

Python Project – Study Aid

(An AI- Assistant for students, capable of interacting and responding to computational, mechanical, common life questions using speech recognition)

Faculty In charge – Dr. Moin Hasan

Department – Python

Lovely Professional University

Team Members-

1) **Prakhar Saxena**

Roll No. - 10

Reg No. - 11911235

2) **Pranjali Jain**

Roll No. - 36

Reg No. - 11909291

3) **Bharat Mundra**

Roll No. - 09

Reg No. - 11911219

Section- K19KX

Session- 2020-2021

Submission Date – 31st October 2020

ACKNOWLEDGEMENT

I would like to express my deep sense of gratitude to the Python Language Department, Lovely Professional University, Punjab, for giving us an opportunity to do this Python Project which helped us to understand the real working and coding environment.

First and for most, we would like to express our deep sense of gratitude to Dr. Moin Hasan (Faculty in charge – Python) , who have been a source of inspiration throughout our course their guidance and advice have made this work possible.

We are very grateful to him for his advice and guidance throughout this project. We also wish to present our gratitude to our parents, and all our friends who have directly and indirectly helped to bring out this project.

Contents

S.No.	Title	Page No.
1.	Abbreviations	4
2.	Abstract - Introduction	5
3.	Skills- StudyAid	6
4.	Packages Required	6-7
5.	Speech API, Driver Code	8-11
6.	Individual Skill Description	12-19
7.	Research Gap	19
8.	Conclusion and Demo link	20
9.	Bibliography	21
10	Contribution Page	22

Abbreviations used in Study Aid

These are the list of abbreviations, along with their full-forms for better understanding of the project.

- **pyttsx3** - `pyttsx3` is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline, and is compatible with both Python 2 and 3.
- **OS**- The OS module in python provides functions for interacting with the operating system. OS, comes under Python's standard utility modules.
- **Json- JavaScript Object Notation**, an open standard file format, and data interchange format, that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and array data types
- **Sapi5- The Speech Application Programming Interface or SAPI** is an API developed by Microsoft to allow the use of speech recognition and speech synthesis within Windows applications.
- **elif** – Else if loop
- **str**- string
- **REST**- REpresentational State Transfer is emerged for standard design for creating web services and web APIs.
- **PIP** - pip is the package installer for Python. We can use pip to install packages from the Python Package Index and other indexes.

ABSTRACT

As his 2016's challenge, Facebook co-founder and CEO Mark Zuckerberg announced that he'll be coding his own AI personal assistant. While we do not know about the progress made by him in this endeavour, there are many existing resources that inspired us to create something similar for ourselves.

An AI personal assistant is a piece of software that understands verbal or written commands and completes task assigned by the client.

We built the AI Assistant bot to work as a study aid for students. In this period of pandemic most of the education has been transferred to the online mode making it very hectic for everyone. This AI Bot acts as an aid for students through which they can jot down their notes or ask the bot to dictate the notes. The bot can also help the student to solve various computational and mathematical question along with answering various general questions that the student had in mind.

INTRODUCTION

We built it using Python language. We considered it for this project as it is one of the most popular programming languages and it comes with several choices to get started our own AI Assistant project. There are many open libraries for speech recognition and synthesis available in Python which could play a very important role in development of this project.

We call our AI assistant **Steve** and have coding process in three parts: *Steve's Mouth*, *Steve's Ears*, and *Steve's Brain*.

As the name suggests, Steve's Mouth deals with text-to-speech conversion process. While many Python libraries are available that offer voice recognition and speech synthesis, we chose to move ahead with pyttsx — an offline, free and open source resource. It is also updated to work with Python 3. To use it, we need to install JPercent's version of `pyttsx` by running the command `pip install pyttsx`. For Windows, we had to install PyWin32 and Microsoft Speech API.

