providing a good infrastructure with well-equipped labs. Our Principal **Dr. K K Rajan** deserves special mention for the moral support he has given to us. We assert beyond the confines of simple gratitude to the Head of the Department of CSE **Mr. Amel Austine**, project coordinator **Mrs. Rini Simon**, project guide **Ms. Asha Joseph** for their very able guidance and valuable cooperation. Also, we would like to express our gratitude for their guidance and valuable suggestions and encouragement in this topic. Unflinching support and constant encouragement from our friends helped us a long way to complete the mini project report work. We thank them from the depth of our heart.

DECLARATION

We hereby declare that the mini project report entitled “SRUTHI – A MALAYALAM VOICE ASSISTANT FOR NATIVE LANGUAGE USERS” submitted by us to the Department of Computer Science and Engineering, Viswajyothi College of Engineering Technology, Vazhakulam, Muvattupuzha in partial fulfillment of the requirement for the award of the degree of B.Tech in Computer Science and Engineering is a record of bonafide mini project work carried out by us under the guidance of Ms. Asha Joseph. We further declare that the work reported in this mini project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree in this college.

Place: Vazhakulam

Date:

ALEN GEORGE

JOEL RAJU

ROSHAN ROY

SAN BABY FRANCIS

ABSTRACT

Today, though most people have access to the internet, language is still a barrier to many. Most systems use English as their primary language and many non-English speaking users would find it difficult to work on these systems. This is especially true for the elderly who have little knowledge of English. This paper aims to bridge this gap by providing an interface so that the users can interact with the computer in their native language. Sruthi aims to be a voice recognition system that would process real time speech in Malayalam, perform the required computational task and would provide the real time feedback as speech in Malayalam. It is intended to act like a voice assistant that can recognise the native language of the user.

Keywords: voice recognition system, real time speech, native language, real time feedback

CONTENTS

LIST OF FIGURES AND TABLES	ii
-----------------------------------	-----------

LIST OF ABBREVIATIONS	iii
------------------------------	------------

1. INTRODUCTION	1
1.1 PROBLEM DEFINITION	2
1.2 OBJECTIVE	2
1.3 SCOPE	2
2. LITERATURE SURVEY	3
3. METHODOLOGY	5
3.1 SPEECH RECOGNITION	5
3.2 KEYWORD DETECTION	5
3.3 PERFORM REQUIRED ACTION	6
3.3.1 READ OUT THE NAME	6
3.3.2 READ OUT CURRENT DATE AND TIME	6
3.3.3 PLAY YOUTUBE VIDEOS	6
3.3.4 TELL JOKES	7
3.3.5 READ OUT WIKIPEDIA ARTICLES	7
3.4 PROVIDE FEEDBACK AS SPEECH	7
3.5 DEPENDENCIES	7
3.6 WORKFLOW	7

3.7	OBSERVATIONS AND GRAPHS	8
3.7.1	PERFORMANCE	9
3.8	GRAPHICAL USER INTERFACE	9
3.9	PACKAGE DESCRIPTION	10
3.9.1	SPEECH RECOGNITION	10
3.9.2	GTTS	12
3.9.3	PLAYSOUND	12
3.9.4	DATETIME	13
3.9.5	WIKIPEDIA	13
3.9.6	PYWHATKIT	14
3.9.7	TKINTER	14
3.9.8	OS	15
3.9.9	PYAUDIO	15
3.10	WORKING	16
3.10.1	STAGE 1	16
3.10.2	STAGE 2	16
3.10.3	STAGE 3	16
3.10.4	STAGE 4	17
4.	RESULT	18
4.1	END RESULT	18
4.2	PRODUCT SCREENSHOTS	20

5. CONCLUSION	25
APPENDIX – I	26
REFERENCES	iv

LIST OF FIGURES AND TABLES

1.1 Overview of Sruth	2
3.1 Keyword Detection	6
3.2 Block diagram showing Workflow	8
3.3 Histogram showing accuracy of each command	8
3.4 Pie chart showing overall accuracy	9
3.5 Splash image of Sruthi	10
3.6 Button that initiates Sruthi	10
3.7 Overview of Google Speech-to-Text engine	11
3.8 Block diagram of TTS engine	12
3.9 Different stages of Sruthi	16
4.1 Change directory to the folder that contains sruthi.py	18
4.2 Run the python script	19
4.3 Graphical User Interface	19
4.4 Sruthi listens to the user input	20
4.5 Sruthi tells her name	20
4.6 Sruthi responds to a greeting	21
4.7 Sruthi reads out the current date	21
4.8 Sruthi reads out the current time	22
4.9 Sruthi reads out information from Wikipedia	22
4.10 Sruthi is asked to play a song	23
4.11 The requested youtube video is played	23
4.12 Sruthi tells a joke	24
A.1 Supported voice commands	26

LIST OF ABBREVIATIONS

NLP – Natural Language Processing

IoT – Internet of Things

TTS – Text -to-Speech

STT – Speech-to-Text

API – Application Programming Interface

GUI – Graphical User Interface

OS – Operating System

Chapter 1

INTRODUCTION

Voice assistants like Google Assistant, Cortana, etc. has become an integral part of our lives. Be it booking tickets, searching the web or telling jokes these voice assistants has got all covered. They significantly made our lives easier and hassle free. In fact, internet-based voice recognition systems are one the most used technological inventions of recent times and will continue to transform our lives as we move forward. But something made us think. Are these voice assistants accessible to all? We found out that though these systems are of great help to the upper sections of the society who are educated and knows how to use technology, there are significant portion of the society who are ignorant about the use of technology and doesn't know how to speak in English. These include the elderly who have trouble using English and even the blind who doesn't know English. Hence, we decided to come up with a voice recognition system that would understand the native language of the user (this project is designed to understand Malayalam) and provide natural feedback as speech in their native language. Though certain voice assistants like Google Assistant supports local language recognition, it is limited to smartphones. We designed a voice recognition software for personal computers where people spent their time doing productive work. We have included certain basic functionality to our system that would ensure hassle free and productive experience to our users. Our software would also be useful to those technology enthusiasts who despite knowing English, would like to interact with their computer in their native language. Our system is primarily intended to reduce the gap between the users ensuring the benefits of technology to all, no matter the language they speak or the disabilities they have.

1.1 PROBLEM DEFINITION

Today, though most people have access to the internet, language is still a barrier to many. Most systems use English as their primary language and many non-English speaking users would find it difficult to work on these systems. This is especially true for the elderly who have little knowledge of English and the differently abled who have trouble using these systems. This project aims to bridge this gap by providing an interface so that the users can interact with the computer in their native language.

1.2 OBJECTIVE

To create a voice recognition system that would process real time speech in Malayalam, perform the required computational task and would provide the real time feedback as speech in Malayalam.

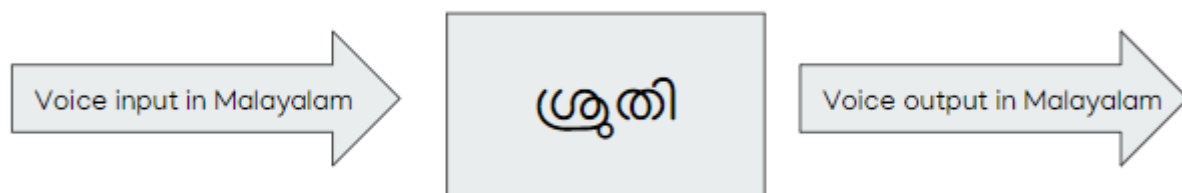


Figure 1.1: Overview of Sruthi

1.3 SCOPE

1. Sruthi is a voice assistant aiming to help those users whose native language is Malayalam.
2. It will help native language speakers to better understand technology and easily interact with their desktop devices.
3. It will also help visually challenged Malayalam speakers to utilize the features of their computer.

Chapter 2

LITERATURE SURVEY

The development of voice assistants was started in 1962 at the Seattle world's fair where IBM presented a device called shoebox IBM that could recognize spoken digits and then return them back through igniting lamps labelled next to the digits 0-9. It had the ability to perceive a total of 16 words. Currently voice assistants like Google Assistant, Siri, Cortana, Alexa etc., are some of the commonly used voice recognition systems across the globe. All of these voice assistants perform the same intended function – that is, voice initialized processing, and all of these developments have been a result of the same new age technology- Artificial intelligence.

It is also known that ever since 2010, the use of these voice assistants has increased exponentially. This is mainly due to the recent developments in Natural Language Processing (NLP), huge demand of smartphones and better internet speeds. However, these systems tend to remain exclusive to mobile devices (Google Assistant) and IoTs (Alexa). Though there exist voice assistants for Desktop computers, they tend to only support English and a very few most spoken languages.

Kurian Benoy and Jiby J Puthiyidam in their paper titled “A Study of Text to Speech Systems for Non-English Languages” discusses the methodologies to design a TTS system that supports foreign languages, especially Indian languages [1]. Preena Johnson, Jishna K C, and Soumya S in their paper titled “Speech to Text Conversion in Malayalam” provide great insights on creating a STT system that supports Malayalam language. In fact, it is easier to recognize a phonetic language such as Malayalam. On other hand, languages such as English is often difficult to recognize as it has several words that pronounce differently when spoken.

