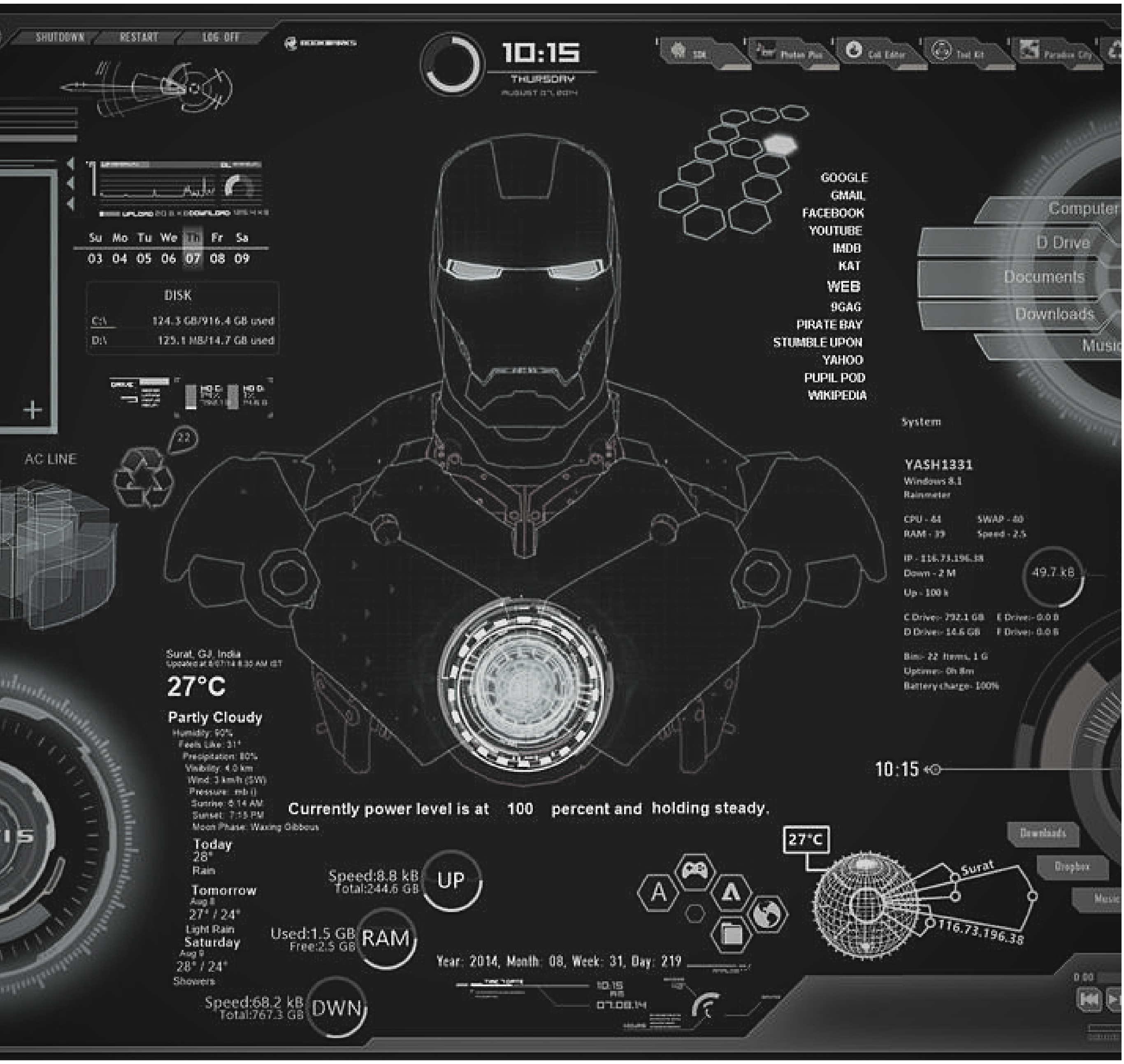
J.A.R.V.I.S VOICE COMMAND ON COMPUTER



TEAM DETECTIVE CLUELESS - PROGRAMMING REPORT

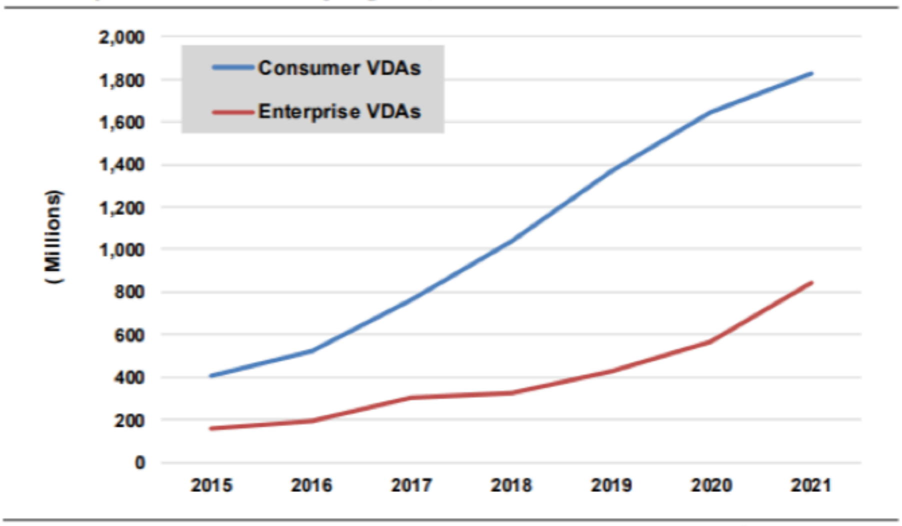
Julie CAI-NGO 516120990086, Mewaine PETARD 719071990002 , Ambre-Mei RACINE 516120990088, Yoshihiro TERAGUCHI 716120290030

1) Background

Nowadays, digitalisation and automatisation have contributed significantly to improve people's daily life. However, as of today, there are still possibilities to optimise the usage of technology and even to adapt it to people with particular conditions. The creation of an intelligent personal assistant could help to plan and execute basic tasks, such as creating to-do-list, arrange meetings, report news, sharing information, etc. In order to achieve it, it has access to a large amount of information on the device or online and it can take different form such as a chatbot or a scheduler, etc. In addition, a voice command for computer could theoretically replace the usage of the keyboard and the mouse. This could help professionals, students and even handicapped people. It can completely revolutionise and optimise the way people work or study and contribute to time saving and increase productivity.

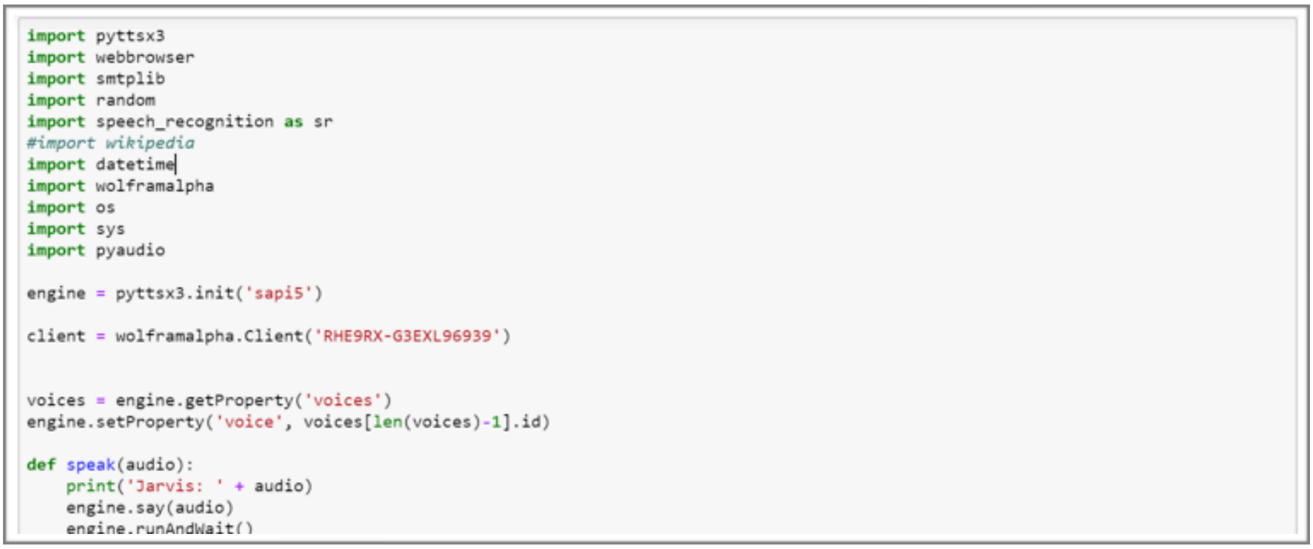
As for the background of this project, a voice assistant uses voice recognition, speech synthesis and natural language processing (NLP) as a digital assistant to help provide a service to the users through a particular application. In the first place, voice assistant is used in our everyday life through many devices such as our smartphones, tablets or even in our cars. Many applications and operations systems are now using this technique to boost and innovate their company. It is used in many different environments such as healthcare, retail or education. Technology is constantly evolving and changing and our society today is sensitive to any added value a company has to offer. For example, Siri is one of the most famous voice assistant on the market today and was launched by Apple when the iPhone 4S came out in October 2011. Similarly, many other companies have launched their own version of voice assistant such as Google with Google Now or Amazon with Alexa. According to Tractica reports, the virtual digital assistant market will grow for more than 390$ million worldwide in 2015 to 1.8 billion by the end of 2021. The number of users is increasing daily and is expected to attain 843 million of users by 2021. The overall speaker market is still in the midst of a major growth cycle and its development is only at the beginning phase.

