

Route Planning and Optimization

In this section we talk about the future scope of the project, which includes route planning for the end-user.

First, we include all libraries and read data as usual.

```
In [1]: import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```
In [2]: ride = pd.read_csv("rideshare_kaggle.csv")
x = ride[["latitude", "longitude"]]
```

Based on previous modules, we can divide the dataset into 12 active neighborhoods or hubs of sort. These 12 neighbourhood coordinates are stored and used for route planning.

```
In [3]: coords = {
    'Haymarket Square': [42.364, -71.060],
    'Back Bay' : [42.3503, -71.0810],
    'North End': [42.3647, -71.0542],
    'North Station': [42.3661, -71.0631],
    'Beacon Hill': [42.3588, -71.0707],
    'Boston University': [42.3505, -71.1054],
    'Fenway': [42.3505, -71.1054],
    'South Station': [42.3519, -71.0551],
    'Theatre District': [42.352, -71.065],
    'West End': [42.3661, -71.0631],
    'Financial District' : [42.3559, -71.0550],
    'Northeastern University': [42.3398, -71.0892]
}

def parse_lat(document):
    ret = [0,0]
    if document:
        lat = coords[document][0]
    return lat
```

```
def parse_long(document):
    ret = [0,0]
    if document:
        lon = coords[document][1]
    return lon
```

```
In [4]: s_coords1 = ride["source"].apply(parse_lat)
s_coords2 = ride["source"].apply(parse_long)
d_coords1 = ride["destination"].apply(parse_lat)
d_coords2 = ride["destination"].apply(parse_long)
```

```
l1 = s_coords1.append(d_coords1, ignore_index = True)
l2 = s_coords2.append(d_coords2, ignore_index = True)
```

```
C:\Users\bhaws\AppData\Local\Temp\ipykernel_10344\3532609333.py:6: FutureWarning: The
series.append method is deprecated and will be removed from pandas in a future versio
n. Use pandas.concat instead.
    l1 = s_coords1.append(d_coords1, ignore_index = True)
C:\Users\bhaws\AppData\Local\Temp\ipykernel_10344\3532609333.py:7: FutureWarning: The
series.append method is deprecated and will be removed from pandas in a future versio
n. Use pandas.concat instead.
    l2 = s_coords2.append(d_coords2, ignore_index = True)
```

```
In [5]: x = pd.concat([l1,l2],axis=1)
print(x)
```

```
          0      1
0    42.3640 -71.0600
1    42.3640 -71.0600
2    42.3640 -71.0600
3    42.3640 -71.0600
4    42.3640 -71.0600
...
1386137  42.3647 -71.0542
1386138  42.3647 -71.0542
1386139  42.3647 -71.0542
1386140  42.3647 -71.0542
1386141  42.3647 -71.0542
```

```
[1386142 rows x 2 columns]
```

We have used a Google Maps plugin to help us plan routes

```
In [6]: pip install -U googlemaps
```

```
Requirement already satisfied: googlemaps in c:\users\bhaws\anaconda3\lib\site-packages (4.10.0)
Requirement already satisfied: requests<3.0,>=2.20.0 in c:\users\bhaws\anaconda3\lib\site-packages (from googlemaps) (2.27.1)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\bhaws\anaconda3\lib\site-packages (from requests<3.0,>=2.20.0->googlemaps) (2.0.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\bhaws\anaconda3\lib\site-packages (from requests<3.0,>=2.20.0->googlemaps) (1.26.9)
Requirement already satisfied: idna<4,>=2.5 in c:\users\bhaws\anaconda3\lib\site-packages (from requests<3.0,>=2.20.0->googlemaps) (3.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\bhaws\anaconda3\lib\site-packages (from requests<3.0,>=2.20.0->googlemaps) (2021.10.8)
Note: you may need to restart the kernel to use updated packages.
```

```
In [7]: import googlemaps
```

```
gmaps = googlemaps.Client(key='AIzaSyB-Dvs504EXQNMt1L-S79Yi12I58dCejLQ')
```

