This is the explanation for the “scripts/ETL.py” and “notebooks/log_analysis.ipynb” files that I made for the task company gave me.

- Read the data:
  First, I read the text file with python open method then read the text file lines of them strip them, clean them using my function(“parse_logfile”) and put them into a list, then create a pandas data frame out of the list.

  - Parse and clean data:

    I created a function called “parse_logfile” which takes a single log entry string and process them to get us a structured data in python dictionary format. The output of this function includes IP Address, Timestamp, Request Method, URL, Status code, Response Size and Query Parameters.

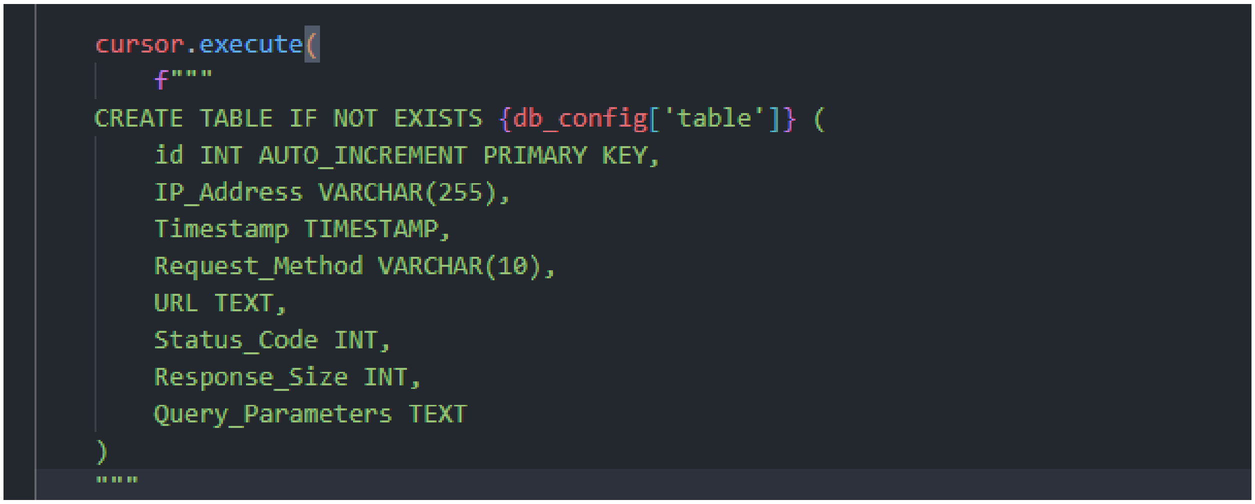
    I also use a handling error system to have access to the logs that have issues during the process. If there are any issues the script will create a log file named “parsing_errors.log”.

    Then I implement this function on every logs of the text file and put the output on the another list, so we have a list of python dict and turn the list into a pandas data frame.

- Connecting to MySQL:

  - Configuration:
    I created the “config.ini” file to be able to modify the MySQL database parameters through it.

  I read the parameters that needed to connect to the database through “config.ini” file and connect to MySQL through its DBAPI(I use mysql-connector-python). Create a cursor and execute some raw sql queries to see if database or table exists or not , if they doesn’t exist create them.



```
cursor.execute(
    f"""
CREATE TABLE IF NOT EXISTS {db_config['table']} (
    id INT AUTO_INCREMENT PRIMARY KEY,
    IP_Address VARCHAR(255),
    Timestamp TIMESTAMP,
    Request_Method VARCHAR(10),
    URL TEXT,
    Status_Code INT,
    Response_Size INT,
    Query_Parameters TEXT
)
"""
```

