ESILV

Webscrapping
First Project
Report

# Eco-responsible Travel

***Students :***
Joalie CORNELIE
Dounia BOUGAMZA

***Professor :***
Nedra MELOULLI

24 janvier 2025

# Table des matières

# 1 Introduction

## 1.1 Goal of the project

The project focuses on leveraging web scraping and NLP techniques to collect and analyze data from diverse sources. It emphasizes practical experience with handling complex websites and multilingual content while addressing real-world challenges. The collected data is processed and visualized to derive meaningful insights, combining technical, analytical, and developmental skills.

## 1.2 Problem definition

In a context where mass tourism is often criticized for its negative impacts on the environment and local communities, this project proposes an innovative travel guide designed to promote eco-tourism and sustainable tourism.

The goal is to raise awareness among travelers about the importance of exploring less-frequented destinations while respecting the environment and local cultures. By leveraging data collected through web scraping, the project offers an alternative and responsible vision of tourism, highlighting activities that align with these values.

We have selected a variety of destinations tailored to different traveler profiles, whether they are hiking enthusiasts or prefer seaside vacations. Additionally, our solution includes a curated selection of restaurants, enabling users to choose based on their tastes and preferences.

Our tool aims to recommend personalized locations and experiences that meet the desires of each user while promoting a more respectful and mindful approach to travel.

# 2 Methods used

Before initiating data collection and scraping, the first step was to review the terms and conditions as well as the required authorizations for each website we intended to use. This verification ensured that our approach complied with the rules set by the various platforms.

We followed these steps :


FIGURE 1 – Check Auhtorisation

## 2.1 Webscrapping

Web scraping is a technique for automatically extracting data from websites. It involves using scripts or tools to collect and structure information from web pages for analysis or integration into specific applications.

### 2.1.1 BeautifulSoup

We proceed first by using Beautiful Soup. Here are the following step that we did :

— We inspected the page by analyzing the website's HTML structure using the browser's inspect tool to identify the data we wanted to extract.


FIGURE 2 – Inspection

— Then we sent an HTTP Request to retrieve the webpage's HTML content.
— We located the desired data using BeautifulSoup's methods to search for specific HTML elements or tags.

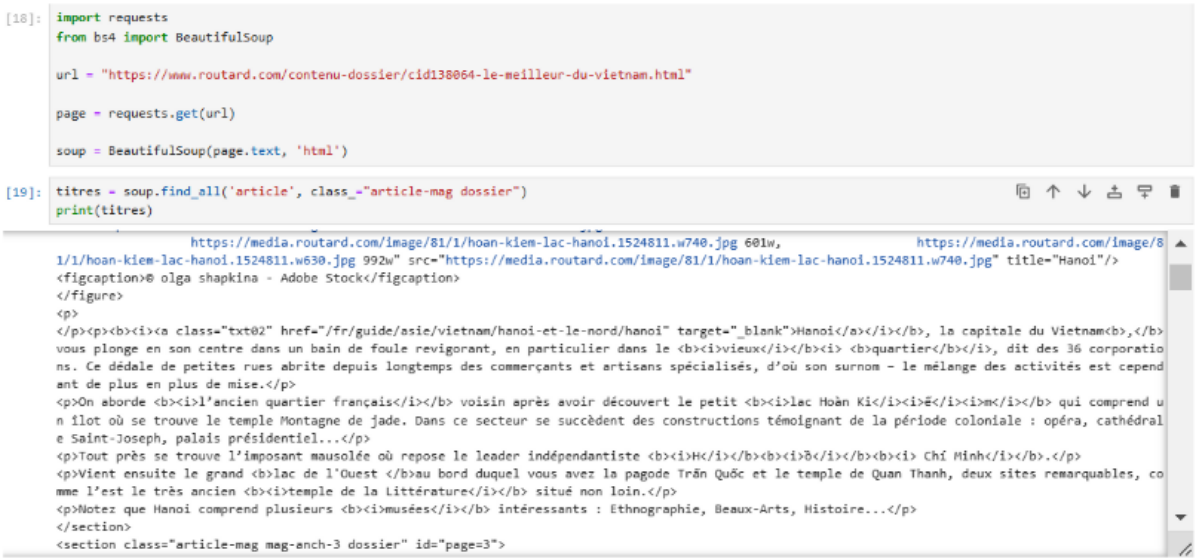


FIGURE 3 – Data collection

— The final step was to extract the relevant data (text, links, images), organize it for

further processing or storage and store it into a dataframe then a CSV file.

