In [1]:
```
import requests
res = requests.get('http://localhost:9200')
print(res.content) #Check if elastic search node is up and running.
```

```
b'{\n  "name" : "1vVZ8vC",\n  "cluster_name" : "elasticsearch",\n  "cluster_uu
id" : "sIHxkUftQV26kA1RWD_ZkQ",\n  "version" : {\n    "number" : "6.2.4",\n
"build_hash" : "ccec39f",\n    "build_date" : "2018-04-12T20:37:28.497551Z",\n
    "build_snapshot" : false,\n    "lucene_version" : "7.2.1",\n    "minimum_w
ire_compatibility_version" : "5.6.0",\n    "minimum_index_compatibility_versio
n" : "5.0.0"\n  },\n  "tagline" : "You Know, for Search"\n}\n'
```

In [2]:
```
from elasticsearch import Elasticsearch
es = Elasticsearch([{'host': 'localhost', 'port': 9200}])  #Connect to elastise
arch node setup through terminal on port 9200
import folium
from folium import plugins #Import Folium Heat Map library
```

In [3]:
```
import json
from collections import defaultdict
list_of_issues = [json.loads(line) for line in open('SPM587SP18issues.json')] #
From charting_issues.ipynb
#list_of_issues


filteredData = list() #Creating a new dictionary for 'labels', since in the ori
ginal json file,
#each of the labels are just string values instead of key value pair, thus cann
ot be queried directly.
for data in list_of_issues:
    temp_List = dict()
    if len(data['labels']):
        temp_List.update({'Author' : data['Author']}) #The new dictionary only
 has Author, Issue number and various labels as key value pair
        temp_List.update({'issue_number' : data['issue_number']})
        for i in data['labels']:
            temp_List.update({i.split(':')[0] : i.split(':')[1]})
        filteredData.append(temp_List)
```

In [4]:
```
print(len(filteredData))
filteredData[0] #Similar output for all the 234 issue records.
```

```
234
```

Out[4]:
```
{'Author': 'HSP18SCM50W',
 'Category': 'Inquiry',
 'DetectionPhase': 'Field',
 'OriginationPhase': 'Coding',
 'Priority': 'High',
 'Status': 'inProgress',
 'issue_number': 475}
```

In [5]:
```
from elasticsearch import Elasticsearch, helpers #Code taken from the tutorial
 provided
es = Elasticsearch()
actions = list()
for data in filteredData:
    action = {
        '_index' : 'issues_fin', #Name the database as issues_fin
        '_type' : 'github repo issues',
        '_id' : data['issue_number'], #use issue number as id for each dictiona
ry entry
        '_source' : data
    }
    actions.append(action)
helpers.bulk(es,actions) #Pushing the new filtered list into elastic search dat
abase named issues_fin
```

Out[5]:
```
(234, [])
```

In [6]:
```
query1 = { #First query that queries elastic search that returns all the issues
    'size' : 500,
    'query' : {
        'match_all' : {}
    }
}
```

In [7]:
```
output1 = es.search(index = 'issues_fin', body = query1, scroll = '1h') #All th
e issues from es stored in this variable
```

In [8]:
```
print(output1.keys()) #Prints all the keys in the output1 dictionary
output1['hits']['hits'][0] #Similar output for rest of the issues.
```

```
dict_keys(['_scroll_id', 'took', 'timed_out', '_shards', 'hits'])
```

Out[8]:
```
{'_id': '470',
 '_index': 'issues_fin',
 '_score': 1.0,
 '_source': {'Author': 'RSP18SCM19N',
  'Category': 'Enhancement',
  'DetectionPhase': 'Testing',
  'OriginationPhase': 'Design',
  'Priority': 'Major',
  'Status': 'Completed',
  'issue_number': 470},
 '_type': 'github repo issues'}
```

