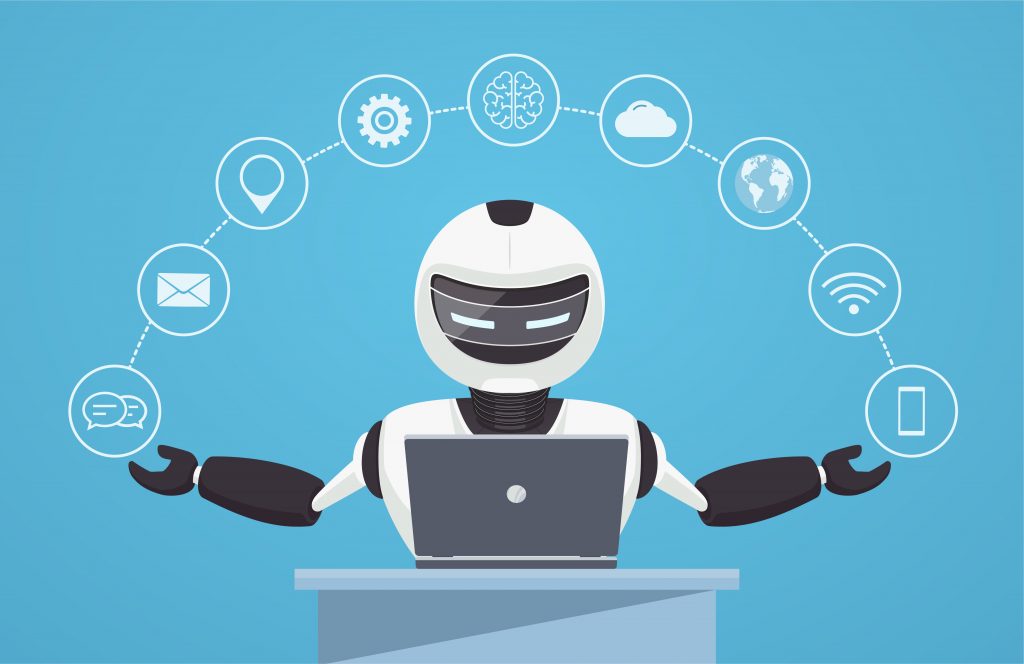


**LUCRARE PENTRU OBȚINEREA**

**ATESTATULUI PROFESIONAL LA INFORMATICĂ**

**BOT ASSISTANT**



***Profesor coordonator: Realizator:***

***Belbe Maria Zaharie Andrei***

***Mărieș Iuliana An școlar 2019-2020***

Cuprins

1. **Introducere** ……………………………………………………………pag. 3
2. **Studiu asupra problemei ce stă la baza programului**………………… pag. 4-5

2.1 Persoanele vizate pentru folosirea programului………… pag. 5

1. **Prezentare subprograme**……………………………………………….. pag. 6-19

3.1 Prezentarea librăriilor folosite ………………………… pag. 6-7

3.2 Prezantare interfață grafică si loading screen …..…….. pag. 7-10

3.3 Funcția speak()…………………………………………. pag. 11

3.4 Funcția cunoastere()……………………………………. pag. 11

3.5 Funcția inceput()……………………………………….. pag. 12

3.6 Funcția uitare()…………………………………………. pag. 13

3.7 Funcția login()………………………………………….. pag. 13

3.8 Funcțille de mesagerie pe Facebook/Messanger……….. pag. 14

3.8.1 Funcția defemail()……………………... pag. 14

3.8.2 Funcția facebook()…………………….. pag. 14-16

3.8.3 Fucția mesagerie().................................. pag. 17-18

3.9 Funcția note().................................................................... pag. 19

**4. Prezentare modul comanda vocală**……………………………………. pag. 20

4.1 Funcția comandaA(Pentru limba engleză)……………… pag 20 4.2 Funcția comandaR(Pentru limba română)………………. pag 20

1. **Prezentarea subprogram principal- assistant(comanda)**…………….. pag 21-23
2. **Codul sursă pentru intreg programul Bot Assitant** ………………….. pag 24-34
3. **Resurse folosite**…………………………………………………………...pag 34

**Introducere**

**Cap. 1**

**Bot Assistant** este un program conceput sa usureze si să automatize activitătile unei persoane când vine vorba de domeniul virtual. De asemenea programul urmărește in special creearea unei alternative care sa ducă la o viata cât mai normal pentru persoanele cu diverse dizabilități precum cele motorii ( lipsa mainilor sau paralizia) care ii impiedică pe oameni să beneficieze de internet si calculator in general.

**Bot Assistant** reprezintă un program dezvoltat pe baza algoritmului speech recognition prin intermediul căruia comenzile aspura unui calculator sunt transpuse in comenzi vocale prin care utilizatorul poate să controleze acțiunile calculatorului fară a avea nevoie de tastură sau mouse, ci doar de un microfon și voce.

Astfel asistentul a fost denumit si poate fi apelat sub numele de “**Snarky**”.



**Studiul asupra problemei pe care programul creeat ar trebui sa o diminueze / resolve.**

**Cap. 2**

***Modele de dizabilitate***  
Dizabilitatea este perceputa si abordata prin prisma a doua modele: medical si social.  
  
 ***Modelul medical*** defineste persoanele cu dizabilitati din perspectiva bolii sau a conditiei lor medicale. In acest model, dizabilitatea este percepută ca problemă a individului, acesta din urma fiind dependent de ceilalti si avand nevoie de tratament adecvat problemei sale. Modelul medical determina excluderea sistematica a persoanelor cu dizabilitati din societate.  
  
 ***Modelul social***este promovat de Uniunea Europeană si pune accentul pe mediul social neadaptat nevoilor persoanelor cu deficiente, de unde rezulta si dificultatile cu care acestea se confrunta. Astfel, modelul social nu mai percepe dizabilitatea ca o problema individuala, ci ca un fapt social, generat de politică, practică, atitudini si mediul inconjurător.

Având modelele de dizabilitate explicate mai sus putem definii programul Bot Assistant ca fiind o solutie pentru adaptarea mediului social la nevoile persoanelor cu deficiențe.

Câteva exemple concludente în acest sens : verificare unui email sau a mesajelor primate pe diverse retele de socializare precum facebook, accesul la informatiile de pe internet, sau deschiderea programelor din propriul sistem de operare( Windows/ Linux/ Mac).

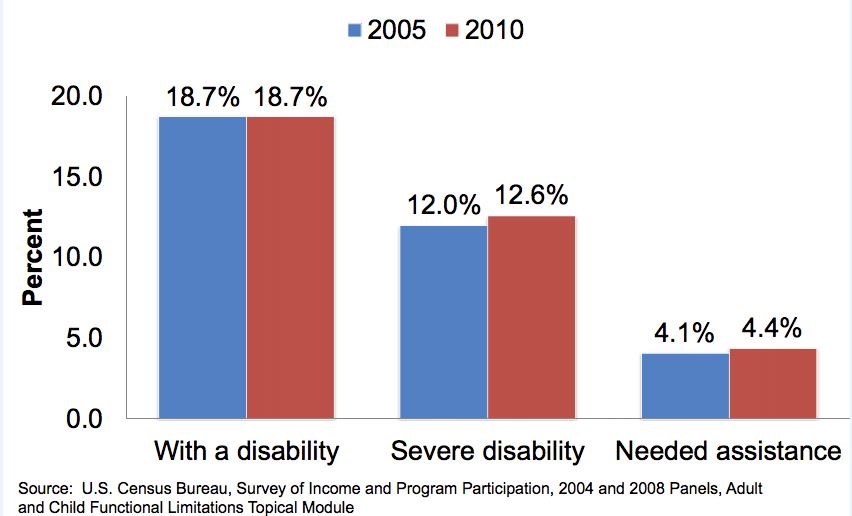
**Persoanele vizate pentru folosirea acestui tip de program.**

Deși programul este menit sa rezolve problemele persoanelor cu anumite dizabilități din mediul social, acesta poate fi folosit pentr a imbunătăți viața oricarei categorii sociale datorită proceselor automatizate pe care acesta le oferă.

