# Quandoo Data Engineering Task

## Problem statement

https://gitlab.com/quandoo-recruitment/data-engineer

## Assumptions

1. Fetch the restaurant data from the tripadvisor.com platform
2. We are fetching the restaurant data from the first three pages(This value can be changed in the scraper.py script)

Eg.
https://www.tripadvisor.com/RestaurantSearch?Action=PAGE&geo=187323&sortOrder=relevance&o=a
https://www.tripadvisor.com/RestaurantSearch?Action=PAGE&geo=187323&sortOrder=relevance&o=a30

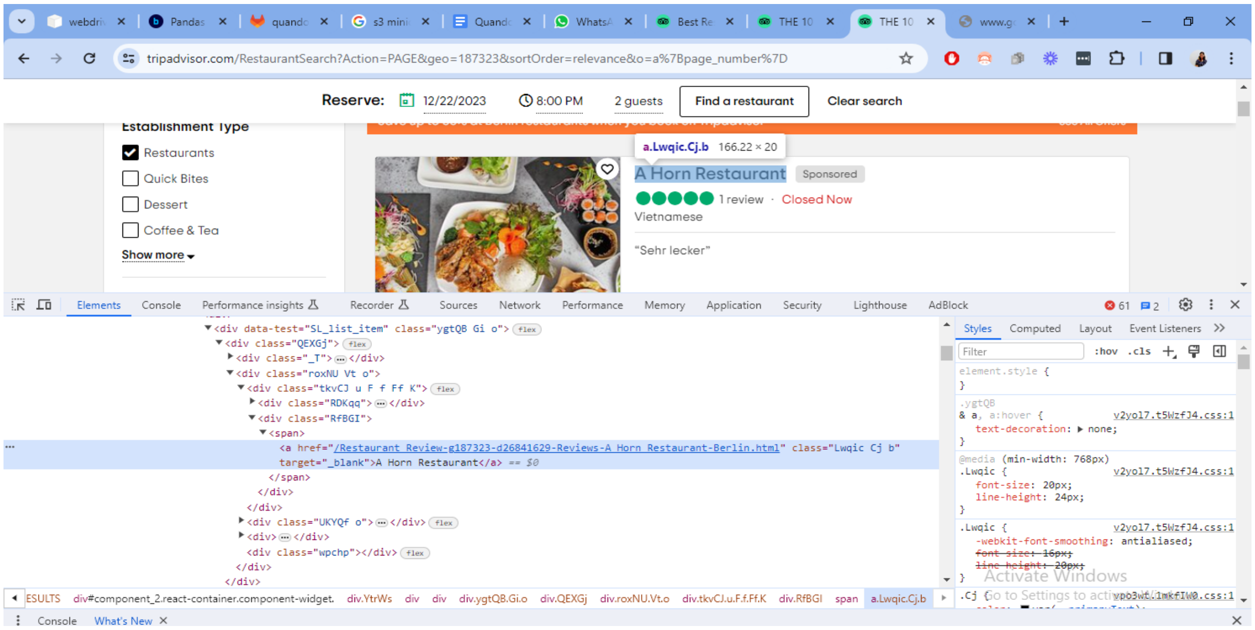
https://www.tripadvisor.com/RestaurantSearch?Action=PAGE&geo=187323&sortOrder=relevance&o=a60

# Understanding the problem statement

We need to fetch the restaurant data from the tripadvisor.com.
Link:

"https://www.tripadvisor.com/RestaurantSearch?Action=PAGE&geo=**%s**"+"&sortOrder=relevance&o=a**%s**"

Logic:
In the link, the first %s denotes geo_id, the second denotes the offset which is a multiple of 30 (value 0 means the first page, value 30 means the second page, value 60 means the third page, and so on.) and the third %s is the name of the place (eg. London_England).