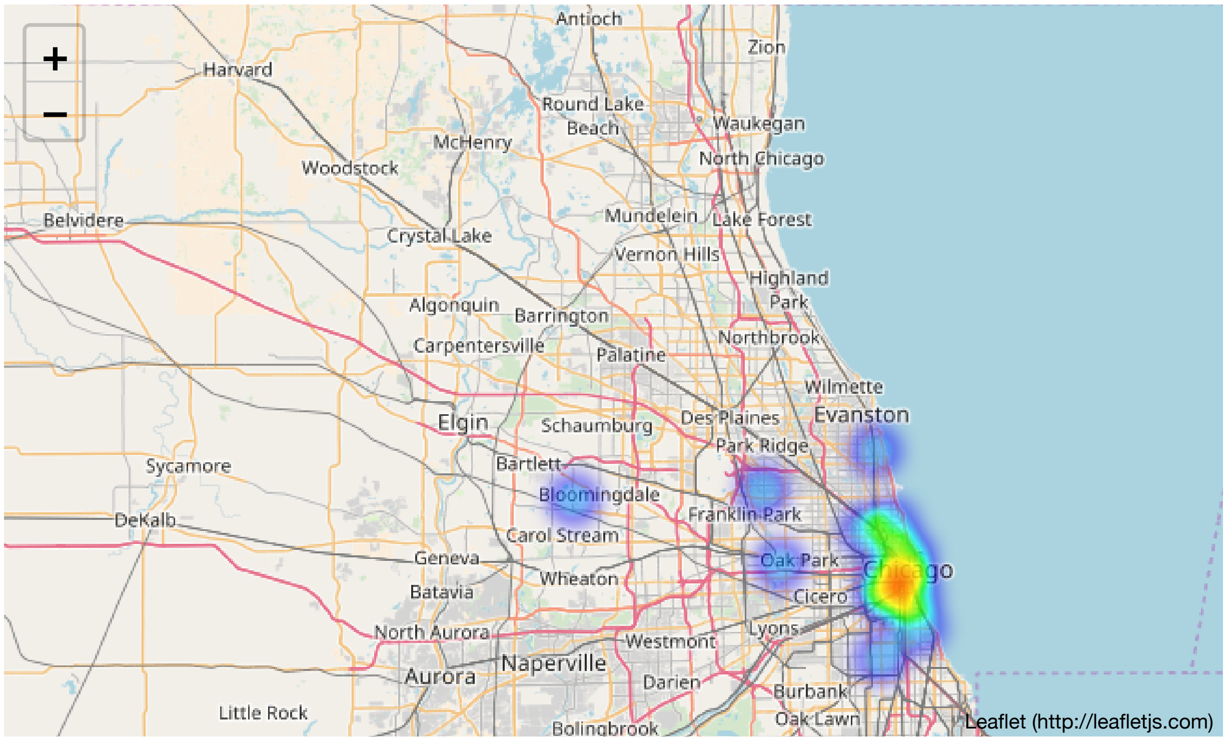
In [9]:
```
output1Lat_Long = [] #Extracting only Latitiude and Longitude labels from outpu
t1 and storing them into array
exists = False
for i in (output1['hits']['hits']): #Format of elasticsearch, after querying th
is is how labels are accessed.
    temp = []
    try: #Since many issues do not have 'Latitude' or 'Longitude' labels, it tr
ies to obtain them, if not then pass
        temp.append(float(i['_source']['Latitude']))
        temp.append(float(i['_source']['Longitude']))
        output1Lat_Long.append(temp)
    except:
        pass
```

