In [1]:
```
import numpy as np
import requests
```

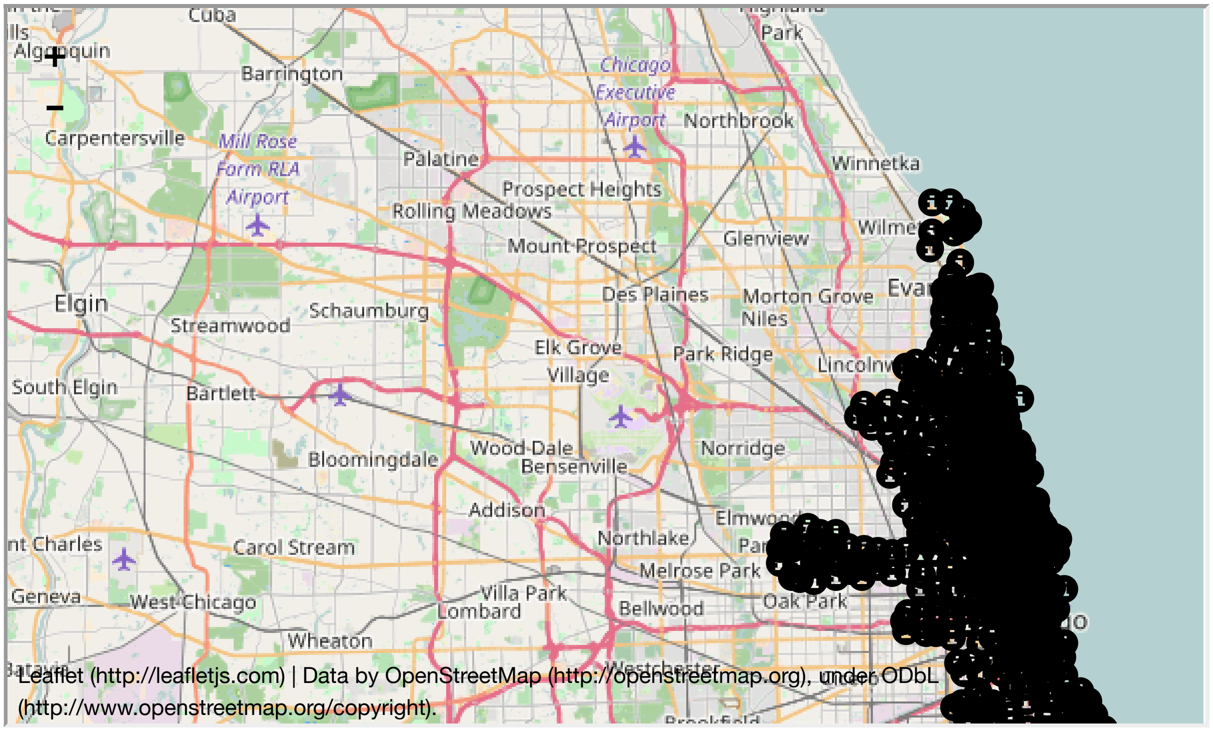
In [2]:
```
# Extract via real-time feed
station = requests.get('https://feeds.divvybikes.com/stations/stations.j
son').json()
station_list = station['stationBeanList']

latitude, longitude, station_name, num_dock, num_bike = \
    zip(*[(s['latitude'], s['longitude'], s['stationName'], s['totalDock
s'], s['availableBikes'])
          for s in station_list])
```

In [15]:
```
# Visualize the Divvy stations in a map
import folium
avg_latitude, avg_longitude = np.mean(latitude), np.mean(longitude)

divvy_map = folium.Map(location=[avg_latitude, avg_longitude], zoom_star
t=10)
for lat, lng, name, d, b in zip(latitude, longitude, station_name, num_d
ock, num_bike):
    folium.Marker(location=[lat, lng],
                  icon=folium.Icon(color='blue' if b > 0 else 'gray'),
                  popup='{} ({}/{})'.format(name, b, d)).add_to(divvy_ma
p)
divvy_map
```

Out[15]:



In [53]:

```
#Question 1: Which Divvy station has the most number of stations within
 its 2km? How many stations?
#Hint: Try np.sum() and np.argmax().#Calculate the distance matrix

# Convert degree to radian
a = np.radians(latitude)
o = np.radians(longitude)
# Pre-compute sin and cos
cos_a = np.cos(a)
sin_a = np.sin(a)
delta_o = np.subtract.outer(o, o)
cos_delta_o = np.cos(delta_o)
sin_delta_o = np.sin(delta_o)
cos_a_sin_a = np.outer(cos_a, sin_a)
# Modified haversine formula
r = 6372.795
distance = r * np.arctan2(np.sqrt((cos_a * sin_delta_o) ** 2 +
                                  (cos_a_sin_a - cos_a_sin_a.T * cos_del
ta_o) ** 2),
                          np.outer(sin_a, sin_a) + np.outer(cos_a,
cos_a) * cos_delta_o)
# Numerical
n = len(latitude)
distance.flat[::n + 1] = 0

print(distance)
```

```
[[  0.           0.93408936   2.02254058 ...,  12.82441235  10.33954292
    3.05707656]
 [  0.93408936   0.           1.23010354 ...,  13.54292122  11.26849271
    3.90447122]
 [  2.02254058   1.23010354   0.         ...,  13.79852614  12.1281266
    5.07623024]
 ...,
 [ 12.82441235  13.54292122  13.79852614 ...,   0.           7.31325836
   12.09082659]
 [ 10.33954292  11.26849271  12.1281266  ...,   7.31325836   0.
    8.01416603]
 [  3.05707656   3.90447122   5.07623024 ...,  12.09082659   8.01416603
    0.        ]]
```

In [60]:

```
#2. Create a transformed distance matrix: 1 for distance < 2, 0 for dist
ance > 2:
l = len(distance[0])
boolean_dist = [[0 for i in range(l)] for j in range(l)]

for i in range(l):
    for j in range(l):
        boolean_dist[i][j] = 1 if distance[i][j] <= 2 else 0
```

In [96]:

```
#3. sum the number of stations within 2km including the station itself
sum_of_station = []
for i in boolean_dist:
    sum_of_station.append(np.sum(i))

#4. Get the result
#the index of the Divvy station that has the most number of stations wit
hin its 2km
max_index = np.argmax(sum_of_station)
print("The index of this station is " + str(max_index))

#the number of stations
max_number_of_station = sum_of_station[max_index] - 1
print("The number of station associated with this station is " + str(max
_number_of_station))

#this Divvy station's location
max_location = (latitude[max_index],longitude[max_index])
print("The location of this station is " + str(max_location))
```

```
The index of this station is 147
The number of station associated with this station is 99
The location of this station is (41.885837, -87.6355)
```

In [95]:

```
#Question 2: Which two Divvy stations are closest to each other? How far
 apart?
#Hint: Try np.amin() and np.where().

#1. Find the minimum distance for each station
min_dist_each_station = []
for i in distance:
    min_dist_each_station.append(np.amin(i[np.nonzero(i)]))

min_dist = np.amin(min_dist_each_station)
print("The closest distance is " + str(min_dist))

#2. Find the two locations associated with this minimum distance
min_dist_loc1 = int(np.where(min_dist_each_station == min_dist)[0])
print("The index of first location is " + str(min_dist_loc1))

min_dist_loc2 = int(np.where(distance[min_dist_loc1] == min_dist)[0])
print("The index of second location is " + str(min_dist_loc2))

#3. Find the coordinates of these two locations
location_1 = (latitude[min_dist_loc1], longitude[min_dist_loc1])
location_2 = (latitude[min_dist_loc2], longitude[min_dist_loc2])
print("The latitude and longitude of the two locations are " + str(locat
ion_1) + " and " + str(location_2))
```

```
The closest distance is 0.103476949854
The index of first location is 157
The index of second location is 69
The latitude and longitude of the two locations are (41.882091, -87.639
833) and (41.882242, -87.641066)
```