Google Maps uses Geocoding which is used to convert addresses to gps coordinates and vice versa.

```
In [8]: geocode_result = gmaps.geocode('singapore 018956')
print(geocode_result[0]["formatted_address"])
print(geocode_result[0]["geometry"]["location"]["lat"])
print(geocode_result[0]["geometry"]["location"]["lng"])
```

```

print()

#full address
geocode_result = gmaps.geocode("10 Bayfront Ave, Singapore 018956")
print(geocode_result[0]["formatted_address"])
print(geocode_result[0]["geometry"]["location"]["lat"])
print(geocode_result[0]["geometry"]["location"]["lng"])
print()

#place name/restaurant name
geocode_result = gmaps.geocode('124/7B saraswati apartments, pune')
print(geocode_result[0]["formatted_address"])
print(geocode_result[0]["geometry"]["location"]["lat"])
print(geocode_result[0]["geometry"]["location"]["lng"])
print()

```

Singapore 018956
1.282275
103.858322

10 Bayfront Ave, Singapore 018956
1.2824499
103.8578807

Saraswati Apartments, Keshav Nagar, Malwadi, Hadapsar, Pune, Maharashtra 411028, India
18.5089229
73.9364299

In [9]: reverse_geocode_result = gmaps.reverse_geocode((42.34505139, -71.08509892))

print(reverse_geocode_result[0]["formatted_address"])

41 Clearway St, Boston, MA 02115, USA

We input the the 12 different neighborhoods in our dataset in the form of a list which we will later use in a function to convert into google maps locations and then create a route.

In [10]: locations = [
'Haymarket Square, Boston',
'Back Bay, Boston',
'North End, Boston',
'North Station, Boston',
'Beacon Hill, Boston',
'Boston University, Boston',
'Fenway, Boston',
'South Station, Boston',
'Theatre District, Boston',
'West End, Boston',
'Financial District, Boston',
'Northeastern University, Boston']

markers = ["color:blue|size:mid|label:" + chr(65+i) + "|" + r for i, r in enumerate(locations)]

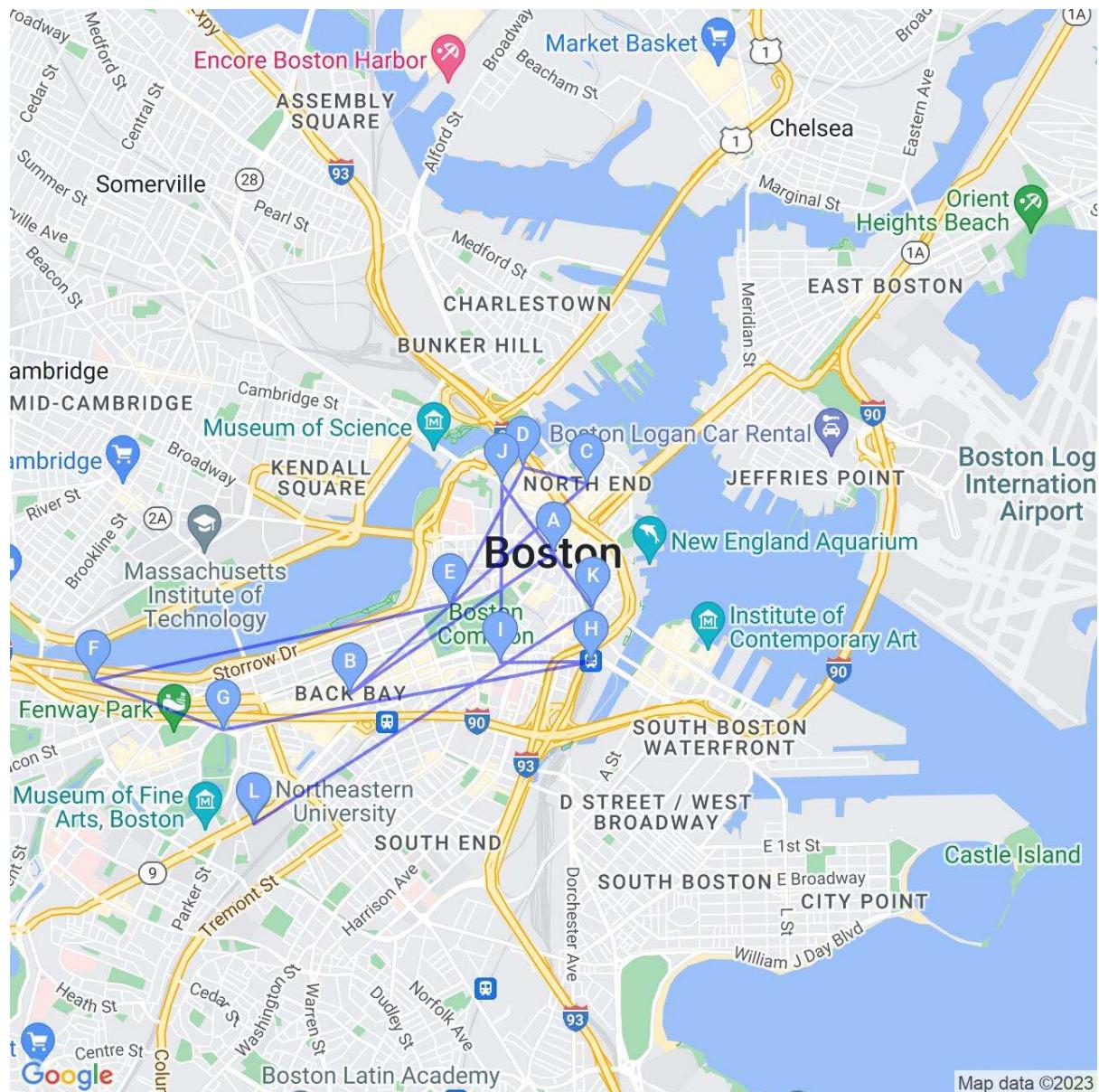
```
In [11]: result_map = gmaps.static_map(  
            center=locations[0],  
            scale=2,  
            zoom=13,  
            size=[640, 640],  
            format="jpg",  
            maptype="roadmap",  
            markers=markers,  
            path="color:0x0000ff|weight:2|" + "|".join(locations))
```

```
In [12]: with open('driving_route_map.jpg', 'wb') as img:  
    for chunk in result_map:  
        img.write(chunk)
```

