“La Sapienza” University of Rome
Faculty of information engineering, information technology and statistics
Department of informatics, automation and control engineering
"ANTONIO RUBERTI"
Degree program: Artificial Intelligence and Robotics

SAPIENZA
UNIVERSITÀ DI ROMA

HOMEWORK

Task-oriented SDS

Student:	**Student:**	**Student:**
Nagothu Jeevana Bhagyasri	Nandrajog Akash	Medasana Poonam
Matricola: 1847507	**Matricola:** 1734002	**Matricola:** 1874750

Teacher: NARDI, Daniele

Rome, 2019

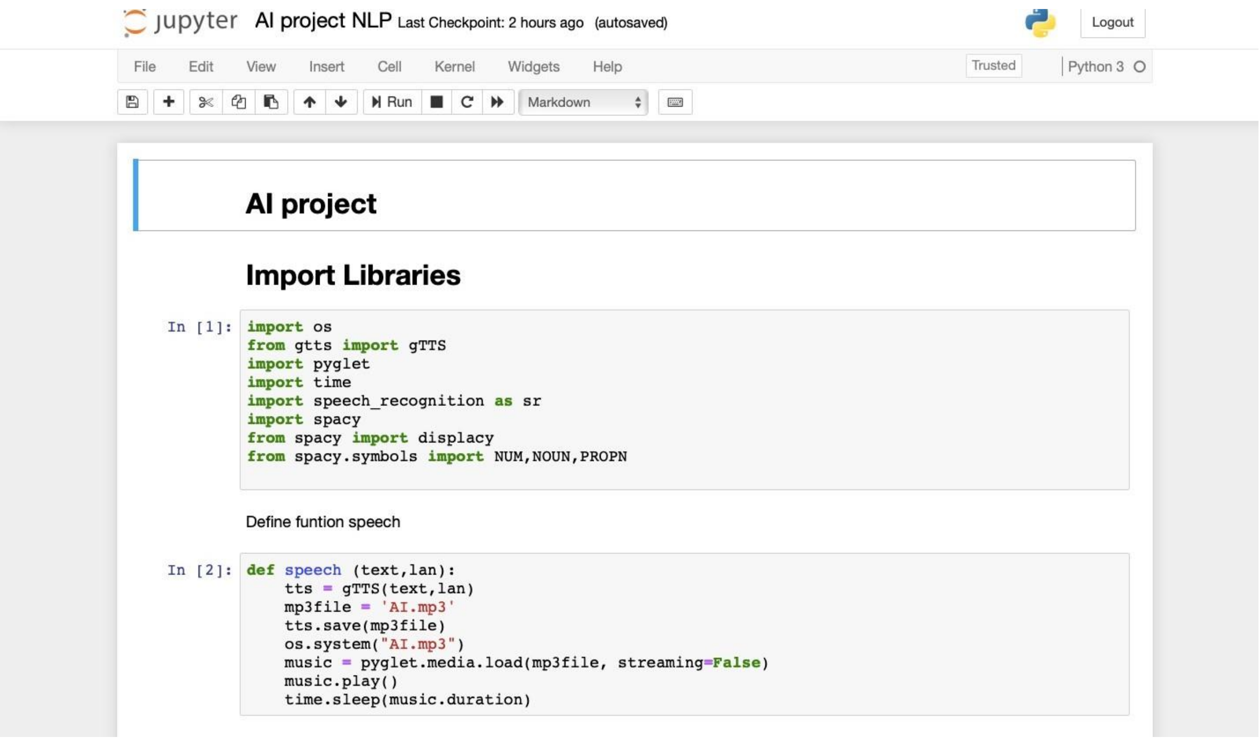
Exercise: to implement task-oriented SDS according to a preferable scenario.

Instruments: Google Speech Recognition API (1), SPACY Parser (2), Jupyter Notebook (3).

Mechanism of the program: a program written in "**Jupyter notebook**" provides a dialog with a user. It recognizes user's voice and translates it into a text using "**SpeechRecognition API**", after that works out this data and "understands its meaning" using "**Spacy Parser**" and gives an answer based on output data of "Spacy Parser".

Scenario: A "Waiter Robot" it serves and responds according to the user who entered into the café to have something based on their interests.

The report is given as a collection of screenshots from Jupyter Notebook.



