University of St. Gallen,
School of Management, Economics, Law, Social Sciences,
International Affairs and Computer Science (HSG)

Explore Switzerland: A Yelp-Integrated Trip Planner

Gianluca Amstutz (20-620-613) - g_97
Laurin Curschellas (20-613-964) - BugHunter
Alexander Gede Vogel (19-615-806) - alexander_v

GitHub Link: https://github.com/BugHunter19/group1233.git

Introduction to Programming
Coding Language: Python
Dr. Mario Silic
26.05.2023

Table of Contents

1. Introduction: Project idea and aim

The goal of this project is to develop a user-friendly program in Python, which can be used in everyday life. The program should provide a comprehensive solution for travelers, to enable them to efficiently plan their journeys. The idea arose due to the absence of comparable apps that offer both, travel options and activities or even accommodation possibilities. This program's aim is to simplify traveling by offering real-time information and review-based recommendations, enabling users to make well-informed decisions and optimize their traveling experience. By integrating transportation connections, activity suggestions and hotel recommendations, the system assists users in organizing their travel plans and finding the best options based on their preferences.

All of this can be achieved in just three simple steps. In the first step, the desired destination is entered. Then, the user provides their starting point or current location. In the final step, the preference regarding the activity (e.g. bar, restaurant, cafe) can be specified. With this information, the program searches for the next connection as well as the top 5 restaurants (or bars or cafes) in the vicinity and creates a list for the user.

In an additional step, it is possible to specify the duration of the stay if one wishes to stay longer than a day. In this case, a list of hotels nearby will also be provided.

2. Code Explanation

This chapter briefly breaks down the entire code and explains, how each step was programmed. All the explanations are included in the code as well.

The first part of the code imports the necessary modules for working with dates and times in Python and sets the timezone to Europe/Zurich. The next part continues with additional imports and functionalities related to data retrieval and geolocation. The "request" module is commonly used for making HTTP requests and interacting with web APIs, the "JSON" module provides functions for working with JSON data. The line "from geopy.geocoders import Nominatim" imports the "Nominatim" class from the "geopy. geocoders" module. This can be used to translate coordinates into the names of the cities.

Afterwards, a "Connection" class is created, representing a transportation connection between two locations. It displays the required data regarding the departure and arrival times in Unix timestamps and the connection information such as train number and platform. An "if" statement is added, which checks the types of arguments that have been passed into the method. When the provided input satisfies the aforementioned conditions, the class instance variables are accordingly set with these inputs. The class has the attributes "destination_x", "destination_y", "departure_time", "arrival_time" and "transport_means". The first method ("__init__self, destination_x, destination_y, departure, arrival, transport_means") initializes the "Connection" object with the provided parameters, validates the types and assigns values to the corresponding attributes. The method "__str__(self)" returns a string representation of the connection object, displaying the transport means, departure time and arrival time. The "get_unix_departure_time(self)" and "get_unix_arrival_time(self)" methods calculate and return the departure/arrival time as a Unix timestamp.

After that, the "find_connection" function retrieves transportation connections between two locations using the Open Data transport API. Again, origin, destination, departure date and time are taken as parameters. First, the "find_connection" function and the parameters for the open API are defined. Then, "r" sends a "GET request" to the specified URL with the parameters that were specified. After that, the code retrieves the current date and time using "datetime.now()" and formats it into strings. Then, the code prompts the user to enter the destination and origin locations using the "input" function. Finally, the "find_connection" is called with the provided data and the returned "Connection" is stored in the "last_conn" variable.

The next part of the code involves a "find_activities" function, which retrieves a list of activities (e.g. restaurants, cafes and bars) using the Yelp Fusion API. The parameters for this function are latitude, longitude, open time and radius. The activities are sorted by rating and the name and price level of the

top five get extracted. Again, an “input” function is provided, allowing the user to select the preferred type of activity.