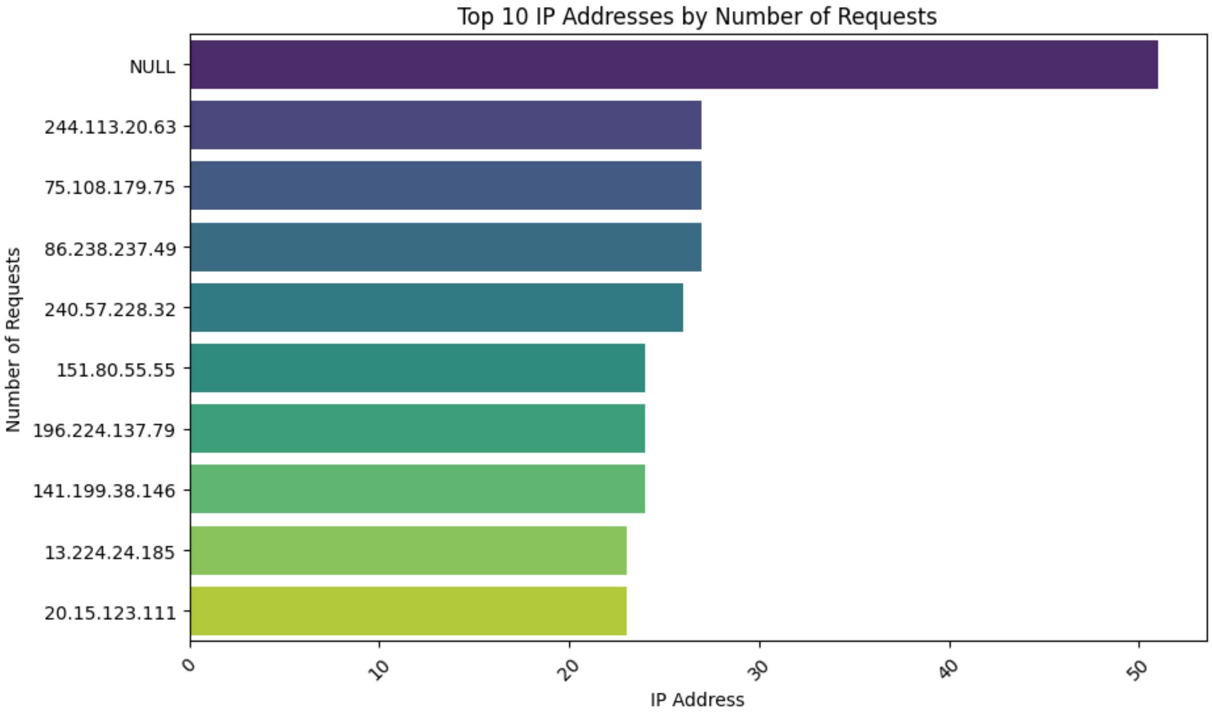
I implement a error handling here too, so we can check if there are any errors occurred during inserting these logs into MySQL database in log file named “insertion_error.log”(these log file created if there is at least one error).

- Logs:
  I use logging module of python to have access to the events occurred during executing of the script. The output will save in log file named “ETL_process.log”.

- Visualizing data:

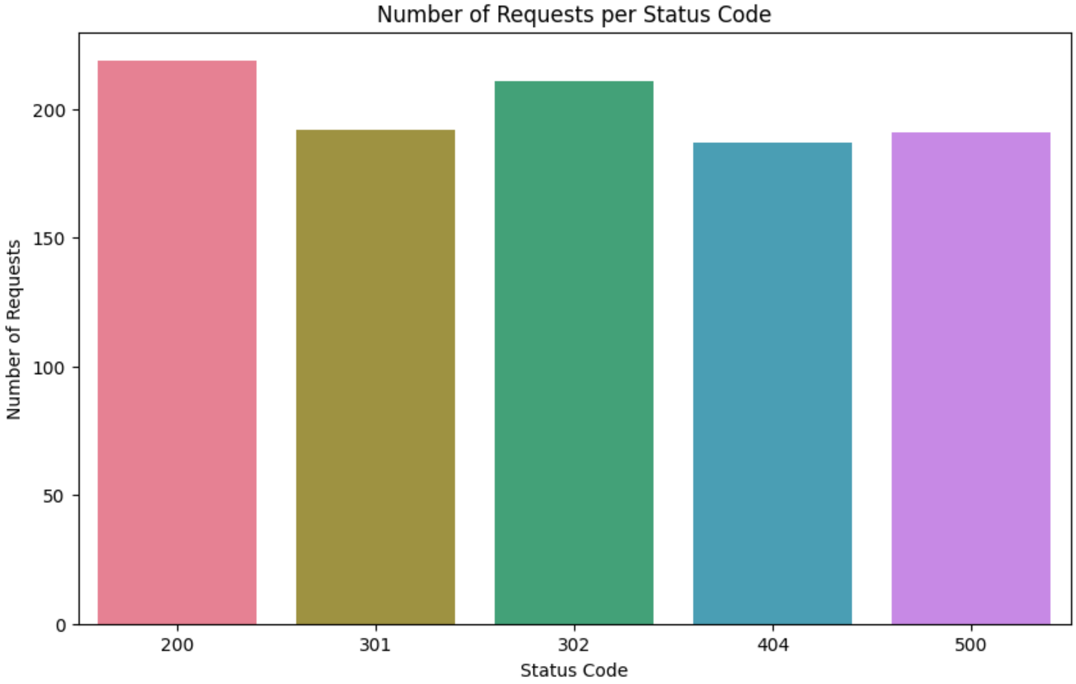
  Now that the data is more structured and cleaner than it was, it's time to analyze and visualize it. Before jumping to the visualization part, I should mention that I saved the data frame as a CSV file and reviewed it in Excel. I filtered it to see the distinct values of the fields. Most of them looked fine at first glance, but it seemed odd to have null values in the IP addresses. I just wanted to mention this.

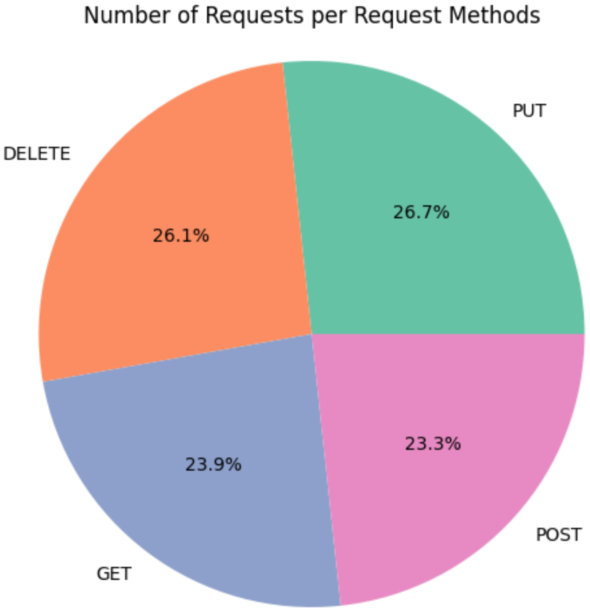
  I created a notebook file named “log_analysis.ipynb” to visualize our database using common Python tools such as Matplotlib and Seaborn. My main idea is to see how the number of requests is distributed among different categories, So I connected to the database once again and put them in a data frame using pandas read_sql function. Then I visualized the distribution of the number of requests and IP addresses.