Plotting of 12 Possible Hubs on Google Maps

The following map shows a random route showing the comparison of all 12 hubs in one single route. It is clearly infeasible since the routes are straight lines to the next stop.

```
In [13]: from IPython.display import Image, display  
i = Image('driving_route_map.jpg', width=600, height=600)  
display(i)
```



Map data ©2023

```
In [14]: waypoints = [
    'Back Bay, Boston',
    'North End, Boston',
    'North Station, Boston',
    'Beacon Hill, Boston',
    'Boston University, Boston',
    'Fenway, Boston',
    'South Station, Boston',
    'Theatre District, Boston',
    'West End, Boston',
    'Financial District, Boston',
]

results = gmaps.directions(origin = 'Haymarket Square, Boston',
                           destination = 'Northeastern University, Boston',
                           waypoints = waypoints,
                           optimize_waypoints = True)

for i, leg in enumerate(results[0]["legs"]):
    print("Stop:" + str(i),
          leg["start_address"],
          "=> ",
```

```

        leg["end_address"],
        "distance: ",
        leg["distance"]["value"],
        "traveling Time: ",
        leg["duration"]["value"]
    )
}

Stop:0 Boston, MA, USA ==> West End, Boston, MA, USA distance: 843 traveling Time: 291
Stop:1 West End, Boston, MA, USA ==> North Station, 135 Causeway St, Boston, MA 02114, USA distance: 88 traveling Time: 33
Stop:2 North Station, 135 Causeway St, Boston, MA 02114, USA ==> North End, Boston, MA, USA distance: 1091 traveling Time: 522
Stop:3 North End, Boston, MA, USA ==> Financial District, Boston, MA, USA distance: 1565 traveling Time: 501
Stop:4 Financial District, Boston, MA, USA ==> South Station, 700 Atlantic Ave, Boston, MA 02110, USA distance: 543 traveling Time: 158
Stop:5 South Station, 700 Atlantic Ave, Boston, MA 02110, USA ==> Boston Theater District, Boston, MA, USA distance: 1948 traveling Time: 643
Stop:6 Boston Theater District, Boston, MA, USA ==> Beacon Hill, Boston, MA, USA distance: 1564 traveling Time: 535
Stop:7 Beacon Hill, Boston, MA, USA ==> Back Bay, Boston, MA, USA distance: 1453 traveling Time: 356
Stop:8 Back Bay, Boston, MA, USA ==> Boston, MA 02215, USA distance: 2044 traveling Time: 450
Stop:9 Boston, MA 02215, USA ==> Fenway-Kenmore, Boston, MA, USA distance: 2970 traveling Time: 325
Stop:10 Fenway-Kenmore, Boston, MA, USA ==> 360 Huntington Ave, Boston, MA 02115, USA distance: 542 traveling Time: 97