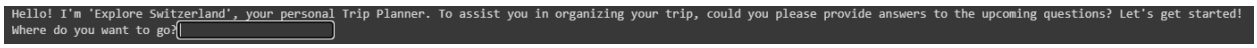
As a final step, two tables are created by using the “tabulate” module. The first table displays information the connection (departure, destination, train information) and selected activity. The second table shows the top 5 activities with their names, ratings and price levels.

The last part of the code provides an additional option for defining the return journey and finding the top 5 hotels in the destination city, if the return journey is not on the same day. Again, it uses the Yelp API and takes latitude, longitude and radius as parameters. Two “input” functions are used to determine the date and time of the return journey. The output is a table, which displays the name, rating and price class of the hotels. The rating sort is not strictly sorted by the rating value (e.g. stars or points given), but by an adjusted rating value that also takes into account the number of ratings, similar to a Bayesian average, in order to prevent the skewing of results to businesses with a single review. At last, the message “I hope you enjoyed my service. Have a great Journey!” is displayed.

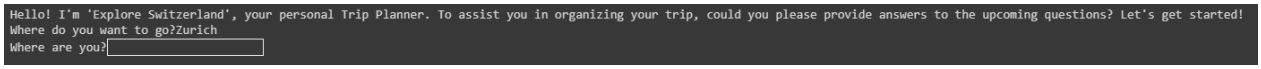
3. Example Output

This chapter presents two example outputs of the code. The first output demonstrates the option with the same-day return, while the second one utilizes the hotel search function and a next-day return.

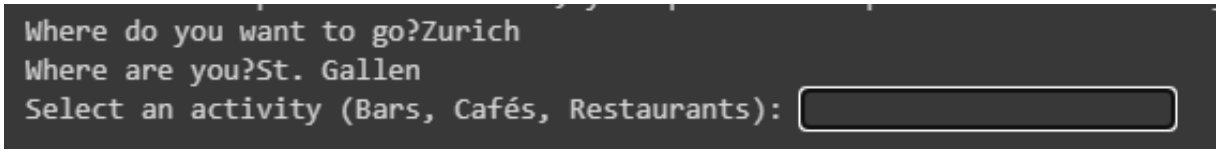
Firstly, the program greets the user with the following words: “Hello! I’m ‘Explore Switzerland’, your personal Trip Planner. To assist you in organizing your trip, could you please provide answers to the upcoming questions? Let's get started!”. Also, the first input field pops up, allowing the user to set a destination.



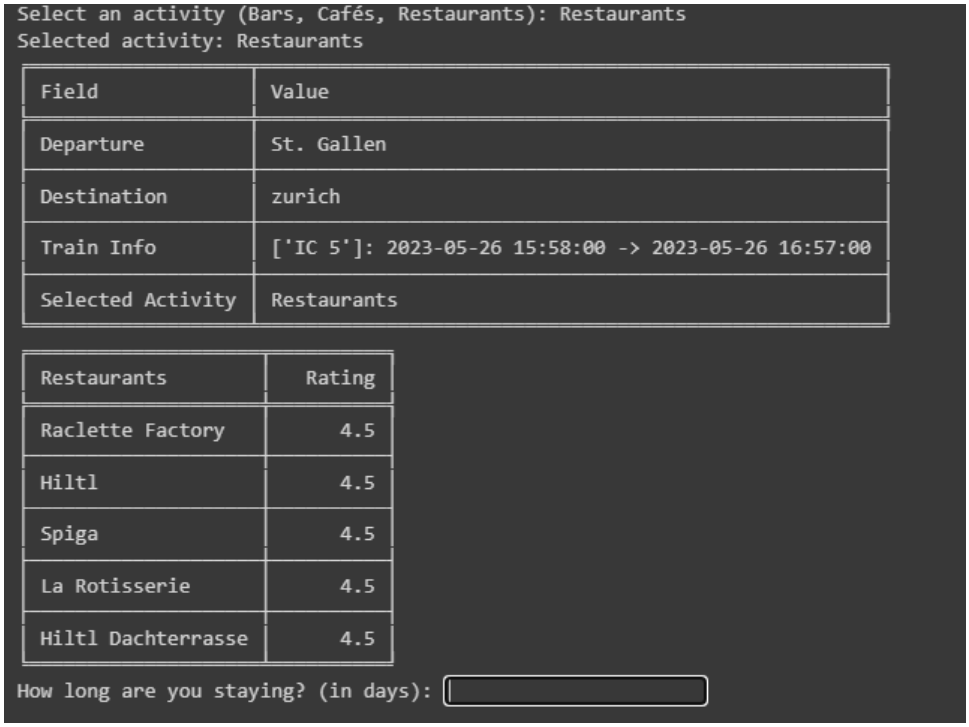
In this example, Zurich was set as the destination. Immediately after entering the destination, another input field allows the user to set his location.



