TI101I - PROGRAMMING IN PYTHON

CHATBOT PROJECT REPORT 2023/2024

TEAM :
KRYLOV RUDOLF & POUILLAUDE AGATHE
L1 INT-2 – PROMO 2028
TEAM CHATBOT

eFrei

PARIS PANTHÉON-ASSAS UNIVERSITÉ

02

ABOUT THE PROJECT

The purpose of text analysis and natural language processing is to develop intelligent systems like chatbots. This project delves into a method centered around the frequency of word occurrences, aiming to create a system capable of generating coherent responses based on the content of a given corpus of documents.

Whether you're an experienced programmer looking to refresh your definitions, or a beginner looking to understand what a chatbot is in the first place, we've got you covered. Let's dive in.

THE LOGIC BEHIND THE CHATBOT

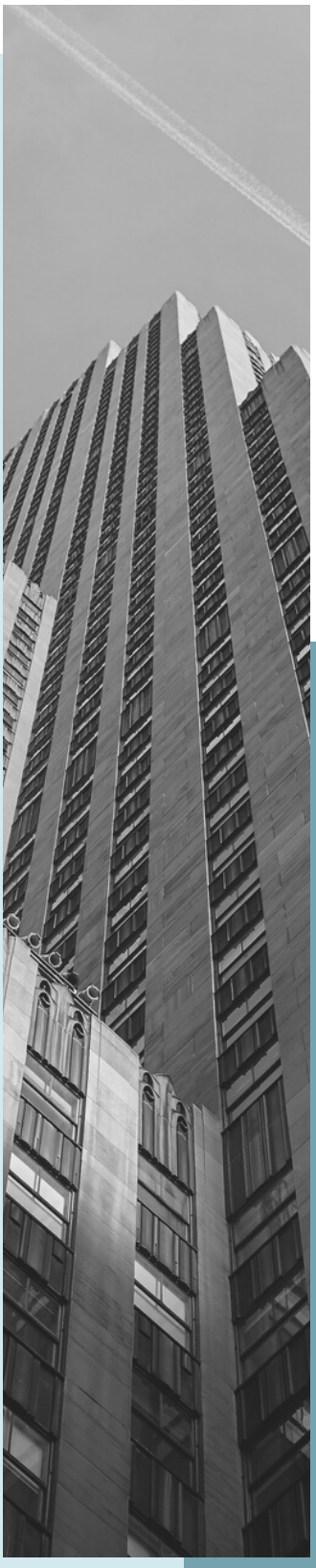
Foundational algorithm : encompassing key stages, each contributes to the system's ability to comprehend and respond effectively. Beginning with data pre-processing, the program undertakes tasks such as text cleanup, punctuation removal, and tokenization, preparing the corpus for subsequent analysis.

Pivotal step : involving the creation of a Term Frequency-Inverse Document Frequency (TF-IDF) matrix, this matrix encapsulates the essence of each unique word in the corpus, assigning vectors to words based on their relevance across documents. This forms the groundwork for understanding the significance of words in the overall context.

User's question : When faced with a user's question, the chatbot replicates the pre-processing steps applied to the corpus, generating a TF-IDF vector for the question. The subsequent similarity calculation, often utilizing cosine similarity, enables the identification of words in the corpus most closely aligned with the inquiry.

Strategical selection : The system then strategically selects the best answer by assessing the presence of similar words and their corresponding TF-IDF scores. This process ensures that the response is not only relevant but also reflective of the underlying context of the user's question.

03



By integrating these components, the text analysis system presented here provides a foundational understanding of how word frequency and TF-IDF matrices can be leveraged to construct intelligent responses. While acknowledging the simplicity of this approach, it establishes a crucial framework for comprehending the intricacies of question-answering systems, paving the way for more advanced methodologies in the evolving landscape of artificial intelligence.