However, it was the paper titled “Voice Assistant using Python” by Nivendita Singh, Diwakar Yagyasan, Surya Vikram Singh, Gaurav Kumar and Harshit Agarwal which gave us an inspiration to build this project [3]. In the paper, the authors try to implement a feature rich voice assistant using python language. Their system could search the web, play multimedia, etc. in response to a voice input in English. Further we came across several papers that had similar or related idea. “Research Paper on on Desktop Voice Assistant” by Vishal Kuamar Dhanraj, Lokesh Kripilani and Semal Mahajan [4], “Voice Assistant using Artificial Intelligence” by Preethi G, Abhishek K, and Vishwaa D A [5] were some of them. We drew inspiration from these projects to build a similar voice assistant but that would support Malayalam as user input along with several additional features such as searching for articles, telling jokes, etc.

Chapter 3

METHODOLOGY

Sruthi is programmed using Python. This voice recognition system listens to the Malayalam speech input provided by the user, looks for relevant keyword and performs the required action. It also provides feedback as speech in Malayalam. Our proposed voice recognition system has the following phases:

- Step 1: Speech Recognition
- Step 2: Keyword Detection
- Step 3: Perform Required Action
- Step 4: Provide Feedback as Speech

3.1 SPEECH RECOGNITION


Initially, the Malayalam speech provided by the user is taken as input in real time. The source of the audio input is set to the default microphone which may be the built-in microphone or an external mic. The voice input received is recognized in real time by the Google STT engine which generates a text/string after analyzing the audio sample. This engine has built-in support for various languages including Malayalam. The Speech-to-Text conversion is implemented by using the *speech_recognition* library that uses the Google Speech-to-Text API.

3.2 KEYWORD DETECTION

The voice recognition system uses pattern matching or keyword detection to predict the action that has to be performed. The string obtained after speech recognition is checked for any matching keywords that would imply a certain action. The keyword is checked across the string

by using a simple if-else-if ladder. If found the required action must be performed.

Voice Input	Keyword
ഇന്നത്തെ കാലാവസ്ഥ എങ്ങനെയാണ്?	കാലാവസ്ഥ
കാലാവസ്ഥ നല്ലതാണോ?	കാലാവസ്ഥ
കാലാവസ്ഥ മോശമാകുമോ?	കാലാവസ്ഥ



Action Performed
Reads out the current weather

Figure 3.1: Keyword Detection

3.3 PERFORM REQUIRED ACTION

This is one of the most important phases of the voice assistant. In this phase the required computational task or the action expected by the user is performed. By the end of the previous phase, we obtained a keyword that required to be served. In this phase, we perform the action corresponding to that keyword. Some of the functionalities that Sruthi supports include:

3.3.1 READ OUT THE NAME

The voice assistant is programmed to read out its name when asked for it. This action will be performed when Sruthi encounters the keyword << പേര് >>

3.3.2 READ OUT CURRENT DATE OR TIME

Sruthi can read out the current date or time when asked for it. This simple action is performed with the help of *datetime* module in python that would generate current date and time. The keywords that would initiate the required action are << തീയതി >> or << സമയം >>

3.3.3 PLAY YOUTUBE VIDEOS

It is possible to play any youtube videos in response to the voice input provided by the user. This feature is made possible by using the *pywhatkit* library that has added support to integrate well with youtube and would redirect the user to the specific youtube video by opening a browser window. The keywords that are used are << കേൾക്കണം >> or << കാണണം >>

3.3.4 TELL JOKES

The voice assistant is programmed to tell jokes to the user when asked for it. We developers have carefully curated a list of Malayalam jokes and have written it to a text file - *jokes.txt* such that each line corresponds to a joke. Sruthi would randomly select a line number using the *randint()* method of the *random* library and would read out the joke written at that specific line. The keyword that would initiate this feature is << തമാശ >>

3.3.5 READ OUT WIKIPEDIA ARTICLES

Another interesting feature that Sruthi supports is that it integrates well with Wikipedia and would read out the Wikipedia articles on demand. The *wikipedia* library supports a variety of features including scraping articles from Wikipedia. The keyword used is << ആരാൺ >>

3.4 PROVIDE FEEDBACK AS SPEECH

This is the final phase of the voice recognition system. It involves providing the real time feedback to the user as speech after performing the required action. Each actions mentioned above must provide a voice output so that the user can hear the results in their native language (e.g., jokes, current date or time, etc.). Hence this final phase involves Text-to-Speech conversion so that the users can listen to the output generated by the above actions. This TTS conversion is performed using the *gtts* library that supports generation of Malayalam speech from text with the help of Google Text-to-Speech API.

3.5 DEPENDENCIES

1. *python 3.7 >=*
2. *speech_recognition*
3. *gtts*
4. *playsound*
5. *datetime*
6. *wikipedia*
7. *pywhatkit*
8. *tkinter*

3.6 WORKFLOW

Initially, the user input in Malayalam is obtained via microphone. This audio sample is then processed by the Google STT engine in order to obtain a text that is equivalent to the given voice

input. This text is then checked for keywords. If any keywords are found, the action specific to the identified keyword is performed. The results are then provided to the user as speech. This requires conversion of the text to its speech equivalent which is performed with the help of Google TTS engine.

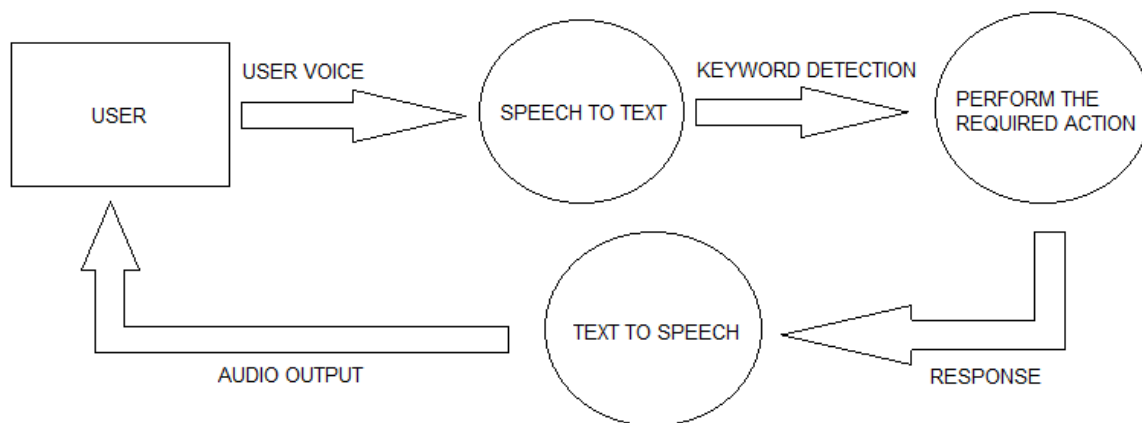


Figure 3.2: Block diagram showing Workflow

3.7 OBSERVATIONS AND GRAPHS

We tested the system for accuracy and errors. The system performed well in various test conditions. However, the system was unable to recognize certain keywords during some of our tests. The overall accuracy of the voice recognition system is found out to be 80%. Individual success/fail rate of each keyword/command was also measured and is plotted below:

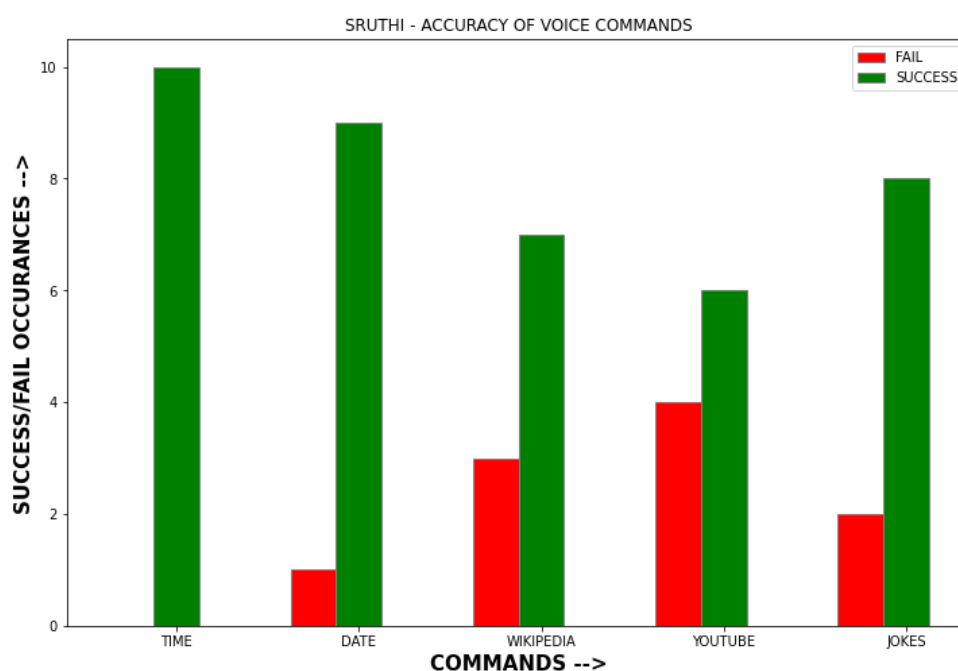


Figure 3.3: Histogram showing accuracy of each command

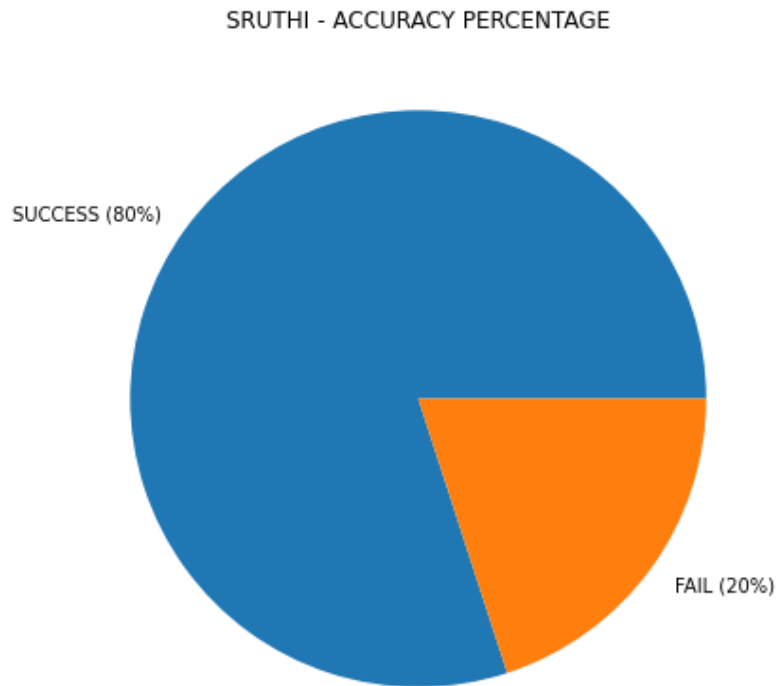


Figure 3.4: Pie chart showing overall accuracy

3.7.1 PERFORMANCE

1. Better accuracy, we were able to achieve an accuracy of 80%
2. Less time consuming in finding the results
3. Better efficiency compared to other systems
4. Improved usability due to simple and minimal interface

3.8 GRAPHICAL USER INTERFACE

The frontend (GUI) of Sruthi is built using the *tkinter* library. This library ensures that the user interface would remain the same across various operating systems including Windows, Linux and MacOS. The design of user interface is intentionally made minimal. This would thus support a diverse set of users ensuring accessibility to all. The design consists of a splash image that shows the logo along with the name of the developers. A button is placed at the bottom that initiates the voice assistant. When the button is clicked, the assistant would prompt/ask the user to speak. This feature would help the blind as they receive voice feedback when the program initiates. Sruthi would then listen to the speech input provided by the user.



Figure 3.5: Splash image of Sruthi



Figure 3.6: Button that initiates Sruthi

3.9 PACKAGE DESCRIPTION

3.9.1 SPEECH RECOGNITION

The *speech_recognition* library acts as a wrapper for several popular speech APIs and is thus extremely flexible. One of these—the Google Web Speech API—supports a default API key that is hard-coded into the *speech_recognition* library. That means we can have access to full features without having to sign up for a service. Recognizing speech requires audio input, and *speech_recognition* makes retrieving this input really easy. Instead of having to build scripts for accessing microphones and processing audio files from scratch, *speech_recognition* will get our program running in just a few minutes. The flexibility and ease-of-use of the *speech_recognition* package make it an excellent choice for any Python project. However, support for every feature of each API it wraps is not guaranteed. *speech_recognition* is compatible with Python 2.6, 2.7 and 3.3+ and it will work out of the box and all we have to do is to work with existing audio files. Specific use cases, however, require a few dependencies. Notably, the *pyaudio* package is needed for capturing microphone input.

Google Speech-to-Text API synchronous recognition request is the simplest method for performing recognition on speech audio data. Speech-to-Text can process up to 1 minute of speech audio data sent in a synchronous request. After Speech-to-Text processes and recognizes all of the audio, it returns a response. A synchronous request is blocking, meaning that Speech-to-Text must return a response before processing the next request. Speech-to-Text typically processes audio faster than realtime, processing 30 seconds of audio in 15 seconds on average. In cases of poor audio quality, our recognition request can take significantly longer. Audio is supplied to Speech-to-Text through the audio parameter of type `RecognitionAudio`. We must specify the sample rate of our audio in the `sampleRateHertz` field of the request configuration, and it must match the sample rate of the associated audio content or stream. Sample rates between 8000 Hz and 48000 Hz are supported within Speech-to-Text. Google Speech-to-Text's recognition engine supports a variety of languages and dialects. We can specify the language (and national or regional dialect) of our audio within the request configuration's `languageCode` field, using a BCP-47 identifier.

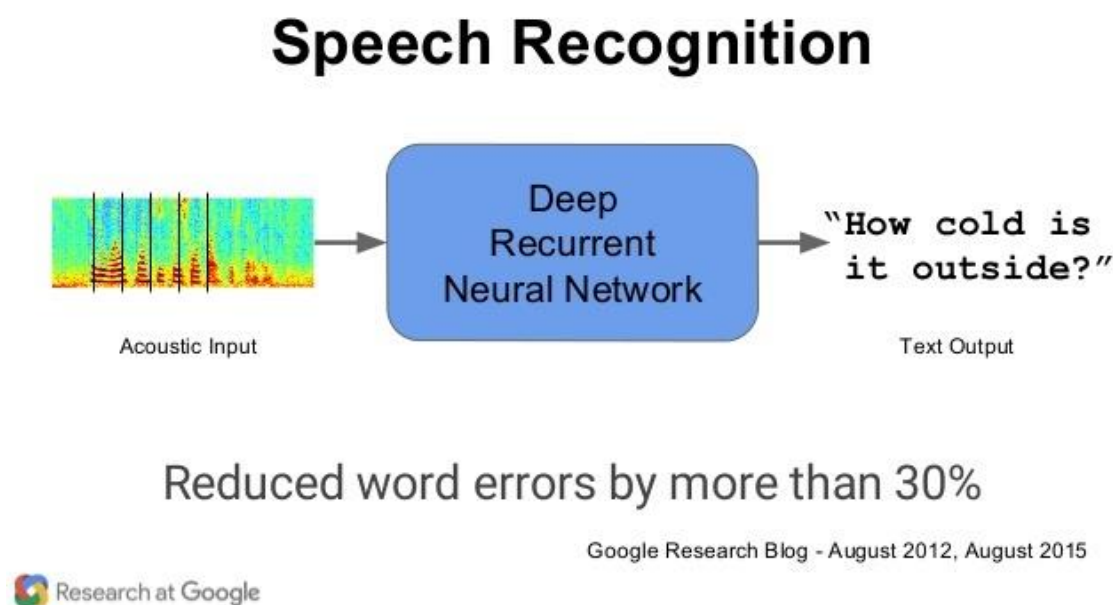


Figure 3.7: Overview of Google Speech-to-Text engine

Google Speech-to-Text can use one of several machine learning models to transcribe our audio file. Google has trained these speech recognition models for specific audio types and sources. When we send an audio transcription request to Speech-to-Text, we can improve the results that you receive by specifying the source of the original audio. This allows the Speech-to-Text API to process our audio files using a machine learning model trained to recognize speech audio from that particular type of source.

3.9.2 GTTS

Google Text-to-Speech or *gtts* is a Python library and command line tool to interface with Google Translate Text-to-Speech API. The Text-to-Speech (TTS) is the process of converting words into a vocal audio form. The program, tool, or software takes an input text from the user, and using methods of natural language processing understands the linguistics of the language being used, and performs logical inference on the text. This processed text is passed into the next block where digital signal processing is performed on the processed text. Using many algorithms and transformations this processed text is finally converted into a speech format. This entire process involves the synthesizing of speech.

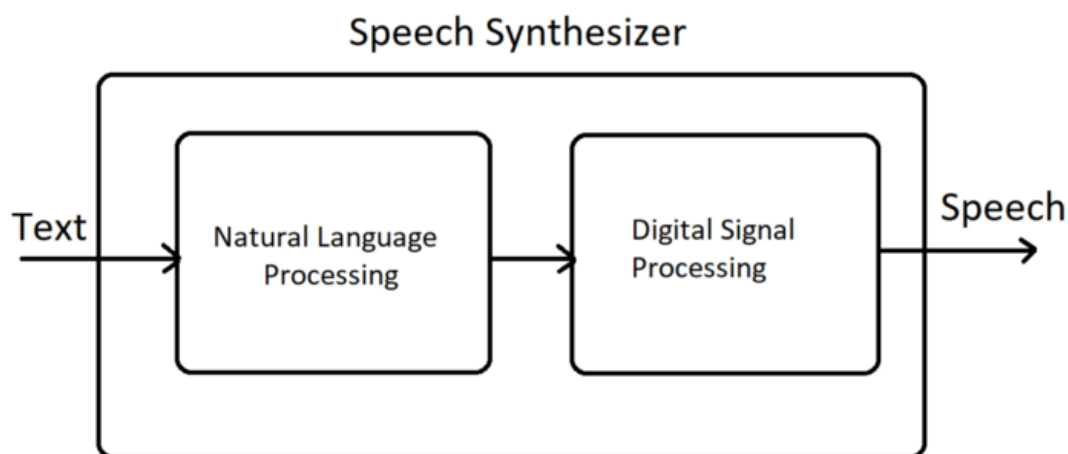


Figure: 3.8: Block diagram of TTS engine

The *gtts* module can be used extensively on other languages such as German, Hindi, French, etc., as well. This is extremely useful when there is a communication barrier and the user is unable to convey his/her messages to people. Text-to-Speech is a great help to the visually impaired people or people with other disabilities as it can help them by assisting in the text to speech translation. There are also many ideas possible with the *gtts* module and it can be used for other languages as well.

