GATHER N GO

(A new approach for an extension of cab services)

Souparna Das, Giridhari Lal Gupta

Abstract

This paper presents a solution to a common problem which the travelers face while travelling from a crowded place. The common problems are high fares, unavailability of enough cabs and long waiting time. All these problems can be solved with our approach which can be used as a real business model. The current strategies of cab booking, cab sharing, cab pooling can be extended by this solution to make it easier and more convenient for the travelers. Also, if we look at it from the environmental perspective, it will help in the reduction of usage of petrol and diesel which are main source of pollution and global warming.

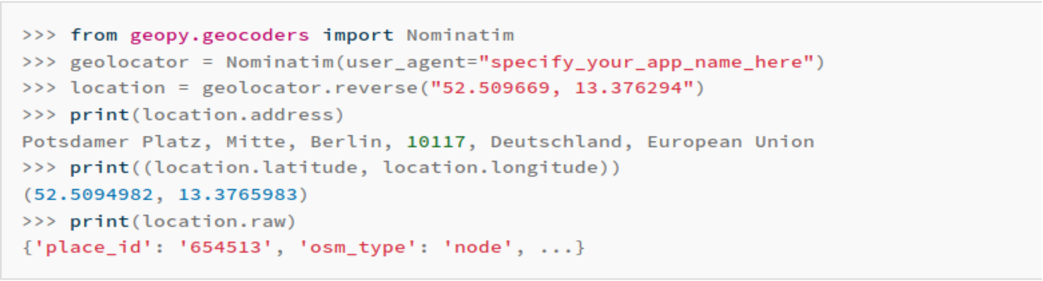
1.Introduction

In the urban areas, the crowded places are high in number. There are many such examples where people gather together, spend time, and then travel to other places. What we mean by large gathering is what we see in busy railway stations, airports, shopping malls in particular days, stadiums, parks, movie theatres and many more. People are seen standing and waiting for cabs, and the fares are usually high when the demand is high in these situations. While urban demands of trips are continuously increasing, the usage of the vehicles must be optimized (in terms of capacity). Well-planned, efficiently operated, and cost-effective transportation system management (TSM) strategies can improve mobility of existing systems for transportation users, especially in urban environments, where a good optimization of the infrastructure is needed (considering the high cost of building new facilities and the continuously increasing demand resulting from economic and population growth). The problem we were focusing on is based on cab booking when the traveler is in a crowded place. If the traveler is alone and he wants to book a cab, he might have to wait for a cab for a long time, and even if he waits, the fare he has to pay will be quite high, and also if a single person will be booking the cab, the capacity of the cab is wasted. There are solutions like cab sharing which reduces the problem but it also comes with a cost of longer travel time, or waiting for other travelers or traveling more distance than needed. To solve this problem, we can take the help of the high number of travelers in a crowded place. In urban areas, many people travel to a particular region for work and also the ever-growing societies and colonies where huge number of people lives in surroundings. So, we can find where people gather together and check the people whose destination is in the same path. We can make them communicate with each other and decide that they can share a cab together and the fare will be distributed among them according to kilometers they travel. This will reduce the time for waiting for separate cabs, the capacity of the cabs will be used more efficiently and also this is a cost-effective solution for the people who wants to travel frequently from crowded places. Reducing over usage of cabs will also create less pollution in urban cities which is an ever-increasing problem. Our solution focusses on minimizing the cost, reduce travel time and convenient for users to use because with a single click they can easily find the people going to the same direction.

2. Implementation details as part of the project

- The user must register to the website with the following information:
    - Name
    - Username
    - DOB
    - Gender
    - Email
    - Phone number (should be unique)
    - Password
- Once he registers to the website, he can login with the following details:
    - Phone Number
    - Password
- Once he login, he will be asked to turn on “location” of the device. So, we get the current location of the user with GPS.
- From the GPS location, we get the Latitude and Longitude of the user.
    - Use “geopy” library of python to get the current address of the user. “geopy.geolocator.reverse()” will give the address as a comma separated value.
    - Store the address in database.
    - Group Users based on PINCODE

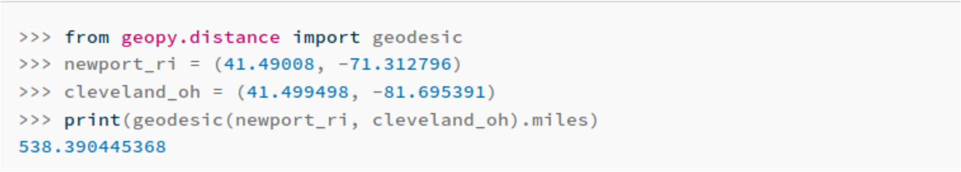
Python example for getting address from GPS location:



```
>>> from geopy.geocoders import Nominatim
>>> geolocator = Nominatim(user_agent="specify_your_app_name_here")
>>> location = geolocator.reverse("52.509669, 13.376294")
>>> print(location.address)
Potsdamer Platz, Mitte, Berlin, 10117, Deutschland, European Union
>>> print((location.latitude, location.longitude))
(52.5094982, 13.3765983)
>>> print(location.raw)
{'place_id': '654513', 'osm_type': 'node', ...}
```

- The user needs to enter his destination now.
    - Use “geopy” again to get the GPS location (Latitude & Longitude) from the entered destination.
- Once we get his/her destination, we will filter the users based on the route they will travel. For example, the users who will go with the same route are clustered together.
- Once we get the clustering, we will use Nearest-Neighbors algorithm to sort out the people who are nearest to that user based on the GPS location.