### 2.1.2 API

We used the Google Maps API to supplement the data collected by web scraping, in particular to obtain detailed information about restaurants corresponding to the selected destinations. Using the API allowed us to extract structured and up-to-date information. Here are the steps we followed :

**Obtaining the API Key**

Before using the Google Maps API, we created a project in the Google Cloud Console, activated the Google Maps Places API service, and generated an API key.

**Request Definition**

We configured the requests to search for restaurants near the identified destinations. Each request included :

— The geographical coordinates (latitude and longitude) of the destination.
— A search radius to limit the results.
— Relevant keywords, such as `"restaurant"`, to target dining establishments.

```python
def run_api(lat, lng):
    url = "https://places.googleapis.com/v1/places:searchNearby"
    api_key = "AIzaSyCBn-917H8fNTQ7grtFZ_nhfT0oiscCQ-8"

    payload = {
        "includedTypes": ["restaurant"],
        "maxResultCount": 20,
        "rankPreference": "DISTANCE",
        "locationRestriction": {
            "circle": {
                "center": {
                    "latitude": lat,
                    "longitude": lng,
                },
                "radius": 1000
            }
        }
    }
```

FIGURE 4

**Data Collection**

We used the `place_search` method of the Google Maps API to extract the following information :

— The name of the restaurant.
— The exact address.
— The average rating and number of reviews.
— The types of cuisine offered (if available).

```
headers = {
    "Content-Type": "application/json",
    "X-Goog-Api-Key": api_key,
    "X-Goog-FieldMask": (
        "places.displayName,places.formattedAddress,places.rating,"
        "places.nationalPhoneNumber,places.servesVegetarianFood,places.priceLevel,"
        "places.regularOpeningHours,places.userRatingCount"
    )
}

response = requests.post(url, json=payload, headers=headers)
if response.status_code == 200:
    return response.json()
else:
    print(f"API call failed for coordinates ({lat}, {lng}) with status code {response.status_code}")
    return None
```

FIGURE 5

**Data Processing and Storage**
The retrieved data was organized into a dataframe and then saved in CSV format for later use in the Streamlit application.

**Quota Management**
One of the challenges associated with using the API was managing usage quotas (number of requests per day). We optimized our queries by filtering the data to reduce the number of necessary API calls.

## 2.2 NLP

For this project, we chose to use the NLP methods studied in class to process our collected data.
The first step involved translating the retrieved data, as the websites we scraped were primarily in French. Consequently, all the descriptions and presentations of the destinations were written in this language. Since recommendation algorithms tend to perform better with English-language data, we found it appropriate to carry out a preliminary translation.
To achieve this, we selected a high-performance translation algorithm from Hugging Face, specifically designed for French-to-English translations.