De exemplu deschiderea prin comandă vocală a browserului sau căutarea automată a informațiilor legate de un anumit subiect pe google.

De asemnea programul a fost conceput, intr-un mod minimal, să poată interactiona direct cu utilizatorul prin anumite funcții predefinite de vorbire, el fiind capabil să răspunde la intrebări precum „how are you?” ,”how was your day?”, ”can you tell me a joke?”, etc.

După cum se poate observa si în imaginea de mai jos oamenii cu orice fel de dizabilitate reprezinta un total de 35% din populatie in 2010. Deci există o mare nevoie de astfel de tipuri de program.

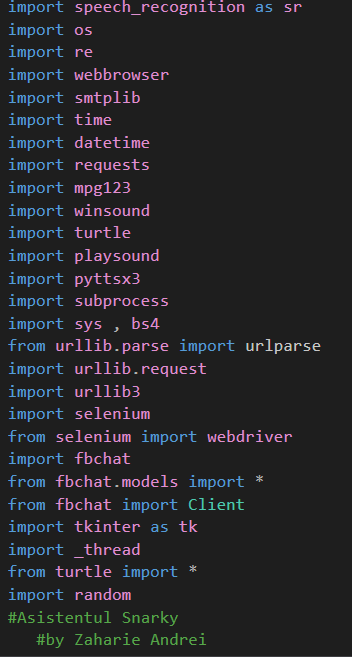


**Prezentare program**

**Cap. 3**

Programul este creeat preponderant in python si folosește atât librariile interne cât si unele librării externe precum speech\_recognition, webbrowser, fbchat.

**3.1 Prezentarea librariilor folosite**

1.Prima librărie si anume speech\_recognition contine functiile prin care se poate realiza comanda vocală si este pusă la dispoziție in open source de care firma cu acelasi nume.

2.Cea de a doua librarie respective cea de a 3 a sunt 2 librării standard ale limbajului python si sunt asemănătoare cu <windows.h> din C++.

3.Webbrowser reprezinta o librărie nativ provenita din C++ care contine functiile pentru operarea in browser.

4.Smtplib reprezintă o librărie care se ocupa cu procesarea sirurilor de caractere fiind asemanatoare cu <cstring> in C++.

5.Următoarele 2 librării gestionează timpul sistemului fiind necesare pentru functiile de sleep si alte procese.

6. Librăria requests reprezintă calea de acces a programului spre executia de linkuri(downloaduri)

7.Urmatoarele librarii mpg123, winsound sunt legate de sistemul de sunat al windowsului.

8.Libraria turtle este specifică pentru construirea unei interfate grafice (vezi pagina 8).

9. Librăria pyttsx3 stă la baza dezvoltării si funcționării acestui program alături de speechrecognition deoarece aceasta face pozibilă transformarea unui text in format vocal ceea ce rezultă in posibilitatea Bot Assitantului de a socializa cu persoanele.

10.Librăria subprocces face legatura dintre programul in sine si procesele pe care le transmite la sistemul de operare astfel reusind sa deschidă anumite programe.

11.Librăriile urlib se ocupa de procesarea url-rilor (linkurilor)

12. Librăria selenium se ocupă de accesarea codului html,css, php din browser facand posibila conecarea la nivel vizual pe o anumita retea de socializare precum facebook

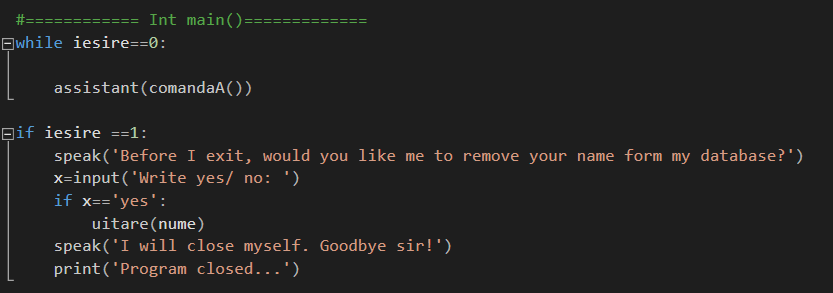
13.Librăria fbchat este creeată de facbook si contine functii predefinite ce pot fi accesate pentru a indeplini anumite cerinte precum trimiterea automată a unui mesaj, poze, video, etc.

14.Librăria random contine functii specifice care aleg aleatoriu elemente dintr- o listă sau interval de numere, cuvinte .

15.Libraria \_thread oferă posibilitatea de a rula programul pe mai multe fire de execuție numite threaduri care faciliteaza procesele in paralel de care vom avea nevoie pentru loading screen ce este prezentat in capitolul 3.1 la pag 8.

**3.2 Prezentare Interfață grafică**

Interfata grafică este prezantă doar la inceputul programului si anume in loading screen deoarece programul este crreat pentru a putea fi rulat cu usurintă in background si pentru a consuma cât mai putine resurse ale calculatorului.

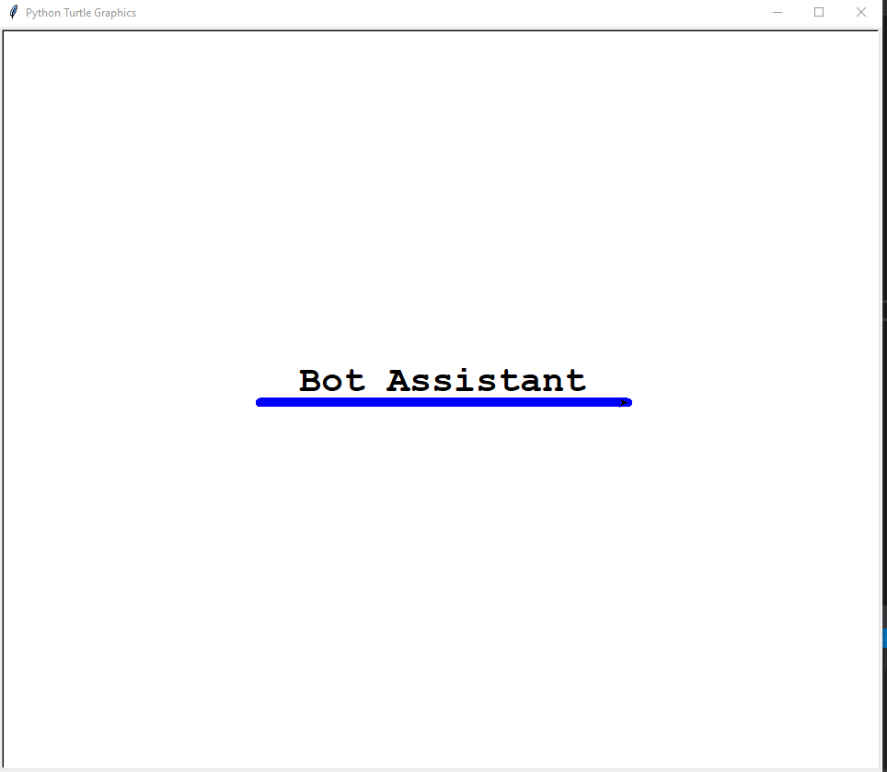
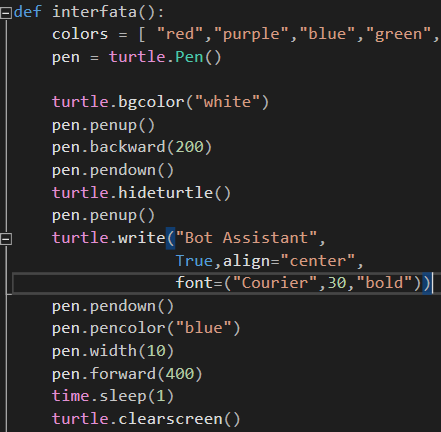
 Iar o interfață grafică prezentă pe tot parcursul rulării programului ar fi ingreunat procesele avand in vedere faptul că intreg programul ruleaza intr-o buclă infinită cu posibilitatea de sistare a programului la comanda vocală „close” după cum puteti observa si in exemplul de mai jos.