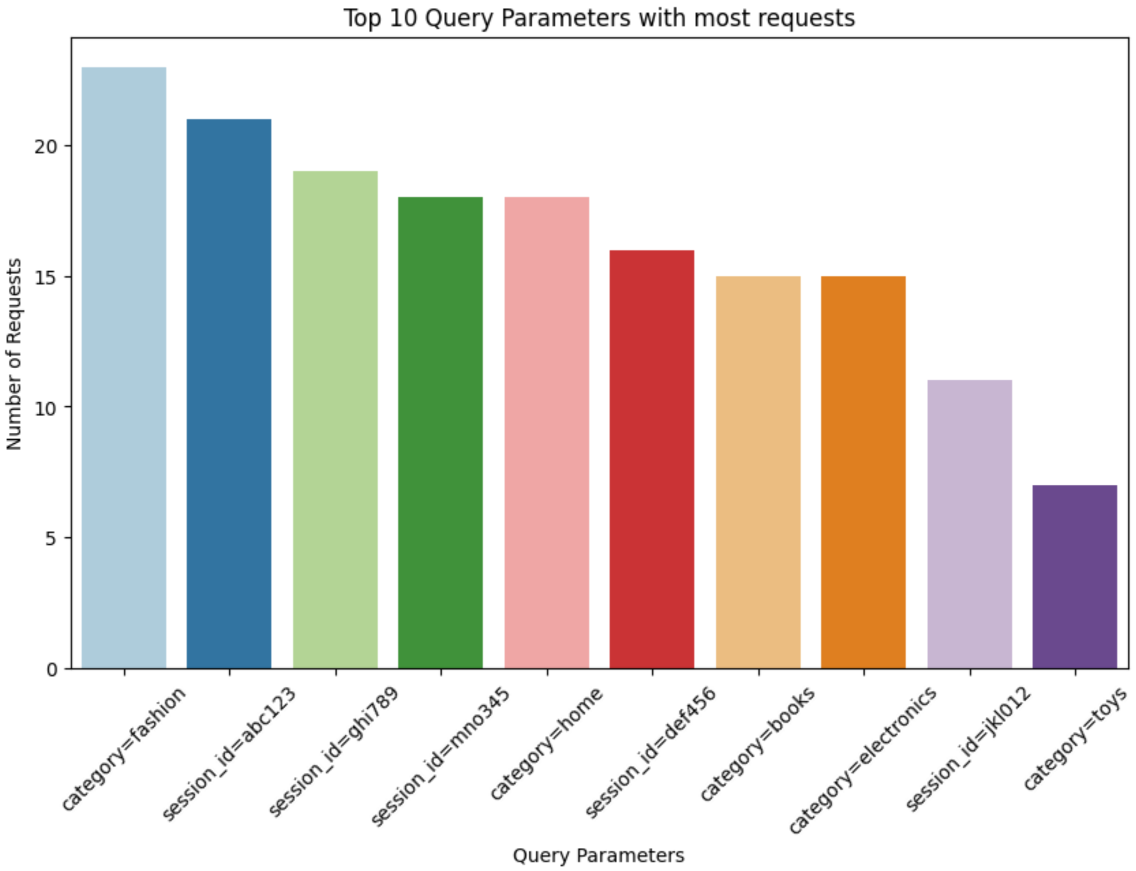


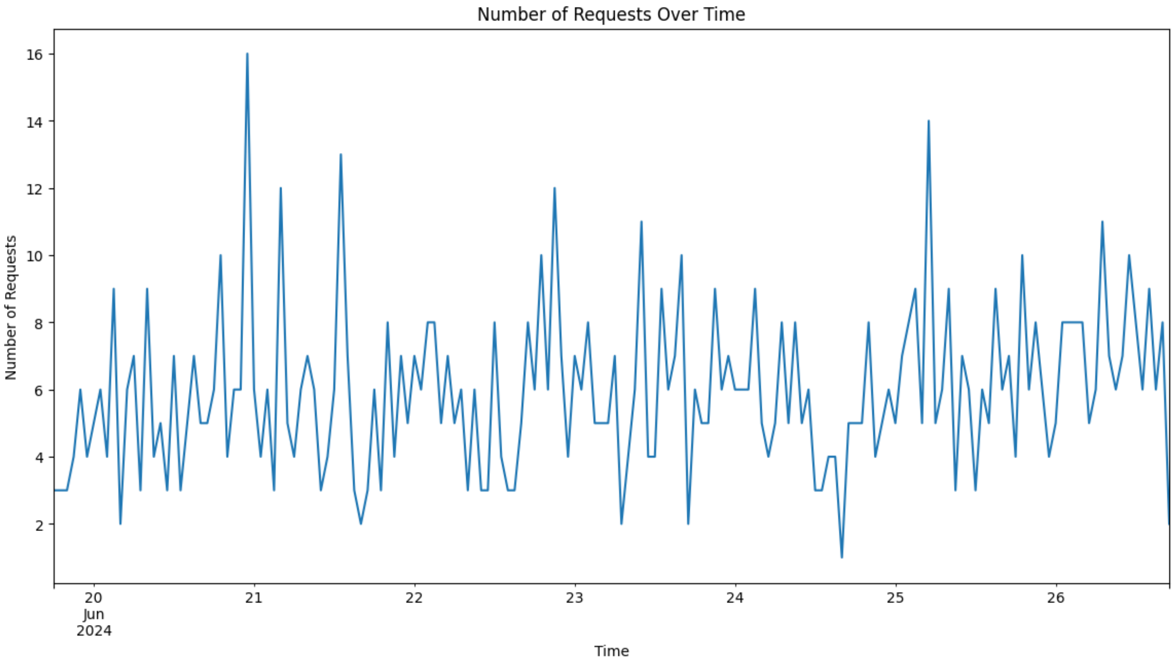
These are top 10 IP addresses that have the most requests.

