# Uni Virtual Assistant

Mariam Yekini (Project Leader) & Morgan Aldore

# Goal

- To create a student virtual assistant that is capable of providing students with information that can aid in their success in university/college
  - Take notes, search wikipedia, tell time, etc.
- Reduce the anxiety around forgetting to complete tasks
- Take away small burdens so that the student can focus on their success in college

# Contribution - Mariam (commands.py)

recognizeAudio()

takeCommand()

```python
# anything written in the function will be converted into speech
def recognizeAudio(audio):

    #specific voice id for chosen voice
    en_voice_id = "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_EN-GB_HAZEL_11.0"

    # Use female English voice
    engine.setProperty('voice', en_voice_id)
    engine.say(audio)
    engine.runAndWait() #Without this command, speech will not be audible to us.
    engine.stop()

#activates microphone, and then converts the audio to text
def takeCommand():
    rec = sr.Recognizer()

    #open microphone and record
    with sr.Microphone() as source:
        rec.adjust_for_ambient_noise(source,duration=1) #adjust audio to account for ambient noise
        print('How can I help you Mariam?')
        audio = rec.listen(source, timeout=10) #timeout if no speech is detected after 10 seconds

    #data = ''
    try:
        data = rec.recognize_google(audio, language='en-in')
        print('You said: ' + data)

    #conditionals to cover errors that may occur if there's too much background noise/didn't capture voice
    except Exception as e:
        recognizeAudio('Could you say that again please...')
        return 'None'

    return data
```

# Contribution - Mariam (commands.py)

getCurrentDate()

greetingType()

```
#function gives the current date
def getCurrentDate():

    text = text.lower()
    now = datetime.datetime.now()
    todaysDate = datetime.datetime.today()
    weekday = calendar.day_name[todaysDate.weekday()]
    monthNumber = now.month
    dayNumber = now.day

    #a list of months
    calendarMonths = ['january', 'february', 'march', 'april', 'may',
    'june', 'july', 'august', 'september', 'october', 'november', 'december']

    #list of days
    calendarDays = ['monday', 'tuesday', 'wednesday', 'thursday', 'friday', 'saturday', 'sunday']

    #day extensions
    dayExtensions = ['rd', 'th', 'st']

    #a list of ordinal numbers
    ordinalNumbers = ['1st', '2nd', '3rd', '4th', '5th', '6th', '7th', '8th', '9th','10th', '11th', '12th',
    '13th', '14th', '15th', '16th', '17th', '18th', '19th', '20th', '21st', '22nd', '23rd', '24th', '25th'
    '26th', '27th', '28th','29th', '30th', '31st']

    return 'Today is '+weekday+' '+ calendarMonths[monthNumber - 1] + ' the '+ ordinalNumbers[dayNumber - 1]+ '. '


#Function to make Uni greet according to the time
def greetingType():
    hour = int(datetime.datetime.now().hour)
    if hour>=0 and hour<12:
        recognizeAudio('Good Morning!')
    elif hour>=12 and hour<18:
        recognizeAudio('Good Afternoon!')
    else:
        recognizeAudio('Good Evening!')

    recognizeAudio("How can I help you Mariam?")

```

# Contribution - Mariam (main.py)

main()

```
14  if __name__ =="__main__":
15      #this is the first method that is executed before taking any commands
16      greetingType()
17
18
19      while True:
20          #converting user data into lowercase
21          data = takeCommand().lower()
22
23          #if who is is found in the data, Uni will search wikipedia
24          if 'who is' in data:
25              recognizeAudio('Searching wikipedia')
26              data = data.replace('wikipedia', '')
27              results = wikipedia.summary(data, sentences=2)
28              recognizeAudio('According to Wikipedia')
29              print(results)
30              recognizeAudio(results)
31
32          #if date is found in data, Uni will tell us the date
33          elif 'date' in data:
34              getCurrentDate = datetime.datetime.today().strftime('%b, %d %Y')
35              print(getCurrentDate)
36              recognizeAudio(getCurrentDate)
37
38          #if time is found in data, Uni will tell us the time
39          elif 'time' in data:
40              getTime = datetime.datetime.now().strftime('%I:%M' '%p')
41              print(getTime)
42              recognizeAudio(getTime)
```

# Contribution - Mariam (main.py)

main()

```
        #if write is found in data, Uni will take notes
        elif "write" in data:
            recognizeAudio("What should i write Mariam?")
            note = takeCommand()
            file = open('uni.txt', 'w')
            recognizeAudio("Should i include the date and time")
            snfm = takeCommand()
            if 'yes' in snfm or 'sure' in snfm:
                strTime = datetime.datetime.now().strftime("%I:%M")
                file.write(strTime)
                file.write(" :- ")
                file.write(note)
            else:
                file.write(note)

        #if show note is found in data, Uni will open up the text file
        elif "show note" in data:
            recognizeAudio("Showing Notes")
            file = open("uni.txt", "r")
            print(file.read())
            recognizeAudio(file.read(6))

        #if joke is found in data, Uni will tell jokes
        elif 'joke' in data:
            recognizeAudio(pyjokes.get_joke())

        #if where is is found in data, Uni will do a google search
        elif "where is" in data:
            data = data.replace('where is', '')
            location = data
            recognizeAudio('Let me look that up for you')
            recognizeAudio(location)
            webbrowser.open("https://www.google.nl/maps/place/" + location + '')

        #if youtube is found in data, Uni will open YouTube
        elif 'youtube' in data:
            recognizeAudio('Opening Youtube\n')
            webbrowser.open('youtube.com')

        #if blackboard is found in data, Uni will open the UMBC Blackboard website
        elif 'blackboard' in data:
            recognizeAudio('Opening Blackboard\n')
            webbrowser.open('https://blackboard.umbc.edu/')

        #if who are you is found, Uni will introduce herself
        elif "who are you" in data:
            recognizeAudio("My name is Uni and I am your virtual assistant")
```

# Contribution - Mariam - (main.py)

main()

```
91          #if how are you is found, Uni will tell me how she is and also ask me
92          elif 'how are you' in data:
93              recognizeAudio('I am fine, Thank you')
94              recognizeAudio('How are you Mariam?')
95
96          #if fine or good is found, she will say im glad
97          elif 'fine' in data or "good" in data:
98              recognizeAudio('I\'m glad to hear that')
99
100         #if birthday is found, she will tell me my birthday
101         elif 'my birthday' in data:
102             recognizeAudio('Your birthday is July 11th')
103
104         #if thank you is found, Uni will go on standy
105         elif 'thank you' or 'thanks' in data:
106             recognizeAudio("I am glad that I could be of assistance!")
107             exit()
```

# Contribution - Morgan (Webbrowser.py)

```python
import speech_recognition as sr
import webbrowser as web



def main():

    path = "C:/Program Files/Google/Chrome/Application/chrome.exe %s"


    r = sr.Recognizer()

    with sr.Microphone() as source:
        r.adjust_for_ambient_noise(source)

        print("Please say something ")

        audio = r.listen(source)

        print("Reconizing Now ... ")

```

# Contribution - Morgan (Webbrowser.py)

```
21
22
23
24          try:
25              dest = r.recognize_google(audio)
26              print("You have said : " + dest)
27
28              web.get(path).open(dest)
29
30          except Exception as e:
31              print("Error : " + str(e))
32
33
34  if __name__ == "__main__":
35      main()
```

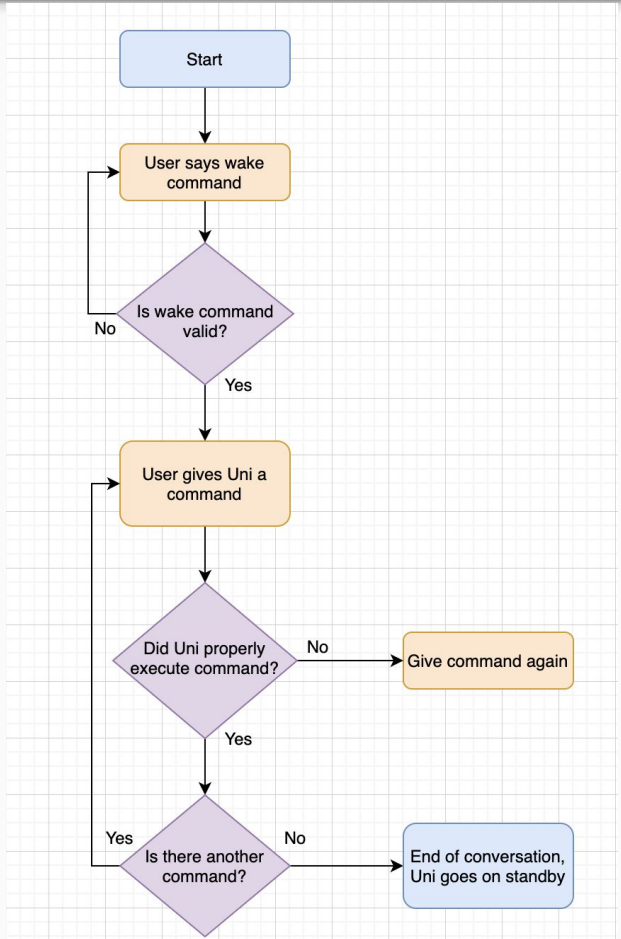
# Contribution - Morgan (Weather.py)

```
# import required modules
import requests, json

# Enter your API key here
api_key = "0cbfef3eb526b59af8909029bb7b59a3"

# base_url variable to store url
base_url = "http://api.openweathermap.org/data/2.5/weather?"

# Give city name
city_name = input("Enter city name : ")

# complete_url variable to store
# complete url address
complete_url = base_url + "appid=" + api_key + "&q=" + city_name

# get method of requests module
# return response object
response = requests.get(complete_url)

# json method of response object
# convert json format data into
# python format data
x = response.json()

# Now x contains list of nested dictionaries
# Check the value of "cod" key is equal to
# "404", means city is found otherwise,
# city is not found
if x["cod"] != "404":

    # store the value of "main"
    # key in variable y
    y = x["main"]
```