```

Example Route Covering all 12 hotspots

The following map shows an example route which passes through all 12 neighborhoods in our dataset. It is clearly not optimised and simply shows one of the available orders in which we can cover all spots.

```

In [15]: marker_points = []
waypoints = []

#extract the location points from the previous directions function

for leg in results[0]["legs"]:
    leg_start_loc = leg["start_location"]
    marker_points.append(f'{leg_start_loc["lat"]},{leg_start_loc["lng"]}')
    for step in leg["steps"]:
        end_loc = step["end_location"]
        waypoints.append(f'{end_loc["lat"]},{end_loc["lng"]}')
last_stop = results[0]["legs"][-1]["end_location"]
marker_points.append(f'{last_stop["lat"]},{last_stop["lng"]}')

markers = [ "color:blue|size:mid|label:" + chr(65+i) + "|"
            + r for i, r in enumerate(marker_points)]
result_map = gmaps.static_map(
            center = waypoints[0],
            scale=2,
            zoom=13,
            size=[640, 640],

```

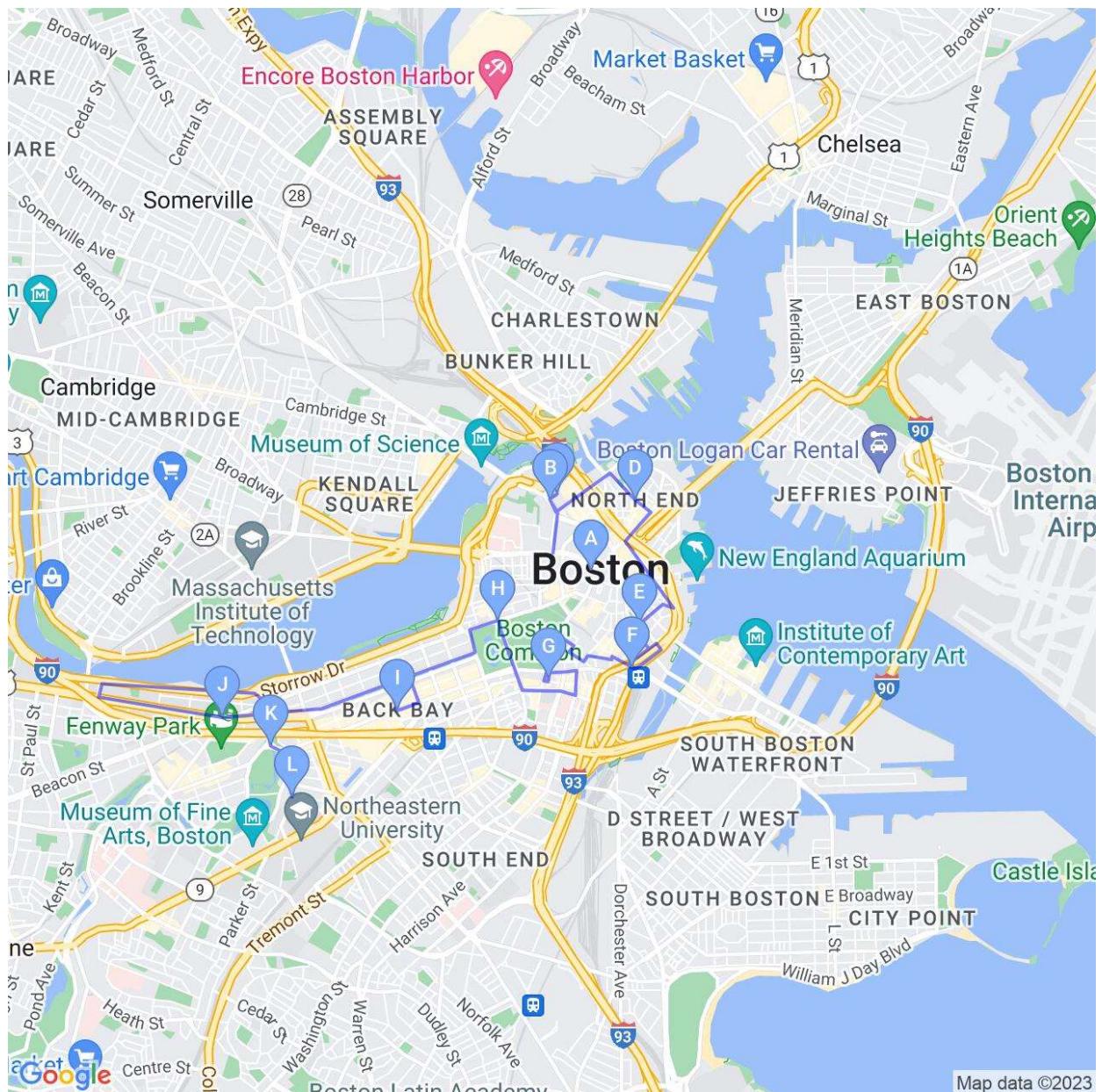
```

format="jpg",
maptype="roadmap",
markers=markers,
path="color:0x0000ff|weight:2|" + "|".join(waypoints))