Total Unique Active VDA Users by Segment, World Markets: 2015-2021

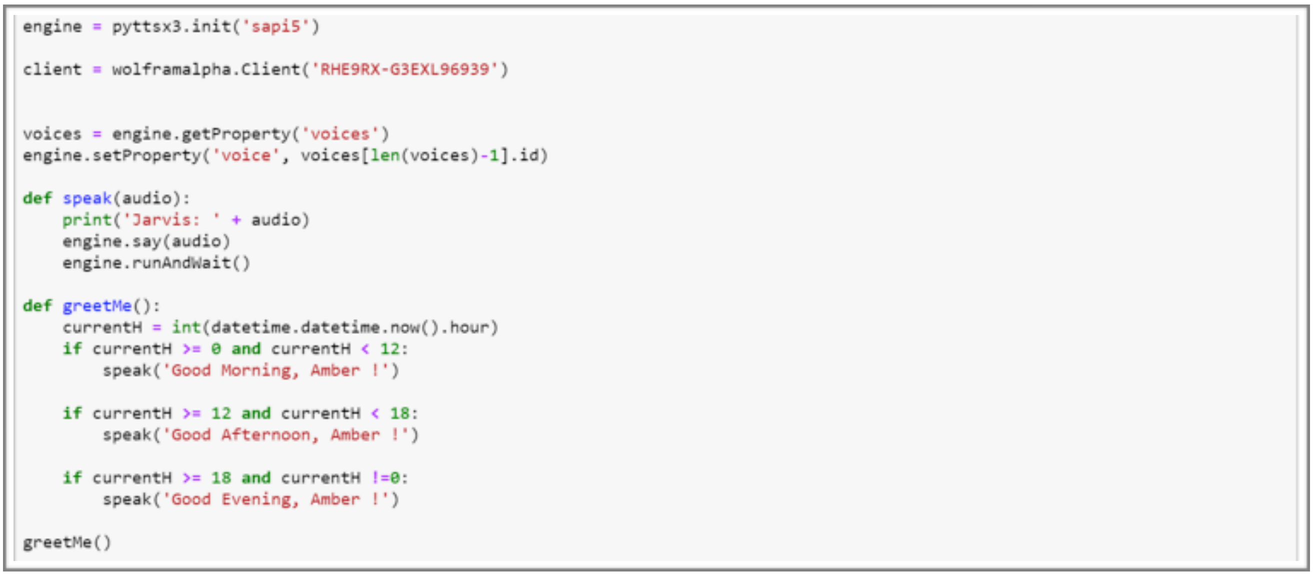


(Source: Tractica)

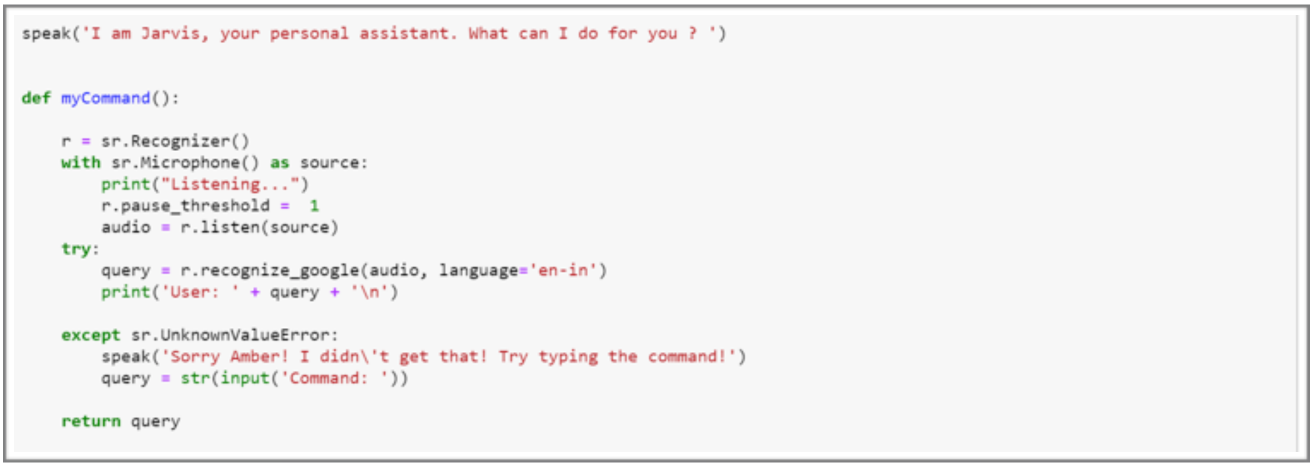
2) Program processing



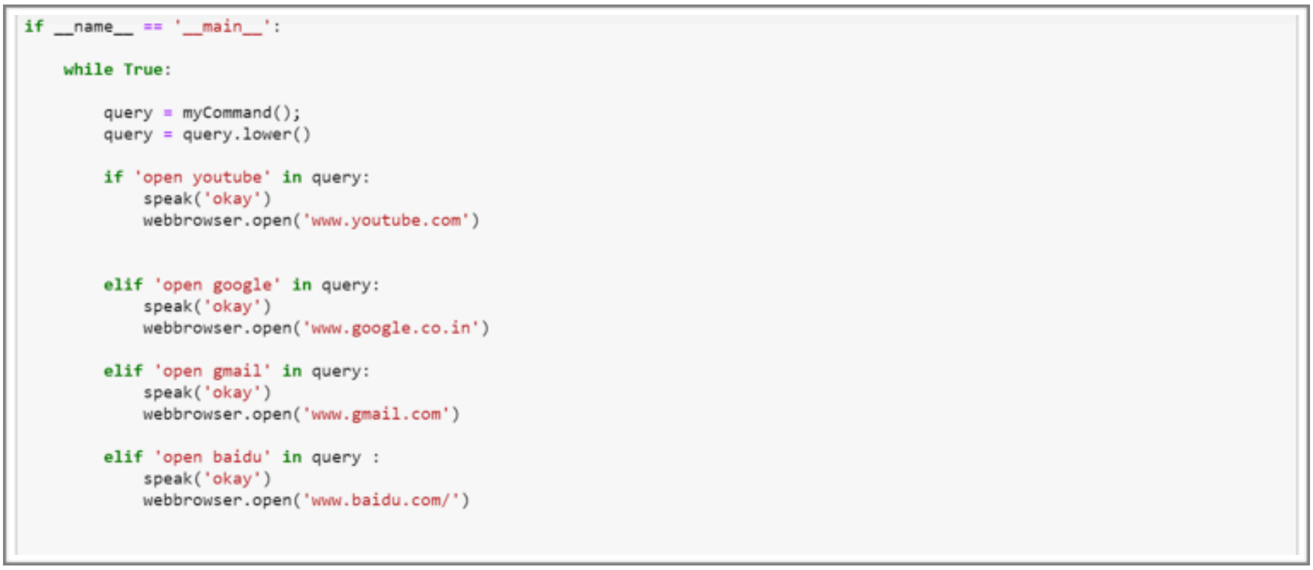
```
import pyttsx3
import webbrowser
import smtplib
import random
import speech_recognition as sr
#import wikipedia
import datetime
import wolframalpha
import os
import sys
import pyaudio

engine = pyttsx3.init('sapi5')

client = wolframalpha.Client('RHE9RX-G3EXL96939')


voices = engine.getProperty('voices')
engine.setProperty('voice', voices[len(voices)-1].id)

def speak(audio):
    print('Jarvis: ' + audio)
    engine.say(audio)
    engine.runAndWait()
```



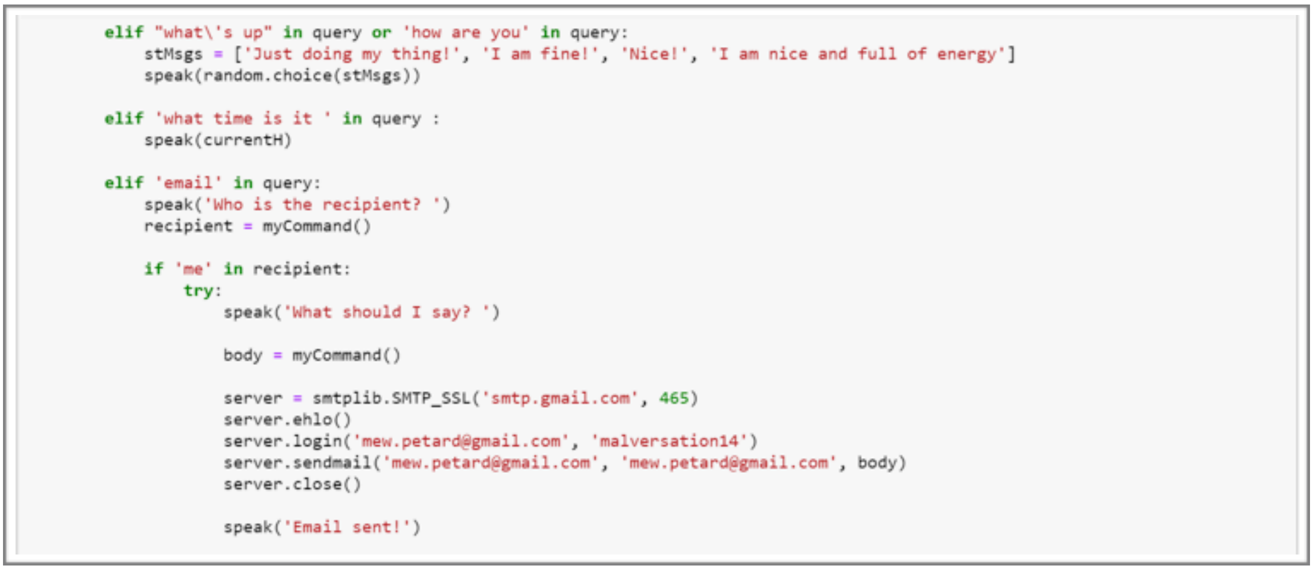
```
engine = pyttsx3.init('sapi5')

client = wolframalpha.Client('RHE9RX-G3EXL96939')


voices = engine.getProperty('voices')
engine.setProperty('voice', voices[len(voices)-1].id)

def speak(audio):
    print('Jarvis: ' + audio)
    engine.say(audio)
    engine.runAndWait()

def greetMe():
    currentH = int(datetime.datetime.now().hour)
    if currentH >= 0 and currentH < 12:
        speak('Good Morning, Amber !')

    if currentH >= 12 and currentH < 18:
        speak('Good Afternoon, Amber !')

    if currentH >= 18 and currentH !=0:
        speak('Good Evening, Amber !')

greetMe()
```



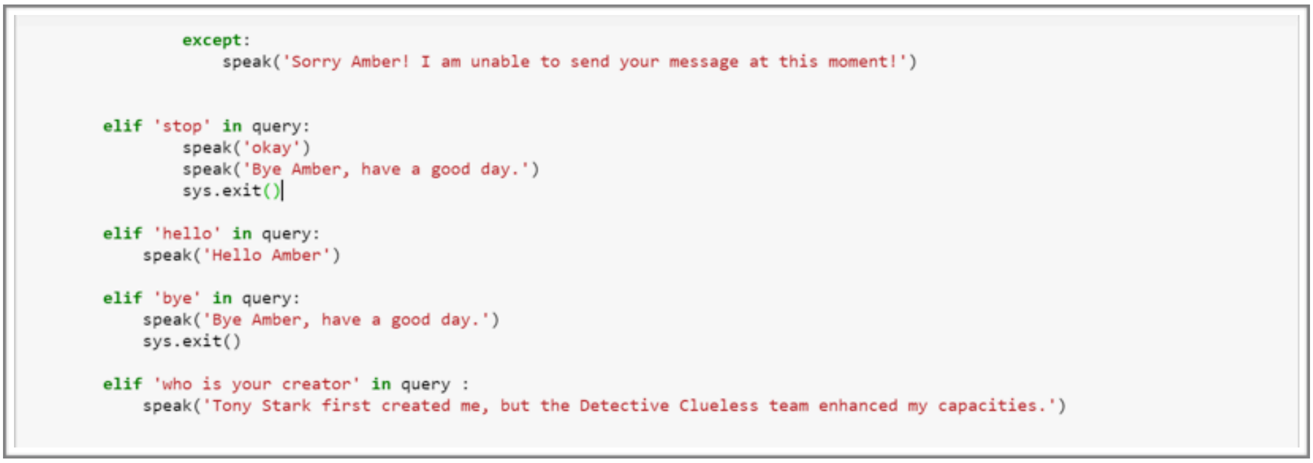
```
speak('I am Jarvis, your personal assistant. What can I do for you ? ')


def myCommand():

    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening...")
        r.pause_threshold =  1
        audio = r.listen(source)
    try:
        query = r.recognize_google(audio, language='en-in')
        print('User: ' + query + '\n')

    except sr.UnknownValueError:
        speak('Sorry Amber! I didn\'t get that! Try typing the command!')
        query = str(input('Command: '))

    return query
```



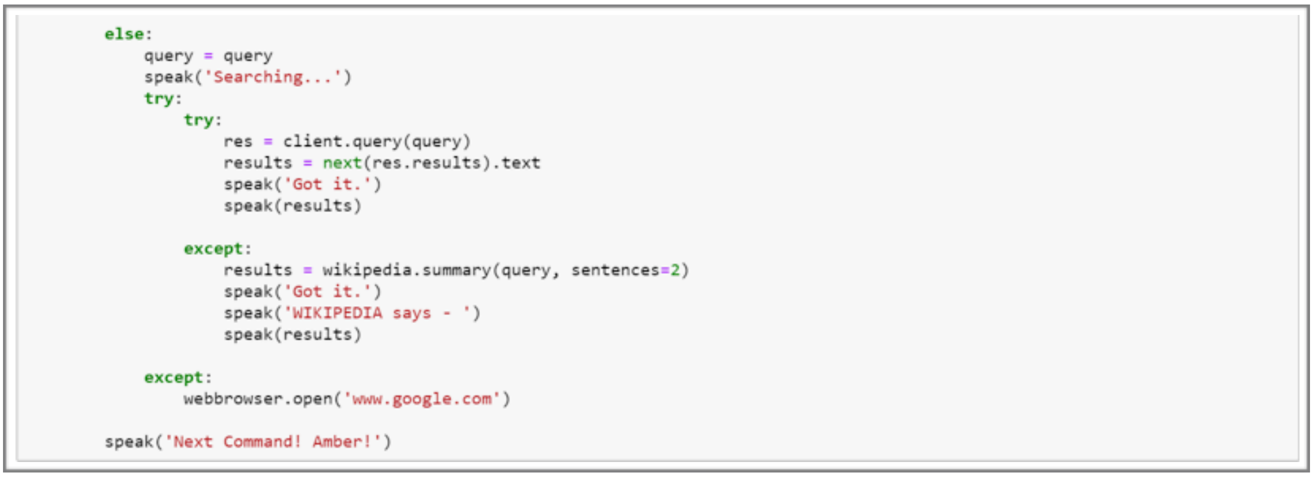
```
if __name__ == '__main__':

    while True:

        query = myCommand();
        query = query.lower()

        if 'open youtube' in query:
            speak('okay')
            webbrowser.open('www.youtube.com')


        elif 'open google' in query:
            speak('okay')
            webbrowser.open('www.google.co.in')

        elif 'open gmail' in query:
            speak('okay')
            webbrowser.open('www.gmail.com')

        elif 'open baidu' in query :
            speak('okay')
            webbrowser.open('www.baidu.com/')
```



```
        elif "what\'s up" in query or 'how are you' in query:
            stMsgs = ['Just doing my thing!', 'I am fine!', 'Nice!', 'I am nice and full of energy']
            speak(random.choice(stMsgs))

        elif 'what time is it ' in query :
            speak(currentH)

        elif 'email' in query:
            speak('Who is the recipient? ')
            recipient = myCommand()

            if 'me' in recipient:
                try:
                    speak('What should I say? ')

                    body = myCommand()

                    server = smtplib.SMTP_SSL('smtp.gmail.com', 465)
                    server.ehlo()
                    server.login('mew.petard@gmail.com', 'malversation14')
                    server.sendmail('mew.petard@gmail.com', 'mew.petard@gmail.com', body)
                    server.close()

                    speak('Email sent!')
```



```
                except:
                    speak('Sorry Amber! I am unable to send your message at this moment!')


        elif 'stop' in query:
                speak('okay')
                speak('Bye Amber, have a good day.')
                sys.exit()

        elif 'hello' in query:
            speak('Hello Amber')

        elif 'bye' in query:
            speak('Bye Amber, have a good day.')
            sys.exit()

        elif 'who is your creator' in query :
            speak('Tony Stark first created me, but the Detective Clueless team enhanced my capacities.')
```



```
        else:
            query = query
            speak('Searching...')
            try:
                try:
                    res = client.query(query)
                    results = next(res.results).text
                    speak('Got it.')
                    speak(results)

                except:
                    results = wikipedia.summary(query, sentences=2)
                    speak('Got it.')
                    speak('WIKIPEDIA says - ')
                    speak(results)

            except:
                webbrowser.open('www.google.com')

        speak('Next Command! Amber!')
```

Decrypting the code :

Several packages must be installed prior to the writing of the code, some of them are general and known such as random, but most are new :
The building of Jarvis relies on several functions , we :

Name of the package	**Functionnality**
Pyttsx3	Text to speech module
Pyaudio	Sets up the portaudio system
smtplib	Installs the Simple Mail Transfer Protocol
webbrowser	Allows web pages to be charged
Speech_recognition	Google Based speech recognition module
wolframalpha	Mathematical Calculations and Trivia

greetMe( )

Uses the datetime package to get the time of the day and determines the proper greeting used by Jarvis depending on the time of the day.

myCommand( ) :

The main function, this function is designed to record the user speaking and use the speech recognition created by google to get an input named ‘query’. Each functionality that follows is a sub–function into the mainCommand( )

Open a webpage :

By entering certain links to webpage and creating if statements depending on keywords, we could have designed specific functions permitting to vocally open Google, Baidu, Youtube home pages.

Send an email :

The command relies mainly on the Simple Mail Transfer Protocol (smtp)

For this code, the email address of the expedient is a gmail address. Due to security matters, we had to allow gmail connections from a less secure app.

Then we log onto the server by entering the email address and password as string.

To ensure secured connection, we write the command server.ehlo()

The email we send in this program is fairly simple, no object, it only send a text the user dictates.

Wolfram Alpha:

For students, Wolfram Alpha is a website which offers calculations, mathematical and physical computations but also some more general factual question such as the weather, and trivia.

To implement Wolfram, we must have a subscription, either pay or free version, and to get the api code to copy paste in the code. As WA is a website, an internet access is required to get the function to work.

For the times when the results cannot be found on WA, the program will use the except rule to do the research on Wikipedia instead. Here no subscription is needed and the Wikipedia module can easily be installed beforehand. It is important to note that due to the recent censorship of Wikipedia in China, a VPN is needed to access this function.

3) Problems encountered

In order for the program to run successfully, we had to pay attention to several factors. First of all, we chose to use Google speech recognition module, however, as we are in China, we are facing the censorship and need to overcome the great firewall. As a result, in order to run the program, our work depends on whether we have a VPN connection or not. In addition, using a VPN significantly slows down the processing of the code, which could lead to interruptions. Moreover, our program is designed to send email via SMTP, which correspond to a Gmail server. Hence, a VPN is also needed. However, mail boxes nowadays have high security checks and a connection from a different country blocks the access to the mail box.

Regarding the coding part, we needed to install and try many packages in order to find the ones functioning correctly with our program. Py.Audio is normally built in some Python versions, but not on Python 3.7, so it was challenging to integrate it. We had to install it manually. In order to do so, we had to search for the method and the right version on Stark Overflow. Finally, if a function is created, keywords have to be written with strings without capital letters, otherwise the program will not work.

4) Perks of our project

During the implementation of our computer voice command also called J.A.R.V.I.S, we encountered a few issues however we quickly realised that there are major benefits that come from our project idea.

The first one would be the fact that the information we can acquire concrete and official data from J.A.R.V.I.S. Information such as date or time are based on the real-time effect. Simple daily tasks can be done quicker since it uses shortcuts and it is very simple to use it. You simply need a WIFI or cellular network but once you have a stable connection, you can just launch the program and J.A.R.V.I.S will simply ask you what command do you need and you will not have to type or even touch your keyboard since your voice is the only tool that you need.

Secondly, J.A.R.V.I.S can be easily accessible by anyone no matter the age or background. Anybody can find a use for a computer voice command whether it is to simply open a new tab or to play a specific music you would like to listen to. As previously said, its use is so simple that you do not have to be a computer wizard to use it.

Thirdly, the program can be modified to suit a particular need or use. Depending on the outcome you would like to have, you can add or delete a few lines of code to suit your expectation. The code is so simple since there is no object, contacts are predefined and since the program addresses himself to a specific person, you can go back to the program and change the name of the responder. It can easily be customised and comes with a wide range of functions. The best way to explore would be for you to just try and run the program with the command you like it to answer to.

Finally, J.A.R.V.I.S's voice can easily be changeable in a different language since it is not mandatory for it to be in English. Depending on the number you insert in the program, you can choose the language you would like it to speak :

```
voices = engine.getProperty(voices)
engine.setProperty(voice; voices[len(voices)-1].id)
```

Therefore, it would have to be determined in the settings (Program) if you would like J.A.R.V.I.S to speak a different language. Different possibilities could be French, English but with a French accent or even Chinese. For the user, the most significant advantage is the use of Google's database. Voice recognition is based on Google's massive database therefore its accuracy is stronger.

5) Limits of our project

Our project is a very basic voice command program. It is limited to a number of commands: opening web-pages, doing factual research on the internet and sending emails. It can not solve complex problems yet and the program is limited to searches on Wikipedia and Wolfram, so answers generated depend on those websites' accuracy. Moreover, the program is only able to open websites (ex: Google and Youtube) with the web-page function, but is not able to search content on those pages.

The SMTP only works with a Gmail mail box and will need to be adapted in order to function with other mail boxes. Moreover, it can only send simple emails, without attachments (ex: images, files, etc.). Furthermore, even if the voice command recognises other languages, it only asks questions in the language it has been configured in and only transcript speech in the set-up language.

Moreover, to be fully functional in order to help people efficiently, the voice command have to be easy to launch and convenient to use. However, in our case, we need to have a very powerful and accurate microphone and avoid parasitic noises. It is actually only working with an external microphone. Hence, even if it can theoretically help people, it is not developed enough yet to practically sustain the user's need in real life.

6) Conclusion

After launching the program a few times, we often had the same issue due to the VPN connection. Connection was quite slow but we managed to run it successfully after multiple attempts. Apart from that, it was also difficult to add the feature that could allow us to send an email without a few failures first. However, as students who have never studied programming before this semester, this was a victory and a successful achievement that we are very proud of. Obviously, this is simply a primitive version of the project and idea that we got and it can evidently be improved accordingly to its usage from future users.