Funcțiile prezente in aceasta imagine precum assistant(), comandaA(), speak() și uitare() vor fi explicate ulterior la capitolul “Prezentarea Subprogramelor”.

**3.2 Prezentare Loading Screen**

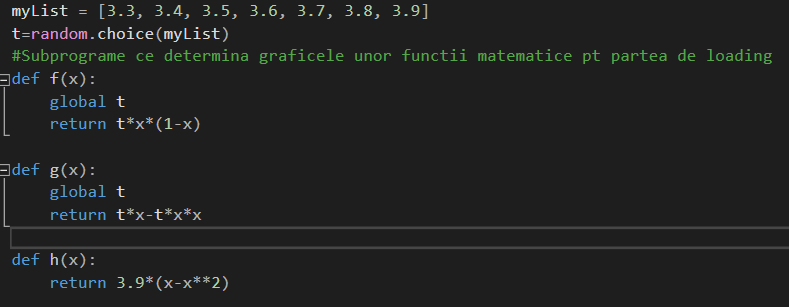
Loading screenul a fost adăugat pentru a oferi o experiență grafică cât timp librăriile si intreg codul sunt incărcate in background.

Astfel la rularea programului prima data va apărea o fereastră cu numele proiectului sub care se pozitioneaza un loading bar albastru la fel ca si in imaginea de mai jos.

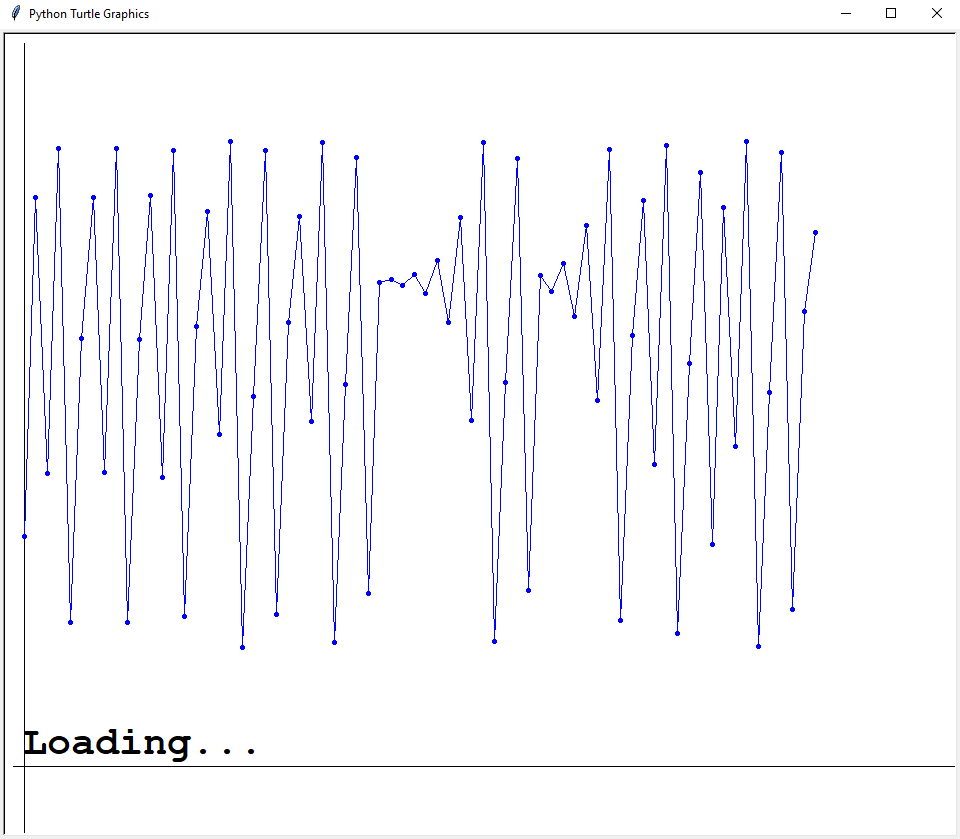


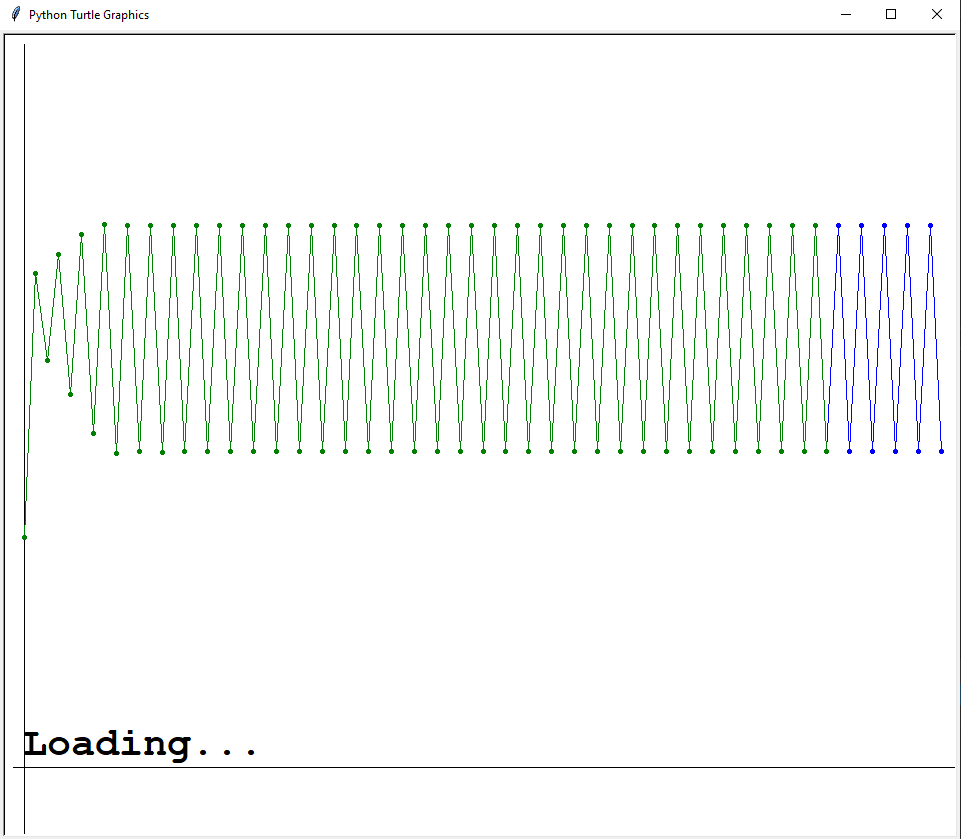
In acest fragment de cod putem observa funcțiile librăriei turtle specificată in capitolul 3.1 si funcțiile predefinite ale acesteia astfel. Prin atribuirea unei variabile numită pen clasa turtle.Pen() putem schita intreaga fereastră de loading.

Așadar functiile prezentate in imagine sunt destul de intuitive , *pen.peup*() reprezintă sistarea afisarii pe ecran a actiunii ce se desfăsoară(deci programul continua să funcționeze), *pen.pendown()* determină afișarea acțiunilor care sunt facute pe ecran. *Turtle.bgcolor()* determină setarea unei color de background, *turtle.write()* determina procesarea si afișarea unei accțiuni dacă functia pen.pendown() are valoare de adevăr, *pen.pencolor()* seteaza culoarea cu care urmează să trasăm liniile, iar *turtle.clearscreen()* determină stergerea afisărilor prezente pe ecran.