03

List of Part II functions

- words_in_question(Returns a list of words in the question after cleaning.
- special_words_in_question -> set: Asks the user to enter a question, removes special words, and returns the list of words in the question without the special words.
- question_word_in_corpus -> dict:Takes the directory of the corpus as input, finds common words between the question and the corpus, and returns a dictionary of common words with their frequencies.
- tf_question(Calculates term frequency (TF) for the words in a given file, considering only words in the question.)
- calculate_tfidf_question -> list: Calculates TF-IDF values for words in the question across all documents in the corpus.
- display_tfidf_matrix -> None : Prints the TF-IDF matrix therefore the user can read it in a better way.

- scalar_product(tfidf_question_list_of_values,tfidf_corpus_list_of_values) -> list: Computes the scalar product of two lists of values.

- norm_vector -> float: Computes the norm of a vector.

- cosine_similarity(tfidf_question_list_of_values, tfidf_corpus_list_of_values) -> list: Computes cosine similarity between two lists of TF-IDF values.

- matrix_cosine_similarity(tfidf_question_matrix, tfidf_corpus_matrix) -> list: Computes cosine similarity between the question and the corpus then add the values in a matrix for each word of the corpus.

- calculate_relevance(matrix_cosine_similarity, doc_number) -> float: Calculates the relevance
  of a document.

04

LIST OF FUNCTION

- list_of_file (List all the file present in the initial folder)

- president_last_name (Return the presidents' last name from the title of the file)
- get_names (Associate the presidents' name to their last name)
- president_full_names (Display the name and last name)

- folder_cleaned (Create a "Cleaned" folder or delete the files in it)
- file_cleaned (re-write the text in each file)

- tf (Calculate the word frequency in the file)
- idf (Create a dictionary of the idf of each word)
- calculate_tfidf (Create a matrix were each word is associate to its tfidf in each text)

QUESTIONS

- least_imp_words (Research into the TFIDF matrix the least important words)
- most_imp_words (Display the most important words)
- most_repeated_word_by (Display the most repeated words of the president choosen
- mentioned_nation (Display the presidents that mention nation in their speeche)
- mentioned_climate (Display the first president that mentioned the climate)
- words_mentioned_by_all_presidents (Display the words that all the president mentioned)

05

PARIS PANTHÉON-ASSAS UNIVERSITÉ

PROJECT GOALS:

The Python programming project aims to introduce students to fundamental concepts in text processing while providing hands-on experience in developing a basic chatbot. The primary focus is on using word frequency to generate intelligent answers from a corpus of texts, employing the TF-IDF (Term Frequency-Inverse Document Frequency) method.

In the part one, we saw first Data Pre-processing, then TF-IDF Matrix generation and finally functionalities likes displaying the least important words in the corpus or the most repeated ones.

