SE 226-ADVANCED PROGRAMMING

PROJECT REPORT

1. *Introduction*

The "Best Hotel for You" project is a Python application with a graphical user interface designed to help users choose hotels in European cities. The goal of this project is to create an application that allows users to select a city, set check-in and check-out dates, and view a list of the best hotels based on certain criteria. This project uses web scraping techniques to collect hotel data from Booking.com and presents the information in an interface. It also allows the prices of the hotels shown as projects to be selected in Euro or TL.

2. *Implementation*:

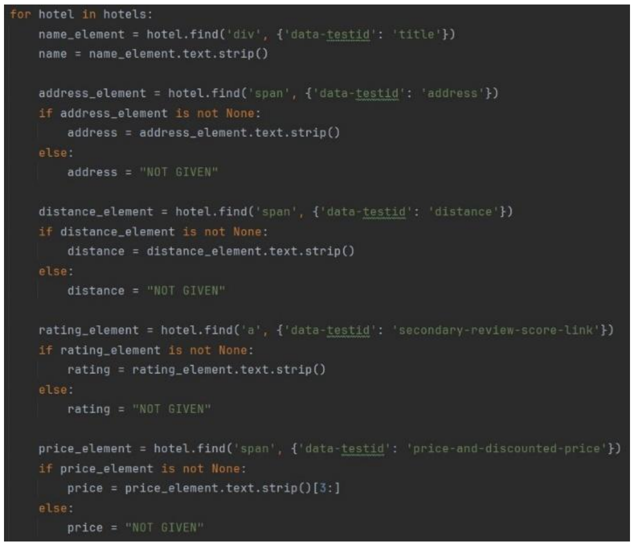
2.1 Web Scraping Requirements:

The web scraping codes of the project are included in scrapping.py and some of the components included are as follows:

- HTTP Requests: Utilizes the requests (import requests) library to send HTTP requests to Booking.com, querying hotel information based on user inputs.

- HTML Parsing: Uses "BeautifulSoup" to parse the HTML content of Booking.com pages and extract relevant hotel attributes such as name, address, distance to city center, rating, and price. To do this, first download "beautiful soup" from the terminal (pip install beautifulsoup4). It is then imported and used.

- Data Processing: Valid information on the site is obtained by writing condition statements. If the requested information is not valid, that information is defined as "NOT GIVEN". However, considering that the name information would be found in any way, the status of NOT GIVEN was not checked.



```python
for hotel in hotels:
    name_element = hotel.find('div', {'data-testid': 'title'})
    name = name_element.text.strip()

    address_element = hotel.find('span', {'data-testid': 'address'})
    if address_element is not None:
        address = address_element.text.strip()
    else:
        address = "NOT GIVEN"

    distance_element = hotel.find('span', {'data-testid': 'distance'})
    if distance_element is not None:
        distance = distance_element.text.strip()
    else:
        distance = "NOT GIVEN"

    rating_element = hotel.find('a', {'data-testid': 'secondary-review-score-link'})
    if rating_element is not None:
        rating = rating_element.text.strip()
    else:
        rating = "NOT GIVEN"

    price_element = hotel.find('span', {'data-testid': 'price-and-discounted-price'})
    if price_element is not None:
        price = price_element.text.strip()[3:]
    else:
        price = "NOT GIVEN"
```

Figure 1Data Processıng

2.2 GUI Development Requirements:

The graphical user interface (GUI) codes of the project are included in GUI.py and some of the components included are as follows:

- Integration: Constructs an GUI using Tkinter, featuring dropdown menus for city selection, entry fields for check-in and check-out dates, and radio buttons for currency selection (Euro or Turkish Lira). As stated in lines 24-34 in the GUI.py, 10 city options will be shown to the user. As stated in the GUI.py, city selection, date selection boxes are created, and the user is asked to enter them.