 După incărcarea completă a casetei albastre fereastra va fi inlocuită cu o altă fereastră dinamică de incărcare a programului. Pentru ca procesul de incărcare să nu devină repetitiv și implicit plictisitor fereastra de loading afisează graficele unor funcții alese complet aleatoriu cu functia random() la inceputul rulării programului. In acest sens am implementat 3 tipuri de functii diferite si o listă de de 7 elemente (numere reale) cu care urmeaza sa fie inmulțiți sau ridicați la putere termenii funcțiilor respective dupa cum se poate observa în exemplul de mai jos.

Astfel având 3 functii si 7 elemente in listâ probabilitatea ca aceeasi funcție să fie afișată de două ori consecutiv este de 1/21 = 0.047.

Prin apelarea repetată a functiilor pentru mai multe valori vom obtine grafice precum cele din exemplele de mai jos.



Codul pentru interfața grafică și implicit a loading screenului:

def f(x):

global t

return t\*x\*(1-x)

def g(x):

global t

return t\*x-t\*x\*x

def h(x):

return 3.9\*(x-x\*\*2)

def jumpto(x, y):

penup(); goto(x,y)

def line(x1, y1, x2, y2):

jumpto(x1, y1)

pendown()

goto(x2, y2)

def coosys():

line(-1, 0, N+1, 0)

line(0, -0.1, 0, 1.1)

def plot(fun, start, color):

pencolor(color)

x = start

jumpto(0, x)

pendown()

dot(5)

for i in range(N):

x=fun(x)

goto(i+1,x)

dot(5)

def interfata():

{ colors = [ "red","purple","blue","green","orange","yellow"]

pen = turtle.Pen()

turtle.bgcolor("white")

pen.penup()

pen.backward(200)

pen.pendown()

turtle.hideturtle()

pen.penup()

turtle.write("Bot Assistant",

True,align="center",

font=("Courier",30,"bold"))

pen.pendown()

pen.pencolor("blue")

pen.width(10)

pen.forward(400)

time.sleep(1)

turtle.clearscreen()

pen.pencolor("blue")

pen.width(10)

turtle.hideturtle()

pen.penup()

turtle.write("By Zaha Tech Industries",True,align="center",font=("Cornerstone",20))

pen.pendown()

pen.goto(-200,0)

pen.goto(200,0)

time.sleep(2)

turtle.clearscreen()

turtle.hideturtle()

pen.penup()

setworldcoordinates(-1.0,-0.1, N+1, 1.1)

speed(0)

turtle.write("Loading...",True,align="left",font=("Courier",30,"bold"))

hideturtle()

coosys()

plot(f, 0.35, "blue")

plot(f, 0.35, "green")

#plot(h, 0.35, "red")

turtle.clearscreen()

turtle.write("Done",True,align="left",font=("Courier",30,"bold"))

turtle.bye()

* 1. **Functia speak()**

Subprogramul speak() care primește ca parametru o variabilă de tip string transformă cu ajutorul librăriei pyttsx textul in format vocal. Astfel orice text va fi enuntat de program cu o voce deja existentă in librăria pyttsx.

def speak(audio):

global vocea

engine = pyttsx3.init()

voices = engine.getProperty('voices')

engine.setProperty('voice', voices[vocea].id)

print(audio)

engine.say(audio)

engine.runAndWait()

In imaginea prezentată mai sus putem observa variabila vocea care poate lua valori naturale( 0 sau 1) reprezentând numărul de oridine al vocilor din vectorul de voci (voices). Acest vector cotine o voce feminine si una masculine care pot fi setate cu prin intemediul funcției engine.setProperty(). Cu această functie pot fi setate de asemenea si viteza de vorbire sau pauzele in vorbire pe care programul sa le facă.

* 1. **Funcția cunoastere()**

Subprogramul cunoastere care primeste ca parametru o variabilă de tip string verifică intr-un fișier dacă persoana este la prima utilizare a programului.Dacă fisierul este gol atunci programul il va rug ape utilizator sa iși introduce un nume pe care ulterior Bot assitantul il va apela,după care va deschide automat o listă (help) cu comenzile pe care utilizatorul le poate folosi.

def cunoastere():

f= open('recunoastere.txt','r+')

if os.path.getsize('recunoastere.txt')==0:

speak('Hey are you new? How would you like me to call you?')

x=input('Name: ')

f.write(x)

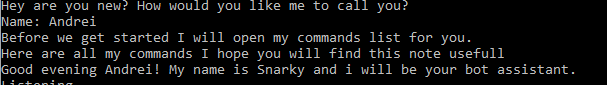
speak("Before we get started I will open my commands list for you.")

subprocess.Popen(["notepad.exe", "Help.txt"])

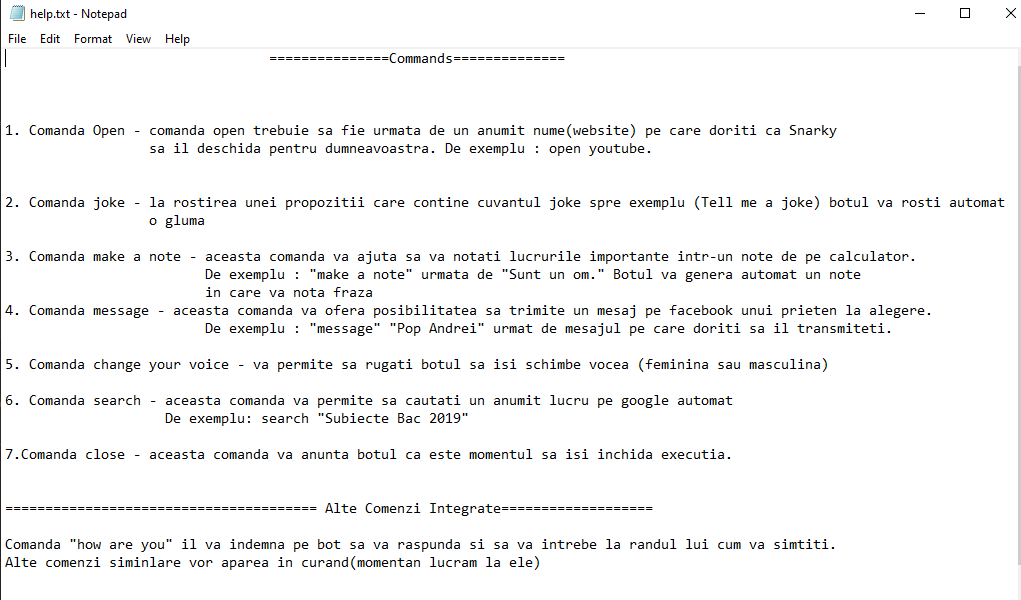
speak("Here are all my commands I hope you will find this note usefull ")



f.close()

După intropducere numelui de utilizator

La prima utilizare programul deschide automat noteul help in care sunt prezentate comenzile acestuia.



* 1. **Funcția inceput()**

Subprogramul inceput(),la fel ca si functia de cunoastere() verifică in acelasi fisier dacă există un nume pe care il va copia si apela pentru utilizatorul existent. Cu ajutorul librăriei time Bot assistentul va verifica in ce perioadă din zi este și la inceperea programului va saluta corespunzător (Good morning pana in orele 10 a.m, Hello pentru intervalul ora 10 – 18 si Good evening pentru orele de la 18 pana la 00).