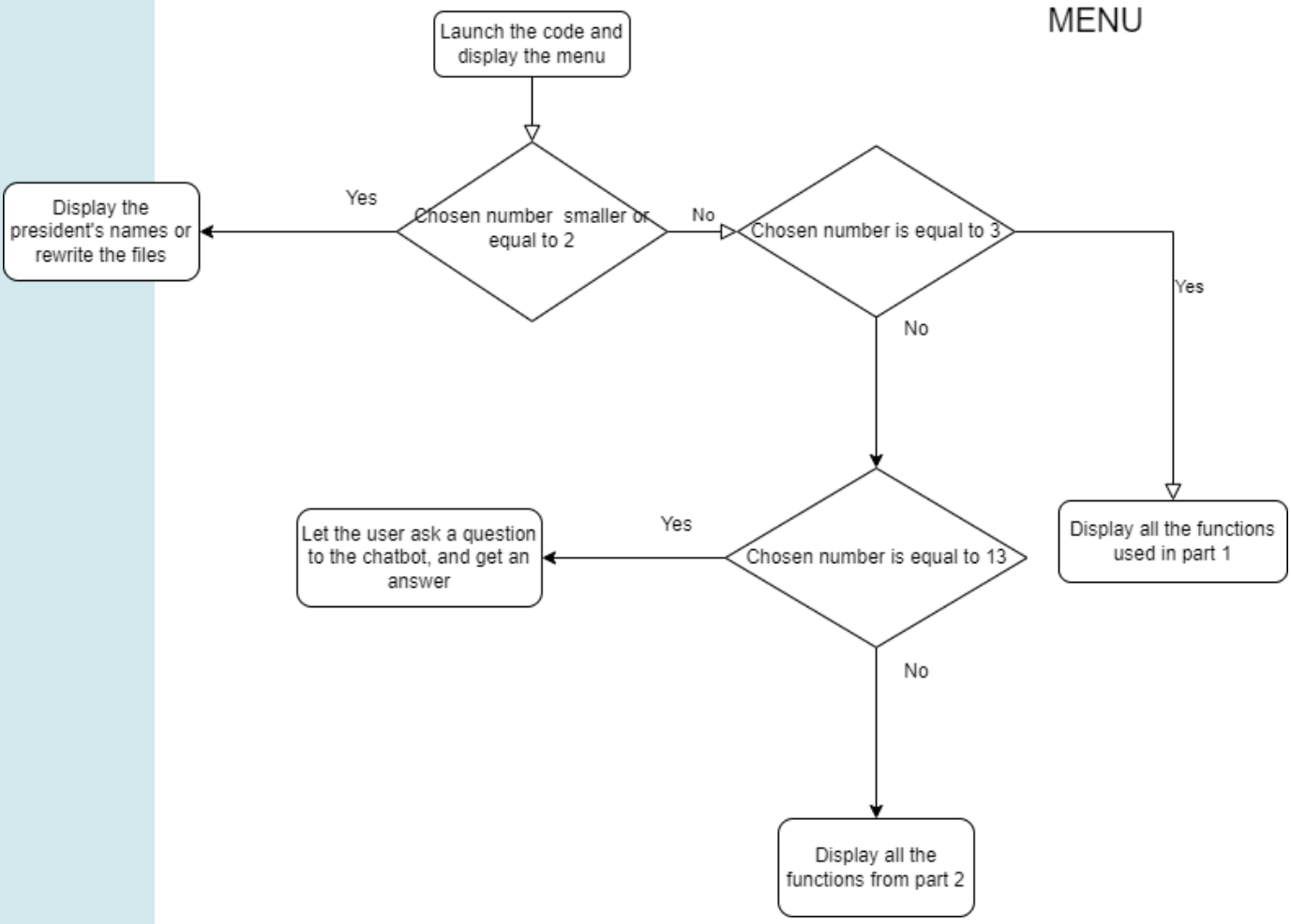
In the second part, we saw question analysis and response generation before a creating an interactive menu

For this project, we mainly used lists, matrixes and some dictionaries. This advantage of dictionary is that for a key we can have a value associated, which facilate the return and display and keep us more organize. But matrixes are more useful here because it is more practical to make as many lists as word in the document and to associate it to a list of values (based on the text), such as : [word, [txt1, txt2, ...]]

Our main problem was to understand how worked the calcul of the cosinus similarities, we thought that we need the coordinates (x, y) of the question and of the texts to calculate the norme and then the vector. We spend many time on the calculate_cosine_similarity to understand how it works.

LEARNING OUTCOMES

- We learned and upgraded through this project our python fundamentals' knowlege including 1D and 2D lists, functions, file handling and string manipulation.
- We also discovered text processing techniques like handling punctuation, lowercasing and tokenization.
- We learnt the TF-IDF method, the Git collaboration and how to develop a simplified chat bot.
- We also improved our problem-solving skills by encountering a lot of challenges, and our code organization and ergonomics.