tripadvisor.com/RestaurantSearch?Action=PAGE&geo=187323&sortOrder=relevance&o=a%7Bpage_number%7D

```
▼<div data-test="SL_list_item" class="ygtQB Gi o"> flex
  ▼<div class="QEXGj"> flex
    ▶<div class="_T">…</div>
    ▼<div class="roxNU Vt o">
      ▼<div class="tkvCJ u F f Ff K"> flex
        ▶<div class="RDKqq">…</div>
        ▼<div class="RfBGI">
          ▼<span>
             <a href="/Restaurant_Review-g187323-d26841629-Reviews-A_Horn_Restaurant-Berlin.html" class="Lwqic Cj b"
             target="_blank">A Horn Restaurant</a> == $0
           </span>
         </div>
       </div>
      ▶<div class="UKYQf o">…</div> flex
      ▶<div>…</div>
       <div class="wpchp"></div> flex
     </div>
   </div>
```

```
ESULTS  div#component_2.react-container.component-widget.  div.YtrWs  div  div  div.ygtQB.Gi.o  div.QEXGj  div.roxNU.Vt.o  div.tkvCJ.u.F.f.Ff.K  div.RfBGI  span  a.Lwqic.Cj.b
```

```
element.style {
}
.ygtQB
& a, a:hover {                    v2yol7.t5WzfJ4.css:1
   text-decoration: ▶ none;
}
@media (min-width: 768px)
.Lwqic {                          v2yol7.t5WzfJ4.css:1
   font-size: 20px;
   line-height: 24px;
}
.Lwqic {                          v2yol7.t5WzfJ4.css:1
   -webkit-font-smoothing: antialiased;
   ~~font-size: 16px;~~
   ~~line-height: 20px;~~
}
.Cj {                             vpo3wt.1mkfIW0.css:1
```

We can use the scraper.py program to fetch the data from the URL.
Here is the list of data attributes for the table restaurant_db.

Table 1: restaurant_db

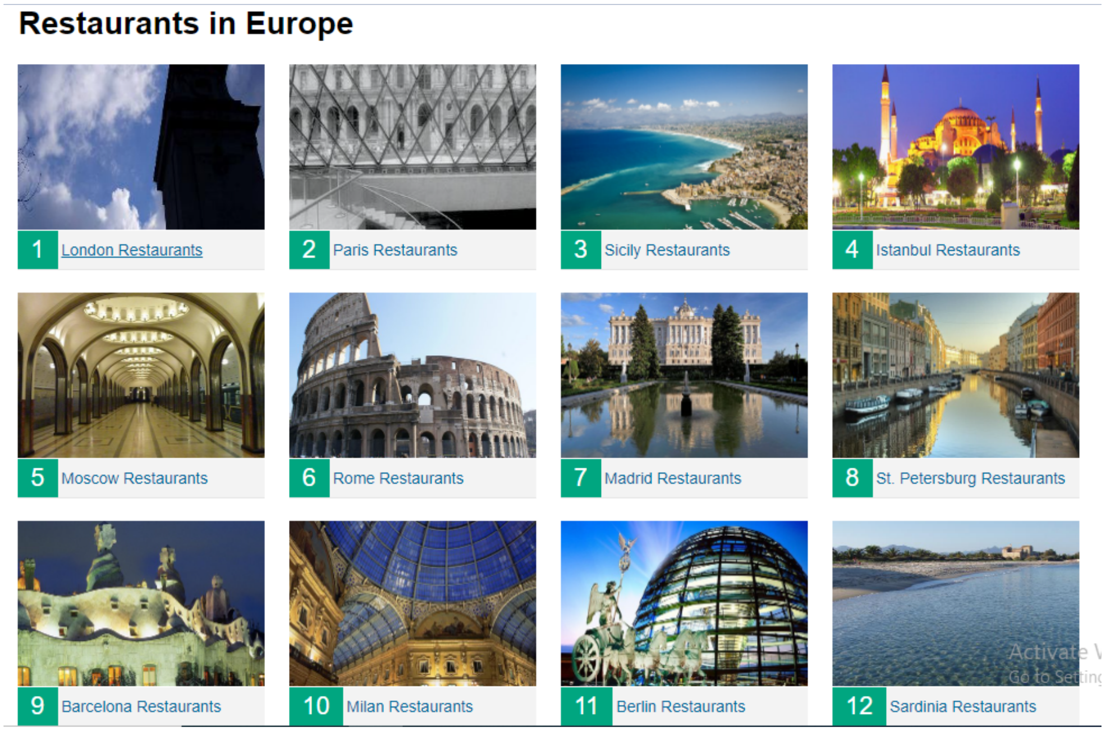
| Attribute | Data type |
|---|---|
| geo_id | varchar(20) |

| | |
|---|---|
| url | varchar(255) |
| restaurant_id | varchar(255) PRIMARY KEY |
| rest_name | varchar(255) |
| fetch_count | int |
| time_of_fetching_data | varchar(255) |
| rating | float |
| address | varchar(255) |
| telephone | varchar(255) |
| website | varchar(255) |
| tags | varchar(255) |
| CUISINES | varchar(255) |
| Special_Diets | varchar(255) |
| Meals | varchar(255) |
| is_michelin | varchar(3) |
| neighborhood | varchar(255) |
| restaurant_rank | varchar(255) |
| Food_Rating | float |
| Value_Rating | float |
| Service_Rating | float |
| Atmosphere_Rating | float |
| PRICE_RANGE | varchar(255) |
| FEATURES | text |
| total_reviews | int |
| menu_link | varchar(255) |
| menu_link_available | varchar(3) |

Table 2: geo_info

| place (varchar(255)) | geo_id(int) |
|---|---|
| Berlin | 187323 |
| London_England | 186338 |
| Paris | 187147 |

Remarks:
Populate the geo_info table with the data scraped using
https://www.tripadvisor.com/Restaurants-g4-Europe.html
This is out of the scope as of now.