Codul functiei inceput()

def inceput():

f= open('recunoastere.txt','r')

nume=f.readline()

f.close()

timp=datetime.datetime.now()

if timp.hour < 10:

audio='Good morning ' +nume+'! My name is Snarky and i will be your bot assistant.'

speak(audio)

elif timp.hour>=10 and timp.hour<18:

audio='Hello '+nume +'! My name is Snarky and i will be your bot assistant.'

speak(audio)

time.sleep(0.5)

speak('How was your day so far?')

elif timp.hour>=19:

audio='Good evening '+nume+'! My name is Snarky and i will be your bot assistant.'

speak(audio)

return nume

* 1. **Funcția uitare()**

Subprogramul uitare este apelat in momentul in care utilizatorul doreste sa oprească programul. “Snarky” il va intreba dacă dorește sa ii retină numele in fieșier sau dorește sa il șteargă. Următoarea pornire a programului va depinde de acest lucru.

def uitare(nume):

f= open('recunoastere.txt','r+')

lines=f.readlines()

f.seek(0)

for line in lines:

if line.strip("\n")!= nume:

f.write(line)

f.truncate()

f.close()

**3.7 Functia login(user,parola)**

Subprogramul login care primește ca parametrii un user ce reprezintă emailul contului de facebook si parola pe care le introduce automat pentru conectarea pe facebook.(funcția este specializată pe conectarea in browser.Următoarele funcții sunt specializate pe conectarea si trimiterea de mesaje.)

def login(user , parola):

driver = webdriver.Chrome('/Users/Dell/source/repos/Snarky2.0/chromedriver')

driver.get("https://www.facebook.com/")

driver.find\_element\_by\_id('email').send\_keys(user)

driver.find\_element\_by\_id('pass').send\_keys(parola)

time.sleep(1)

driver.find\_element\_by\_id('loginbutton').click()

**3.8 Funcțiile de mesagerie pe Facebook/ Messanger**

3.8.1 Functia defemail(comandaA()) – primeste ca parametru o comanda vocala care reprezintă emailul pe care utilizatorul il rosteste și ii atribuie “@yahoo.com” la final iar mai apoi intreabă utilizatorul dacă emailul pe a fost inregistrat correct.

def defemail():

email=comandaA()

speak("Is this the right email?")

print(email+”@yahoo.com”);

x=comandaA()

if(x==”da”)

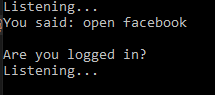
return true

else

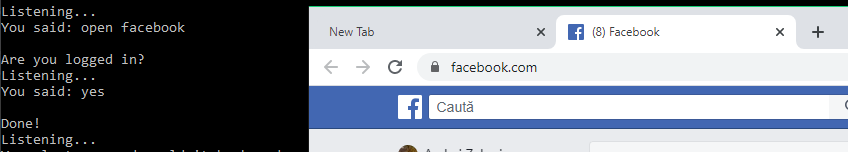
return false

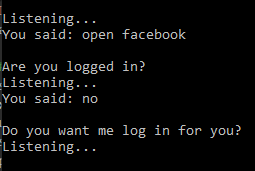
3.8.2 Funcția facebook(url)- primeste ca parametru adresa url a siteului facebook.

Subprogramulo contine o serie de verificări pe care programul le face inainte de a deschide web siteul in browser.

**Pasul 1**. Când utilizatorul dorește să deschidă facebook ul va fi intrebat de cître program daca este logat.

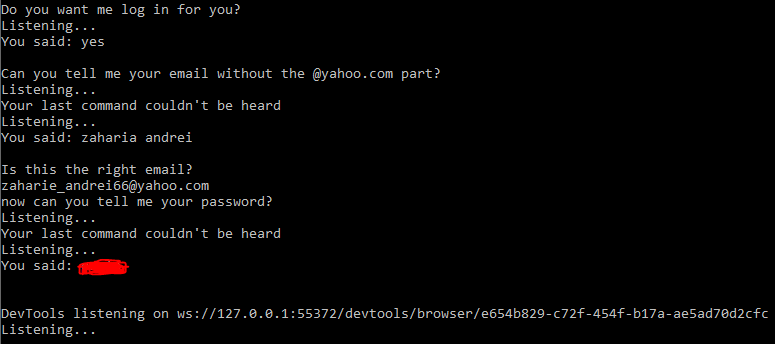
**Pasul 2.** În funcție de răspunsul la prima intrebare programul va continua pe două ramuri diferite . Dacă raspunsul a fost afirmativ programul va deschide facebookul in browser.

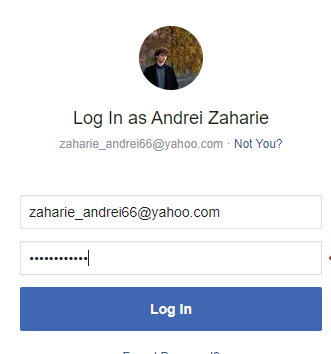


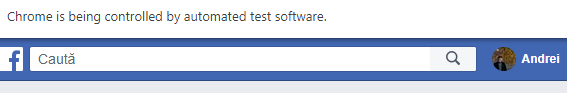
Dacă răspunsul este negativ atunci programul va continua prin a intreba utilizatorul dacă dorește ca acesta sa il ajute să se conecteze.

Dacă raspunsul este negativ(“no”) atunci programul va deschide facebookul in browser și îl va lăsa peutilizator să se logheze manual.

În cazul în care răspunsul este pozitiv(“yes”) atunci programul va inițializa procesul de colectare a datelor necesare pentru conectare( email și parolă)





 După cum se poate observa in exemplul de mai sus după rostirea emailului si parolei(care este colorată cu roșu din motive de singuranță personală) programul va deschide automat pagina de facebook si va introduce informațiile userului.

Observație: In cazul in care utilizatorul sesizează că emailul sau parola nu ai fost înregistrate corect prin răspunsul negativ la îintrebările („Is this the right email”/”Is thie the right password?”) procesul va fi reluat si repetat până când emailul si parola vor fi înregistrate corect.

Codul sursă a subprogramului facebook():

def facebook(url):

speak('Are you logged in?')

raspuns=comandaA()

if 'no' in raspuns:

speak("Do you want me log in for you?")

ras=comandaA()

if'yes' in ras:

#webbrowser.close(url)

speak("Can you tell me your email without the @yahoo.com part?")

x=defemail()

user= 'x'+'@yahoo.com'

speak("now can you tell me your password?")

parola=comandaR()

login(user,parola)

if 'yes' in x:

user= user+'@yahoo.com'

speak("now can you tell me your password?")

parola=comandaR()

login(user,parola)

elif 'no' in x:

speak("Can you repeat if please")

email=comandaR()

speak("Is this the right email?")

print(email+'@yahoo.com');

x=comandaA()

if 'yes' in x:

user= email+'@yahoo.com'

speak("now can you tell me your password?")

parola=comandaA()

login(user,parola)

elif'no' in x:

speak("I can't understand can you write it for me,please?")

else:

webbrowser.open(url)

else:

webbrowser.open(url)

print('Done!')

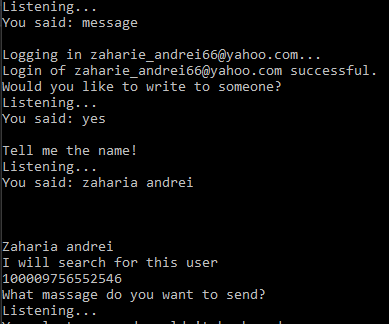
3.8.3 Funcția mesagerie()- este create cu ajutorul funcțiilor din librăriei fbchat care reprezintă un API creeat de cei de la facebook special pentru limbajul python.

Când funcția este apelată aceasta face conexiunea cu ajutorul contului de facebook al userului și permite transformarea mesajelor vocale pe care programul le primeste in mesaje de tip text care mai apoi sunt trimise unei persoane sau unui grup ales de către utilizator.