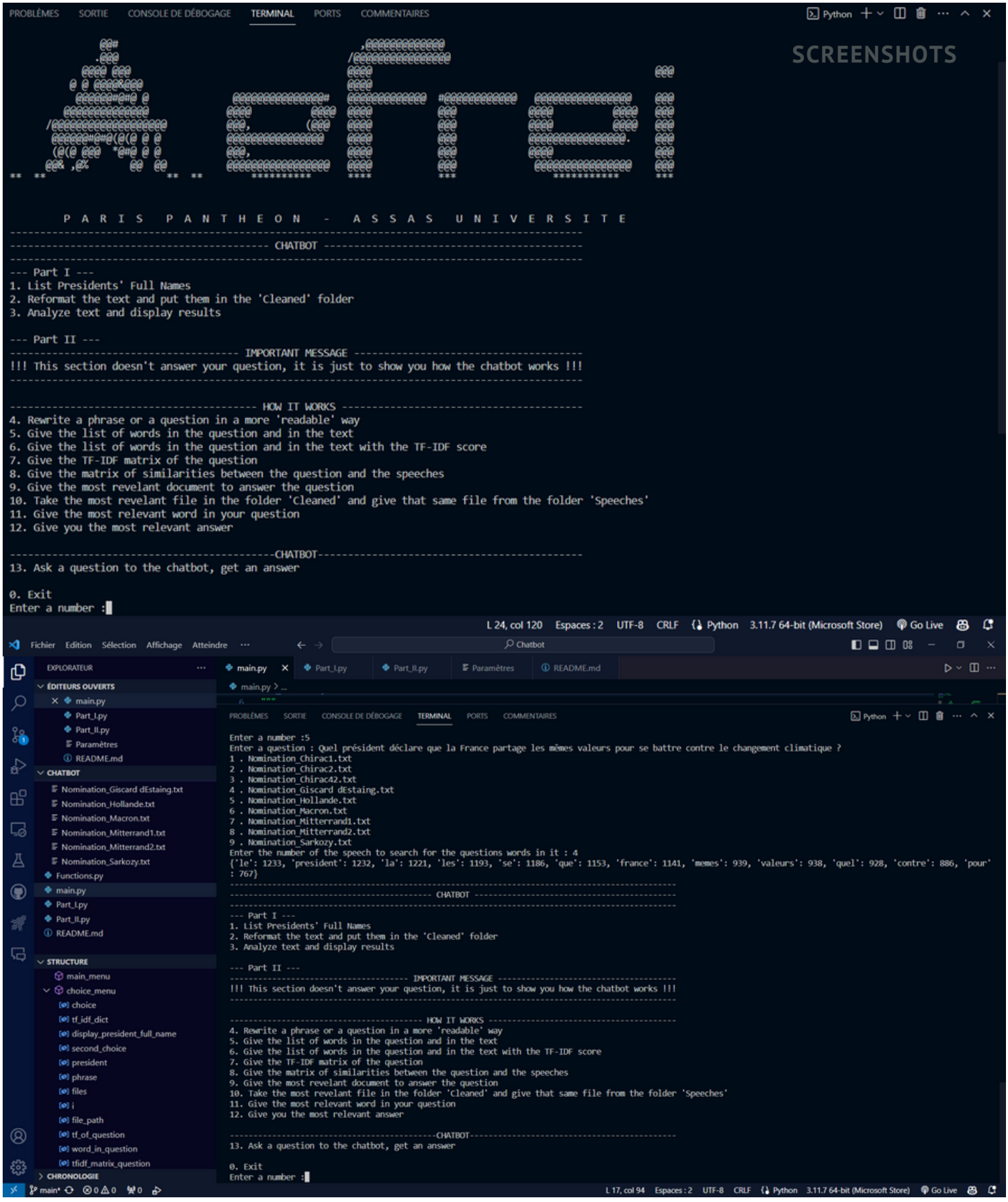


07

efrei
PARIS PANTHÉON-ASSAS UNIVERSITÉ

RESULTS :

SCREENSHOTS



08

efrei
PARIS PANTHÉON-ASSAS UNIVERSITÉ