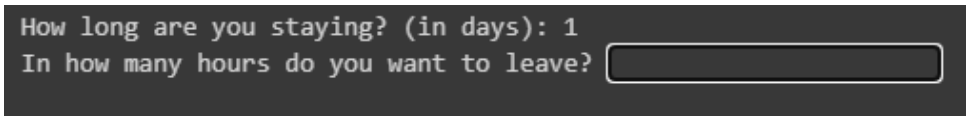
After setting the starting point of the trip (e.g. St. Gallen), the program asks for the desired activity, giving the user a choice out of “Bars”, “Cafés” and “Restaurants”. In this example, the option “Restaurants” was selected.



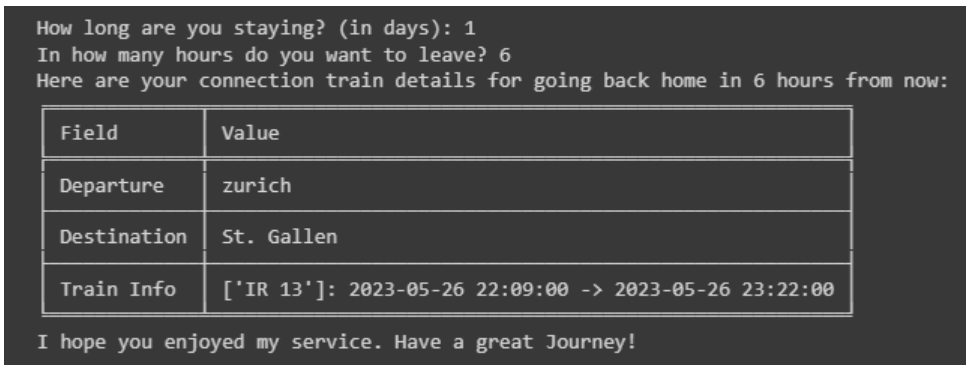
As a result, this output is generated, displaying the starting and destination point, the train connection, the activity, as well as the restaurants with their respective ratings. Furthermore, there is an option to define the duration of the journey.



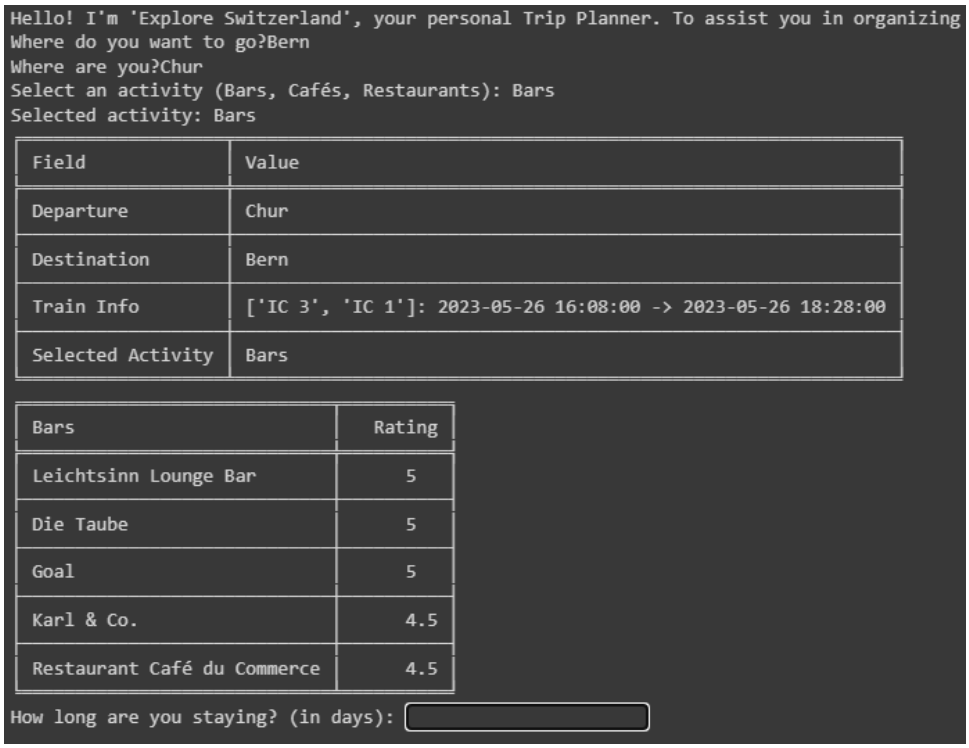
In this example, since the return is scheduled for the same day, the program prompts the user to input the number of hours until the return.



After entering the information (in this example 6 hours), an output with a train connection to head back to the starting point is displayed.



The first steps of the second example are similar to the ones in the first example. In order to get different outputs, the parameters were changed to “Bern”, “Chur”, and “Bars”.



Alternatively from the first example, the trip is extended to two days here, generating an output with the top five hotels in Bern, including their respective ratings and price classes.

```
How long are you staying? (in days): 2
```

Hotels	Rating	Price
Hotel Bellevue Palace Immobilien	5	$$$$
Bern Backpackers Hotel Glocke	4	$$
Hotel Belle Epoque Jörg Musfeld	4.5	$$
Hotel Bern	4	$$$
Hotel Restaurant Goldener Schlüssel Bern . Marianne & Jost	3.5	$$$

```
I hope you enjoyed my service. Have a great Journey!
```

References

The inspiration and main source of this project was the course “Fundamentals of Computer Science” at the University of St. Gallen, particularly Assignment 6, which was discussed in the exercise group. In addition, opendata.ch (https://transport.opendata.ch/docs.html) was used, which provided the documentation for the OpenData Transport API. The API-Key[1] for this program was created on the Yelp Development Portal (https://docs.developer.yelp.com/docs/fusion-intro). In order to solve minor problems or answer questions we encountered whilst coding, ChatGPT (https://chat.openai.com/) and GitHub (https://github.com/) were taken into consideration.

[1] OZKjiwrpl5cRuxxx8WYPNYCOgC7TlbolDx_V0TTwoTH6tbWrKW-AsG2uxRioInqrxn7Nk2spjF9i0QzuRf8DG4DOqOxcfQsack070Ey3AQk43NaIRUgaKtBdKy2NYXYx

Declaration of authorship

We hereby declare:

- that we have written this thesis without any help from others and without the use of documents or aids other than those stated above;

- that we have mentioned all the sources used and that we have cited them correctly according to established academic citation rules;

- that we have acquired any immaterial rights to material we may have used, such as images or graphs, or that we have produced such materials ourselves;

- that the topic or parts of it are not already the object of any work or examination of another course unless this has been explicitly agreed to with the faculty member in advance and is referred to in the thesis;

- that we will not pass on copies of this work to third parties or publish them without university's written consent if a direct connection can be established with the University of St. Gallen or its faculty members;

- that we are aware that our work can be electronically checked for plagiarism and that we hereby grant the University of St. Gallen copyright in accordance with the Examination Regulations insofar as this is required to administrative action;

- that we are aware that the university will prosecute any infringement of this declaration of authorship and, in particular, the employment of a ghost writer, and that any such infringement may result in disciplinary and criminal consequences which may result in my expulsion from the university or our beings stripped of our degrees.

By uploading this academic term paper, we confirm through our conclusive action that we are submitting the Declaration of Authorship, that we have read and understood it and that it is true.

St. Gallen, 26th of May

Gianluca Amstutz, Laurin Curschellas, Alexander Gede Vogel