Observație: Mesajele pot fi trimise indifferent daca aplicatia facebook este deschisa sau nu in browser( funcția mesagerie() fiind complet independent față de procesele funțiilor facebook() si defeamil().).

Prezentarea subprogramului.

Când utilizatorul dorește sa trimită un mesaj subprogramul ii va cere numele complet al persoanei sau al grupului catre care doreste să trimită mesajele după care va transforma numele ca acesta sa se potrivească cu formatul pe care il cere facebookul si anume ( numele să inceapă cu litere de mari si să fie separate prin exact un spațiu).



In exemplul de mai sus se observă un număr format din 16 cifre care repreintă user idul persoanei de pe facebook.

După incheierea fazei de logare si găsire a grupului sau persoanei dorite va continua un loop format dintr-un While care va permite trimiterea unui număr nedefinit de mesaje până cand utilizatorul va rosti cuvântul cheie care permite sistarea executiei acestui subprogram si anume cuvântul “stop”.

Observație: Utilizarea cuvântului stop nu va produce oprirea subprogramului in cazul in care acesta se află intr-o propoziție. De exmplu “M-am oprit la stop” nu va produce oprirea Whileului.

Codul sursă a subprogramului mesagerie():

client1= Client('email,'parola') – email si parola de la contul de facebook

speak("Would you like to write to someone?")

raspuns=comandaA()

if 'yes' in raspuns:

speak('Tell me the name!')

raspuns = comandaR()

raspunsok=raspuns[0].upper()

raspuns=raspuns[:0]+raspuns[1:]

raspunsok=raspunsok+raspuns+ " "

#raspuns = comandaR()

raspunsok=raspunsok+raspuns[0].upper()

raspuns=raspuns[:0]+raspuns[1:]

h=""

for i in range(0,len(raspuns)):

if (raspuns[i] !=" "):

h=h+raspuns[i]

raspunsok=raspunsok+h

x=client1.searchForUsers(raspunsok,2)

print('\n')

print(raspunsok)

speak("I will search for this user")

x=client1.searchForUsers("raspunsok",2)

print(x[0].uid)

userid=x[0]

speak('What massage do you want to send?')

l=1

while l!=0:

mesaj = comandaR()

if 'stop'!= mesaj:

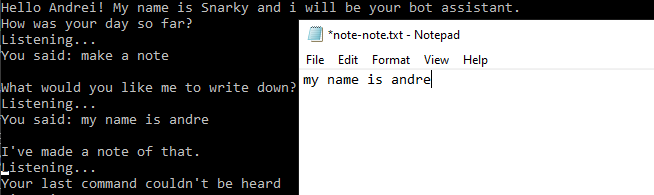
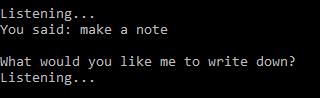
client1.send(fbchat.models.Message(mesaj),userid.uid)

else:

l=0

**3.9 Funcția note()**

Funcți note(audio) primește ca parametru comanda vocală transformată in text după care creeaza automat un notepad în care scrie comanda pe care utilizatorul si-a dorit să o noteze, după care afisează notepadul pentru ai oferi utilizatorului o siguranță că acest fișier a fost creeat si salvat.



Codul subprogramului note()

def note(audio):

date = 'note'

file\_name = str(date).replace(":", "-") + "-note.txt"

with open(file\_name, "w") as f:

f.write(audio)

subprocess.Popen(["notepad.exe", file\_name])

**Modulul de recunoaștere vocală și transformarea acestuia in text.**

**Cap. 4**

**Prezentare-** Acest modul face parte din biblioteca sppech\_recognition care a fost prezentată ulterior si biblioteca pyttsx (python text to speech) care are ca rol transformarea comenzii vocale in text.

**4.1 comandaA():** Acest subprogram folosește mai multe funcții predefinite precum clasa Microphone() care cotine opțiuni de ajustare a sunetului de fundal, durata ascultarii, marja de erroare acceptată etc. Dar si clasa Recognizer() care reușesște sa înregistreze comanda vocala și să o interpreteza cu ajutorul modulul de inteligență artificială pus la dispoziție de Google.

**Precizare**: Clasa recognize\_google() conține doi parametrii care reprezinta imputul pe care trebuie sa il interpreteze, in cazul nostru comanda vocală și un cel de al doilea parametru ce reprezintă limba în care care programul trebuie sa caute si să interpreteze comanda.

***Codul sursă***

def comandaA():

r = sr.Recognizer()

with sr.Microphone() as source:

print('Listening...')

r.pause\_threshold = 1

r.adjust\_for\_ambient\_noise(source, duration=1)

audio = r.listen(source)

try:

comanda = r.recognize\_google(audio).lower()

print('You said: ' + comanda + '\n')

#revenirea in loop in cazul in care comanda nu a fost recunoscută

except sr.UnknownValueError:

print('Your last command couldn\'t be heard')

comanda = comandaA()

return comanda

**4.1 comandaR():** Subprogramul este aproape identic cu cel prezentat pentru limba engleză, singurele modificări aduse fiind opțiune pentru limba română.

comanda = r.recognize\_google(audio, language='ro-RO').lower()

Observație : In cazul în care cel de al doilea parametru din subprogramul r.recognize\_google nu este completat acesta va interpreta mesajele in limba engleză.

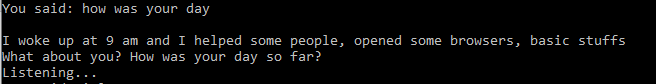
**Subprogramul principal assistant(comanda)**

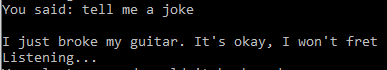
**Cap. 5**

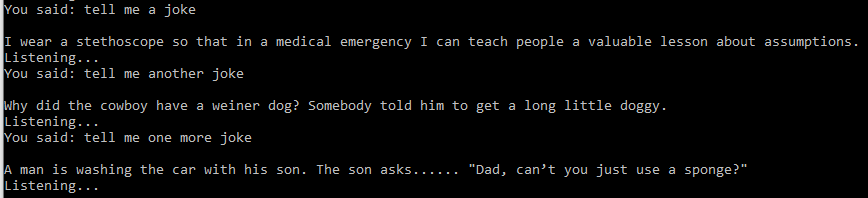
Acest subprogram conține si face posibilă funcționarea tuturor subprogramelor prezentate anterior. Acesta are o structură asemănătoare ca funcționalitate cu cea a swichului. Asftel intr-un while acest subprogram este apelat de fiecare dată când o noua comandă vocală este inregistrată iar mai apoi cu ajutorul unui set de ifuri si elifuri(echivalentul else if in c++) verifică comanda pe care mai apoi o execută apelând subprogramul specific acelei accțiuni.

Pe lângă subprogramele amintite mai sus in assistant() se pot regăsi și câteva comenzi dedicate pentru mentinerea socializârii dintre utilizator și bot. Astfel acesta are câteva intrebări definite la care poate răspunde si interacționa cu utilizatorul.

Exemple





De asemenea unele dintre aceste module de interacționare au fost realizate dinamic pentru a nu se repeta si implicit a distruge incercarea de umanizare a botului. Astfel funcții precum **tell me a joke** sau **how was your day** se vor schimba random in cazul primei cereri si in funcție de ora la care este pusa intrebarea in cazul celei de a doua.