```

```
In [16]: with open('driving_route_map2.jpg', 'wb') as img:
    for chunk in result_map:
        img.write(chunk)

i = Image('driving_route_map2.jpg')
display(i)
```



Example Routing as per Customer Demand

The following code blocks show, in small part, what our project is capable of. We can input from the user a pickup and a destination, and we can plan routes such that the ride travels through the closest possible hubs. We could also use the following for transit route planning passing through any combination of the 12 hotspots.

For now, we can input just the pickup and dropoff hotspot and google maps returns the most optimal route.

```
In [17]: print('1. Haymarket Square')
print('2. Back Bay')
print('3. North End')
print('4. North Station')
print('5. Beacon Hill')
print('6. Boston University')
print('7. Fenway')
print('8. South Station')
print('9. Theatre District')
print('10. West End')
print('11. Financial District')
print('12. Northeastern University')

print("-----")
s = input("Select Source Option: ")
d = input("Select Destination Option: ")
source = locations[int(s)-1]
destination = locations[int(d)-1]
print("-----")
print('Selected Source: ',source)
print('Selected Destination: ',destination)

results = gmaps.directions(origin = source,
                           destination = destination,
                           optimize_waypoints = True)

marker_points = []
waypoints = []

#extract the location points from the previous directions function

for leg in results[0]["legs"]:
    leg_start_loc = leg["start_location"]
    marker_points.append(f'{leg_start_loc["lat"]},{leg_start_loc["lng"]}')
    for step in leg["steps"]:
        end_loc = step["end_location"]
        waypoints.append(f'{end_loc["lat"]},{end_loc["lng"]}')
last_stop = results[0]["legs"][-1]["end_location"]
marker_points.append(f'{last_stop["lat"]},{last_stop["lng"]}')

markers = [ "color:blue|size:mid|label:" + chr(65+i) + "|"
           + r for i, r in enumerate(marker_points)]

result_map2 = gmaps.static_map(
    center = waypoints[0],
    scale=2,
    zoom=13,
    size=[640, 640],
    format="jpg",
    maptype="roadmap",
    markers=markers,
    path="color:0x0000ff|weight:2|" + "|".join(waypoints))

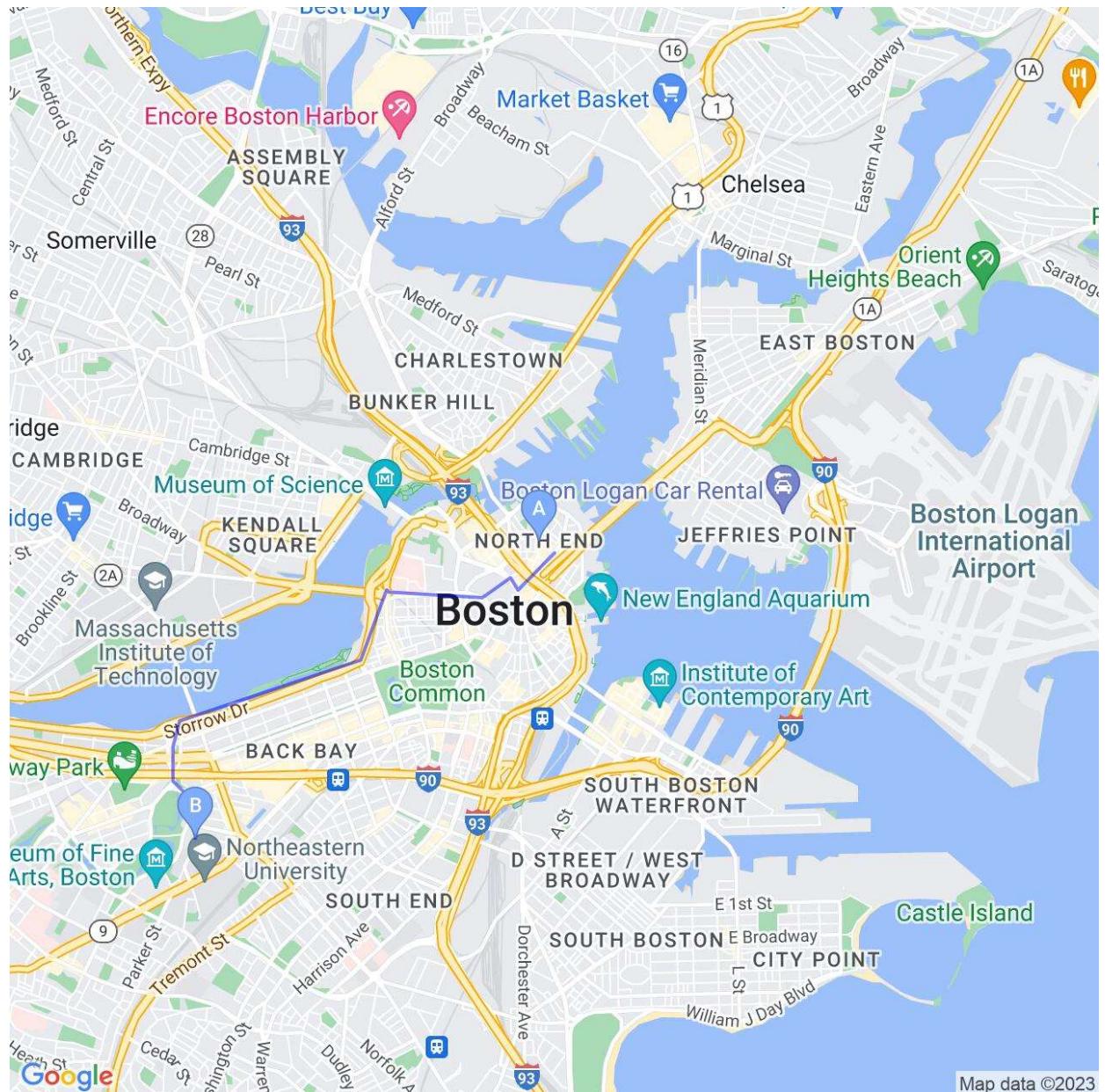
with open('driving_route_map2.jpg', 'wb') as img:
```

```
for chunk in result_map2:  
    img.write(chunk)  
  
i = Image('driving_route_map2.jpg')  
display(i)
```

1. Haymarket Square
2. Back Bay
3. North End
4. North Station
5. Beacon Hill
6. Boston University
7. Fenway
8. South Station
9. Theatre District
10. West End
11. Financial District
12. Northeastern University

Select Source Option: 3
Select Destination Option: 12

Selected Source: North End, Boston
Selected Destination: Northeastern University, Boston



The above map shows a route from the North End hub to the Northeastern University hub.

In []: