

Bachelor of Science in Computer Science & Engineering



Voice Assistant with Python

by

Md. Rejwan Kabir Hamim

ID: 1804126

Department of Computer Science & Engineering

Chittagong University of Engineering & Technology (CUET)

Chattogram-4349, Bangladesh.

February, 2023

Chittagong University of Engineering & Technology (CUET)
Department of Computer Science & Engineering
Chattogram-4349, Bangladesh.

Student Name : Md. Rejwan Kabir Hamim

Session : 2020-2021

ID : 1804126

Submitted To : Ashim Dey
: Assistant Professor
Department of Computer Science & Engineering

: Avishek Das
: Lecturer
Department of Computer Science & Engineering

Department : Computer Science & Engineering
Program : B.Sc. Engineering

Project Title : **Voice Assistant with Python**

1 Introduction

The advancement of technology has led to a growing demand for voice-based virtual assistants. The use of voice-based virtual assistants has become increasingly popular in recent years, with people seeking a more convenient and efficient way to interact with technology.

In our project, We have developed a virtual voice assistant using some powerful python libraries and packages. The voice assistant has the capability to perform various tasks including informing about the current time, retrieving information about a person from Wikipedia, opening a website, and playing videos from YouTube. Besides, it can open a desired person profile from codeforces, show problemset of some specific tag and in a rating range from codeforces. It can also open a specific problem from codeforces.

2 Tools and Technologies

The necessary tools to implement this project are illustrated below:

- Personal Computer
- Jupyter Notebook
- Miniconda
- Python libraries:
 - webbrowser
 - speech_recognition
 - pyttsx3
 - pywhatkit
 - datetime
 - wikipedia

- Python Packages:
 - pyaudio
 - pyttsx3
 - pywhatkit

3 Methodology

In this project we have made a virtual voice assistant using python. The methodology used in the development of the voice assistant project was divided into the following stages:

1. Requirements gathering and analysis

In this step, the requirements for the voice assistant were identified and analyzed to determine what functionality was needed to meet the project goals. This included a review of existing voice assistants and their features, as well as a study of the target audience and their needs.

2. Library selection and installation

The necessary libraries for the project were selected and installed. This included speech_recognition for voice recognition, pyttsx3 for text-to-speech conversion, pywhatkit for advanced control of the browser, datetime for working with dates and times, and wikipedia for retrieving data from Wikipedia and pyaudio for audio input.

3. Voice recognition and text-to-speech conversion

The voice assistant was designed to listen to user input through a microphone and convert the audio into text using the speech_recognition library. The text was then converted into speech using the pyttsx3 library.

4. Browser control and Wikipedia integration

The pywhatkit library was used to control the browser. It is also used to play a video from YouTube. We can retrieve information from Wikipedia using the wikipedia library.

5. Implementation and testing


The voice assistant was implemented based on the design. Then it was tested for performance, accuracy, and functionality. The microphone was used as the source of voice input for the assistant.

6. Deployment

The final version of the voice assistant was deployed for use after all necessary refinements were made. The code was thoroughly tested and documented to ensure that it can be easily maintained and updated in the future.

4 Result

Our Virtual assistant will use microphone as source of voice. When we execute our program microphone will be ready for taking voice from user. In this time it will print listening, which means it is ready for receiving voice.



```
listening...
```

Figure 4.1: Voice Assistant ready for listening

We have defined some necessary task which was not defined in existing voice assistant like: Siri or Alexa. We can send different types of voice command and our voice assistant will act according to given command.

We can open a user Codeforces profile by voice command. Here is an example shown in below:

```

listening...
result2:
[]

listening...
result2:
{  'alternative': [  {  'confidence': 0.88646972,
                        'transcript': 'open profile of tourist'},
                      {'transcript': 'profile of tourist'}],
  'final': True}
open profile of tourist
listening...

```

Figure 4.2: Command for opening Tourist profile from codeforces

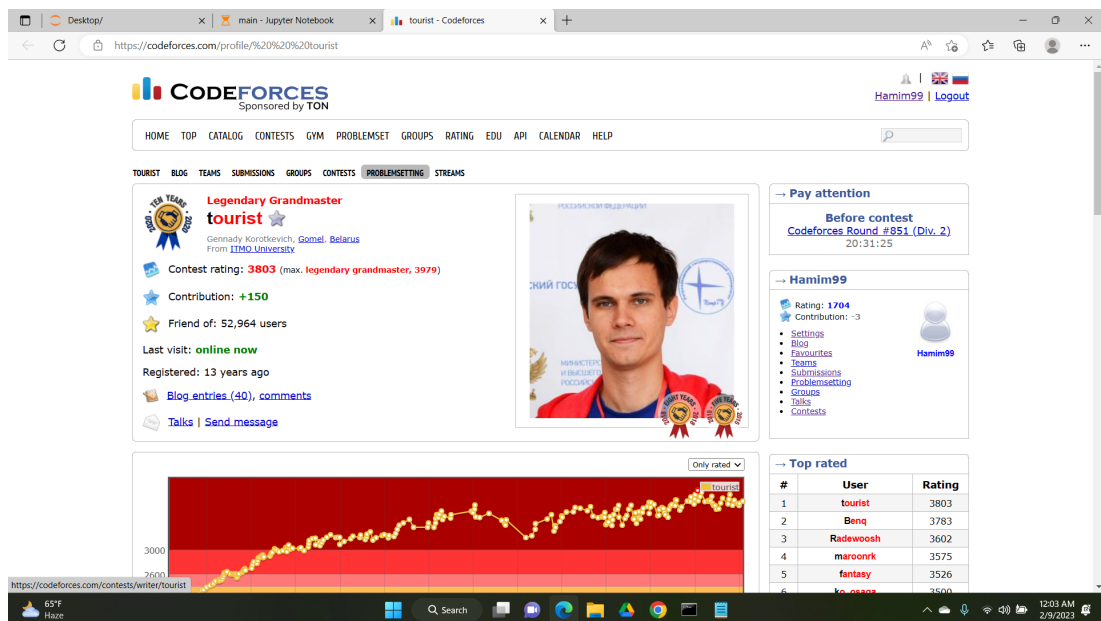


Figure 4.3: Profile of Tourist

Many time, we need to solve topic-wise problem. Which we can do by our voice assistant. It can show problemset of Codeforces with one or multiple tagged. Here is a request for showing dp tagged problem:

```

listening...
result2:
{  'alternative': [  {  'confidence': 0.88528782,
                        'transcript': 'show me problem set of Tag DP'},
                      {'transcript': 'show me problem set of DP'},
                      {'transcript': 'show me problem set off tag DP'}],
  'final': True}
show me problem set of tag dp
listening...

```

Figure 4.4: Command for showing problemset with dp tag

#	Name	Tags	Rating
1792D	Fixed Prefix Permutations	binary search, bitmasks, data structures, hashing, math, sortings	1700
1792C	Min Max Sort	binary search, brute force, greedy, math, two pointers	1500
1792B	Stand-up Comedian	greedy, math	1200
1791G2	Teleporters (Hard Version)	binary search, greedy, sortings	1900
1791G1	Teleporters (Easy Version)	greedy, sortings	1100
1791F	Range Update Point Query	binary search, brute force, data structures	1500
1791E	Negatives and Positives	dp, greedy, sortings	1100
1791D	Distinct Split	brute force, greedy, strings	1000
1790E	Vlad and a Pair of Numbers	bitmasks, constructive algorithms	1400
1790D	Matryoshkas	data structures, greedy, sortings	1200
1790C	Permutation	brute force, implementation, math	1000
1787D	Game on Axis	combinatorics, dfs and similar, dsu, graphs, implementation	1900
1787C	Remove the Bracket	dp, greedy, math	1600

Figure 4.5: Dp tagged problemset

Thus, it can show Codeforces problemset in a specific rating range.

```

listening...
result2:
{  'alternative': [  {  'confidence': 0.87203586,
                      'transcript': 'show me problem Set in rating range '
                                     '900 to 2000'},
                    {  'transcript': 'show me problem Set in writing range '
                                     '900 to 2000'},
                    {  'transcript': 'show me problem Set in writing 900 '
                                     'to 2000'},
                    {  'transcript': 'show me problem Set in rating range '
                                     '900 to'}}],
    'final': True}
show me problem set in rating range 900 to 2000
listening...

```

Figure 4.6: Command for showing problemset in a specific rating range

Our voice assistant can open video from Youtube. Here is an example of opening an YouTube video:

```

webbrowser.open_new("https://codeforces.com/problemset/problem/" + num + "/" + url)

elif 'play' in command:
    song = command.replace('play', '')
    talk('playing ' + song)
    pywhatkit.playonyt(song)
elif 'time' in command:
    time = datetime.datetime.now().strftime('%I:%M %p')
    talk('current time is ' + time)
elif 'who is' in command:
    person = command.replace('who is', '')
    info = wikipedia.summary(person, 1)
    print(info)
    talk(info)
elif 'open' in command:
    url = command.replace('open', '')
    talk('opening ' + url)
    url = url.replace(' ', '')
    webbrowser.open_new(url)
else:
    tmp=1;
    talk("sorry you say nothing")
return tmp

In [8]: while True:
        tmp=run_alice()
        if(tmp==1):
            break

listening...
result2:
{ 'alternative': [ { 'confidence': 0.88687533,
                    'transcript': 'play FIFA World cup theme song'}],
  'final': True}
play fifa world cup theme song
listening...

```

Figure 4.7: Command for opening an Youtube Video

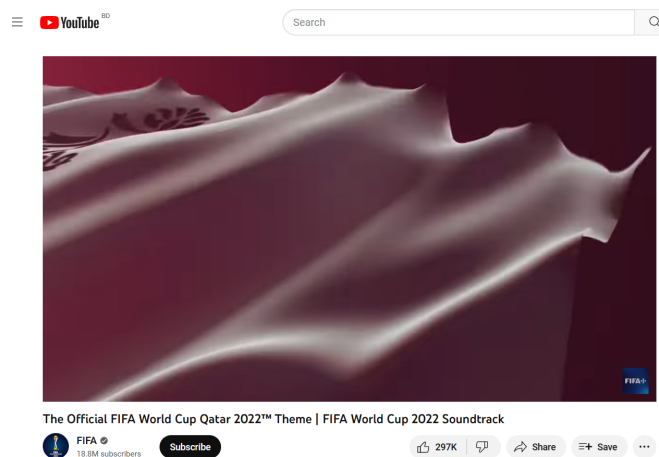


Figure 4.8: Resulting Youtube video

This Assistant can deal with Wikipedia also, It can find Information about famous People or object which are present in wikipedia.

```

listening...
result2:
{ 'alternative': [ { 'confidence': 0.88687539,
                    'transcript': 'who is Mark Zuckerberg'}],
  'final': True}
who is mark zuckerberg
Mark Elliot Zuckerberg (; born (1984-05-14)May 14, 1984) is an American business magnate, internet entrepreneur, and philanthro-
pist.

```

Figure 4.9: Finding information about Mark Zuckerberg from Wikipedia

Besides, Our virtual voice assistant can inform about time, finding a problem from Codeforces. By this, we can also go to desired website.

5 Conclusion

In conclusion, the development of a voice assistant using Python and its associated libraries was a dynamic and enlightening process. The integration of libraries such as `speech_recognition`, `pyttsx3`, `pywhatkit`, `datetime`, and `wikipedia` brought the voice assistant to life, enabling it to recognize voice commands, respond with text-to-speech conversion, control the browser, access information from Wikipedia and even play videos from YouTube.

This project showcases the versatility and capability of Python programming language in developing innovative and practical applications that can enhance the lives of its users. Further modification on this project can make it more powerful and user friendly which will be considered in future.