3.9.3 PLAYSOUND

The *playsound* module contains only a single function named *playsound()*. It requires a single argument: the path to the audio file that we wish to play. It can be a local file, or a URL. There's an optional second argument, *block*, which is set to *True* by default. We can set it to *False* for making the function run asynchronously. It works with both WAV and MP3 files.

3.9.4 DATETIME

In Python, date and time are not a data type of their own, but a module named *datetime* can be imported to work with the date as well as time. Python *datetime* module comes built into Python, so there is no need to install it externally. The *datetime* module supplies classes to work with date and time. These classes provide a number of functions to deal with dates, times and time intervals. Date and datetime are an object in Python, so when we wish manipulate them, we are actually manipulating objects and not string or timestamps. The module is categorized into 6 main classes:

1. *date* – An idealized naive date, assuming the current Gregorian calendar always was, and always will be, in effect. Its attributes are year, month and day.
2. *time* – An idealized time, independent of any particular day, assuming that every day has exactly 24*60*60 seconds. Its attributes are hour, minute, second, microsecond, and tzinfo.
3. *datetime* – It is a combination of date and time along with the attributes year, month, day, hour, minute, second, microsecond, and tzinfo.
4. *timedelta* – A duration expressing the difference between two date, time, or datetime instances to microsecond resolution.
5. *tzinfo* – It provides time zone information objects.
6. *timezone* – A class that implements the tzinfo abstract base class as a fixed offset from the UTC

3.9.5 WIKIPEDIA

Wikipedia is the largest platform on the internet, which contains tons of information. It is an open-source platform which manages by the community of volunteer editors using a wiki-based editing system. It is a multi-lingual encyclopedia. Python provides the *wikipedia* module (or API) to scrap the data from the Wikipedia pages. This module allows us to get and parse the information from Wikipedia. In simple words, we can say that it is worked as a little scrapper and can scrap only a limited amount of data. The Wikipedia module allows us to search a query supplied as an argument using the *search()* method. This method returns a list of all articles that contain the searched query. The *summary()* method returns the article's summary or topic. This method takes the two arguments - title and sentences and returns the summary in the string format.

3.9.6 PYWHATKIT

Python offers numerous inbuilt libraries to ease our work. Among them *pywhatkit* is a Python library for sending WhatsApp messages at a given time specified by the user, it has several other features too. Some of its other features include playing a youtube video, performing google search, getting information about a particular topic, etc. The function *playonyt()* opens youtube in the default browser and plays the video that we mentioned in the function. If we pass the topic name as parameter, it plays any random video on that topic. On passing the URL of the video as the parameter, it opens that exact video. We can perform a Google search using the *search()* function of this library. It opens our browser and searches for the topic that we have specified in our program. In order to obtain information about a particular topic, we can use the *info()* function that would provide information related to our search query.

3.9.7 TKINTER

Python offers multiple options for developing Graphical User Interface. Out of all the GUI methods, Tkinter is the most commonly used method. It is a standard python interface to the Tk GUI toolkit shipped with python. Python with Tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using Tkinter is an easy task. As with most other modern Tk bindings, Tkinter is implemented as a python wrapper around a complete Tcl interpreter embedded in the python interpreter. Tkinter calls are translated into Tcl commands, which are fed to this embedded interpreter, thus making it possible to mix python and Tcl in a single application. The layered approach used in designing Tkinter gives Tkinter all of the advantages of the TK library. Therefore, at the time of creation, Tkinter inherited from the benefits of a GUI toolkit that had been given time to mature. This makes early versions of Tkinter a lot more stable and reliable than if it had been rewritten from scratch. Moreover, the conversion from Tcl/Tk to Tkinter is really trivial, so that Tk programmers can learn to use Tkinter very easily. Python scripts that use Tkinter do not require modifications to be ported from one platform to the other. Tkinter is available for any platform that Python is implemented for, namely Microsoft Windows, X Windows, and Macintosh. This gives it a great advantage over most competing libraries, which are often restricted to one or two platforms. Moreover, Tkinter will provide the native look-and-feel of the specific platform it runs on.

3.9.8 OS

The OS module in python provides functions for interacting with the operating system. OS comes under python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The *os* and *os.path* modules include many functions to interact with the file system. The currently working directory is the folder in which the python script is saved and being run from. The path to the current working directory can be obtained using *os.getcwd()*. In order to create a directory *os.mkdir()* can be used. In order to delete a directory the *os.rmdir()* may be used, it stands for remove directory. While deleting a folder it is important to note that the current working directory cannot be removed – the directory in which the python script is being executed from. *os.popen()* can be used to create a file within the current working directory. To create a file, the contents of the file and the write mode parameter must be provided as arguments to the above function. To change the current working directory *os.chdir()* method is used. This method changes the current working directory to a specified path. It only takes a single argument as a new directory path. The *os.listdir()* method in python is used to get the list of all files and directories in the specified directory. If we don't specify any directory, then the list of files and directories in the current working directory will be returned. Using the OS module we can remove a file in our system using the *os.remove()* method. To remove a file, we need to pass the name of the file as a parameter. The OS module provides us a layer of abstraction between us and the operating system. When we are working with OS module it is important to always specify the absolute path. Depending upon the operating system the code can run on any OS but we need to change the path exactly. If we try to remove a file that does not exist, we will get *FileNotFoundError*.

3.9.9 PYAUDIO

PyAudio provides Python bindings for PortAudio v19, the cross-platform audio library. With PyAudio, we can easily use Python to play and record audio on a variety of platforms, such as Linux, Microsoft Windows, and Apple MacOS. To use PyAudio, first instantiate PyAudio using *pyaudio.PyAudio()* which sets up the portaudio system. To record or play audio, open a stream on the desired device with the desired audio parameters using *pyaudio.PyAudio.open()*. This sets up a *pyaudio.Stream* to play or record audio. Play audio by writing audio data to the stream using *pyaudio.Stream.write()*, or read audio data from the stream using *pyaudio.Stream.read()*. Use *pyaudio.Stream.stop_stream()* to pause playing/recording, and *pyaudio.Stream.close()* to terminate the stream. Finally, terminate the portaudio session using *pyaudio.PyAudio.terminate()*

3.10 WORKING

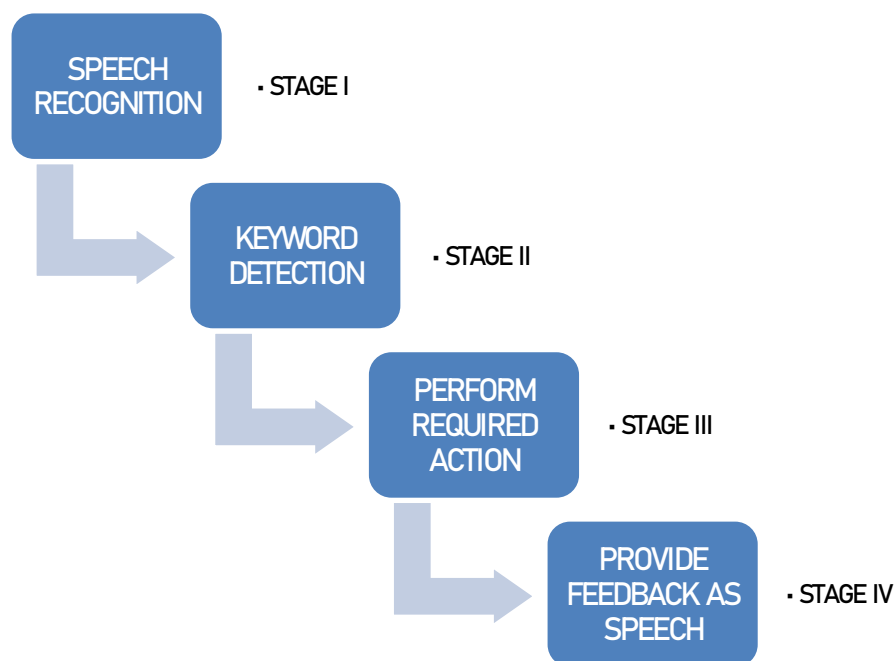


Figure 3.9: Different stages of Sruthi

3.10.1 STAGE 1

The input to this stage is the Malayalam voice command provided by the user. Whenever the user clicks the button, the voice assistant gets initiated. It then instructs the user to speak into the default microphone. The speech input thus provided by the user is processed by the Google STT engine with the aid of the *speech_recognition* library. It then returns a unicode string in Malayalam which is equivalent to the user speech input. This string is passed to the next stage.

3.10.2 STAGE 2

The string obtained after Speech-to-Text conversion is the input to this stage. Now, the string is checked for any keywords. This is done by splitting the string into tokens where each word is matched against a set of predefined keywords. Once found, the next phase begins.

3.10.3 STAGE 3

In this stage, the required action or the necessary computational task corresponding to the matched keyword is performed. For example, if the user instructs the voice recognition system to get information regarding a famous personality, the Wikipedia article that contains information about the personality must be fetched, processed and converted into some suitable format such as string or text file. These tasks are performed in this phase. These instructions

are keyword specific and would vary from keyword to keyword.

3.10.4 STAGE 4

This is the final stage. The voice assistant will have already performed the required task such as fetching a Wikipedia article or the fetching current date or time. Now this information must be read out to the users via speakers. This is accomplished by Text-to-Speech translation of the information obtained in the previous phase and playing the audio/speech using *playsound* library. The TTS conversion is performed with the help of Google TTS engine using *gtts* module.