AI project

Import Libraries

In [1]:
```
import os
from gtts import gTTS
import pyglet
import time
import speech_recognition as sr
import spacy
from spacy import displacy
from spacy.symbols import NUM,NOUN,PROPN
```

Define funtion speech

In [2]:
```
def speech (text,lan):
    tts = gTTS(text,lan)
    mp3file = 'AI.mp3'
    tts.save(mp3file)
    os.system("AI.mp3")
    music = pyglet.media.load(mp3file, streaming=False)
    music.play()
    time.sleep(music.duration)
```

In the above screenshot, Firstly I have imported the packages like

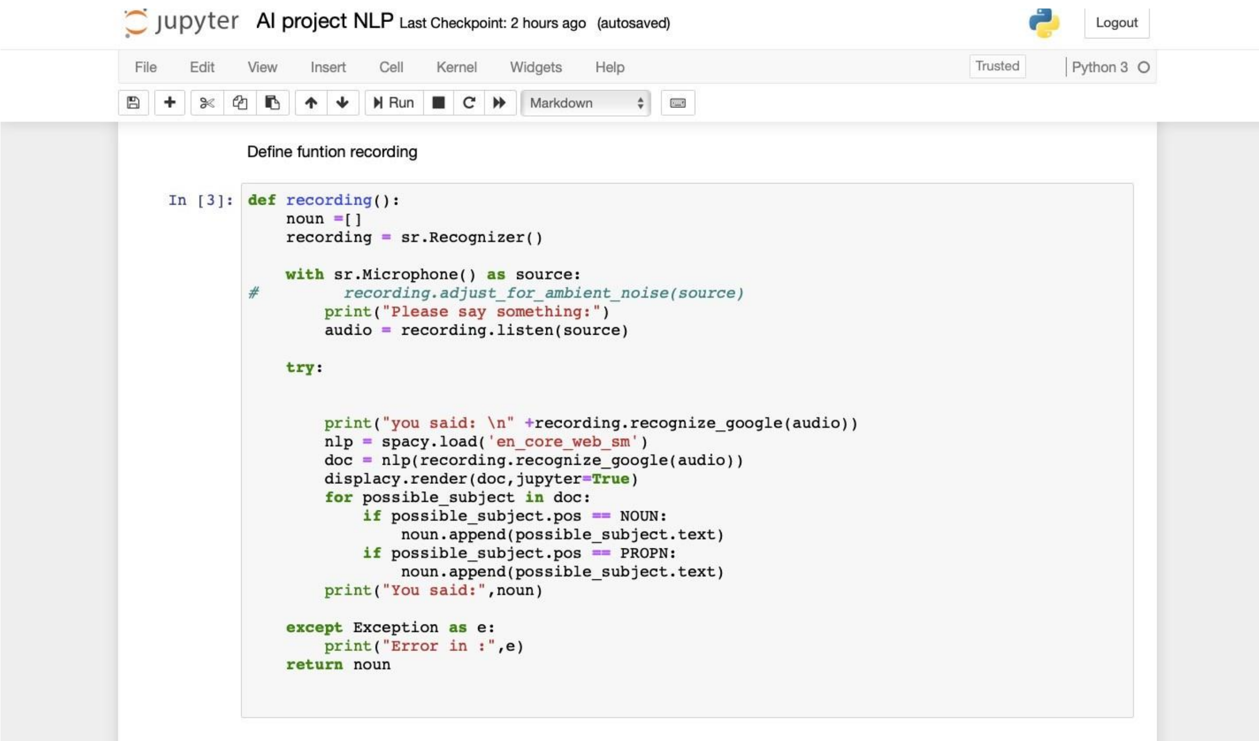
:-**Spacy** to understand the meaning of the text

:-**Speech Recognition** to recognizes users voice and translate it into text

:-**pyglet** to windowing, user interface even handling, and to play sounds and music.

:-**Gtts** Google's text to speech interface inorder to convert string text to spoken text and can be saved as .mp3.