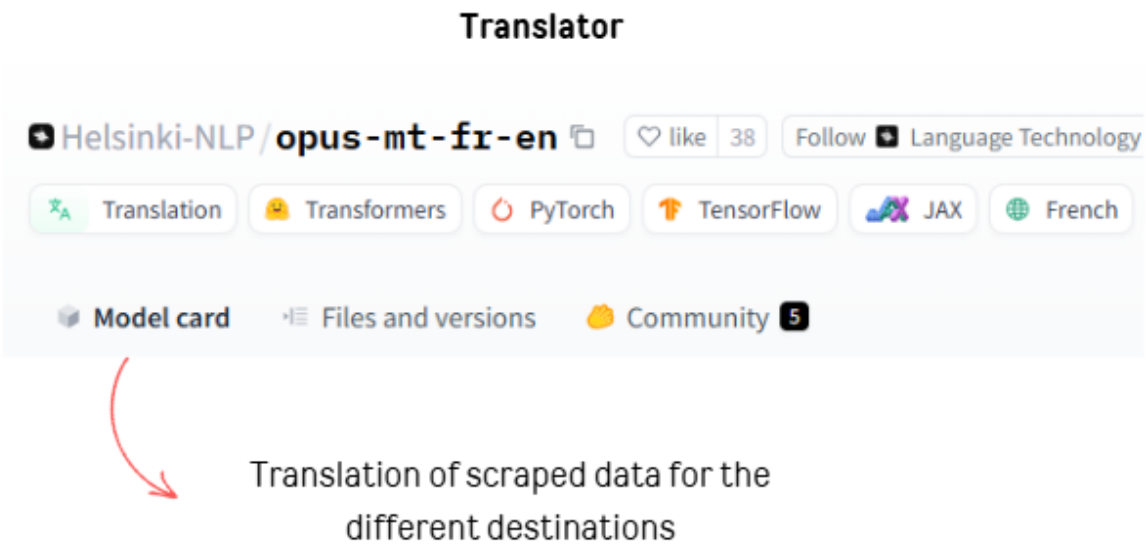


FIGURE 6 – Translation model

We then proceeded to implement our recommendation system. To achieve this, we preprocessed the description text using NLP techniques such as lemmatization and the removal of stop words.
We selected a recommendation model originally designed for recommending anime.
The recommendation process involves vectorizing the user's query and the location descriptions, then calculating a similarity score between the query and the locations. This score determines the relevance of each location to the user's preferences.
We tested several models before selecting this one, with the objective of maximizing the similarity score for more accurate recommendations.

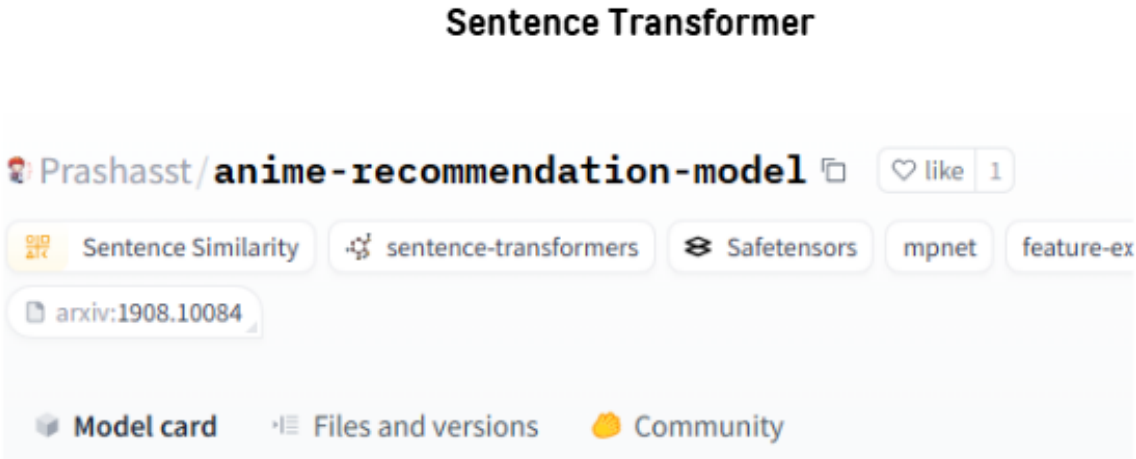


FIGURE 7 – Sentence Trasnformer

# 3 Problems and technical solutions

## 3.1 Problems

During the implementation of the Google Maps Places API, we faced several challenges :

- **Quota Limits :** The API has daily request limits, which constrained the number of queries we could make. This required careful planning and optimization of our requests.
- **Incomplete Data :** Some restaurants in the API results had missing or incomplete information, such as cuisine type or reviews, which impacted the completeness of our dataset.

| | Nom | Address | Note | Telephone | Serves_Vegetarian_Food | Price_Level | Horaires | User_Rating_Count |
|---|---|---|---|---|---|---|---|---|
| 0 | ceylon cafe | VQFC+6PF, Dambulla, Sri Lanka | NaN | 072 438 5915 | NaN | NaN | NaN | NaN |
| 1 | iresha hotel | VQFC+6PF, Dambulla, Sri Lanka | NaN | NaN | NaN | NaN | NaN | NaN |
| 2 | Chefrasul | VQFC+6PF, Dambulla, Sri Lanka | NaN | 075 657 4186 | NaN | NaN | NaN | NaN |
| 3 | The hot kichen | VQFC+6PF, Dambulla, Sri Lanka | NaN | NaN | NaN | NaN | NaN | NaN |
| 4 | traffic police Lanka | VQFC+6PF, Dambulla, Sri Lanka | NaN | NaN | NaN | NaN | NaN | NaN |