***Codul sursă***

def assistant(comanda):

NOTE = ["make a note", "write this down", "remember this", "type this"]

"if statements for executing commands"

if 'change your voice' in comanda:

global vocea

if vocea==1:

vocea=0

speak('I have changed my voice back, sir!')

else:

vocea=1

speak('I have changed my voice for you')

elif 'close' in comanda:

global iesire

iesire = 1

elif comanda in NOTE:

speak("What would you like me to write down? ")

scrie = comandaA()

note(scrie)

speak("I've made a note of that.")

elif 'open ' in comanda:

reg\_ex = re.search('open (.+)', comanda)

if reg\_ex:

domain =reg\_ex.group(1)

url = 'https://www.'+domain +'.com'

if 'facebook' in domain:

facebok(url)

else:

webbrowser.open(url)

print('Done!')

else:

pass

elif 'how are you' in comanda:

speak("I am good how are you?")

elif'how was your day'in comanda:

timp=datetime.datetime.now()

if timp.hour < 10:

audio='Well i did nothing so far. I am not a morning person'

speak(audio)

elif timp.hour>=10 and timp.hour<18:

audio='I woke up at 9 am and I helped some people, opened some browsers, basic stuffs'

speak(audio)

time.sleep(0.5)

speak ("What about you? How was your day so far?")

elif timp.hour>=19:

audio='My day was really good and I did a lot of good things'

speak(audio)

elif 'i love you' in comanda:

speak("Awww.I was not programmed to have feelings but I think I love you too!")

elif 'joke' in comanda:

res = requests.get(

'https://icanhazdadjoke.com/',

headers={"Accept":"application/json"}

)

if res.status\_code == requests.codes.ok:

speak(str(res.json()['joke']))

else:

speak('Hmmm Sorry sir I think I ran out of jokes.')

elif 'search' in comanda:

com = re.search('search (.+)', comanda)

cautare= com.group(1)

cautare\_avansata(cautare)

elif 'message' in comanda:

mesagerie()

elif 'help page' in comanda:

subprocess.Popen(["notepad.exe", "Help.txt"])

speak("Here are all my commands I hope you will find this note usefull ")

#loop to continue executing multiple commands

return 1

**Codul sursă pentru intreg programul Bot Assistant.**

**Cap. 6**

import speech\_recognition as sr

import os

import re

import webbrowser

import smtplib

import time

import datetime

import requests

import mpg123

import winsound

import turtle

import playsound

import pyttsx3

import subprocess

import sys , bs4

from urllib.parse import urlparse

import urllib.request

import urllib3

import selenium

from selenium import webdriver

import fbchat

from fbchat.models import \*

from fbchat import Client

import tkinter as tk

import \_thread

from turtle import \*

import random

#Asistentul Snarky

#by Zaharie Andrei

#==========================================================

'''Toate Drepturile de autor se rezerva lui ZAHA TECH INDUSTIES.

'''

#================================================================

iesire =0

# MODUL SPEAK ============================================

vocea=0

def speak(audio):

"speaks audio passed as argument"

global vocea

engine = pyttsx3.init()

voices = engine.getProperty('voices')

engine.setProperty('voice', voices[vocea].id)

print(audio)

engine.say(audio)

engine.runAndWait()

#Cunoastere persoana=============

def cunoastere():

f= open('recunoastere.txt','r+')

if os.path.getsize('recunoastere.txt')==0:

speak('Hey are you new? How would you like me to call you?')

x=input('Name: ')

f.write(x)

speak("Before we get started I will open my commands list for you.")

subprocess.Popen(["notepad.exe", "Help.txt"])

speak("Here are all my commands I hope you will find this note usefull ")

f.close()

#Interfata grafica==============

N = 80

myList = [3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9]

t=random.choice(myList)

#Subprograme ce determina graficele unor functii matematice pt partea de loading

def f(x):

global t

return t\*x\*(1-x)

def g(x):

global t

return t\*x-t\*x\*x

def h(x):

return 3.9\*(x-x\*\*2)

def jumpto(x, y):

penup(); goto(x,y)

def line(x1, y1, x2, y2):

jumpto(x1, y1)

pendown()

goto(x2, y2)

def coosys():

line(-1, 0, N+1, 0)

line(0, -0.1, 0, 1.1)

def plot(fun, start, color):

pencolor(color)

x = start

jumpto(0, x)

pendown()

dot(5)

for i in range(N):

x=fun(x)

goto(i+1,x)

dot(5)

#=====================

def interfata():

colors = [ "red","purple","blue","green","orange","yellow"]

pen = turtle.Pen()

turtle.bgcolor("white")

pen.penup()

pen.backward(200)

pen.pendown()

turtle.hideturtle()

pen.penup()

turtle.write("Bot Assistant",

True,align="center",

font=("Courier",30,"bold"))

pen.pendown()

pen.pencolor("blue")

pen.width(10)

pen.forward(400)

time.sleep(1)

turtle.clearscreen()

pen.pencolor("blue")

pen.width(10)

turtle.hideturtle()

pen.penup()

turtle.write("By Zaha Tech Industries",True,align="center",font=("Cornerstone",20))

pen.pendown()

pen.goto(-200,0)

pen.goto(200,0)

time.sleep(2)

turtle.clearscreen()

turtle.hideturtle()

pen.penup()

setworldcoordinates(-1.0,-0.1, N+1, 1.1)

speed(0)

turtle.write("Loading...",True,align="left",font=("Courier",30,"bold"))

hideturtle()

coosys()

plot(f, 0.35, "blue")

plot(f, 0.35, "green")

#plot(h, 0.35, "red")

turtle.clearscreen()

turtle.write("Done",True,align="left",font=("Courier",30,"bold"))

turtle.bye()

#PREZENTARE ==============================

def inceput():

f= open('recunoastere.txt','r')

nume=f.readline()

f.close()

timp=datetime.datetime.now()

if timp.hour < 10:

audio='Good morning ' +nume+'! My name is Snarky and i will be your bot assistant.'

speak(audio)

elif timp.hour>=10 and timp.hour<19:

audio='Hello '+nume +'! My name is Snarky and i will be your bot assistant.'

speak(audio)

time.sleep(0.5)

speak('How was your day so far?')

elif timp.hour>=19:

audio='Good evening '+nume+'! My name is Snarky and i will be your bot assistant.'

speak(audio)

return nume

#RETINERE SAU UITARE PERSOANA=======

def uitare(nume):

f= open('recunoastere.txt','r+')

lines=f.readlines()

f.seek(0)

for line in lines:

if line.strip("\n")!= nume:

f.write(line)

f.truncate()

f.close()

# LOGARE AUTOMATA==========================

def login(user , parola):

driver = webdriver.Chrome('/Users/Dell/source/repos/Snarky2.0/chromedriver')

driver.get("https://www.facebook.com/")

driver.find\_element\_by\_id('email').send\_keys(user)

driver.find\_element\_by\_id('pass').send\_keys(parola)

time.sleep(1)

driver.find\_element\_by\_id('loginbutton').click()

#driver.get("https://www.facebook.com/messages/t/10000717453743")

#LOGIN INCERCARE==========================

def login2(use,parola):

headers = {

'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.130 Safari/537.36'

}

login\_data = {

'name': 'zaharie\_andrei66@yahoo.com',

'pass': '',

'form\_id': 'new\_login\_form',

'op': 'Login'

}

with requests.Session() as s:

url = 'https://www.facebook.com/'

r = s.get(url, headers=headers)

soup = BeautifulSoup(r.content, 'html5lib')

login\_data['form\_build\_id'] = soup.find('input', attrs={'name': 'form\_build\_id'})['value']

r = s.post(url, data=login\_data, headers=headers)

#=================================================================================

#CAUTARE AVANSATA ===================

def cautare\_avansata(audio):

res=requests.get('https://google.com/search?q='+ ''.join(audio))

soup=bs4.BeautifulSoup(res.text,"html.parser")

linkElements = soup.select(' a')

linkOpen=min(9, len(linkElements))

for i in range(linkOpen):

if i !=0 and i!=1 and i!=6:

webbrowser.open('https://www.google.com'+linkElements[i].get('href'))

speak('That is everything I found for'+ audio)