For speech recognition, Steve's ears, we have used SpeechRecognition. This great resource offers the freedom to use Sphinx project to convert the audio input into a text. We could have also used Google services and Wit.ai to do the same with the help of SpeechRecognition but we mainly focussed on enabling the bot to be functional in offline conditions also therefore we moved forward with pyttsx.

Now coming to Steve's Brain, it basically comprises of the code we have written to enable it to answer various computational, mathematical and geographical questions. We have incorporated several python libraries for these tasks. Some of the task that Steve can perform are - Wikipedia search, Opening Gmail, YouTube, Google Search, Setting Alarm, Weather update, Making Shopping list, dictating notes, etc.

SKILLS

The implemented voice assistant – ‘*Steve*’ can perform the following task—

It can open YouTube, Gmail, Google chrome and stack overflow. Predict current time, take a photo, search Wikipedia to abstract required data, predict weather in different cities, get top headline news from Times of India and can answer computational and geographical questions too. Along with the above functionalities we have also added a functionality to aid students to jot down notes for them using speech to text conversion whereas it can also dictate answers or text, helping students to write down their assignments.

Packages Required:

To build Steve and to enable it to act as student’s Study-Aid, it is necessary to install the following packages in your/our system using the pip command.

PIP is a package manager for Python packages and modules in Python 3.4 or above, PIP is installed by default.

```
1  import speech_recognition as sr
2  import pyttsx3
3  import datetime
4  import wikipedia
5  import webbrowser
6  import os
7  import time
8  import subprocess
9  from ecapture import ecapture as ec
10 import wolframalpha
11 import json
12 import requests
```

1) Speech Recognition — Speech recognition is an important feature used in house automation and in artificial intelligence devices. The main function of this library is it tries to understand whatever the humans speak and converts the speech to text.

2) pyttsx3 — pyttsx3 is a text to speech conversion library in python. This package supports text to speech engines on Mac OS x, Windows and on Linux.

- 3) **Wikipedia** — Wikipedia is a multilingual online encyclopaedia used by many people from academic community ranging from freshmen to students to professors who wants to gain information over a particular topic. This package in python extracts data required from Wikipedia.
- 4) **ecapture** — This module is used to capture images from our camera
- 5) **datetime** — This is an inbuilt module in python and it works on date and time
- 6) **OS** — This module is a standard library in python and it provides the function to interact with operating system
- 7) **time** — The time module helps us to display time
- 8) **Web browser** — This is an in-built package in python. It extracts data from the web
- 9) **Subprocess** — This is a standard library use to process various system commands like to log off or to restart our PC.
- 10) **Json-** The json module is used for storing and exchanging data.
- 11) **request-** The request module is used to send all types of HTTP request. Its accepts URL as parameters and gives access to the given URL'S.
- 12) **Wolfram Alpha** — Wolfram Alpha is an API which can compute expert-level answers using Wolfram's algorithms, knowledge base and AI technology. It is made possible by the Wolfram Language.

Setting up the speech engine

The **pyttsx3** module is stored in a variable name engine.

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline and is compatible with both Python 2 and 3. An application invokes the `pyttsx3.init()` factory function to get a reference to a `pyttsx3`. Engine instance. it is a very easy to use tool which converts the entered text into speech.

The `pyttsx3` module supports two voices first is female and the second is male which is provided by “sapi5” for windows.

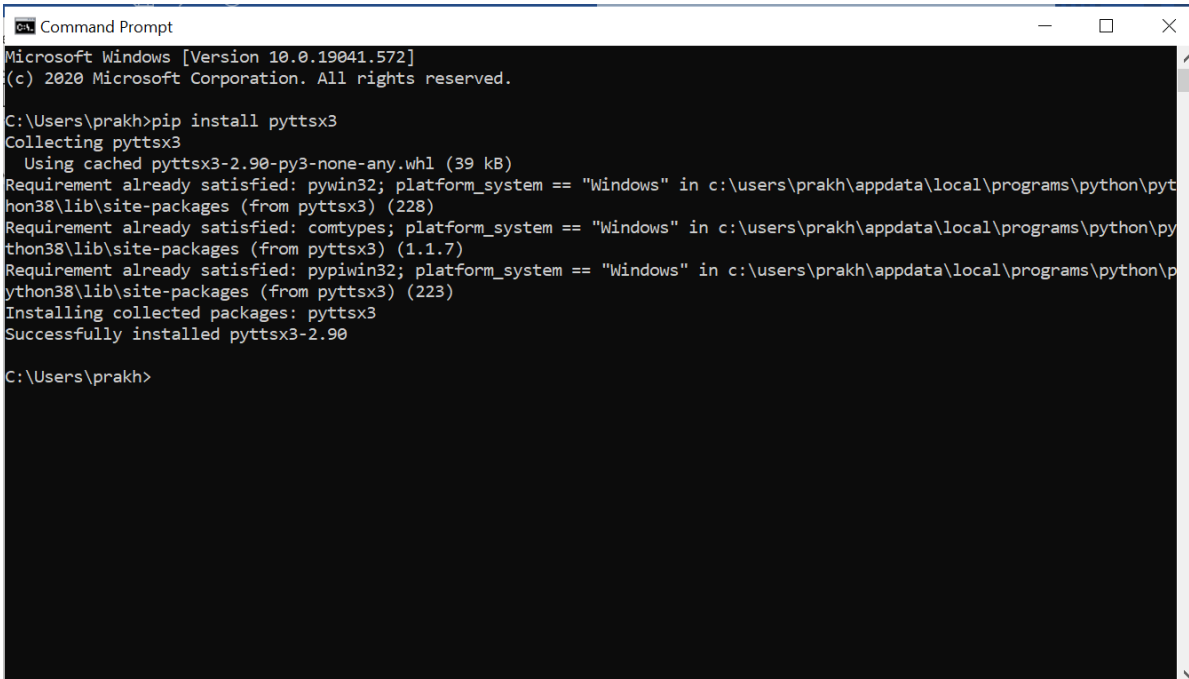
It supports three TTS engines :

- *sapi5* – SAPI5 on Windows
- *nsss* – NSSpeechSynthesizer on Mac OS X
- *espeak* – eSpeak on every other platform

Installation-

To install the `pyttsx3` module, first of all, we have to open the terminal and write-

```
pip install pyttsx3
```



```
C:\Users\prakh>pip install pyttsx3
Collecting pyttsx3
  Using cached pyttsx3-2.90-py3-none-any.whl (39 kB)
Requirement already satisfied: pywin32; platform_system == "Windows" in c:\users\prakh\appdata\local\programs\python\python38\lib\site-packages (from pyttsx3) (228)
Requirement already satisfied: comtypes; platform_system == "Windows" in c:\users\prakh\appdata\local\programs\python\python38\lib\site-packages (from pyttsx3) (1.1.7)
Requirement already satisfied: pypiwin32; platform_system == "Windows" in c:\users\prakh\appdata\local\programs\python\python38\lib\site-packages (from pyttsx3) (223)
Installing collected packages: pyttsx3
Successfully installed pyttsx3-2.90
C:\Users\prakh>
```

Since we had pre-installed `pypiwin32`, therefore the installation went on smoothly but if you receive errors such as No module named `win32com.client`, No module named `win32`, or No module named `win32api`, you will need to additionally install `pypiwin32`. It can work on any platform. Now we are all set to write a program for conversion of text to speech.

Sapi5 is a Microsoft Text to speech engine used for voice recognition. The **Speech Application Programming Interface** or **SAPI** is an API developed by Microsoft to allow the use of speech recognition and speech synthesis within Windows applications.

The voice Id can be set as either 0 or 1,

- **0 indicates Male voice**
- **1 indicates Female voice**

```
1 engine=pyttsx3.init('sapi5')
2 voices=engine.getProperty('voices')
3 engine.setProperty('voice', voices[0].id')
```

Now we defined a function **speak** which converts text to speech. The speak function takes the text as its argument, further initialize the engine.

runAndWait: This function Blocks while processing all currently queued commands. It Invokes call-backs for engine notifications appropriately and returns back when all commands queued before this call are emptied from the queue.

Initiate a function to greet the user:

- Define a function **wishMe** for the AI assistant to greet the user.
- The **now().hour** function abstract's the hour from the current time.
- If the hour is greater than zero and less than 12, the voice assistant wishes you with the message "Good Morning".
- If the hour is greater than 12 and less than 18, the voice assistant wishes you with the following message "Good Afternoon".
- Else it voices out the message "Good evening"

```
1  def wishMe():
2      hour=datetime.datetime.now().hour
3      if hour>=0 and hour<12:
4          speak("Hello,Good Morning")
5          print("Hello,Good Morning")
6      elif hour>=12 and hour<18:
7          speak("Hello,Good Afternoon")
8          print("Hello,Good Afternoon")
9      else:
10         speak("Hello,Good Evening")
11         print("Hello,Good Evening")
```

Setting up the command function for our AI assistant, “Steve”:

Defined a function **takecommand** for the AI assistant to understand and to accept human language. The microphone captures the human speech and the recognizer recognizes the speech to give a response.

The exception handling is used to handle the exception during the run time error and, the **recognize_google** function uses google audio to recognize speech.

```
1  def takeCommand():
2      r=sr.Recognizer()
3      with sr.Microphone() as source:
4          print("Listening...")
5          audio=r.listen(source)
6
7      try:
8          statement=r.recognize_google(audio,language='en-in')
9          print(f"user said:{statement}\n")
10
11     except Exception as e:
12         speak("Pardon me, please say that again")
13         return "None"
14     return statement
15
16 speak("Loading your AI personal assistant Steve")
17 wishMe()
```

The Main function:

The main function starts from here, the commands given by the humans is stored in the variable **statement**.

```
1  if __name__=='__main__':
2
3
4      while True:
5          speak("Tell me how can I help you now?")
6          statement = takeCommand().lower()
7          if statement==0:
8              continue
```

If the following trigger words are there in the statement given by the users it invokes the virtual assistant to speak the below following commands.

```
1  if "good bye" in statement or "ok bye" in statement or "stop" in statement or "bye" in statement:
2      speak('your personal assistant Steve is shutting down,Good bye')
3      print('your personal assistant Steve is shutting down,Good bye')
4      break
5
```

Skill 1 -Fetching data from Wikipedia:

The following commands helps to extract information from Wikipedia.

The **wikipedia.summary()** function takes two arguments, the statement given by the user and how many sentences from Wikipedia is needed to be extracted is stored in a variable **result**.

```
1  if 'wikipedia' in statement:
2      speak('Searching Wikipedia...')
3      statement =statement.replace("wikipedia", "")
4      results = wikipedia.summary(statement, sentences=3)
5      speak("According to Wikipedia")
6      print(results)
7      speak(results)
```

Skill 2 -Accessing the Web Browsers — Google chrome , G-Mail and YouTube:

The web browser extracts data from web. The **open_new_tab** function accepts **URL** as a parameter that needs to be accessed.

The **Python time sleep function** is used to add delay in the execution of a program. We can use this function to halt the execution of the program for given **time** in seconds.

```
1         elif 'open youtube' in statement:
2             webbrowser.open_new_tab("https://www.youtube.com")
3             speak("youtube is open now")
4             time.sleep(5)
5
6         elif 'open google' in statement:
7             webbrowser.open_new_tab("https://www.google.com")
8             speak("Google chrome is open now")
9             time.sleep(5)
10
11        elif 'open gmail' in statement:
12            webbrowser.open_new_tab("gmail.com")
13            speak("Google Mail open now")
14            time.sleep(5)
```

Skill 3 -Predicting time:

The current time is abstracted from **datetime.now()** function which displays the hour, minute and second and is stored in a variable name **strTime**.

```
1
2         elif 'time' in statement:
3             strTime=datetime.datetime.now().strftime("%H:%M:%S")
4             speak(f"the time is {strTime}")
```

Skill 4 -To fetch latest news:

If the user wants to know the latest news, the voice assistant is programmed to fetch top headline news from Time of India by using the web browser function.

Skill 5 -Capturing photo:

The **ec.capture()** function is used to capture images from your camera. It accepts 3 parameters.

Camera index — The first connected webcam will be indicated as index 0 and the next webcam will be indicated as index 1

Window name — It can be a variable or a string. If we don't wish to see the window, type as False

Save name — A name can be given to the image and if we don't want to save the image, type as false

```
1         elif 'news' in statement:
2             news = webbrowser.open_new_tab("https://timesofindia.indiatimes.com/home/headlines")
3             speak('Here are some headlines from the Times of India,Happy reading')
4             time.sleep(6)
5
6         elif "camera" in statement or "take a photo" in statement:
7             ec.capture(0,"robo camera","img.jpg")
```

Skill 6-Searching data from web:

From the **web browser** we can **search** required data by passing the user statement (command) to the **open_new_tab()** function.

User: Hey Steve , please search images of butterfly

The Voice assistant opens the google window & fetches butterfly images from web.

```
1         elif 'search' in statement:
2             statement = statement.replace("search", "")
3             webbrowser.open_new_tab(statement)
4             time.sleep(5)
```

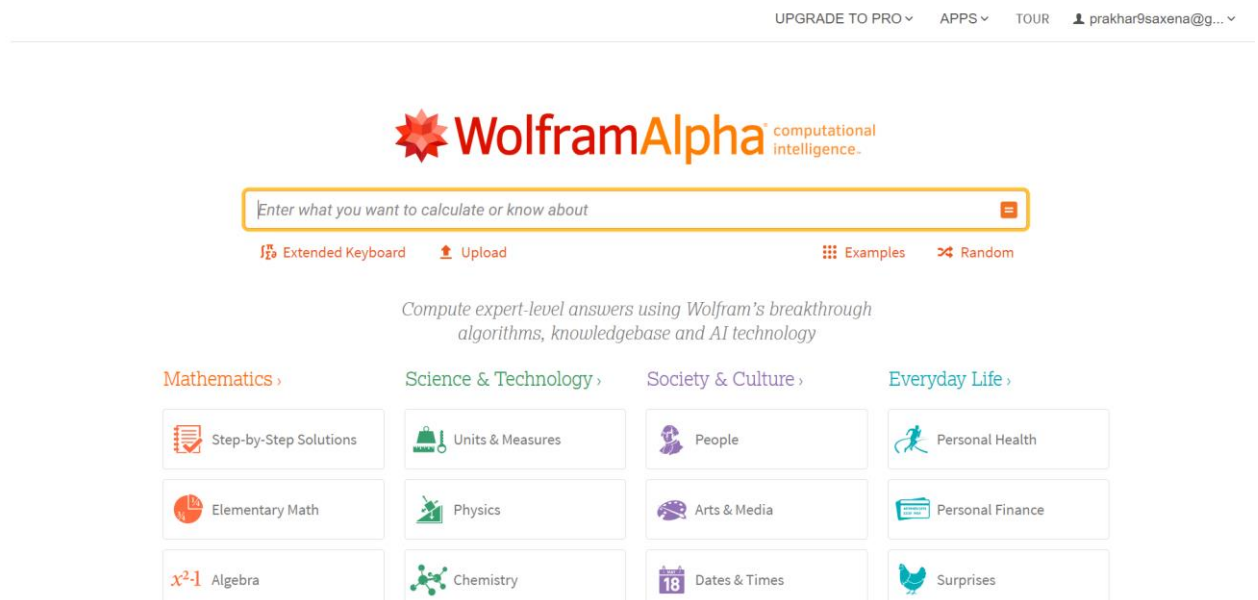
Skill 7- Setting our AI assistant to answer geographical and computational questions:

Here we can use a third-party API called **Wolfram alpha API** to answer computational and geographical questions. It is made possible by the Wolfram Language. The **client** is an instance (class) created for wolfram alpha. The **res** variable stores the response given by the wolfram alpha.

```
1 elif 'ask' in statement or "what" in statement or "who" in statement:
2     speak('I can answer to computational and geographical questions and what question do you want to ask now')
3     question=takeCommand()
4     app_id="R2K75H-7ELALHR35X"
5     client = wolframalpha.Client('R2K75H-7ELALHR35X')
6     res = client.query(question)
7     answer = next(res.results).text
8     speak(answer)
9     print(answer)|
```

To access the wolfram alpha API an unique App ID is required which can be generated by the following ways:

1. Login to the official page of wolfram alpha and we created an account as we did not possess one.



2. Enter details of the project, and created our wolfram API Key

Sign up for FREE access to the Wolfram|Alpha API

Before you get started, please provide us with some additional information:

How are you planning to use the Wolfram|Alpha API?

For my Virtual AI Assistant - Steve
Study- AID|
Python Project

Phone (optional)

+918765186212

☒ I have read and agree with the Wolfram|Alpha API [Terms of Use](#).

Sign up

Human: *Hey Steve, what is the capital of Uttar-Pradesh?*

Steve Voice assistant: *Lucknow, Uttar-Pradesh, India*

Skill 8- Extra features:

It was interesting to program our AI assistant to answer the following questions like what it can and who created it, isn't it?

```
1 elif 'who are you' in statement or 'what can you do' in statement:
2     speak('I am Steve version 1 point 0 your personal assistant. I am programmed to minor tasks like'
3         'opening youtube,google chrome,gmail and stackoverflow ,predict time,take a photo,search wikipedia,predict weather'
4         'in different cities , get top headline news from times of india and you can ask me computational or geographical questions too!')
5
6
7 elif "who made you" in statement or "who created you" in statement or "who discovered you" in statement:
8     speak("I was built by Sir Prakhhar Saxena and Pranjali Ma'am")
9     print("I was built by Mr. Prakhhar Saxena & Ms. Pranjali Jain")
```

Skill 9- To forecast weather:

Now to program our AI assistant to detect weather we need to generate an API key from Open Weather map.

Open weather map is an online service which provides weather data. By generating an API ID in the official website, we can use the APP_ID to make our voice assistant detect weather of all places whenever required. The necessary modules needed to be imported for this weather detection is json and request module.

The **city_name variable** takes the command given by the human using the **takeCommand()** function.

The **get** method of **request** module returns a **response** object. And the **json** methods of response object converts json format data into python format.

The variable **X** contains list of nested dictionaries which checks whether the value of 'COD' is 404 or not that is if the city is found or not.

The values such as temperature and humidity is stored in the main key of variable **Y**.

```

1         elif "weather" in statement:
2             api_key="Apply your unique ID"
3             base_url="https://api.openweathermap.org/data/2.5/weather?"
4             speak("what is the city name")
5             city_name=takeCommand()
6             complete_url=base_url+"appid="+api_key+"&q="+city_name
7             response = requests.get(complete_url)
8             x=response.json()
9             if x["cod"]!="404":
10                y=x["main"]
11                current_temperature = y["temp"]
12                current_humidiy = y["humidity"]
13                z = x["weather"]
14                weather_description = z[0]["description"]
15                speak(" Temperature in kelvin unit is " +
16                    str(current_temperature) +
17                    "\n humidity in percentage is " +
18                    str(current_humidiy) +
19                    "\n description  " +
20                    str(weather_description))
21                print(" Temperature in kelvin unit = " +
22                    str(current_temperature) +
23                    "\n humidity (in percentage) = " +
24                    str(current_humidiy) +
25                    "\n description = " +
26                    str(weather_description))

```

Skill 10- To log off your PC:

The **subprocess.call()** function here is used to process the system function to log off or to turn off our PC. This invokes our AI assistant to automatically turn off our PC.

```

1         elif "log off" in statement or "sign out" in statement:
2             speak("Ok , your pc will log off in 10 sec make sure you exit from all applications")
3             subprocess.call(["shutdown", "/l"])
4
5         time.sleep(3)

```

Skill 11- To Dictate and Jot down notes:

The AI Assistant Steve can be a great study aid for students as it can jot down and record notes by listening to the student. In addition to noting down notes it can also dictate the stored record on the command of the user. We are using file reading and writing for this functionality in Steve.

```
1 elif "write a note" in statement or "write" in statement:
2     speak("What should i write, sir")
3     note = takeCommand()
4     file = open('C:\\Users\\prakh\\Desktop\\notes.txt', 'w')
5     speak("Sir, Should i include date and time")
6     snfm = takeCommand()
7     if 'yes' in snfm or 'sure' in snfm:
8         strTime = datetime.datetime.now().strftime("% H:% M:% S")
9         file.write(strTime)
10        file.write(" :- ")
11        file.write(note)
12    else:
13        file.write(note)
14
15 elif "show note" in statement:
16     speak("Showing Notes")
17     file = open("C:\\Users\\prakh\\Desktop\\notes.txt", "r")
18     print(file.read())
19     speak(file.read(6))
```

RESEARCH GAP

Since we have developed an offline version of AI assistant, therefore we faced few problems related to voice recognition and environmental noise cancellation. The accuracy of inbuilt microphone and speech recognition was quite low therefore initially we were facing some errors. We resolved these issues using try and except block and optimizing the speech recognition module.

CONCLUSION

Firstly, it was great deal to decide the project which best suits to the title "Study Aid". After long discussion we finally decided to develop Steve which is an AI- Assistant for students, capable of interacting and responding to computational, mechanical, common life questions using speech recognition.

Initially building this project we faced a lot of difficulties in the installation of various libraries and after searching a lot we somehow managed to install all the libraries. Then we made a rough plan to what all functionalities will a student require so we came up with some common requirement like open YouTube, Gmail, Google chrome and stack overflow. Predict current time, take a photo, search Wikipedia to abstract required data, predict weather in different cities, get top headline news from Times of India and can even answer computational and geographical questions too. Along with the above functionalities we have also added a functionality to aid students to jot down notes for them using speech to text conversion whereas it can also dictate answers or text, helping students to write down their assignments.

Some of the libraries we have incorporated in our project are-Speech Recognition, pyttsx3 , Wikipedia, Web browser , time.

We tried to make our project lively so that it could help students to LEARN and GROW with STEVE.

BIBLIOGRAPHY

Some of the resources that helped us in completing STEVE and achieve our milestone within stipulated time period are as follows-

- 1) **Stack Overflow**- <https://stackoverflow.com/>
- 2) **W3Schools**- <https://www.w3schools.com/>
- 3) **pip install**- <https://pypi.org/project/pip/>
- 4) **Udemy**- <https://www.udemy.com/topic/python-gui/>
- 5) **YouTube**- <https://www.youtube.com/watch?v=YXPyB4XeYLA>

CONTRIBUTION PAGE

Demo Video - <https://drive.google.com/file/d/1-znHF8Lq9UDERYH30bXYNaN11whCXOhm/view?usp=sharing>

1. Prakhar Saxena

- Idea Discussion
- Coding
- Report Development
- Library installation
- Code debugging

2. Pranjali Jain

- Idea Discussion
- Coding
- Report Development
- Demo Video
- Code Commenting

3. Bharat Mundra

- Idea Discussion
- Development
- Follow-ups