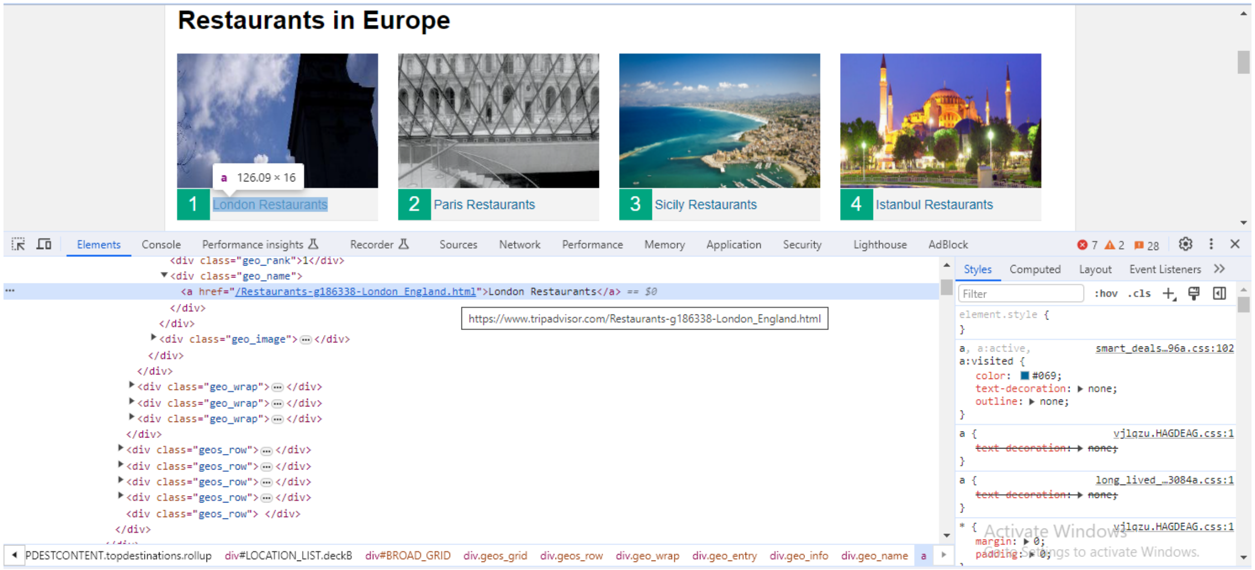
Restaurants in Europe

1 London Restaurants
2 Paris Restaurants
3 Sicily Restaurants
4 Istanbul Restaurants
5 Moscow Restaurants
6 Rome Restaurants
7 Madrid Restaurants
8 St. Petersburg Restaurants
9 Barcelona Restaurants
10 Milan Restaurants
11 Berlin Restaurants
12 Sardinia Restaurants

If we inspect the elements and analyze the pattern, we find that we can get the geo_id by fetching all the div elements with class geo_name and then looping over the div elements to get href attribute of the <a> tag.

We get the link to the restaurant list as well as the geo_id.
Eg. <a href="/Restaurants-g186338-London_England.html">London Restaurants</a>

Link = https://www.tripadvisor.com/Restaurants-g186338-London_England.html
geo_id = 186338
place = London_England



```
            <div class="geo_rank">1</div>
          ▼<div class="geo_name">
              <a href="/Restaurants-g186338-London_England.html">London Restaurants</a> == $0
            </div>
          </div>
        ▶<div class="geo_image"> … </div>
        </div>
      </div>
    ▶<div class="geo_wrap"> … </div>
    ▶<div class="geo_wrap"> … </div>
    ▶<div class="geo_wrap"> … </div>
    </div>
  ▶<div class="geos_row"> … </div>
  ▶<div class="geos_row"> … </div>
  ▶<div class="geos_row"> … </div>
  ▶<div class="geos_row"> … </div>
   <div class="geos_row"> </div>
  </div>
```

https://www.tripadvisor.com/Restaurants-g186338-London_England.html

PDESTCONTENT.topdestinations.rollup div#LOCATION_LIST.deckB div#BROAD_GRID div.geos_grid div.geos_row div.geo_wrap div.geo_entry div.geo_info div.geo_name a

```
element.style {
}
a, a:active,                 smart_deals…96a.css:102
a:visited {
   color: #069;
   text-decoration: ▶ none;
   outline: ▶ none;
}
a {                            vjlqzu.HAGDEAG.css:1
   ~~text-decoration: ▶ none;~~
}
a {                          long_lived_…3084a.css:1
   ~~text-decoration: ▶ none;~~
}
* {                            vjlqzu.HAGDEAG.css:1
   margin: ▶ 0;
   padding: ▶ 0;
```

# Python libraries used

| Python Library | Purpose |
|---|---|
| selenium | To visit the web address and fetch the data |
| pandas | To load all the data into the data frame and export it as CSV file |
| pymysql | To connect to the database and insert the data |
| fast-api | To write the API endpoints |

# Project files

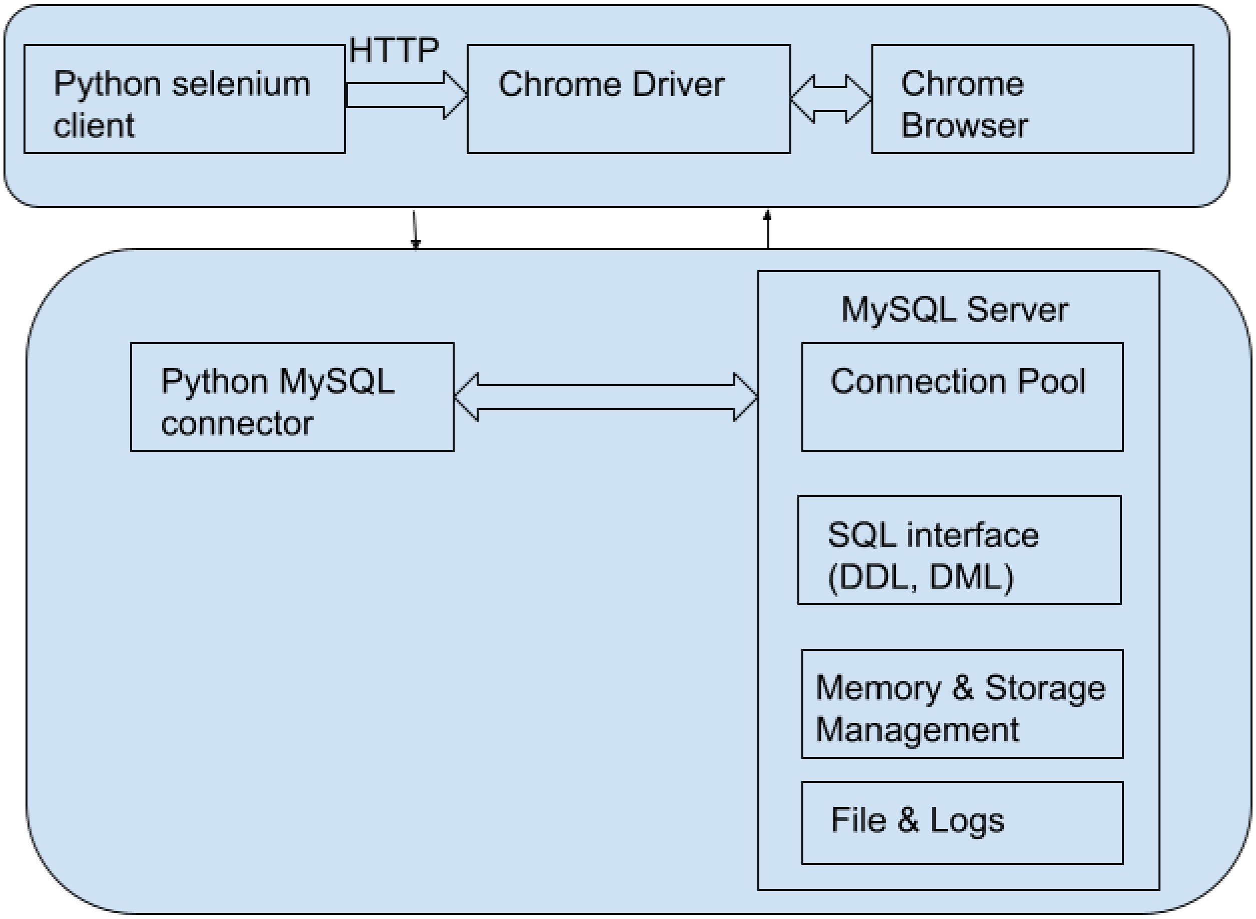
| Sr. | File name | Purpose |
|---|---|---|
| 1. | requirements.txt | This file contains the Python libraries list required for the project. |
| 2. | constants.py | This file contains constant values. |
| 3. | app.py | This file runs the API. |
| 4. | scraper.py | This file scrapes the restaurant's data. |
| 5. | data_storage.py | This file exports scraped data to the database as well as creates the CSV file of the scraped data. |

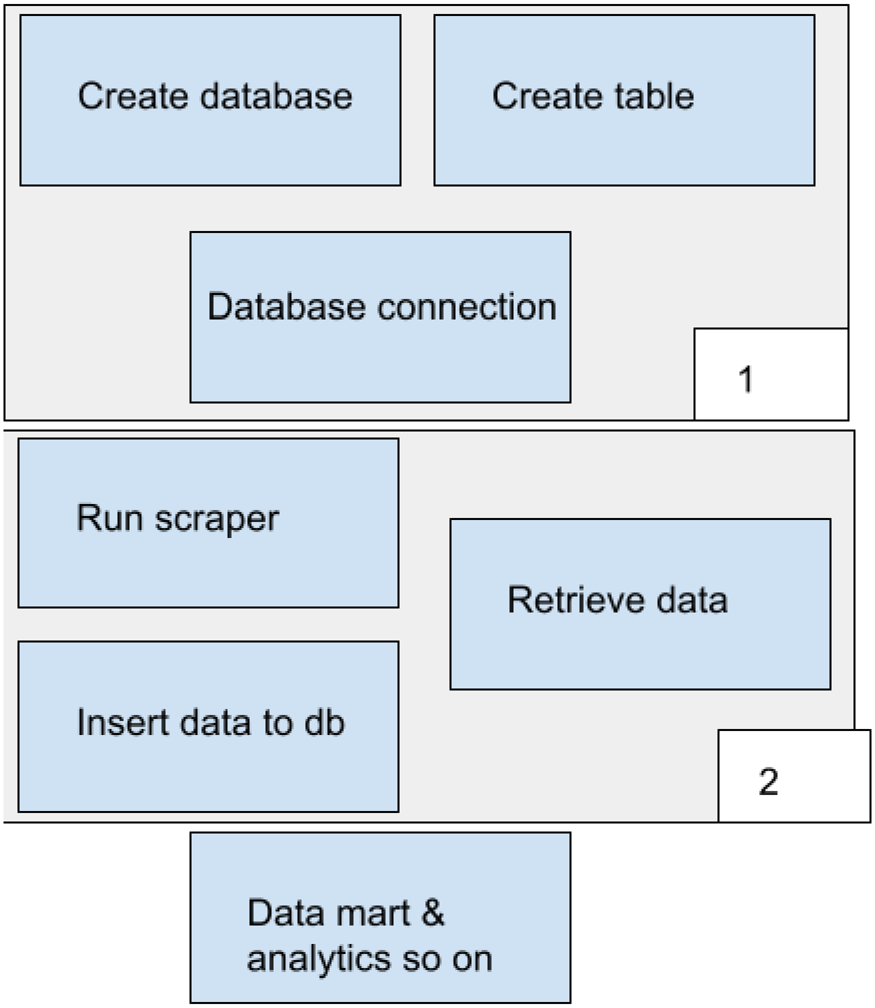
| | | |
|---|---|---|
| 6. | test.py | This file contains the unit test cases. |
| 7. | Dockerfile | This file contains all the commands to assemble the Python project image. |
| 8. | docker-compose.yml | This YAML file defines and runs multi-container Docker applications, such as the Python app and mysql. |
| 9. | datamart.py | Do manipulation of existing data |
| 10. | load_csv_to_db.py | Load CSV file to database |
| 11. | Dockerfile_sql | Docker container mysql |

# API Endpoints

| Sr | Endpoint | Purpose |
|---|---|---|
| 1. | /scrape_data/{geo_id} | To fetch the restaurant data for particular geo_id from web |
| 2. | /retreive_data/{geo_id} | To fetch the data of particular geo_id from database |

