

Brewen Digonnaux--Lanrelec
Isabelle Somphone
PROMO 2028 – L1 INT 3



Python's Project:
My first chatBot

SUMMARY

INTRODUCTION:	3
REQUIRED WORK FOR THIS PROJECT:	3
FILES OF THE PROJECT :	3
FEATURES	4
MAIN MENU:	4
TUTORIAL	5
TESTING	6
1ST OPTION: PART 1 FUNCTIONALITIES:	6
2ND OPTION: CHATBOT:	7
FACILITIES & DIFFICULTIES:	8
CONCLUSION:	8

Introduction:

Nowadays, there are various generative artificial intelligences such as ChatGPT that are used daily to answer many questions as they take information from books, articles, and many more on the internet. To finish our semester 1, we were asked to do a python's project called "My first chatbot". At our level, we will work on words occurrences to generate responses from a collection of texts. This project was divided into 2 parts and a bonus one to be more organized, in the first one was to define the basic functions to use it later for the main code. Then in the second one, it focused on the calculation that the chatbot must have done to spot similarities and generate answers.

Required work for this project:

We were given a directory with 6 different speeches made by French presidents, we had to create a temporary main program that call a menu and access functions and use the TF-IDF method such as:

Main functions required	Functions to develop from the main ones
<ul style="list-style-type: none">• extract president's names from texts given• associate a first name• display a list of names• convert texts in lower cases + stock contents in new files<ul style="list-style-type: none">• new folder -> 'cleaned'<ul style="list-style-type: none">• delete any punctuation characters• careful of (') and (-)	<ul style="list-style-type: none">• display:<ul style="list-style-type: none">• list of least important words• words with highest TF-IDF• indicate:<ul style="list-style-type: none">• words most repeated by Chirac• names of pres. that spoke of the "Nation" + who repeated it the most

Files of the project :

- Cleaned folder: a folder with new contents in lower case.
- Source folder: contains all the most important functions of our project:
 - **functions.py**: contains all the functions that deal with TF-IDF.
 - **menu.py**: main menu.
 - **ui.py**: extra file for some text formatting.
 - **utils.py**: contains all the functions that are needed across files.
 - **vector.py**: useful to understand the CHATBOT FEATURES part.

Brewen Digonnaux—Lanrelec
Isabelle Somphone
PROMO 2028 – L1 INT3

- Speeches folder: folder containing all president's speeches.
- **README.md**: a Markdown document with a general presentation and how to run the software.
- **chatbot.py**: file to run the program.

Features

Main menu:

To propose a great experience for the user and a better design, we chose to put an interesting font for the title 'CHATBOT'. This menu is command line interface (CLI) where you can choose between 2 main functionalities at the beginning: 'Part 1 functionalities' or 'chatbot'.



Figure 1: First choice that the user must make.

The first choice will offer a set of actions to perform (6 actions) that can be made thanks to functions.py and utils.py. You can see we added some extras like the frame (Figure 2), the code for it, is found on the ui.py file. The user must choose between 1 to 6 for what they want to know. This set of actions is possible with the data pre-processing algorithm and TF-IDF matrix.

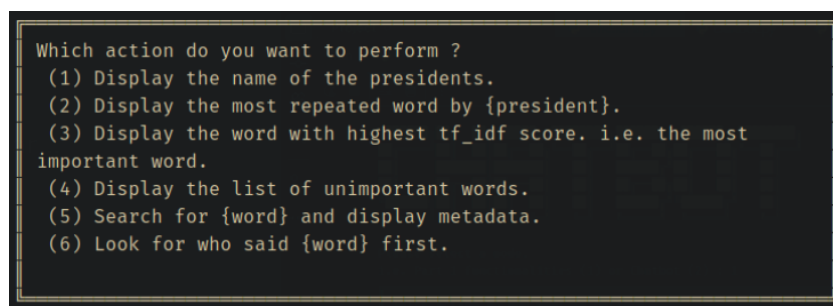


Figure 2: Part 1 functionalities

The second option is the 'chatbot' where you are supposed to ask questions, the chatbot run algorithms and generate answers. When a question is asked, the chatbot will do the same process and calculate a TF-IDF vector for this question. And finally, it will do a similarity calculation, select the best answer, and provide it.

```
Please select a mode.  
i.e. Part 1 functionalities (1) or Chatbot (2) : 2  
  
Ask any question about the president's nominations (warning, the  
question must be in French !)
```

Figure 3: 2nd choice of the user 'chatbot'

We chose to use a class function with the `__init__()` that will initialize all the attributes in this class to be able to call the same functions multiple times and reuse easily the data. We created dictionaries with the president's information. That will create a corpus compiling content from files, then a set '`word_set`' is created and will contains every word in the corpus. After that, it will compute both term frequency (tf) of the corpus, it's inverse document frequency (idf) and the matrix IF-TDF associated.

This whole menu can be display thanks to `menu()` function called in chatbot.py.

Tutorial

As we specified on the README.md file, you can access this program either with PyCharm, in VsCode and finally within a terminal.