FIGURE 8

- **Geographical Coverage :** The API sometimes returned results that were outside the intended search radius, requiring post-processing to filter out irrelevant entries.
- **Rate Limits :** The API enforces rate limits for successive requests, which occasionally caused delays in data collection when making a large number of calls.
- **API Costs :** Although the API provides a free tier, exceeding the usage quota can incur additional costs, requiring close monitoring of our usage.
- **Data Formatting Issues :** The raw data from the API required significant preprocessing to standardize the format, especially for integration with our Streamlit application.

## 3.2 Technical solutions

To address the challenges encountered during the implementation of the Google Maps Places API, we implemented the following solutions :

- **Optimizing API Requests :** We reduced the number of API calls by strategically selecting the search radius and filtering results to focus only on relevant locations. This helped manage quota limits effectively.
- **Handling Missing Data :** For incomplete entries (e.g., missing cuisine types), we implemented fallback strategies such as labeling them as `"Unknown"`
- **Data Preprocessing :** We used Python libraries like `pandas` to clean and organize the raw data from the API into a standardized format compatible with our Streamlit application. This included normalizing text fields and handling null values.

# 4 Results

We presented our results in the form of a Streamlit application.

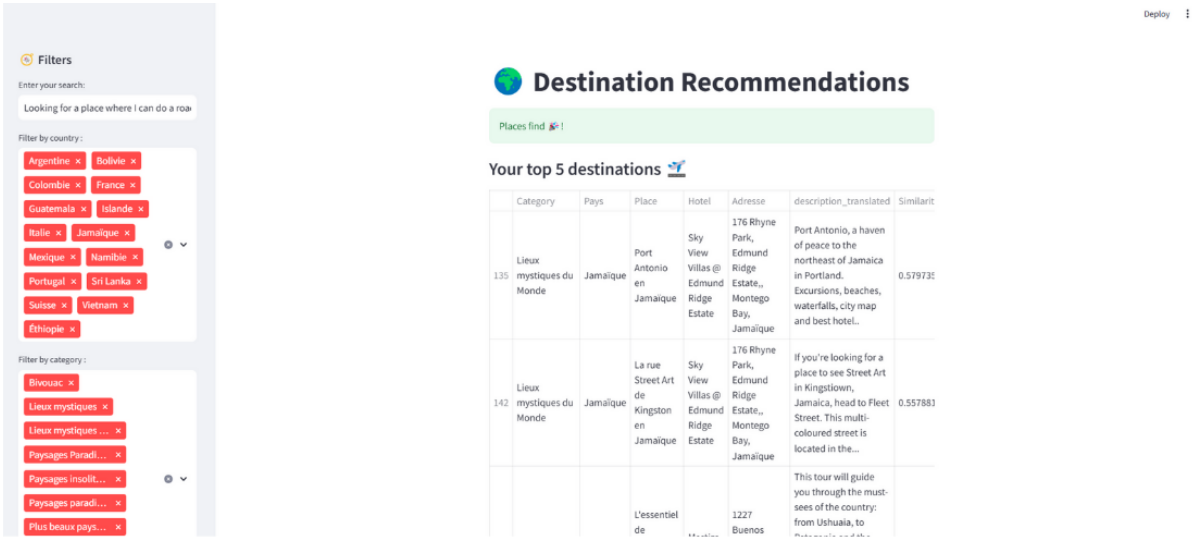


FIGURE 9 – Streamlit Application

# 5 Conclusion

This project allowed us to enhance our web scraping skills, which we had been introduced to during our Python course in the first year of our Master's program. We deepened this expertise by developing an application using our own data while adopting a multidisciplinary approach by integrating NLP techniques. This experience enabled us to combine theory and practice to tackle real-world challenges.

One of the next steps would be to collect more data and identify a more efficient model to provide destination recommendations that better align with users' specific expectations.