In [10]:
```
output1Lat_Long #Array containing all the coordinates.
```

```
Out[10]: [[41.878693, -87.638924],
          [41.838897, -87.646804],
          [41.858415, -87.660926],
          [41.89302, -87.631556],
          [41.878693, -87.638924],
          [41.847095, -87.616767],
          [41.847095, -87.616767],
          [41.847095, -87.616767],
          [41.877817, -87.631247],
          [41.89323, -87.617419],
          [41.847095, -87.616767],
          [41.877846, -87.631296],
          [41.838897, -87.646804],
          [41.838897, -87.646804],
          [41.847095, -87.616767],
          [41.877846, -87.631296],
          [41.878693, -87.638924],
          [41.89323, -87.617419],
          [41.883772, -87.625962],
          [41.917164, -87.686965],
          [41.883772, -87.625962],
          [41.917164, -87.686965],
          [41.880982, -87.630553],
          [41.917164, -87.686965],
          [41.89323, -87.617419],
          [41.891551, -87.607375],
          [40.170101, -92.177847],
          [41.883772, -87.625962],
          [41.917164, -87.686965],
          [41.847095, -87.616767],
          [41.838897, -87.646804],
          [41.89302, -87.631556],
          [41.847095, -87.616767],
          [41.809739, -87.607105],
          [41.847441, -87.679408],
          [41.847095, -87.617419],
          [41.847095, -87.616767],
          [41.89302, -87.631556],
          [41.847095, -87.616767],
          [41.877817, -87.631247],
          [41.877846, -87.631296],
          [41.951072, -88.119872],
          [41.89302, -87.631556],
          [41.853136, -87.63316],
          [41.891551, -87.607375],
          [41.880982, -87.630553],
          [41.883772, -87.625962],
          [40.170101, -92.177847],
          [41.877817, -87.631247],
          [41.877817, -87.631247],
          [40.170101, -92.177847],
          [41.883772, -87.625962],
          [41.883772, -87.625962],
          [41.880982, -87.630553],
          [41.847441, -87.679408],
          [41.852623, -87.611958],
          [41.847095, -87.616767],
          [41.877817, -87.631247],
          [41.89302, -87.631556],
          [41.877846, -87.631296],
          [41.879094, -87.813483],
          [42.004828, -87.67314],
          [41.858415, -87.67314],
          [42.004828, -87.67314],
          [41.858415, -87.660926],
          [41.89302, -87.631556],
          [41.89302, -87.631556],
          [41.838897, -87.646804],
          [41.847095, -87.616767],
          [41.853136, -87.63316],
          [41.917164, -87.686965],
          [41.891551, -87.607375],
          [41.883772, -87.625962],
          [41.877817, -87.631247],
          [41.89323, -87.617419],
          [41.883772, -87.625962],
          [41.853136, -87.63316],
          [41.877817, -87.631247],
          [41.880982, -87.630553],
          [41.853136, -87.63316],
          [41.925573, -87.649249],
          [41.852623, -87.611958],
          [41.852623, -87.611958],
          [41.847095, -87.616767],
          [41.877817, -87.631247],
          [41.878693, -87.638924],
          [41.878693, -87.638924],
          [41.852623, -87.611958],
          [41.877817, -87.631247],
          [41.879094, -87.813483],
          [41.877817, -87.631247],
          [38.591142, -89.984312],
          [41.852623, -87.611958],
          [41.891551, -87.607375],
          [41.877817, -87.631247],
          [41.89323, -87.617419],
          [41.917164, -87.686965],
          [41.877817, -87.631247],
          [41.880982, -87.630553],
          [41.89323, -87.617419],
          [41.877817, -87.631247],
          [41.852623, -87.611958],
          [41.877846, -87.631296],
          [41.877817, -87.631247],
          [41.847095, -87.616767],
          [41.809739, -87.607375],
          [41.847441, -87.631247],
          [41.877948, -87.634926],
          [41.847095, -87.616767],
          [41.847095, -87.616767],
          [38.542048, -89.984333],
          [41.883772, -87.625962],
          [41.847095, -87.616767],
          [41.847095, -87.616767],
          [41.877846, -87.631296],
          [41.879094, -87.813483],
          [41.847095, -87.616767],
          [41.847095, -87.616767],
          [41.883772, -87.625962],
          [41.877846, -87.631296],
          [41.96245, -87.837132],
          [41.847095, -87.616767],
          [41.852623, -87.611958],
          [41.877817, -87.631247],
          [41.852623, -87.611958],
          [41.777989, -87.664548],
          [41.853136, -87.63316],
          [40.170101, -92.177847],
          [41.89323, -87.617419],
          [41.880982, -87.630553],
          [40.170101, -92.177847],
          [41.883772, -87.625962],
          [41.883772, -87.625962],
          [41.880982, -87.630553],
          [41.891551, -87.607375],
          [41.853136, -87.63316]]
```

In [11]:
```
output1HeatMap = folium.Map([41.891551, -87.607375], zoom_start=16) #Co-ordinat
es of Chicago, Navy Pier
output1HeatMap.add_child(plugins.HeatMap(output1Lat_Long, radius = 15)) #Imposi
ng Lat,Long values from issues on the map
```

Out[11]:



In [12]:
```
query2_1 = { #Second query that queries elastic search that returns all the iss
ues that have label DetectionPhase as Field AND Priority as Critical
    'size' : 500,
    'query' : {
        'bool' : {
            'must' : [{'match' : {'DetectionPhase' : 'Field'}}, #'must' means l
ogical AND
                      {'match' : {'Priority' : 'Critical'}}
                     ]
        }
    }
}
```

In [13]:
```
output2_1 = es.search(index = 'issues_fin', body = query2_1, scroll = '1h')#All
 the resultant issues from es stored in this variable
```

In [14]:
```
print(output2_1.keys()) #Prints all the keys in the output2_1 dictionary
output2_1['hits']['hits'][0] #Similar output for rest of the issues.
```

```
dict_keys(['_scroll_id', 'took', 'timed_out', '_shards', 'hits'])
```

Out[14]:
```
{'_id': '12',
 '_index': 'issues_fin',
 '_score': 2.383039,
 '_source': {'Address': '645 N MCCLURG CT',
  'Author': 'HSP18SCM69D',
  'Category': 'Inquiry',
  'DetectionPhase': 'Field',
  'Latitude': '41.893230',
  'OriginationPhase': 'Coding',
  'Priority': 'Critical',
  'Status': 'Approved',
  'issue_number': 12},
 '_type': 'github repo issues'}
```

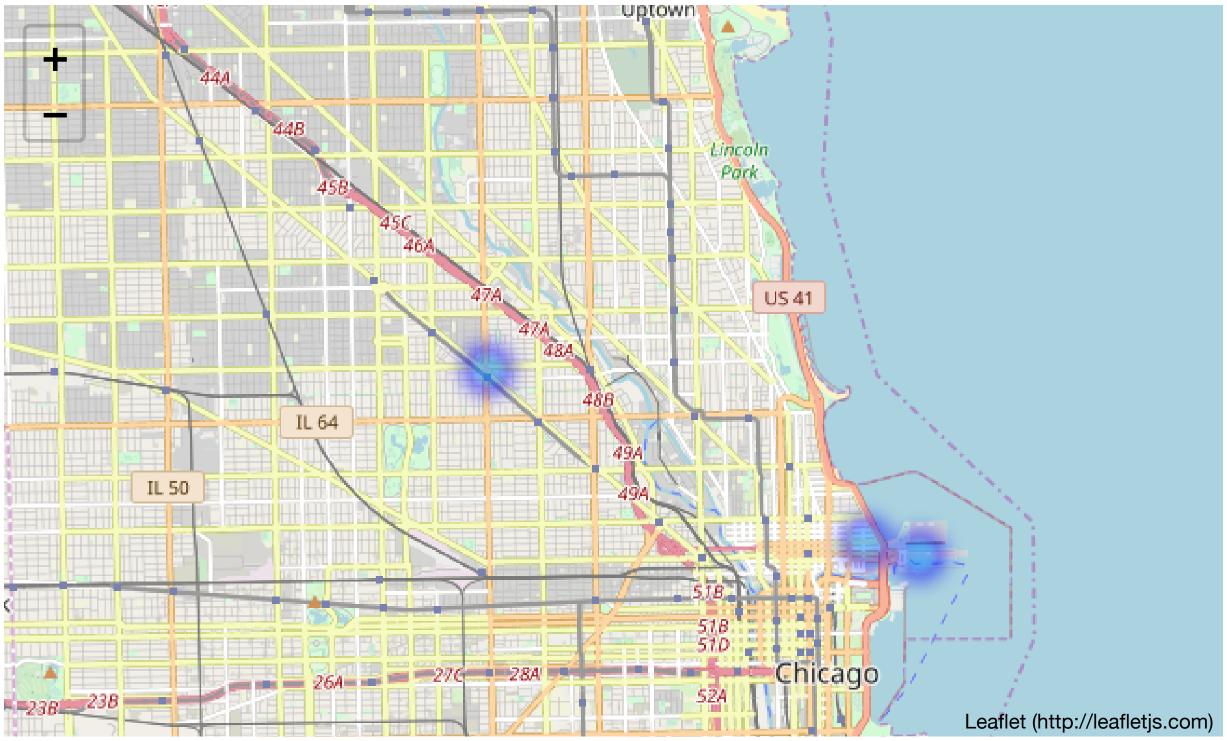
In [15]:
```
output2_1Lat_Long = [] #Extracting only Latitiude and Longitude labels from out
put1 and storing them into array
exists = False
for i in (output2_1['hits']['hits']): #Format of elasticsearch, after querying
 this is how labels are accessed.
    temp = []
    try:
        temp.append(float(i['_source']['Latitude']))#Since many issues do not h
ave 'Latitude' or 'Longitude' labels, it tries to obtain them, if not then pass
        temp.append(float(i['_source']['Longitude']))
        output2_1Lat_Long.append(temp)
    except:
        pass
```