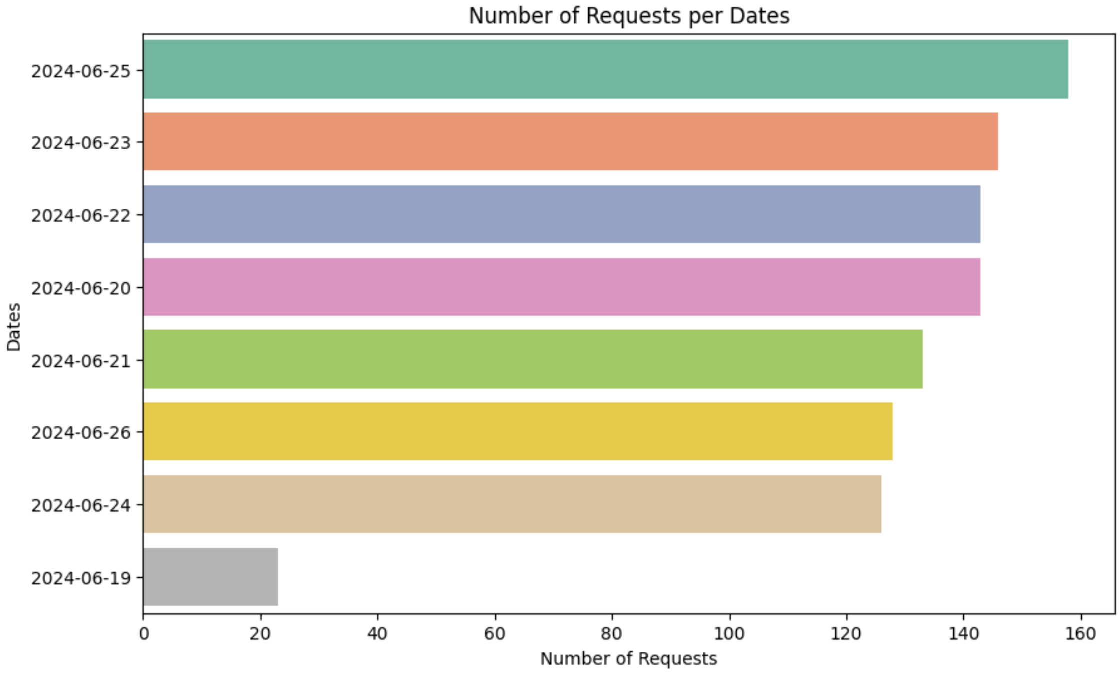
Based on the other fields, I provided descriptive statistics and visualizations, which you can find in the “log_analysis.ipynb” file. Here are the charts for a clearer understanding of the data:






Top 10 Query Parameters with most requests
Number of Requests
0
5
10
15
20
category=fashion
session_id=abc123
session_id=ghi789
session_id=mno345
category=home
session_id=def456
category=books
category=electronics
session_id=jkl012
category=toys
Query Parameters


Number of Requests Over Time
Number of Requests
2
4
6
8
10
12
14
16
20
Jun
2024
21
22
23
24
25
26
Time



These graphs provide a general overview. For a more detailed analysis and specific tasks, We may need to focus on particular fields and other parameters.