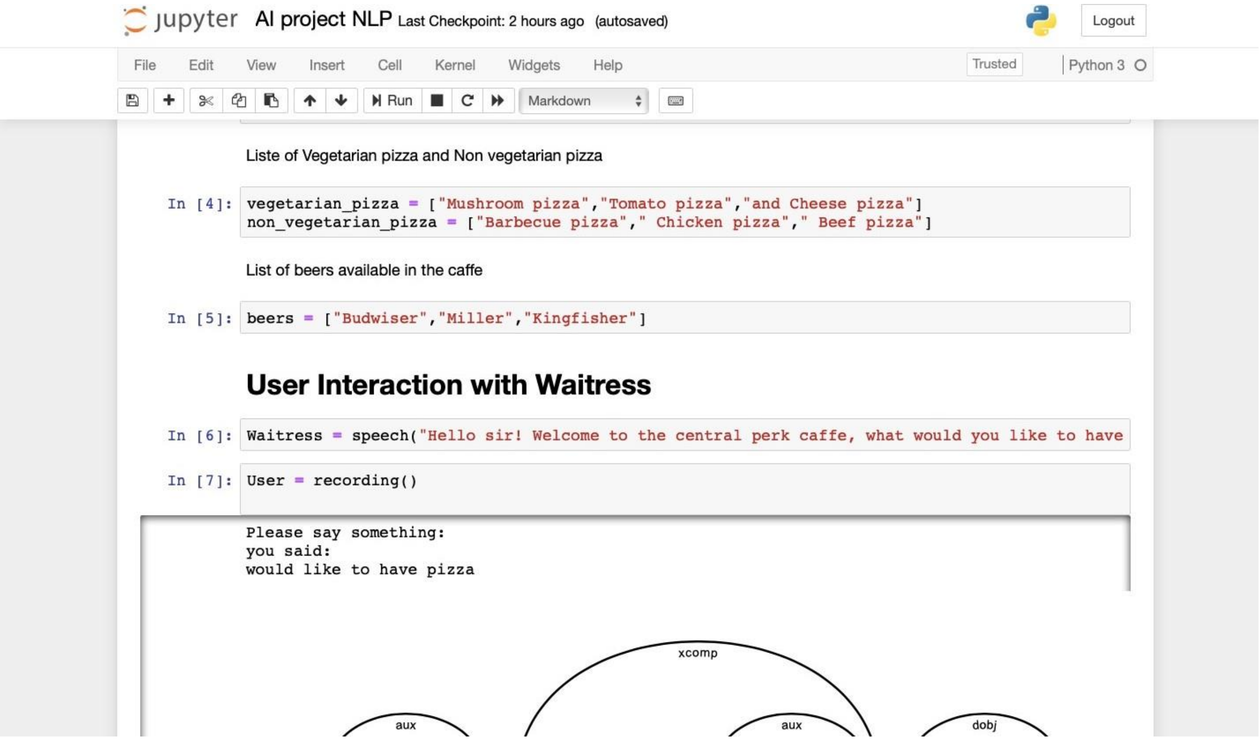
And, I was written module speech which contains the above text..



jupyter AI project NLP Last Checkpoint: 2 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown

Define funtion recording

```
In [3]: def recording():
            noun =[]
            recording = sr.Recognizer()

            with sr.Microphone() as source:
        #           recording.adjust_for_ambient_noise(source)
                print("Please say something:")
                audio = recording.listen(source)

            try:


                print("you said: \n" +recording.recognize_google(audio))
                nlp = spacy.load('en_core_web_sm')
                doc = nlp(recording.recognize_google(audio))
                displacy.render(doc,jupyter=True)
                for possible_subject in doc:
                    if possible_subject.pos == NOUN:
                        noun.append(possible_subject.text)
                    if possible_subject.pos == PROPN:
                        noun.append(possible_subject.text)
                print("You said:",noun)

            except Exception as e:
                print("Error in :",e)
            return noun
```



jupyter AI project NLP Last Checkpoint: 2 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown

Liste of Vegetarian pizza and Non vegetarian pizza

```
In [4]: vegetarian_pizza = ["Mushroom pizza","Tomato pizza","and Cheese pizza"]
        non_vegetarian_pizza = ["Barbecue pizza"," Chicken pizza"," Beef pizza"]
```

List of beers available in the caffe

```
In [5]: beers = ["Budwiser","Miller","Kingfisher"]
```

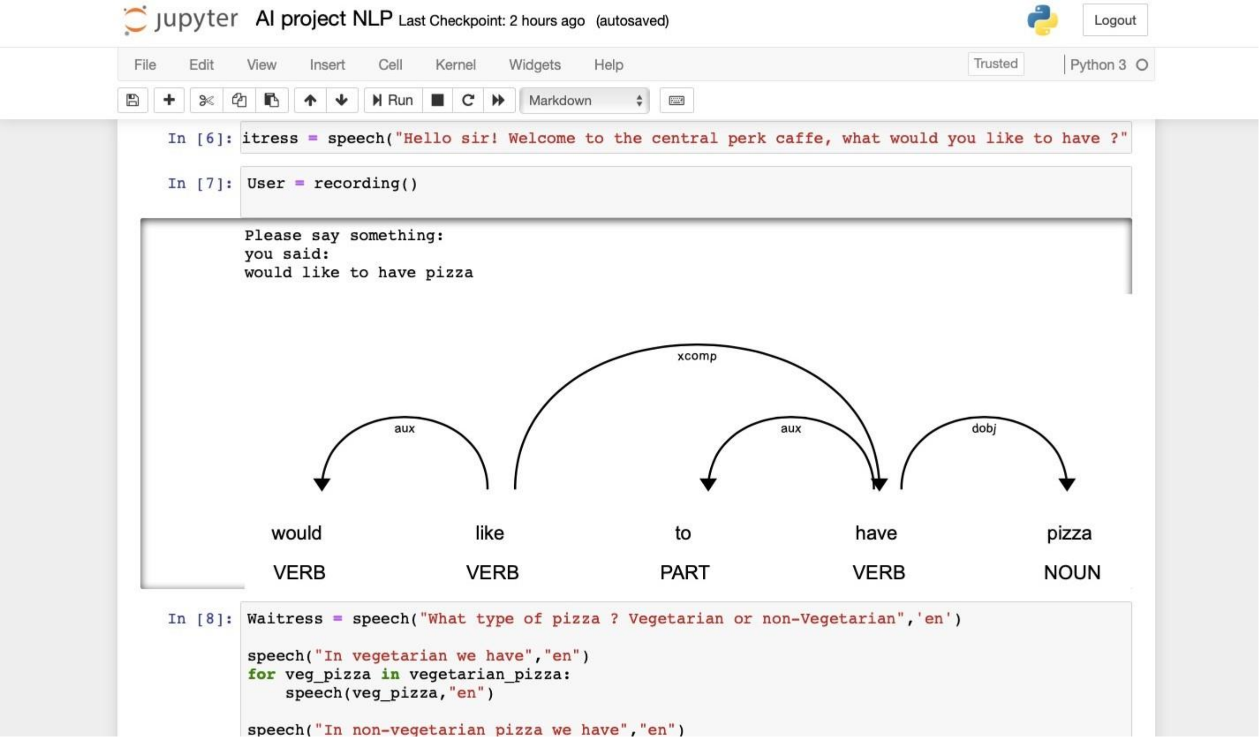
User Interaction with Waitress

```
In [6]: Waitress = speech("Hello sir! Welcome to the central perk caffe, what would you like to have
```

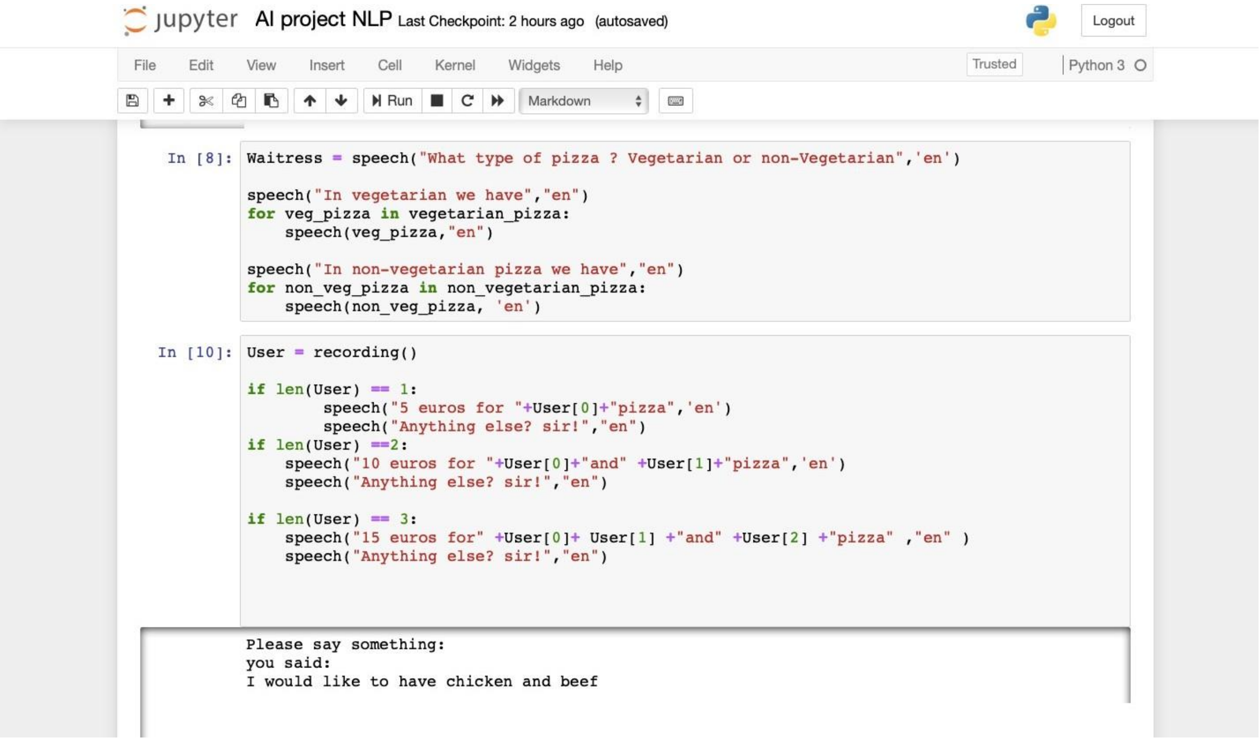
```
In [7]: User = recording()
```

```
Please say something:
you said:
would like to have pizza
```

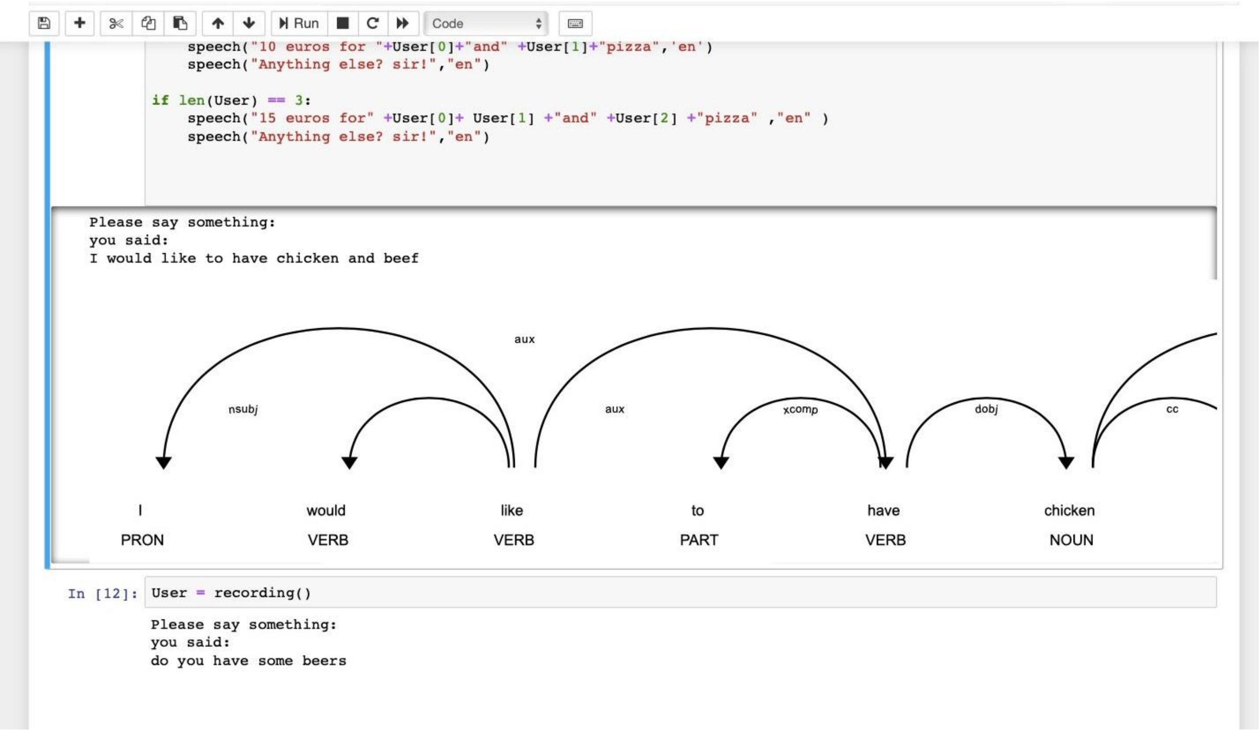
Here, Spacy understands the text and meaning of it and able to differentiate the words as noun, verb, conjunction and preposition.


jupyter AI project NLP Last Checkpoint: 2 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown
xcomp
aux
aux
dobj
would
VERB
like
VERB
to
PART
have
VERB
pizza
NOUN

```
In [6]: itress = speech("Hello sir! Welcome to the central perk caffe, what would you like to have ?"

In [7]: User = recording()

        Please say something:
        you said:
        would like to have pizza

In [8]: Waitress = speech("What type of pizza ? Vegetarian or non-Vegetarian",'en')

        speech("In vegetarian we have","en")
        for veg_pizza in vegetarian_pizza:
            speech(veg_pizza,"en")

        speech("In non-vegetarian pizza we have","en")
```


jupyter AI project NLP Last Checkpoint: 2 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown

```
In [8]: Waitress = speech("What type of pizza ? Vegetarian or non-Vegetarian",'en')

        speech("In vegetarian we have","en")
        for veg_pizza in vegetarian_pizza:
            speech(veg_pizza,"en")

        speech("In non-vegetarian pizza we have","en")
        for non_veg_pizza in non_vegetarian_pizza:
            speech(non_veg_pizza, 'en')

In [10]: User = recording()

         if len(User) == 1:
                 speech("5 euros for "+User[0]+"pizza",'en')
                 speech("Anything else? sir!","en")
         if len(User) ==2:
             speech("10 euros for "+User[0]+"and" +User[1]+"pizza",'en')
             speech("Anything else? sir!","en")

         if len(User) == 3:
             speech("15 euros for" +User[0]+ User[1] +"and" +User[2] +"pizza" ,"en" )
             speech("Anything else? sir!","en")

         Please say something:
         you said:
         I would like to have chicken and beef
```

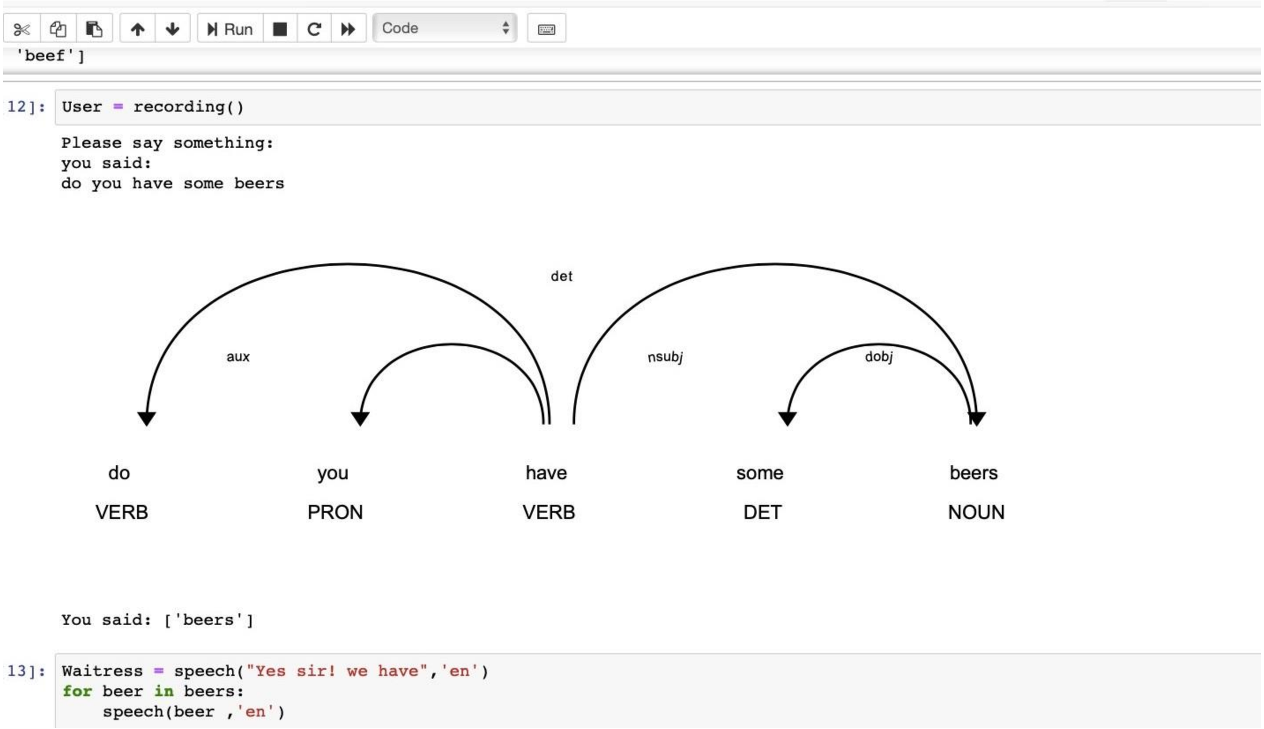


```
        speech("10 euros for "+User[0]+"and" +User[1]+"pizza",'en')
        speech("Anything else? sir!","en")

    if len(User) == 3:
        speech("15 euros for" +User[0]+ User[1] +"and" +User[2] +"pizza" ,"en" )
        speech("Anything else? sir!","en")
```

```
Please say something:
you said:
I would like to have chicken and beef
```

```
In [12]: User = recording()
```

```
Please say something:
you said:
do you have some beers
```



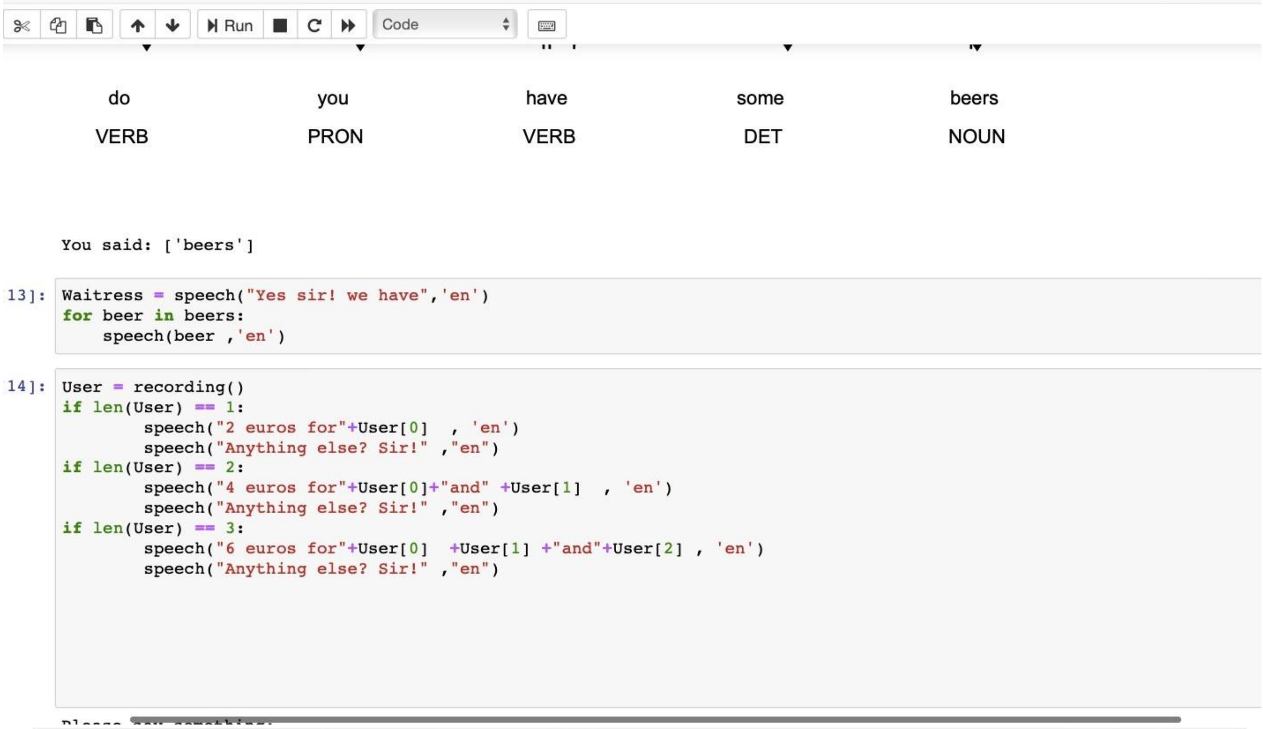
```
'beef']
```

```
12]: User = recording()
```

```
Please say something:
you said:
do you have some beers
```

```
You said: ['beers']
```

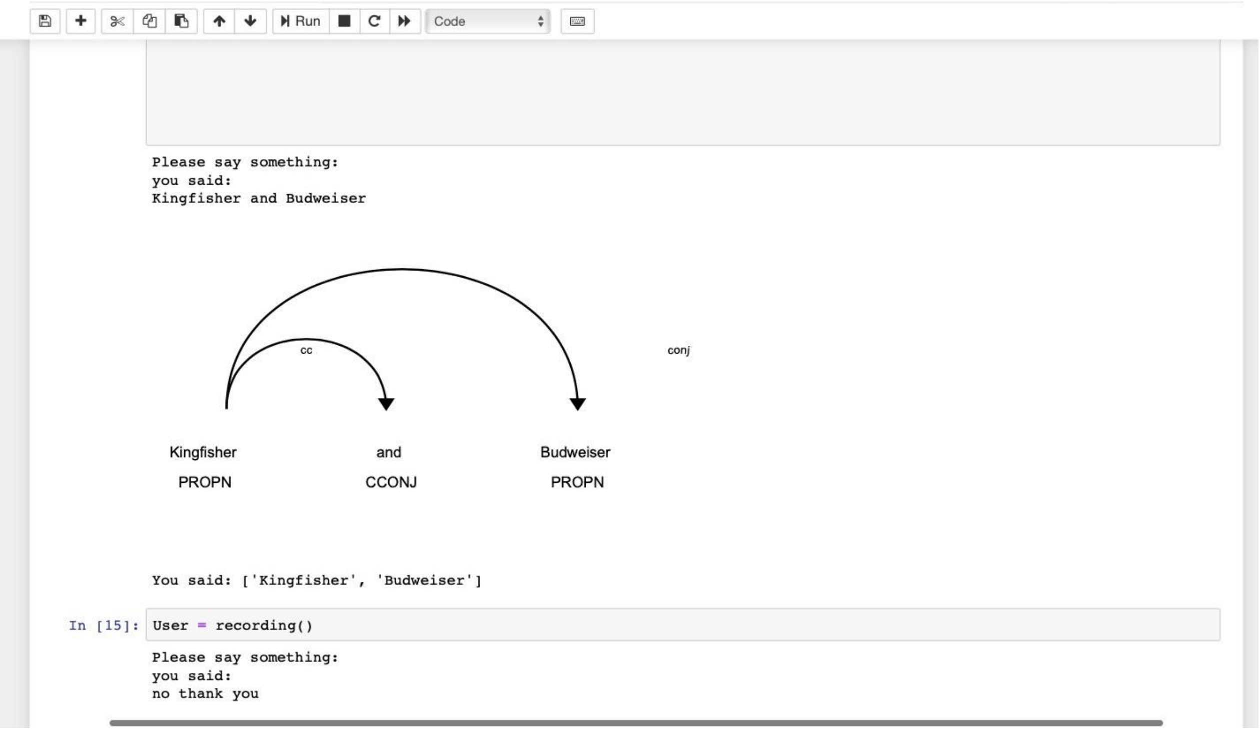
```
13]: Waitress = speech("Yes sir! we have",'en')
     for beer in beers:
         speech(beer ,'en')
```



```
You said: ['beers']
```

```
13]: Waitress = speech("Yes sir! we have",'en')
     for beer in beers:
         speech(beer ,'en')
```

```
14]: User = recording()
     if len(User) == 1:
             speech("2 euros for"+User[0]  , 'en')
             speech("Anything else? Sir!" ,"en")
     if len(User) == 2:
             speech("4 euros for"+User[0]+"and" +User[1]  , 'en')
             speech("Anything else? Sir!" ,"en")
     if len(User) == 3:
             speech("6 euros for"+User[0]  +User[1] +"and"+User[2] , 'en')
             speech("Anything else? Sir!" ,"en")
```

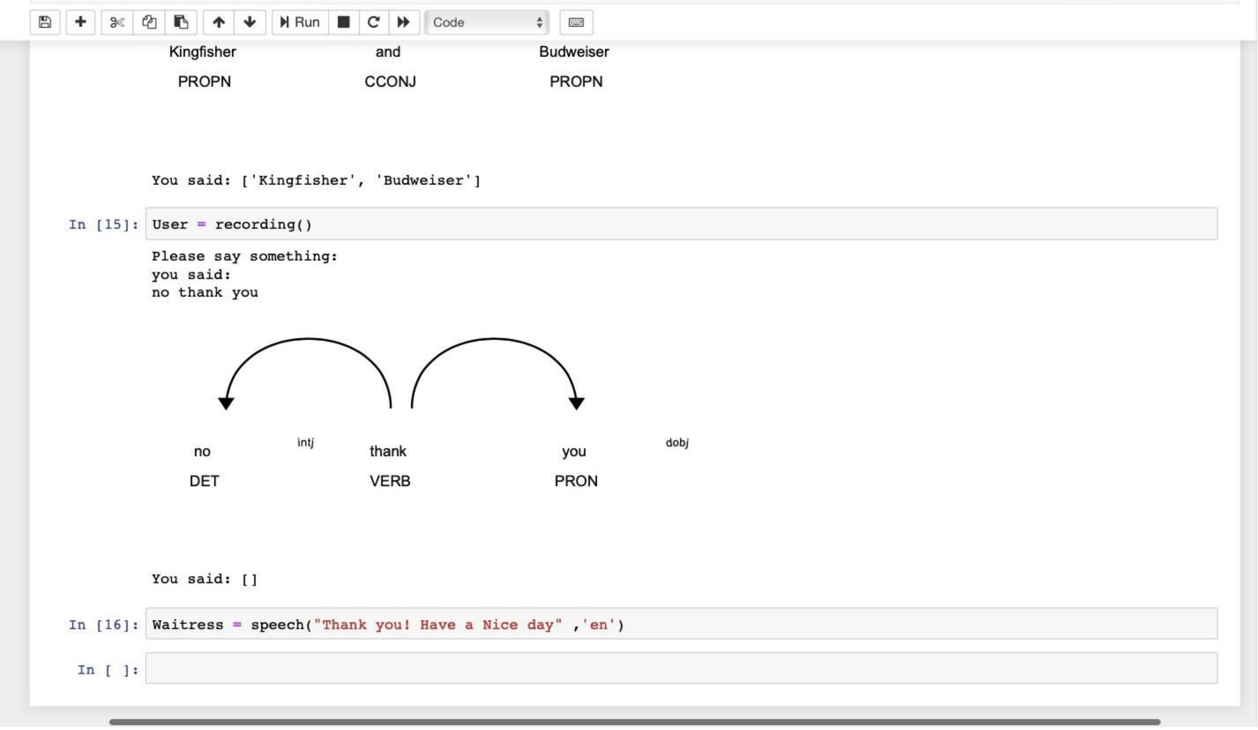


```
Please say something:
you said:
Kingfisher and Budweiser
```

```
You said: ['Kingfisher', 'Budweiser']
```

```
In [15]: User = recording()
```

```
Please say something:
you said:
no thank you
```



```
You said: ['Kingfisher', 'Budweiser']

In [15]: User = recording()

Please say something:
you said:
no thank you

You said: []

In [16]: Waitress = speech("Thank you! Have a Nice day" ,'en')

In [ ]:
```

Conclusion

As we see the robots are becoming the part of everyday life; the use of Waiter Robot can be extended to various functional purposes. There could be so many advantages with these non-human waiters that employers do not have to worry about hiring ,sick time, vacations, or human error.

Resulting program has a good ability for recognizing and understanding human English spoken language (fast Internet connection and silence in environment around are needed). It uses following libraries: speech recognition, speech and spacy.

Possible improvements

Algorithm of the dialogue can be made longer and deeper. Module “speech” can use internal Python *.mp3 player instead of using external program. Also some moments of the program can be improved to increase efficiency of calculations, but for this little demonstration task it is enough.