# Contribution - Morgan (Weather.py)

```
47      # store the value corresponding
48      # to the "temp" key of y
49      current_temperature = y["temp"]
50
51      # store the value corresponding
52      # to the "pressure" key of y
53      current_pressure = y["pressure"]
54
55      # store the value corresponding
56      # to the "humidity" key of y
57      current_humidiy = y["humidity"]
58
59      # store the value of "weather"
60      # key in variable z
61      z = x["weather"]
62
63      # store the value corresponding
64      # to the "description" key at
65      # the 0th index of z
66      weather_description = z[0]["description"]
67
68      # print following values
69      print(" Temperature (in kelvin unit) = " +
70                      str(current_temperature) +
71            "\n atmospheric pressure (in hPa unit) = " +
72                      str(current_pressure) +
73            "\n humidity (in percentage) = " +
74                      str(current_humidiy) +
75            "\n description = " +
76                      str(weather_description))
77
78  else:
79      print(" City Not Found ")
```

# Flowchart


Start
User says wake command
Is wake command valid?
No
Yes
User gives Uni a command
Did Uni properly execute command?
No
Give command again
Yes
Is there another command?
Yes
No
End of conversation, Uni goes on standby

# Concepts

- Functions
  - recognizeAudio()
  - takeCommand()
  - getCurrentDate()
  - greetingType()
  - main()
- If/Elif/Else Statements
- Exception Handling
- For Loops
- While Loops
- Operators

# Datasets

- **Libraries & Packages**
  - Speech Recognition
  - Pyttsx3
    - Microsoft Speech API (SAPI5)
  - Wikipedia
  - Pyjokes

| Stage | Tasks |
| --- | --- |
| 1 | • Ensured PC was working properly<br>• Configuration of IDE<br>• Packages |
| 2 | • Created basic functions needed in the main<br>• Outputted basic functions with string before converting to speech |
| 3 | • Ensured basic functions were implemented correctly<br>• Implemented speech functionality<br>• Ensured main function works properly executes basic functions<br>• Trade Offs |
| 4 | • User Testing<br>• Feedback |
| 5 | • Improvements<br>• Maintenance |

## Testing

| PHASE | | DETAILS | Execution Status | Owner | Last Executed | |
|---|---|---|---|---|---|---|
| 1 | Project Definition and Planning | - Ensure PC is working | Passed | All | 12/12020 | PROJECT END |
| | | - IDE Selection | Passed | All | 9/21/2020 | |
| | | - Github Creation | Passed | Mariam | 9/13/2020 | |
| | | Import Packages | Caution | Mariam + Morgan | 12/1/2020 | |
| | | - Determine feasibility of connecting Uni to student account | Not executed | John | | |
| 2 | Project Initial Development | - Output voice input in string format | Passed | Mariam | 10/10/2020 | |
| | | Output current time in string format | Passed | Mariam | 10/10/2020 | |
| | | - Output current date in string format | Passed | Mariam | 10/10/2020 | |
| | | - Output greeting according to time in string format | Passed | Mariam | 10/10/2020 | |
| | | - Output webbrowser input in string format | Passed | Morgan | 11/30/2020 | |
| | | - Output weather in string format | Passed | Morgan | 12/1/2020 | |
| | | - Output email content in string format | Not executed | John | | |
| 3 | Project Launch & Execution | - Output current time with speech | Passed | Mariam | 12/1/2020 | |
| | | - Output current date with speech | Passed | Mariam | 12/1/2020 | |
| | | - Output greeting according to time with speech | Passed | Mariam | 12/1/2020 | |
| | | - Ensure Uni responds to wake command | Failed | Mariam | 12/1/2020 | |
| | | - Output webbrowser with speech | Failed | Morgan | 12/1/2020 | |
| | | - Output weather with speech | Failed | Morgan | 12/1/2020 | |
| | | - Execute all basic functions and commands through main function | Passed | Mariam | 12/1/2020 | |
| 4 | Project Performance & Control | - Final testing before user testing and feedback | Passed | Mariam | 11/29/2020 | |
| | | - Final version control | Passed | Mariam | 12/1/2020 | |
| 5 | Project Close | - Improvements and testing | Passed | Mariam | 12/1/2020 | |
| | | - Maintenance (continuous testing and debugging) | Passed | Mariam | 12/1/2020 | |

# Testing

# Discussion

**Successful**

- Ran properly on both Windows and OS systems
- Includes many command capabilities
- Incorporated user feedback into final program

**Future Work**

- Uni sometimes doesn't always capture the command correctly
- Wake command
- Connecting Uni to student account
- Create a more robust program
- GUI
- Weather unit in Fahrenheit