How to use

- In PyCharm : Configure usage of `python3.11` with script `chatbot.py` . Ensure the `Emulate terminal in output console` is enabled to avoid error message. To do that, click on the `Current File` in the top left corner of the IDE and select `Edit Configurations...` . Then click on `Create new run configuration` and select `python` . Type `chatbot.py` in the script section. Then click on `Modify options` and add `Emulate terminal in output console` . Finally click `Apply` then `Ok` .
- Within a terminal: type `./chatbot.py`
- In VsCode: with `python` extension installed, open `chatbot.py` and click `run`

Figure 4: Indication on the README.md

Testing

1st option: part 1 functionalities:

To access to the set of actions that will orientate the user, he must type: '1' such as:



```
CHATBOT

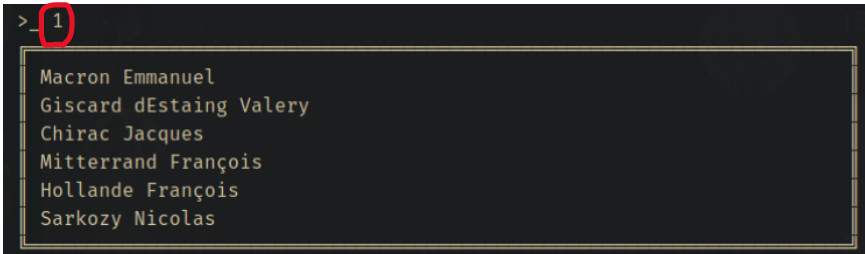
Please select a mode.
i.e. Part 1 functionalities (1) or Chatbot (2) 1

Which action do you want to perform ?
(1) Display the name of the presidents.
(2) Display the most repeated word by {president}.
(3) Display the word with highest tf_idf score. i.e. the most
important word.
(4) Display the list of unimportant words.
(5) Search for {word} and display metadata.
(6) Look for who said {word} first.
```

Figure 5: The user type '1'

It will list all the actions that the user can make. It can display many things such as just the name of the presidents, the most repeated word by one president, the one with the highest TF-IDF, unimportant words, the metadata of one word in particular and who said a word first.

To show how does this work, we will run the program as if the user wanted to display the name of the presidents. As it corresponds to the first action, the user will type as a command '1'. The result will be.



```
>_1
Macron Emmanuel
Giscard dEstaing Valery
Chirac Jacques
Mitterrand François
Hollande François
Sarkozy Nicolas
```

Figure 6: The user wants the 1st action.

Brewen Digonnaux—Lanrelec
Isabelle Somphone
PROMO 2028 – L1 INT3

2nd option: Chatbot:

As it is the second option, the user must type the command '2' to run the chatbot:



Figure 7: The user type '2'

It will then prompt and demand the user any question on the president's nominations, BUT in French. To test it, we will ask him one of the examples given in the directive file for this project:

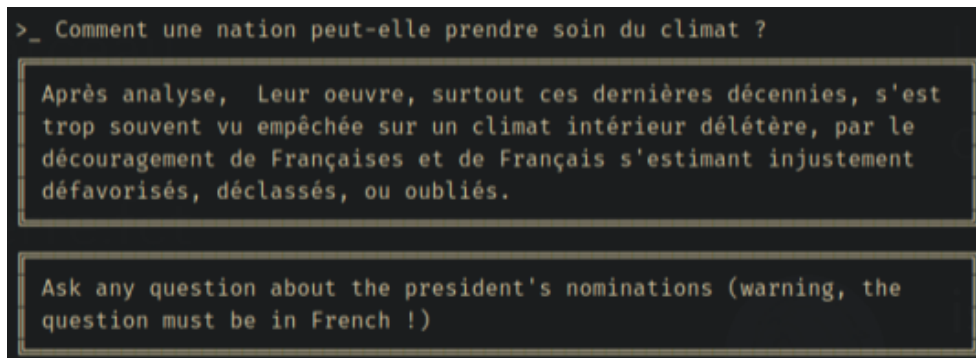


Figure 8: The user asks a question to the chatbot.

It generates an answer and will continue to ask the user to give questions.

For further tests, here below is the link to our git depository:

<https://github.com/sohukia/pychatbot-digonnauxlanrelec-somphone-int3.git>

Facilities & difficulties:

In terms of facilities, we could manipulate vectors when we understood how data were structured. It is great to work in group, to share the burden of the workload even if we didn't have the same level of experience in python programming. This project was also a way to learn as one of us did already have some knowledges. Thanks to communication, the gap between our levels was not disturbing. Technically, naming variables, documenting the code was easy. For duplicate deletion, we learned a new method that consists of converting a list into a dictionary so that the keys are unique, keeping the same order and convert into a list, the keys of this dictionary.

One of the main difficulties was time managing, as we had to balance this project with others course and tests. Furthermore, challenges may arise in a work in group as we could have forgot to commit and push, and the other part would not have the modifications and cannot continue to work on it. In the technical aspect, an issue with the encoding in utf-8 of the input functions as it did not consider the '-' character. Also, when a word was found in the corpus but shouldn't exist, needing a specific instruction to avoid errors with this word. We got some issues with the TF-IDF matrix but solved it with one single line. And finally, code and memory usage optimization were hard to tackle as we had to rewrite the whole codebase to use OOP paradigm which was better and easier to work on.

Conclusion:

As it is a work in pair, this project tested us in our ability to work fairly in group. To organize our project, we must have created a GIT depository to follow all the things we done. Thanks to this project, we learned how to do a chatbot. We enhanced our skills in Python language programming using tools like PyCharm and Visual Studio: code. We also gained benefits in managing things to stay organized thanks to the collaboration in pair.

Overall, this project exclusively done in Python allowed us to improve our technical, organization and even communication skills for time managing.