# Architecture diagram





# How scraped data looks like

{'url': 'https://www.tripadvisor.com/Restaurant_Review-g187147-d19261302-Reviews-Miura-Paris_Ile_de_France.html', 'restaurant_id': 'd19261302', 'geo_id': '187147', 'fetch_count': 1, 'time_of_fetching_data': '23/12/2023 16:28:30', 'total_reviews': 159, 'rating': 5.0, 'rest_name': 'Miura', 'address': "15, rue de l'Arc de Triomphe, 75017 Paris France", 'telephone': '+33 1 47 54 00 28', 'website': None, 'tags': "['$$$$', 'French', 'European', 'Contemporary']", 'CUISINES': "['French', 'European', 'Healthy', 'Contemporary']", 'Special_Diets': '', 'Meals': "['Lunch', 'Dinner', 'Drinks']", 'is_michelin': 'No', 'menu_link_available': 'No', 'menu_link': '', 'neighborhood': '17th Arr. - Batignolles-Monceau0.2 miles from Arc de Triomphe', 'restaurant_rank': '#23 of 14,523 Restaurants in Paris', 'Food_Rating': 0.0, 'Service_Rating': 0.0, 'Value_Rating': 0.0, 'Atmosphere_Rating': 0.0, 'PRICE_RANGE': "['$65 - $94']", 'FEATURES': "['Reservations', 'Seating', 'Serves Alcohol', 'Full Bar', 'Accepts Credit Cards', 'Table Service', 'Private Dining', 'Street Parking', 'Wine and Beer', 'Dog Friendly', 'Non-smoking restaurants', 'Gift Cards Available']"}

# How to reproduce the project

### (A) Using docker-compose.yml

Step 1: Download the project zip file

Step 2: Install the docker

Step 3: Run the docker-compose.yml using the following command

```
docker-compose up
```

If running for the first time, use the following command

```
docker-compose up –build
```

Step 4: Access the endpoints: /scrape_data/{geo_id} and /retreive_data/{geo_id}

### (B) Without using docker

Step 1: Download and extract the project zip file

Step 2: Install Python and Mysql workbench

Step 3: On the command prompt, go to the project folder.

Step 4: Create virtual environment

```
python -m venv venv
```

Step 5: Activate the virtual environment

```
(On windows) venv\Scripts\activate
```

```
pip install -r requirements.txt
```

Step 6: Connect to MySQL database

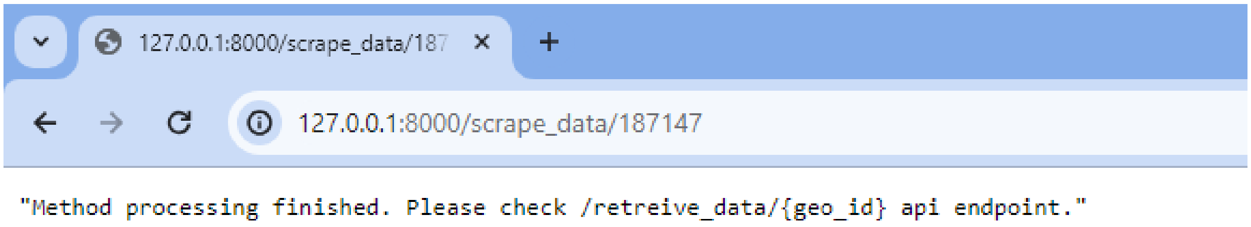
Step 7: Run load_csv_to_db.py to load the data to database

```
python load_csv_to_db.py
```

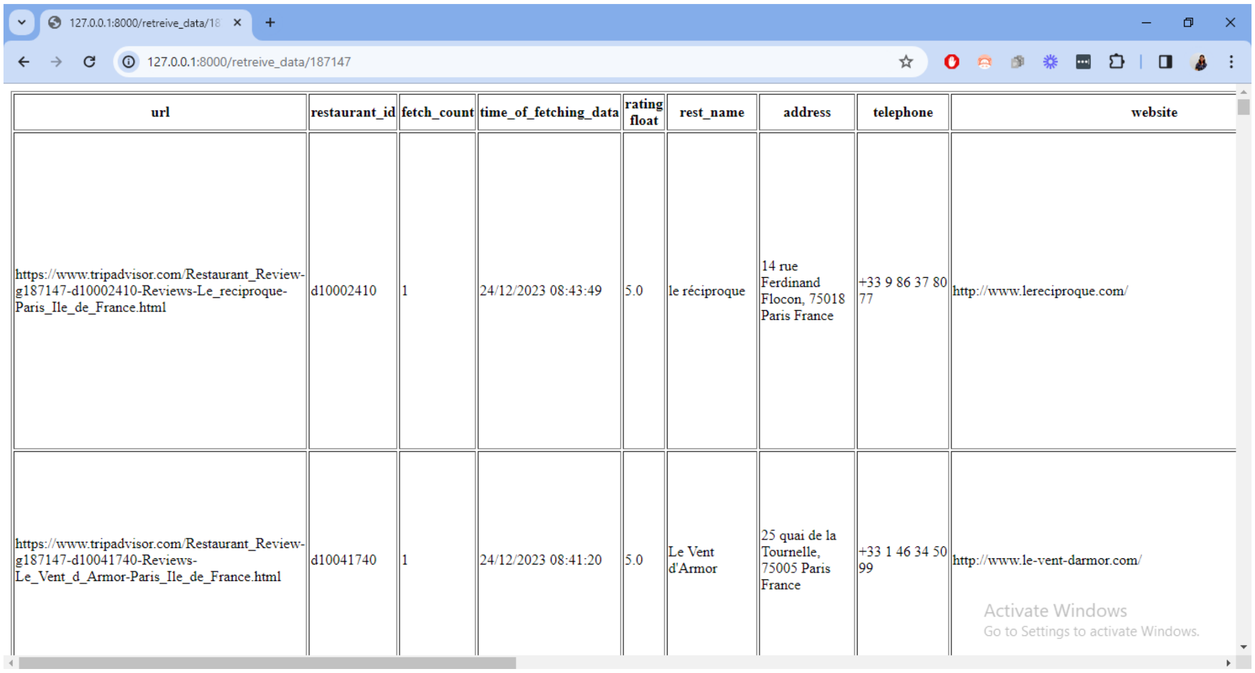
Step 6: Run app.py

```
uvicorn app:app
```

# Screenshots

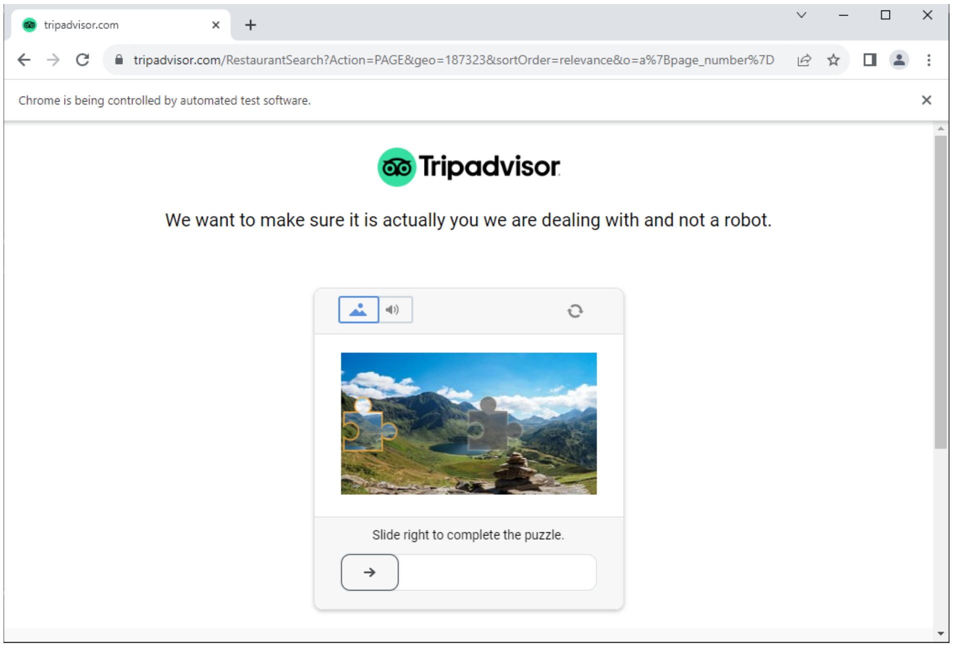

127.0.0.1:8000/scrape_data/187
127.0.0.1:8000/scrape_data/187147

```
"Method processing finished. Please check /retreive_data/{geo_id} api endpoint."
```


127.0.0.1:8000/retreive_data/18
127.0.0.1:8000/retreive_data/187147
Activate Windows
Go to Settings to activate Windows.

| url | restaurant_id | fetch_count | time_of_fetching_data | rating float | rest_name | address | telephone | website |
|---|---|---|---|---|---|---|---|---|
| https://www.tripadvisor.com/Restaurant_Review-g187147-d10002410-Reviews-Le_reciproque-Paris_Ile_de_France.html | d10002410 | 1 | 24/12/2023 08:43:49 | 5.0 | le réciproque | 14 rue Ferdinand Flocon, 75018 Paris France | +33 9 86 37 80 77 | http://www.lereciproque.com/ |
| https://www.tripadvisor.com/Restaurant_Review-g187147-d10041740-Reviews-Le_Vent_d_Armor-Paris_Ile_de_France.html | d10041740 | 1 | 24/12/2023 08:41:20 | 5.0 | Le Vent d'Armor | 25 quai de la Tournelle, 75005 Paris France | +33 1 46 34 50 99 | http://www.le-vent-darmor.com/ |

# Challenges



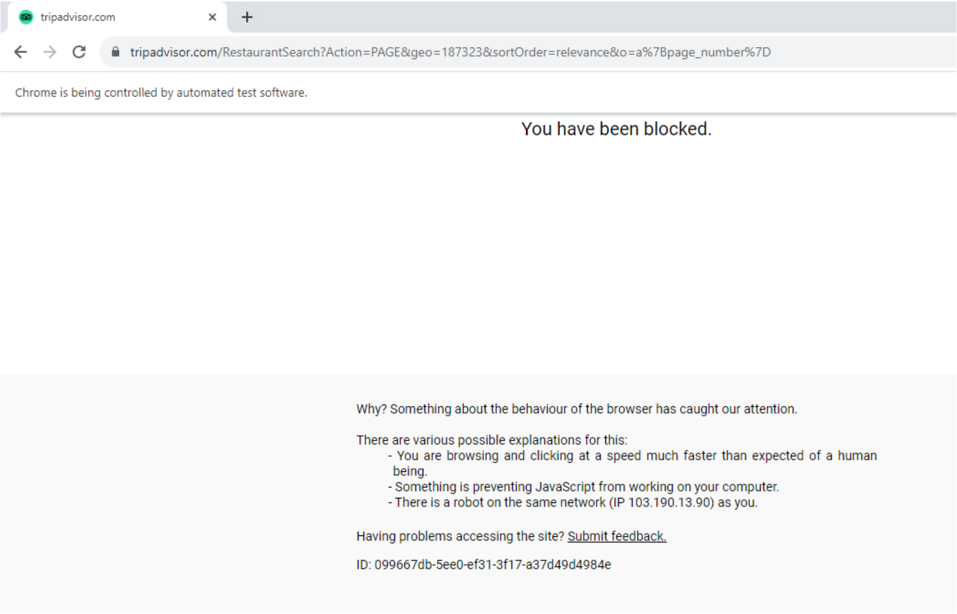
tripadvisor.com

tripadvisor.com/RestaurantSearch?Action=PAGE&geo=187323&sortOrder=relevance&o=a%7Bpage_number%7D

Chrome is being controlled by automated test software.

We want to make sure it is actually you we are dealing with and not a robot.

Slide right to complete the puzzle.



tripadvisor.com

tripadvisor.com/RestaurantSearch?Action=PAGE&geo=187323&sortOrder=relevance&o=a%7Bpage_number%7D

Chrome is being controlled by automated test software.

You have been blocked.

Why? Something about the behaviour of the browser has caught our attention.

There are various possible explanations for this:

- You are browsing and clicking at a speed much faster than expected of a human being.
- Something is preventing JavaScript from working on your computer.
- There is a robot on the same network (IP 103.190.13.90) as you.

Having problems accessing the site? Submit feedback.

ID: 099667db-5ee0-ef31-3f17-a37d49d4984e

# Discussion

1. Why the list data is stored as varchar data types?

Some attributes like CUISINES are stored as varchar data type in the database.

To store a Python list in MySQL, you generally need to convert the list into a format that can be stored in a MySQL column, as MySQL itself doesn't have a native data type for lists.

2. Data mart Implementation idea

Storing the results of datamart.py script into NoSQL database can be a good solution. Here's the explanation.

A NoSQL data mart is a data storage and processing system that utilizes a NoSQL (Not Only SQL) database to store and manage data for analytical purposes. Unlike traditional relational databases, NoSQL databases provide a more flexible schema design, scalability, and the ability to handle large volumes of unstructured or semi-structured data. NoSQL databases often used for data marts include MongoDB, Cassandra, Couchbase, and others.

Here are some key considerations and components when implementing a NoSQL data mart:

**Data Modeling:**

NoSQL databases often use schema-less or dynamic schema approaches, allowing for flexible data modeling.

Document-oriented databases (e.g., MongoDB) store data in JSON-like documents, while key-value stores (e.g., Cassandra) use key-value pairs.

**Scalability:**

NoSQL databases are designed to scale horizontally, allowing for distributed and scalable architectures.

Data marts can handle large volumes of data and high concurrent queries by adding more nodes to the NoSQL cluster.

**Query Language:**

NoSQL databases have their query languages. For example, MongoDB uses a query language based on JSON-like documents.

Queries may be optimized for specific use cases, and aggregations are often performed using the database's native features.

**Data Ingestion:**

Implement mechanisms for efficient data ingestion from various sources into the NoSQL data mart.

Tools like Apache Kafka or custom ETL (Extract, Transform, Load) processes can be used for data integration.

**Indexing:**

NoSQL databases use indexing to optimize query performance.

3. Schedule the sourcer to run (daily/hourly/monthly)

Using AWS service, we can call api scrape_data_endpoint from lambda function and schedule the lambda function (daily/hourly/monthly)