#MAKE A NOTE ===========================================

def note(audio):

date = 'note'

file\_name = str(date).replace(":", "-") + "-note.txt"

with open(file\_name, "w") as f:

f.write(audio)

subprocess.Popen(["notepad.exe", file\_name])

#MODUL DE FACEBOOK=============================================================

def defemail():

email=comandaA()

speak("Is this the right email?")

print('zaharie\_andrei66'+'@yahoo.com');

#x=comandaA()

#return x

def facebok(url):

speak('Are you logged in?')

raspuns=comandaA()

if 'no' in raspuns:

speak("Do you want me log in for you?")

ras=comandaA()

if'yes' in ras:

#webbrowser.close(url)

speak("Can you tell me your email without the @yahoo.com part?")

x=defemail()

user= 'zaharie\_andrei66'+'@yahoo.com'

speak("now can you tell me your password?")

parola=comandaR()

login(user,parola)

'''if 'yes' in x:

user= 'zaharie\_andrei66'+'@yahoo.com'

speak("now can you tell me your password?")

parola=comandaR()

login(user,parola)

elif'no' in x:

speak("Can you repeat if please")

email=comandaR()

speak("Is this the right email?")

print(email+'@yahoo.com');

x=comandaA()

if 'yes' in x:

user= email+'@yahoo.com'

speak("now can you tell me your password?")

parola=comandaA()

login(user,parola)

elif'no' in x:

speak("I can't understand can you write it for me,please?")'''

else:

webbrowser.open(url)

else:

webbrowser.open(url)

print('Done!')

#Final MODUL FACEBOOK==========================================================

#==========================MESAGERIE AUTOMATA FACEBOOK=========================

def mesagerie():

#login('zaharie\_andrei66@yahoo.com','')

client1= Client('zaharie\_andrei66@yahoo.com','')

speak("Would you like to write to someone?")

raspuns=comandaA()

''' if 'yes' in raspuns:

speak('Tell me the name!')

raspuns = comandaR()

raspunsok=raspuns[0].upper()

raspuns=raspuns[:0]+raspuns[1:]

raspunsok=raspunsok+raspuns+ " "

#raspuns = comandaR()

raspunsok=raspunsok+raspuns[0].upper()

raspuns=raspuns[:0]+raspuns[1:]

h=""

for i in range(0,len(raspuns)):

if (raspuns[i] !=" "):

h=h+raspuns[i]

raspunsok=raspunsok+h

x=client1.searchForUsers(raspunsok,2)

print('\n')

print(raspunsok)

speak("I will search for this user")'''

x=client1.searchForUsers("Marozsan Alex",2)

print(x[0].uid)

userid=x[0]

speak('What massage do you want to send?')

l=1

while l!=0:

mesaj = comandaR()

if 'stop'!= mesaj:

client1.send(fbchat.models.Message(mesaj),userid.uid)

else:

l=0

#client1.send(Message(text='mesaj'), thread\_id=userid.uid, thread\_type=ThreadType.USER)

#Comanda vocala pentru asistent (Limba ENGLEZA)

def comandaA():

r = sr.Recognizer()

with sr.Microphone() as source:

print('Listening...')

r.pause\_threshold = 1

r.adjust\_for\_ambient\_noise(source, duration=1)

audio = r.listen(source)

try:

comanda = r.recognize\_google(audio).lower()

print('You said: ' + comanda + '\n')

#revenirea in loop in cazul in care comanda nu a fost recunoscută

except sr.UnknownValueError:

print('Your last command couldn\'t be heard')

comanda = comandaA()

return comanda

#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ #Comanda vocala pentru asistent (Limba ROMANA)

def comandaR():

"listens for commands"

r = sr.Recognizer()

with sr.Microphone() as source:

print('Listening...')

r.pause\_threshold = 1

r.adjust\_for\_ambient\_noise(source, duration=1)

audio = r.listen(source)

try:

comanda = r.recognize\_google(audio, language='ro-RO').lower()

print('You said: ' + comanda + '\n')

#revenirea in loop in cazul in care comanda nu a fost recunoscută

except sr.UnknownValueError:

print('Your last command couldn\'t be heard')

comanda = comandaA();

return comanda

# Interpretarea Comenzilor

def assistant(comanda):

NOTE = ["make a note", "write this down", "remember this", "type this"]

"if statements for executing commands"

if 'change your voice' in comanda:

global vocea

if vocea==1:

vocea=0

speak('I have changed my voice back, sir!')

else:

vocea=1

speak('I have changed my voice for you')

elif 'close' in comanda:

global iesire

iesire = 1

elif comanda in NOTE:

speak("What would you like me to write down? ")

scrie = comandaA()

note(scrie)

speak("I've made a note of that.")

elif 'open ' in comanda:

reg\_ex = re.search('open (.+)', comanda)

if reg\_ex:

domain =reg\_ex.group(1)

url = 'https://www.'+domain +'.com'

if 'facebook' in domain:

facebok(url)

else:

webbrowser.open(url)

print('Done!')

else:

pass

elif 'how are you' in comanda:

speak("I am good how are you?")

elif'how was your day'in comanda:

timp=datetime.datetime.now()

if timp.hour < 10:

audio='Well i did nothing so far. I am not a morning person'

speak(audio)

elif timp.hour>=10 and timp.hour<18:

audio='I woke up at 9 am and I helped some people, opened some browsers, basic stuffs'

speak(audio)

time.sleep(0.5)

speak ("What about you? How was your day so far?")

elif timp.hour>=19:

audio='My day was really good and I did a lot of good things'

speak(audio)

elif 'i love you' in comanda:

speak("Awww.I was not programmed to have feelings but I think I love you too!")

elif 'joke' in comanda:

res = requests.get(

'https://icanhazdadjoke.com/',

headers={"Accept":"application/json"}

)

if res.status\_code == requests.codes.ok:

speak(str(res.json()['joke']))

else:

speak('Hmmm Sorry sir I think I ran out of jokes.')

elif 'search' in comanda:

com = re.search('search (.+)', comanda)

cautare= com.group(1)

cautare\_avansata(cautare)

elif 'message' in comanda:

mesagerie()

elif 'help page' in comanda:

subprocess.Popen(["notepad.exe", "Help.txt"])

speak("Here are all my commands I hope you will find this note usefull ")

#loop to continue executing multiple commands

return 1

#faza de cunoastere si impretenire=====

# Modul de Recunoastere pesoana (in caz ca aceasta este la prima folosire a programului)

interfata()

cunoastere()

nume=inceput()

#SFARSIT faza=========

#============ Int main()=============

while iesire==0:

assistant(comandaA())

if iesire ==1:

speak('Before I exit, would you like me to remove your name form my database?')

x=input('Write yes/ no: ')

if x=='yes':

uitare(nume)

speak('I will close myself. Goodbye sir!')

print('Program closed...')

**Resurse folosite**

**Cap. 7**

Librării python <https://docs.python.org/3/library/index.html>

Facebook api for python <https://facebook-sdk.readthedocs.io/en/latest/>

Messanger api for python <https://fbchat.readthedocs.io/en/stable/api.html>

Time and datetime documation <https://docs.python.org/3/library/time.html>

<https://docs.python.org/3/library/datetime.html>

Turtle library for python <https://docs.python.org/3/library/turtle.html>

Webbrowser documation <https://docs.python.org/2/library/webbrowser.html>

Selenium documation <https://selenium-python.readthedocs.io/>

Urllib3 documation <https://docs.python.org/2/library/urllib.html>

Requests documation <https://2.python-requests.org/en/master/>

Create your assistant <https://microsoft.github.io/botframework-solutions/virtual-assistant/tutorials/create-assistant/csharp/3-create-project/>

Threading <https://realpython.com/intro-to-python-threading/>

<https://www.tutorialspoint.com/python/python_multithreading.htm>

smptlib documantation <https://docs.python.org/3/library/smtplib.html>