Project2： Finding the best nearby destinations using Google API

Shaowei Lin linsw@bu.edu

Project Purpose and Background

With the increasing pace of urban life, there is a growing need to quickly and accurately find nearby services such as restaurants, cafes, or gas stations. Although there are various mapping applications available on the market, they usually provide information that needs to be bigger and more complex, making users need to spend extra time to sift and compare. In addition, these apps are usually designed to fulfill the needs of most of the population. They may need help to meet the more specific needs of individual users (e.g., finding only highly rated restaurants or currently open stores).

With this background in mind, this project aims to provide a straightforward solution that allows users to quickly get highly rated and currently open service establishments around them by entering an address and selecting the type of business they wish to find. The project uses the Google Maps API to provide rich and accurate geographic information and merchant data.

Uses

City planning and analysis: Analyzing the types and ratings of services in a given area can provide some data to support city planning.

Travel planning: Travelers or tour operators can use this tool to better plan their trip by knowing in advance what services are available near their destination.

Market Research: While more comprehensive than specialized market analysis tools, this simple tool can provide small businesses or self-employed individuals with preliminary market information, such as the availability of similar services in the vicinity and how users rate them.

Emergency Response: In an emergency or disaster, it may be critical to quickly locate necessary nearby facilities (e.g., gas stations, hospitals.).

Social Events: Users can use the tool to plan meeting places with friends or family, choosing the most suitable venues based on ratings and distance.

API

In this project, I have used Google Maps APIs, specifically Google Maps Places API and Google Maps Geocoding API. These APIs have the following features:

Geocoding: By entering an address, you can get its corresponding latitude and longitude information, the first step in a location search.

Places Search: This API allows users to search for specific types of locations (e.g., restaurants, gas stations.) within a specified latitude, longitude, and radius. It also provides basic information about the location (e.g., address, rating.).

Distance Calculation (Distance Matrix): This section is used to calculate the actual distance between the input address and the search result.

These APIs are well suited to the needs of this project as they provide comprehensive and accurate data, allowing users to perform personalized and location-accurate searches.

User Interaction

When the program runs, an input box first appears, prompting the user to enter an address. This address is the starting point for the location search.

After entering the address, the program displays a simple menu listing several common locations (e.g., restaurants, coffee shops, gas stations.). The user can select by entering a number corresponding to the option.

Based on the user's inputs and selections, the program lists the relevant locations in the vicinity and sorts them according to ratings. In addition to the basic information (e.g., name, address, rating.), the distance between these locations and the entered address is displayed, as well as the hours of operation, if available.

Code

My Python script utilizes the Google Maps API to implement a nearby location search function. While the code is relatively simple, it fulfills the needs of various practical scenarios.

Main components:

Initializing the Google Maps client: An API key is used to initialize the Google Maps client.

Address Conversion: Using the Google Maps Geocoding API, the address entered by the user is converted to latitude and longitude.

Location Search: Using the converted latitude and longitude, locations are searched within a specific radius (1000 meters) using the Google Maps Places API. Users can also select the type of location they want to search (e.g., restaurant, cafe.).

Sorting and displaying results: Searched locations are sorted by rating from highest to lowest, and their basic information (name, address, rating.) is displayed.

Calculate Distance: Use the Google Maps Distance Matrix API to calculate the distance between each location searched and the entered address.

Get opening hours: If the information is available, the opening hours of each location are also displayed.

User Interaction

Enter Address: The user is asked to enter an address as a starting point for the search.

Select Location Type: The user can select the type of location they want to search for from a predefined list.

Performance and Scalability

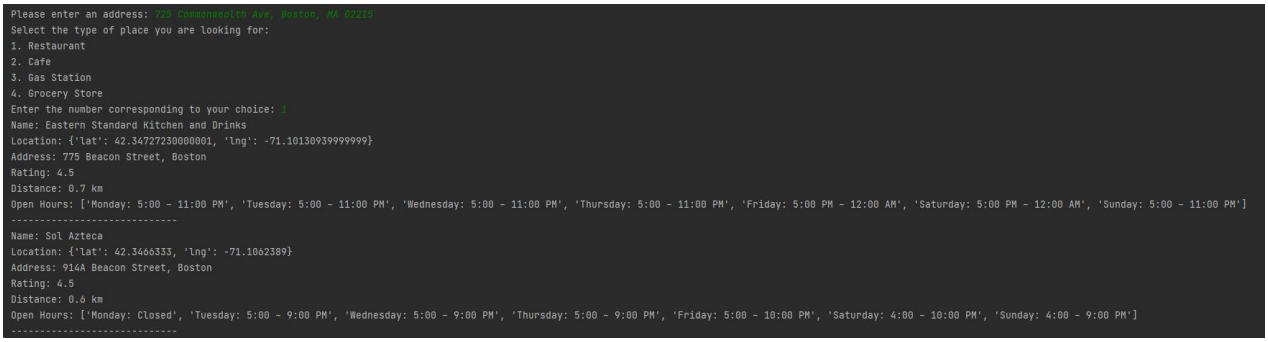
Performance

Program performance is affected by network latency and API limitations due to heavy reliance on the Google Maps API. Location sorting and filtering are done in memory, which can cause performance bottlenecks with large datasets. Optimizing the algorithm or using a

database are possible solutions. And currently, there's a command line program in the user interface section.

In this program, the user can not only search based on the type of location (e.g., restaurant, cafe) but also find out the rating, distance, and hours of operation for each location. In this way, after combining multiple factors, users can make a more appropriate decision.

In this program, the user can not only search based on the type of location (e.g., restaurant, cafe.) but also know the rating, distance, and hours of operation of each location. This way, after combining multiple factors, users can make more appropriate decisions.



From the output of this project, We first entered the address of our class. Then, we selected that we wished to find a restaurant that outputs a ranking of restaurant ratings within 1000m, corresponding to the address and distance from the starting point.

Future Improvements and Expansion Plans

The application could be enhanced by integrating features like tailored location suggestions that offer venue recommendations based on a user's historical searches and preferences. In addition, community-driven recommendations could be implemented, allowing users to share their preferred locations and simultaneously gain access to selections endorsed by their friends and family. Furthermore, smart alerts could offer timely reminders, advising users on the optimal time to arrive at a particular location, considering the venue's operational hours and the user's current geographic position.

Conclusion and Summary

This project is a highly customizable and extensible tool for searching nearby locations. Although there is still room for improvement in performance and user interface, it has successfully addressed the basic need of users to find suitable locations in unfamiliar areas. Key challenges include how to optimize performance further and how to extend it into a more comprehensive and personalized service. This project has the prospect of wide application and commercial value.