In [16]:
```
output2_1Lat_Long
```

Out[16]:
```
[[41.89323, -87.617419], [41.917164, -87.686965], [41.891551, -87.607375]]
```

In [17]:
```
output2_1HeatMap = folium.Map([41.891551, -87.607375], zoom_start=16)#Co-ordina
tes of Chicago, Navy Pier
output2_1HeatMap.add_child(plugins.HeatMap(output2_1Lat_Long, radius = 15))#Imp
osing Lat,Long values from issues on the map
```

Out[17]:



In [18]:
```
query2_2 = {#Second query that queries elastic search that returns all the issu
es that have label DetectionPhase as Field AND Status as Completed
    'size' : 500,
    'query' : {
        'bool' : {
            'must' : [{'match' : {'DetectionPhase' : 'Field'}},
                      {'match' : {'Status' : 'Completed'}}
                     ]
        }
    }
}
```

In [19]:
```
output2_2 = es.search(index = 'issues_fin', body = query2_2, scroll = '1h')
```

In [20]:
```
print(output2_2.keys()) #Prints all the keys in the output2_2 dictionary
output2_2['hits']['hits'][0] #Similar output for rest of the issues.
```

```
dict_keys(['_scroll_id', 'took', 'timed_out', '_shards', 'hits'])
```

Out[20]:
```
{'_id': '22',
 '_index': 'issues_fin',
 '_score': 4.140232,
 '_source': {'Address': '233 W JACKSON',
  'Author': 'HSP18SCM69D',
  'Category': 'Bug',
  'DetectionPhase': 'Field',
  'Latitude': '40.170101',
  'Longitude': '-92.177847',
  'OriginationPhase': 'Coding',
  'Priority': 'High',
  'Status': 'Completed',
  'issue_number': 22},
 '_type': 'github repo issues'}
```

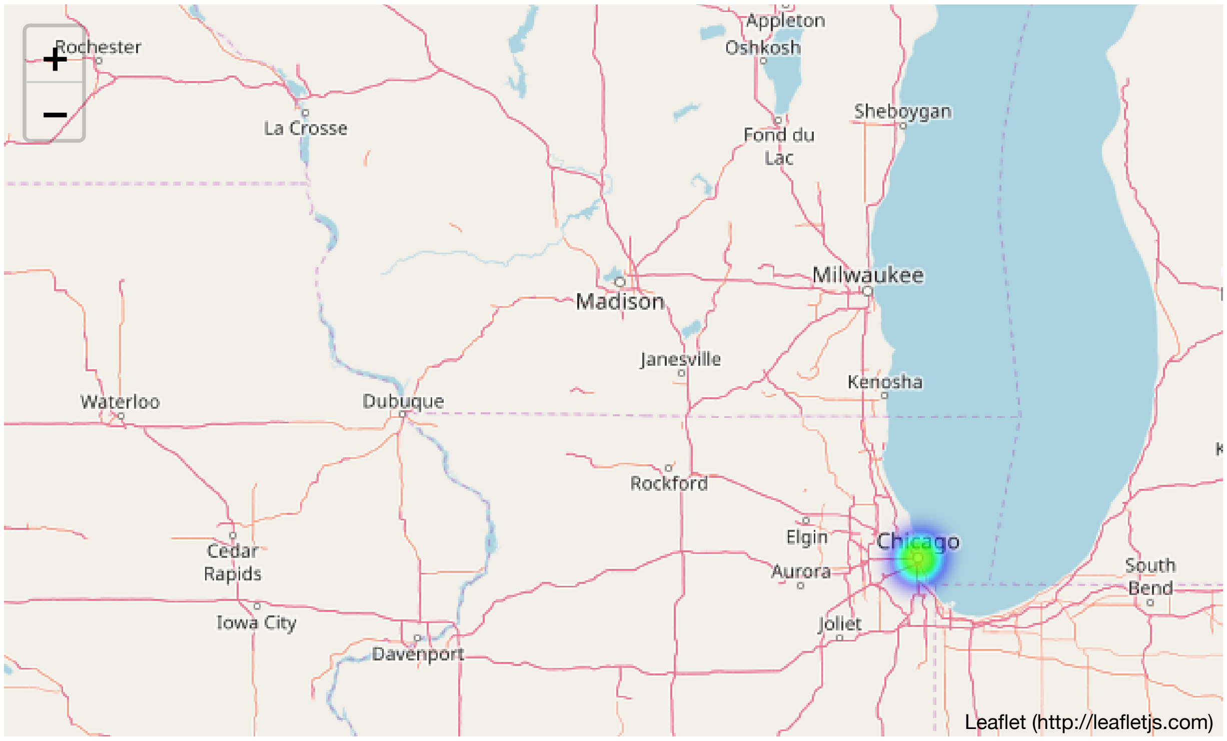
In [21]:
```
output2_2Lat_Long = []
exists = False
for i in (output2_2['hits']['hits']):
    temp = []
    try:
        temp.append(float(i['_source']['Latitude']))
        temp.append(float(i['_source']['Longitude']))
        output2_2Lat_Long.append(temp)
    except:
        pass
```

In [22]:
```
output2_2Lat_Long
```

Out[22]:
```
[[40.170101, -92.177847],
 [41.891551, -87.607375],
 [41.89323, -87.617419],
 [41.853136, -87.63316]]
```

In [23]:
```
output2_2HeatMap = folium.Map([41.891551, -87.607375], zoom_start=16)
output2_2HeatMap.add_child(plugins.HeatMap(output2_2Lat_Long, radius = 15))
```

Out[23]:



In [24]:
```
query2_3= { #Second query that queries elastic search that returns all the issu
es that have label DetectionPhase as Field AND Priority is Critical AND Status
 as Completed
    'size' : 500,
    'query' : {
        'bool' : {
            'must' : [{'match' : {'DetectionPhase' : 'Field'}},
                      {'match' : {'Priority' : 'Critical'}},
                      {'match' : {'Status' : 'Approved'}}
                     ]
        }
    }
}
```

In [25]:
```
output2_3 = es.search(index = 'issues_fin', body = query2_3, scroll = '1h')
```

In [26]:
```
print(output2_3.keys()) #Prints all the keys in the output2_3 dictionary
output2_3['hits']['hits'][0] #Similar output for rest of the issues.
```

```
dict_keys(['_scroll_id', 'took', 'timed_out', '_shards', 'hits'])
```

Out[26]:
```
{'_id': '26',
 '_index': 'issues_fin',
 '_score': 3.0965915,
 '_source': {'Address': '1951 N WESTERN AVE',
  'Author': 'HSP18SCM69D',
  'Category': 'Enhancement',
  'DetectionPhase': 'Field',
  'Latitude': '41.917164',
  'Longitude': '-87.686965',
  'OriginationPhase': 'Requirements',
  'Priority': 'Critical',
  'Status': 'Approved',
  'issue_number': 26},
 '_type': 'github repo issues'}
```