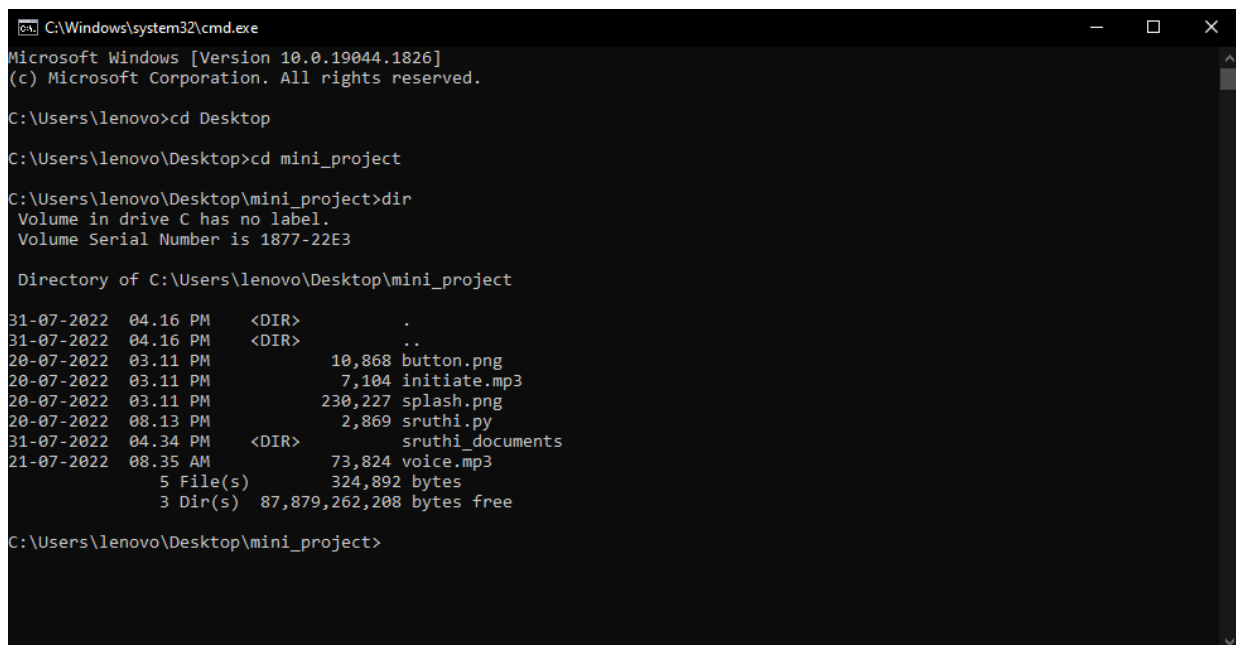
Chapter 4

RESULT

4.1 END RESULT

This section contains screenshots and general instructions that has to be followed while using the voice recognition system. All the screenshots provided below are taken from a Windows operating system environment. The system is also tested to work well with other operating systems.

Step 1. Open the command prompt (in Windows) or any terminal emulator (in Linux and MacOS) and move to the location that contains the python script - *sruthi.py*



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19044.1826]
(c) Microsoft Corporation. All rights reserved.

C:\Users\lenovo>cd Desktop
C:\Users\lenovo\Desktop>cd mini_project
C:\Users\lenovo\Desktop\mini_project>dir
Volume in drive C has no label.
Volume Serial Number is 1877-22E3

Directory of C:\Users\lenovo\Desktop\mini_project

31-07-2022  04.16 PM  <DIR>          .
31-07-2022  04.16 PM  <DIR>          ..
20-07-2022  03.11 PM               10,868 button.png
20-07-2022  03.11 PM                7,104 initiate.mp3
20-07-2022  03.11 PM            230,227 splash.png
20-07-2022  08.13 PM                2,869 sruthi.py
31-07-2022  04.34 PM  <DIR>      sruthi_documents
21-07-2022  08.35 AM            73,824 voice.mp3
               5 File(s)        324,892 bytes
               3 Dir(s)  87,879,262,208 bytes free

C:\Users\lenovo\Desktop\mini_project>
```

Figure 4.1: Change directory to the folder that contains *sruthi.py*

Step 2. Run the python script using the command line - *python sruthi.py*

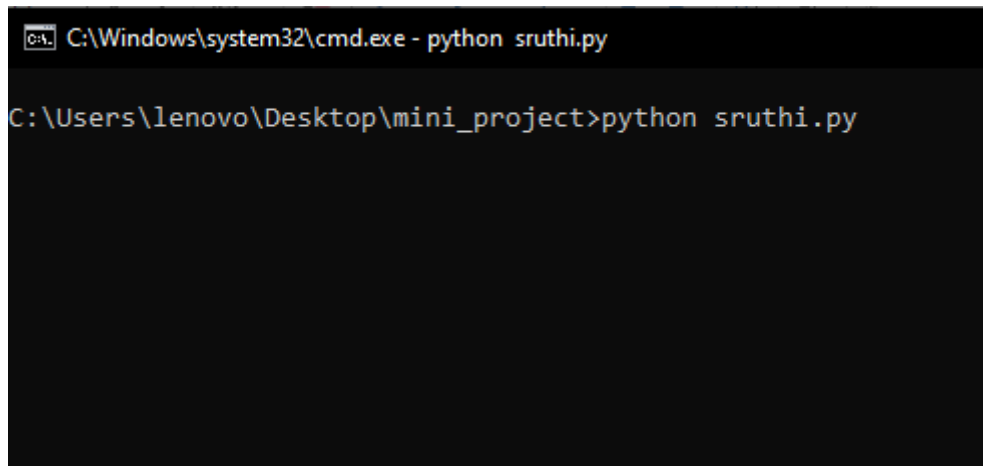


Figure 4.2: Run the python script

Step 3. On clicking the bottom button, the user can provide voice commands to the system

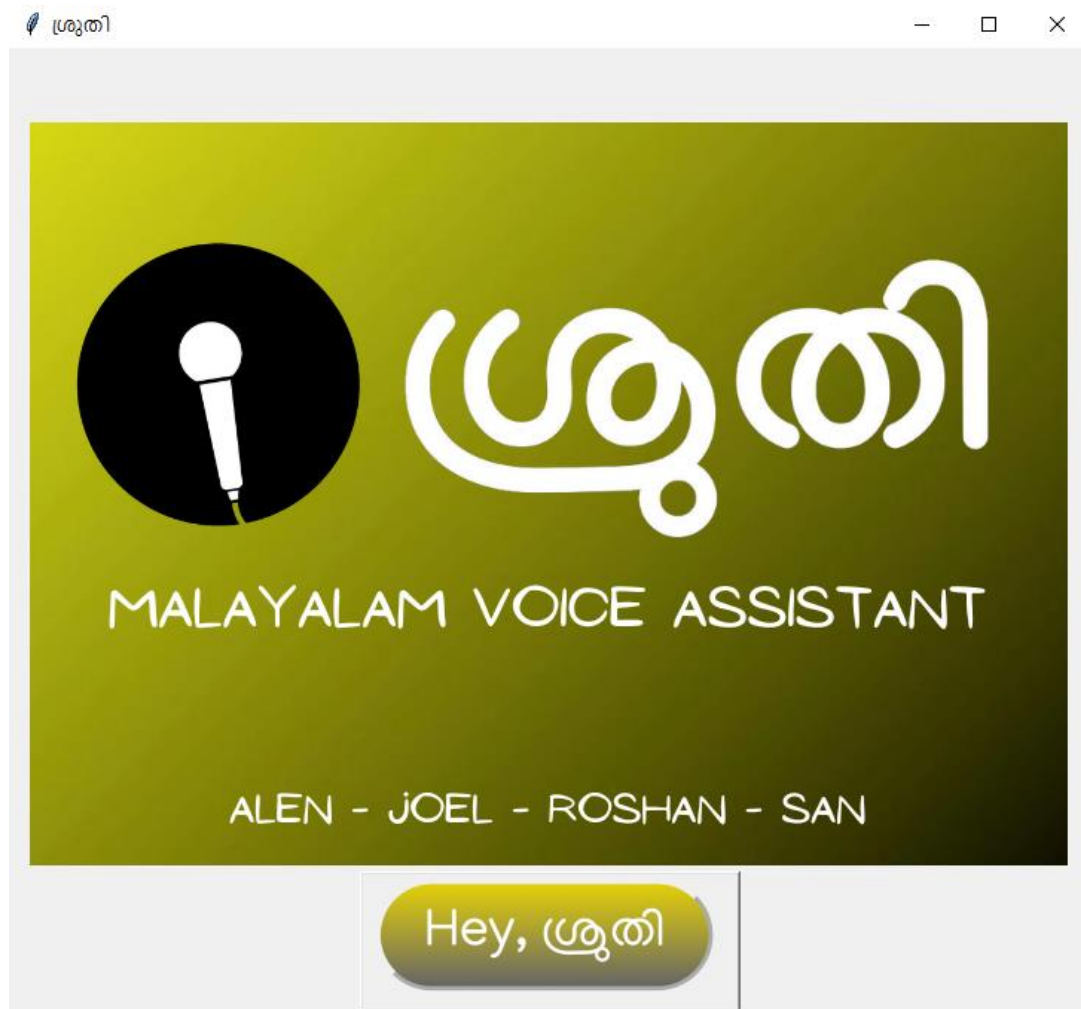


Figure 4.3: Graphical User Interface

4.2 PRODUCT SCREENSHOTS

<< Hey, ശ്രുതി >>

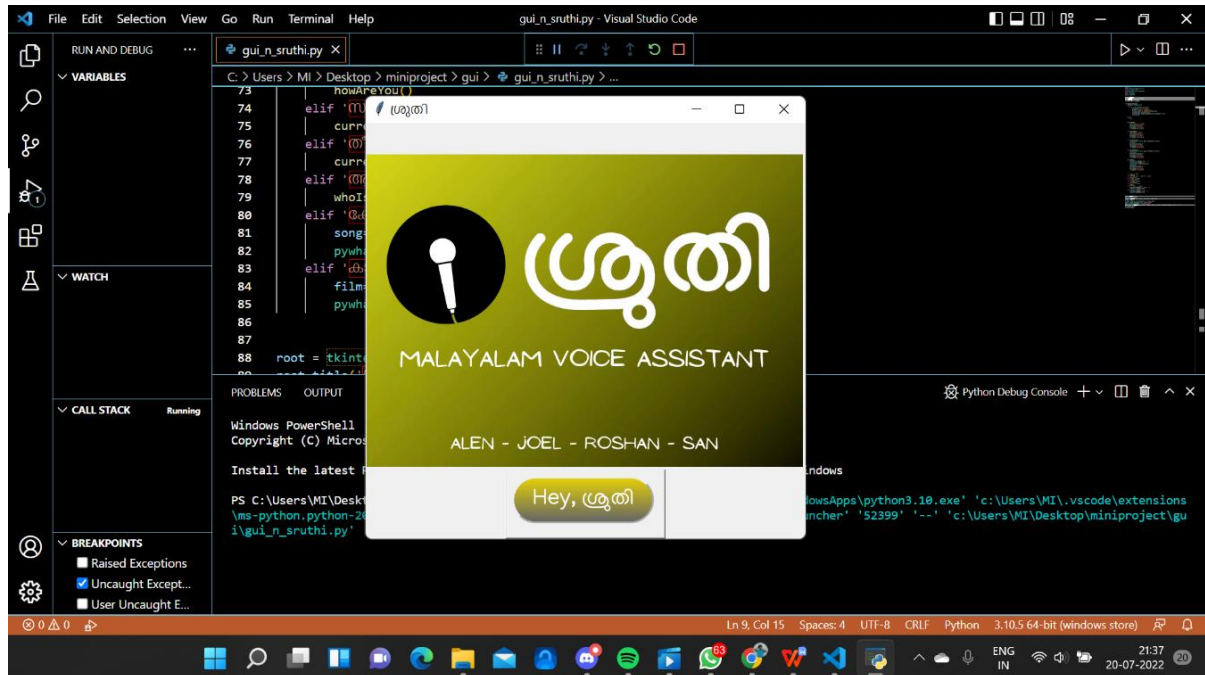


Figure 4.4: Sruthi listens to the user input

<< പേര് >>

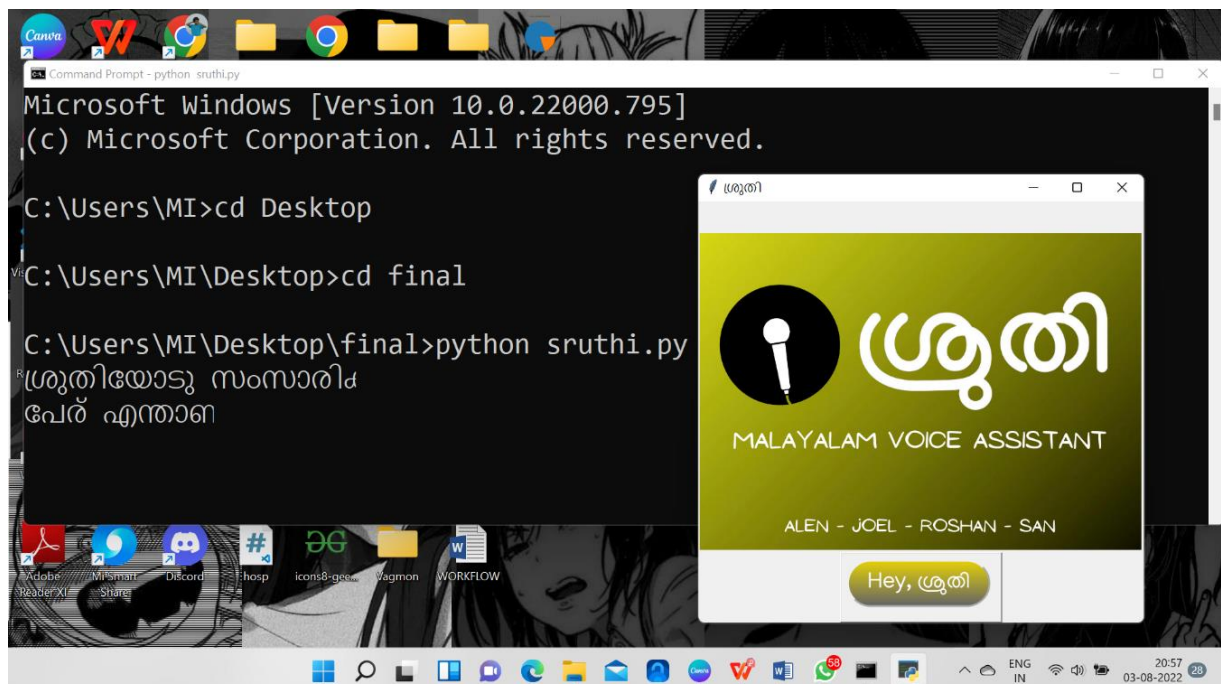


Figure 4.5: Sruthi tells her name

<< എന്താക്കെയുണ്ട് വിശേഷം >>

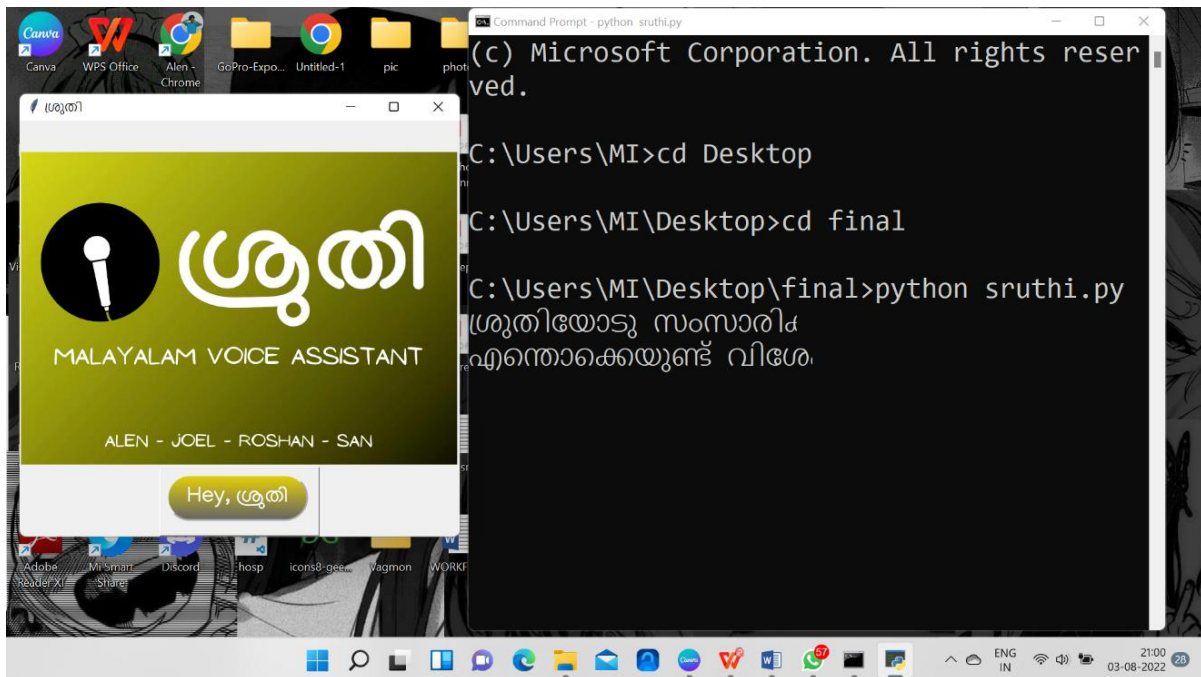


Figure 4.6: Sruthi responds to a greeting

<< തീയതി >>

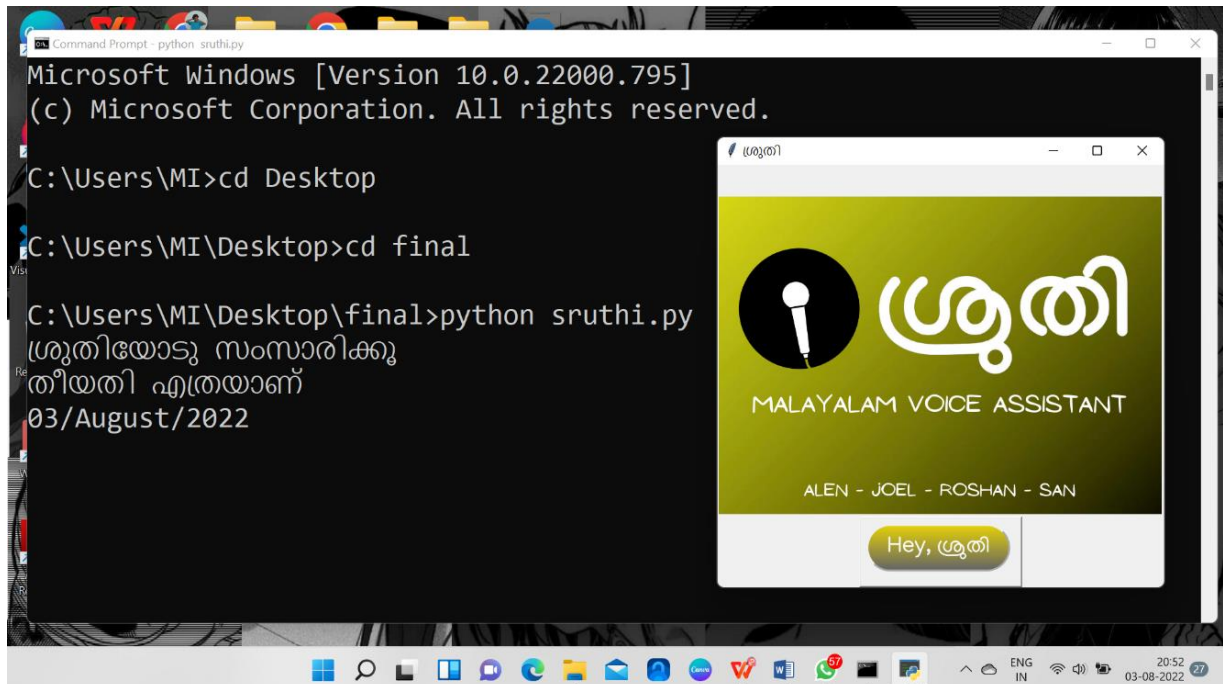


Figure 4.7: Sruthi reads out the current date

<< സമയം >>

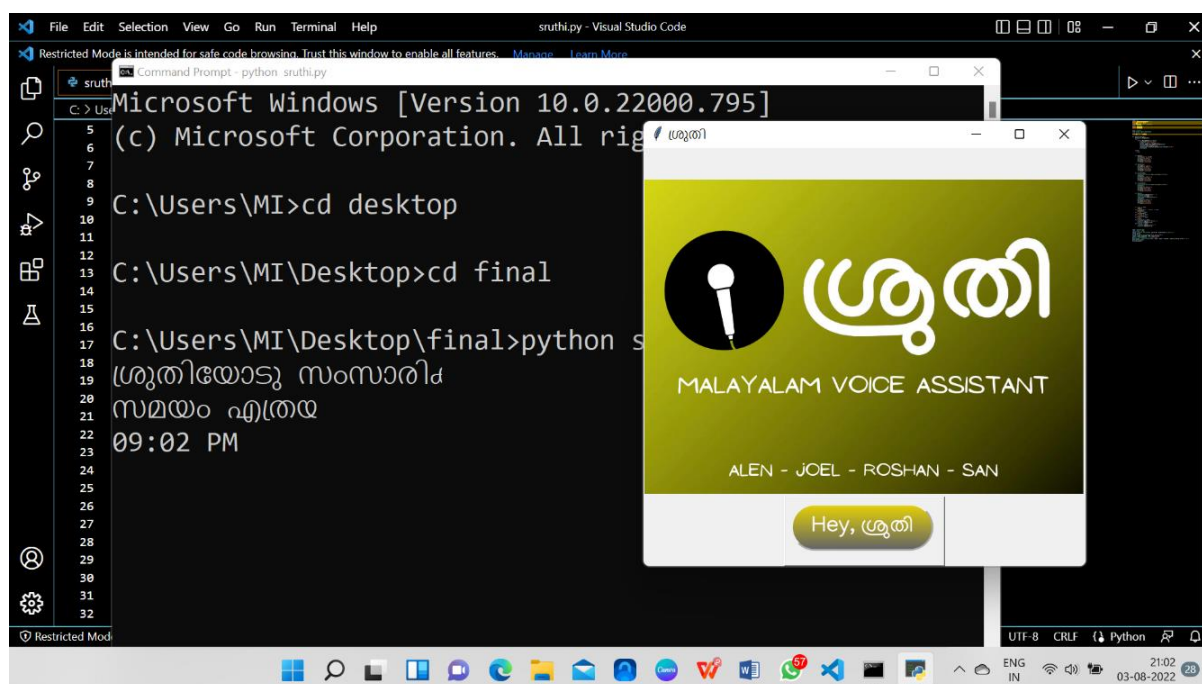


Figure 4.8: Sruthi reads out the current time

<< ആരാൻ >>

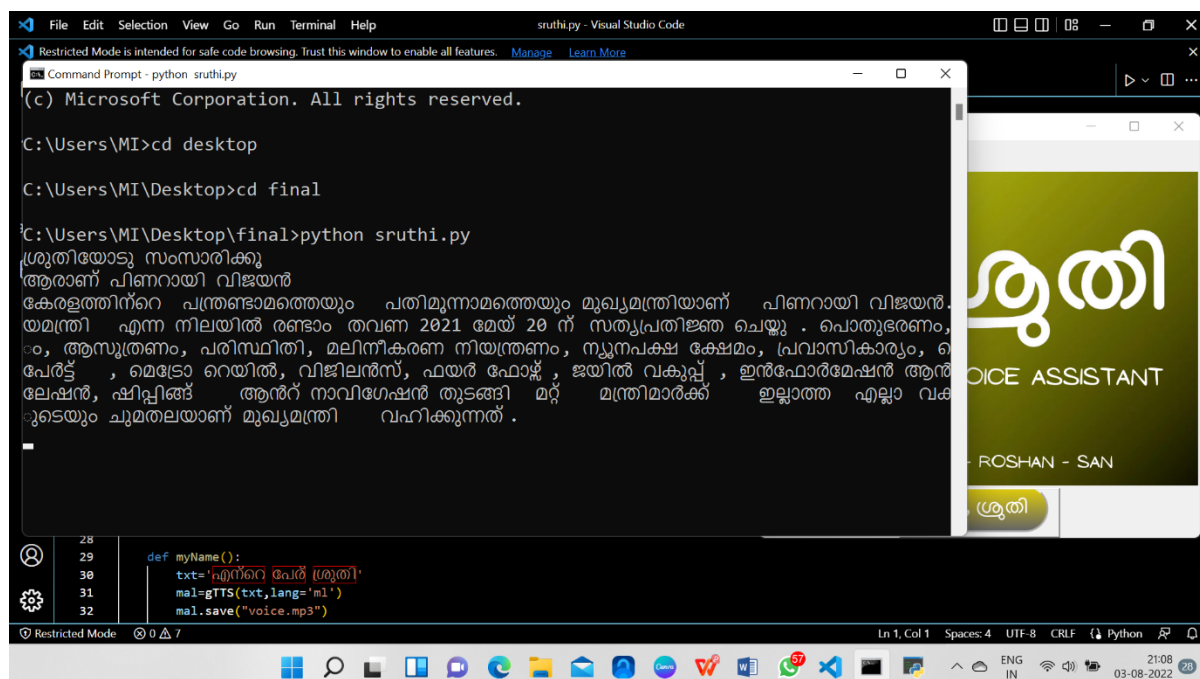


Figure 4.9: Sruthi reads out information from Wikipedia

<< കേൾക്കണം >>

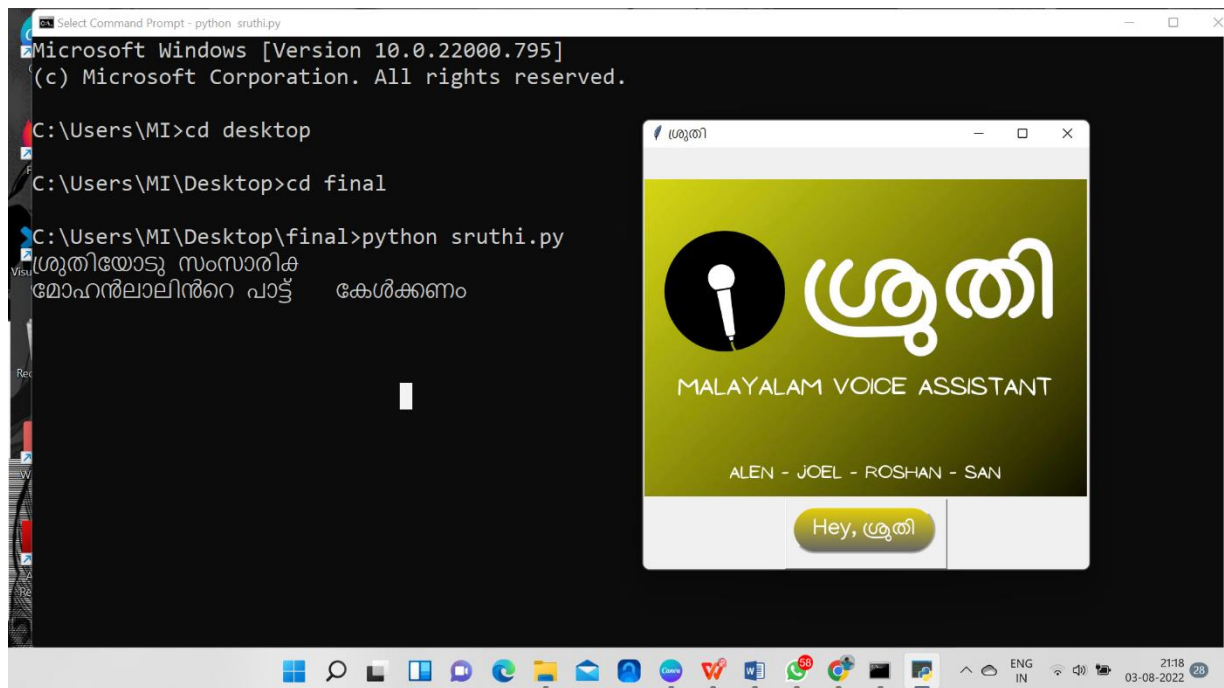


Figure 4.10: Sruthi is asked to play a song

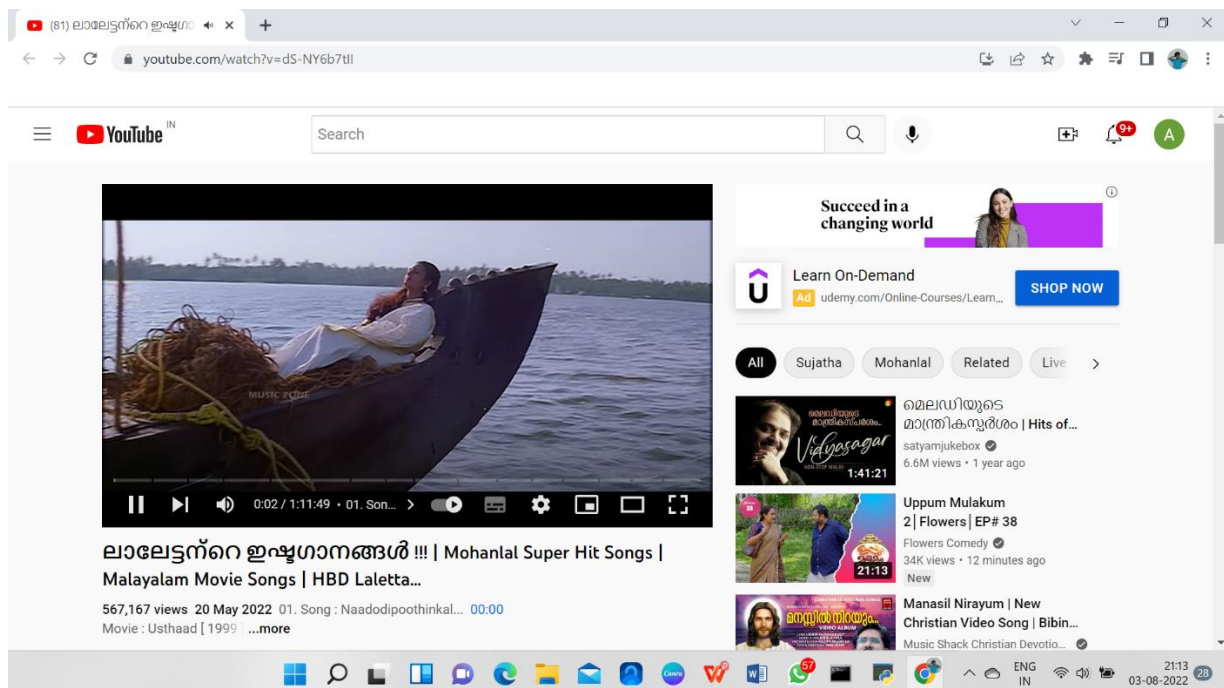


Figure 4.11: The requested youtube video is played

<< തമാശ >>

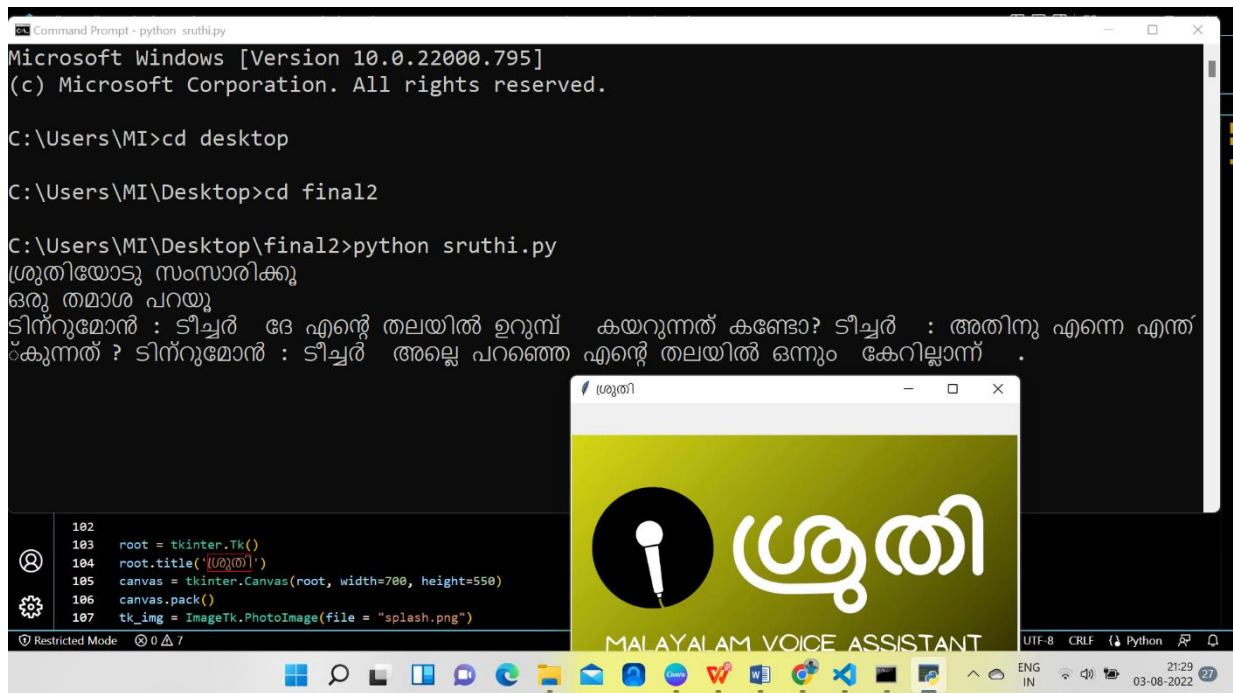


Figure 4.12: Sruthi tells a joke

Chapter 5

CONCLUSION

In short, Sruthi is intended to be a minimal voice assistant that would help visually impaired and those who are unfamiliar with English language. The voice recognition system is easy to use and can be used to perform common tasks like fetching Wikipedia articles, getting the current date and time, entertainment tasks, etc. In future, we would like to add newer features like video calling, email composing, setting up of reminders, etc. We also intend to include AI based recognition techniques that would enable personalized responses and feedbacks. Besides we would like to release our product with an Open-Source license, which would enable everyone to have access to our product and can make modifications according to their needs. In fact, this was one of our primary goals – to create a system that would benefit everyone despite their limitations and disabilities by making this project free for all.

APPENDIX – I

VOICE COMMANDS

The following table shows the complete list of voice commands/keywords that Sruthi currently supports along with a real-world example:

Table A.1: Supported voice commands

Command/Keyword	Example Voice Input
<< പേര് >>	നിന്റെ പേര് എന്താണ്?
<< എന്തൊക്കെയാണ് വിശേഷം >>	നിനക്ക് എന്തൊക്കെയാണ് വിശേഷം ?
<< സമയം >>	സമയം എത്രയായി ?
<< തീയതി >>	ഇന്നത്തെ തീയതി എത്ര ?
<< ആരാണ് >>	ആരാണ് മോഹൻലാൽ ?
<< കേൾക്കണം >> or << കാണണം >>	എനിക്ക് യേശുദാസിന്റെ പാട്ട് കേൾക്കണം
<< തമാശ >>	ഒരു തമാശ പറയാമോ ?

REFERENCES

- [1] Kurian Benoy, Jiby J Puthiyidam – “A Study of Text to Speech Systems for Non-English Languages”, IJRAR June 2019, Volume 6, Issue 2
- [2] Preena Johnson , Jishna K C, Soumya S – “Speech to Text Conversion in Malayalam”, IJLERA July 2017, Volume 2, Issue 7
- [3] Nivendita Singh, Diwakar Yagyasan, Surya Vikram Singh, Gaurav Kumar, Harshit Agarwal – “Voice Assistant Using Python”, IJIRT July 2021, Volume 8, Issue 2
- [4] Vishal Kumar Dhanraj, Lokesh Kripilani, Semal Mahajan – “Research Paper on Desktop Voice Asistant”, IJRES February 2022, Volume 10, Issue 2
- [5] Preethi G, Thiruppugal S, Abishek K, Vishwaa D A – “Voice Assistant using Artificial Intelligence”, IJERT May 2022, Volume 11, Issue 5
- [6] Khushboo Sharma, Disha Bahal, Aman Sharma, Ankita Garg, Neeta Verma – “A Voice Assistant for Disabled Personalities”, IJEAST May 2022, Volume 7, Issue 1
- [7] S Brindha, S Nimrukthi, E Rajasurya, T Stenson – “Intelligent AI based Voice Assistant”, IJCRT June 2022, Volume 10, Issue 6
- [8] Ankush Yadav, Aman Singh, Aniket Sharma, Ankur Sindhu, Umang Rastogi – “Desktop Voice Assistant for the Visually Impaired”, IJRTE July 2020, Volume 9, Issue 2