Python example for measuring distance from given Latitude and Longitude

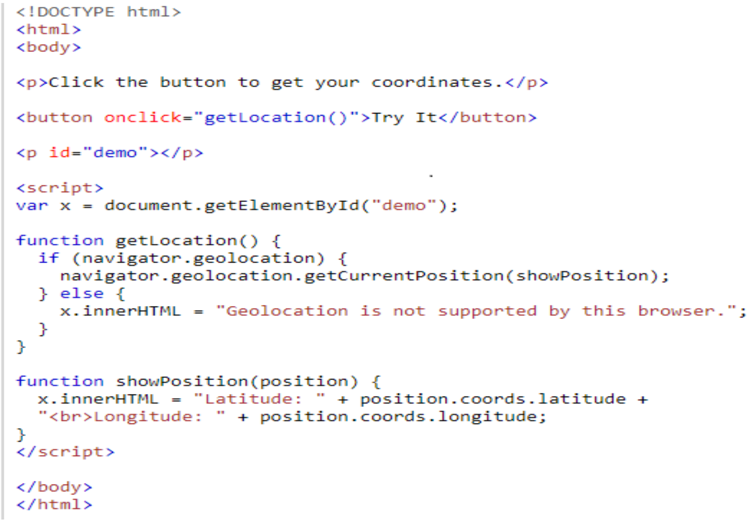


```
>>> from geopy.distance import geodesic
>>> newport_ri = (41.49008, -71.312796)
>>> cleveland_oh = (41.499498, -81.695391)
>>> print(geodesic(newport_ri, cleveland_oh).miles)
538.390445368
```

- The user will now be given a list of other users who will be traveling in the same direction. We will give option to sort the list based on ratings (discussed later) of that user or distance from the cluster center.
- The user will now send request to the other users who want to travel with him. If the other user accepts the request in a given time limit, they will be selected to book a single cab. For now, we will allow at max four people because most of the cabs are 5 seaters.
- The person who will send the request will be the team leader. The team Leader may remove anyone from the formed group or he can assign someone else as a team leader. The others who will accept the requests will be the team members, any member can send request to others to join his team,
- In the request, the number of people who are already present in a group will be mentioned.
- The team members can contact with each other (via group messages, voice chat may be enabled later).
- They will be shown the prices of different companies who provides cab booking like OLA, UBER etc. Our system will automatically detect the farthest point of that trip and accordingly show the price.
- The person who is travelling farthest will be redirected to OLA/UBER as per the preference. Once the booking is confirmed, the estimated price will have to be shared with the team members (for detecting it automatically, we need APIs).
- The members of the team will meet at the pick-up point where the cab will arrive.
- According to the estimated price, our system will calculate fare for each person depending on the distance he will travel. He needs to pay the estimated amount what we calculated before leaving the cab.

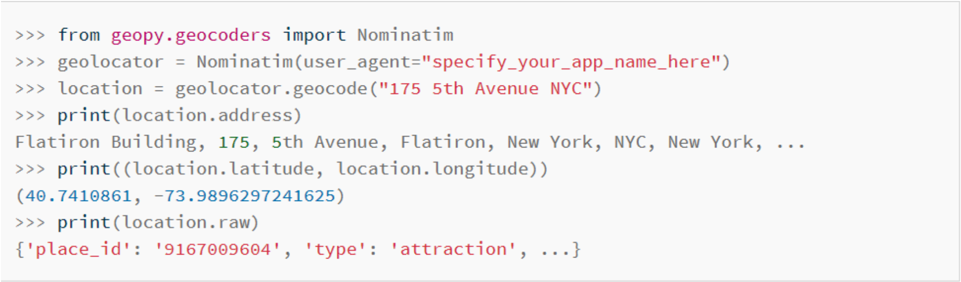
Tools and Flow:

- To make user activate the location of his device and get the GPS coordinates:



```
<!DOCTYPE html>
<html>
<body>

<p>Click the button to get your coordinates.</p>

<button onclick="getLocation()">Try It</button>

<p id="demo"></p>

<script>
var x = document.getElementById("demo");

function getLocation() {
  if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(showPosition);
  } else {
    x.innerHTML = "Geolocation is not supported by this browser.";
  }
}

function showPosition(position) {
  x.innerHTML = "Latitude: " + position.coords.latitude +
  "<br>Longitude: " + position.coords.longitude;
}
</script>

</body>
</html>
```

- Get the latitude and longitude from the entered address:



```
>>> from geopy.geocoders import Nominatim
>>> geolocator = Nominatim(user_agent="specify_your_app_name_here")
>>> location = geolocator.geocode("175 5th Avenue NYC")
>>> print(location.address)
Flatiron Building, 175, 5th Avenue, Flatiron, New York, NYC, New York, ...
>>> print((location.latitude, location.longitude))
(40.7410861, -73.9896297241625)
>>> print(location.raw)
{'place_id': '9167009604', 'type': 'attraction', ...}
```

- Create database details for user travel information:
  - User Phone Number (Foreign Key)
  - Geolocation: Starting
  - Geolocation: Destination
- After grouping users based on their PINCODEs, check and cluster them using their destination if it is on the same route.
- Show the list to each of the users on that clusters.
- Once the users have selected their team, and booked the cab, store the details in the following database tables:
  - Fare
  - Team ID
  - Members (Phone Numbers)

  Calculate the fares for each member and store it to a table:

  - Phone Number
  - Fare to Pay
- To make the fare distribution clear, show the fare to all the members of the team.