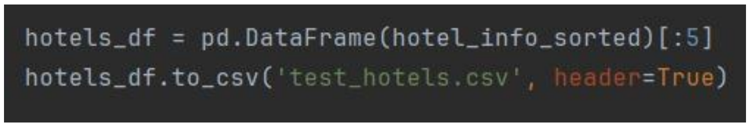
- User Interaction: Provides error handling mechanisms to validate user inputs. When any error or unacceptable input is received, an exception message is displayed on the interface. The check_date() method on line 87 checks the validity of the entered date. Between lines 105 and 121, there are codes that control the inputs and display the necessary messages to the user. The first thing to check is that the city has been selected and the dates have been entered. If the dates are entered later, the dates are checked. These checks are that check-in and check-out are not the same, check-out is not before the check-in, and the entered date is not a past date.

- Data Presentation: Displays the top 5 hotels in the GUI interface with detailed information including name, address, distance, rating, and price based on specific sorting criteria (rating). Whether there is an internet connection is checked in the Scrapping class. In case there is no Internet, the "Name" information on line 64 is updated to "Connection Error". In the GUI class, "Name" information is checked on line 127. If it is defined as "Connection Error", an error message is displayed on the screen and the code is terminated.

2.3 Hotel Information Storage/Display Requirements

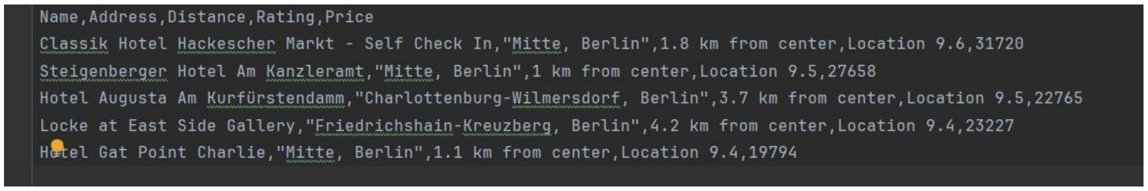
The hotel information storage and display functionality within the "Best Hotels for You" project involve the following components:

- Data Export: Sorted hotel data with all attributes is saved in a text or CSV file (test_hotels.csv) for further analysis or reference.



```
hotels_df = pd.DataFrame(hotel_info_sorted)[:5]
hotels_df.to_csv('test_hotels.csv', header=True)
```

Figure 2 CSV



```
Name,Address,Distance,Rating,Price
Classik Hotel Hackescher Markt - Self Check In,"Mitte, Berlin",1.8 km from center,Location 9.6,31720
Steigenberger Hotel Am Kanzleramt,"Mitte, Berlin",1 km from center,Location 9.5,27658
Hotel Augusta Am Kurfürstendamm,"Charlottenburg-Wilmersdorf, Berlin",3.7 km from center,Location 9.5,22765
Locke at East Side Gallery,"Friedrichshain-Kreuzberg, Berlin",4.2 km from center,Location 9.4,23227
Hotel Gat Point Charlie,"Mitte, Berlin",1.1 km from center,Location 9.4,19794
```

Figure 3 CSV-2

3. ***Challenges***:

- Sorting and Handling Rating Information: One of the significant challenges faced during the project was dealing with the hotel ratings, which included both numerical values and textual descriptions (e.g., "Comfort 8.2"). This complexity made it difficult to sort the hotels based on their ratings. This required additional parsing and validation steps to ensure that the ratings were correctly interpreted and sorted.

- Web Scraping Limitations and HTML Structure Variability: Web scraping from Booking.com presented challenges due to the variability in the HTML structure of the website. Handling missing data and ensuring that absent information was correctly marked as "NOT GIVEN" was another challenge for the scraping process.

- Date Validation and Exception Handling: Identifying invalid dates and preventing code corruption was difficult and important. In this case, it was necessary to provide error messages to the user. Another exception was that the code would not crash if there was no internet. One of the time-consuming parts was the need to plan for this situation in both classes (GUI, Scrapping).

4. ***Conclusion***:

Overall, the project is favorable in terms of functionality, User Interface and Data Integrity, demonstrating web scraping and GUI usage, providing a useful tool for users looking for hotel accommodation in European cities.