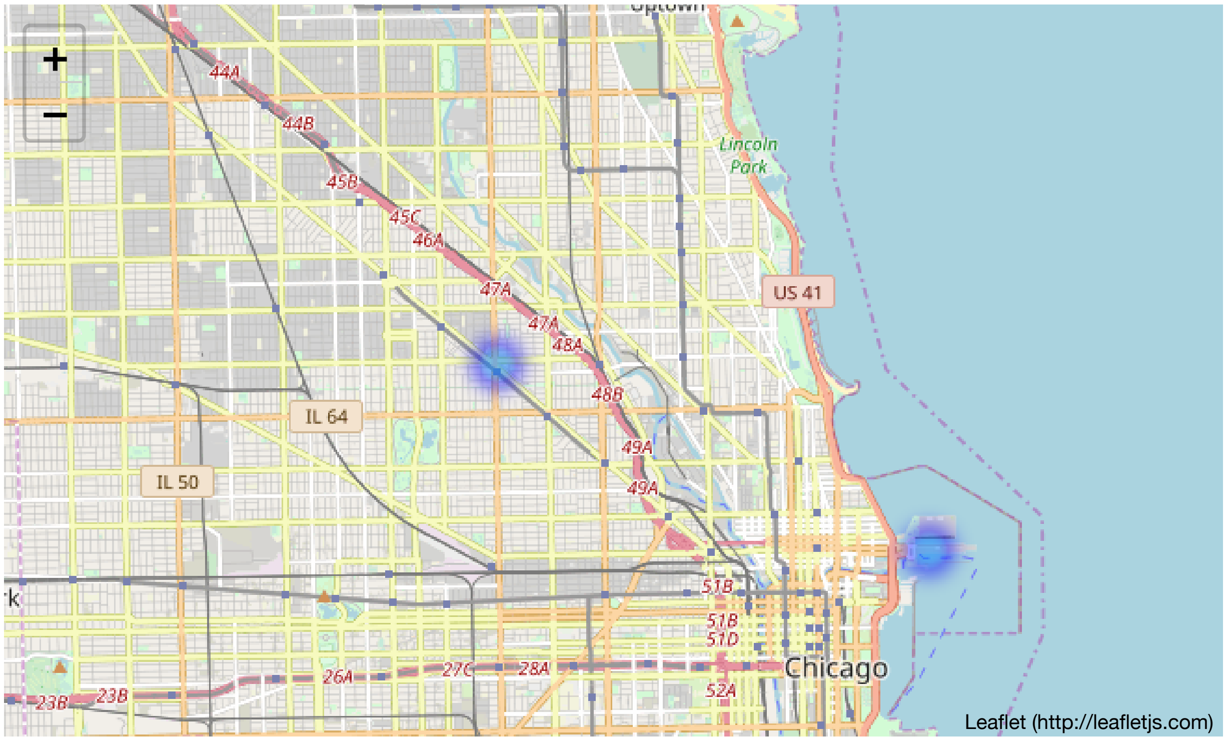
In [27]:
```
output2_3Lat_Long = []
exists = False
for i in (output2_3['hits']['hits']):
    temp = []
    try:
        temp.append(float(i['_source']['Latitude']))
        temp.append(float(i['_source']['Longitude']))
        output2_3Lat_Long.append(temp)
    except:
        pass
```

In [28]:
```
output2_3Lat_Long
```

Out[28]:
```
[[41.917164, -87.686965], [41.891551, -87.607375]]
```

```
In [29]: output2_3HeatMap = folium.Map([41.891551, -87.607375], zoom_start=16)
         output2_3HeatMap.add_child(plugins.HeatMap(output2_3Lat_Long, radius = 15))
```

Out[29]:



```
In [30]: query2_4= {#Second query that queries elastic search that returns all the issue
         s that have label DetectionPhase as Field AND Priority is Critical OR High AND
          Status as Completed OR inProgress
             'size' : 500,
             'query' : {
                 'bool' : {
                     'must' : [{'match' : {'DetectionPhase' : 'Field'}},
                               {'match' : {'Priority' : 'Critical OR High'}}, #OR here i
         s logical OR
                               {'match' : {'Status' : 'Approved OR inProgress'}}
                              ]
                 }
             }
         }
```

```
In [31]: output2_4 = es.search(index = 'issues_fin', body = query2_4, scroll = '1h')
```

```
In [32]: print(output2_4.keys()) #Prints all the keys in the output2_4 dictionary
         output2_4['hits']['hits'][0] #Similar output for rest of the issues.

         dict_keys(['_scroll_id', 'took', 'timed_out', '_shards', 'hits'])

Out[32]: {'_id': '475',
          '_index': 'issues_fin',
          '_score': 5.081758,
          '_source': {'Author': 'HSP18SCM50W',
           'Category': 'Inquiry',
           'DetectionPhase': 'Field',
           'OriginationPhase': 'Coding',
           'Priority': 'High',
           'Status': 'inProgress',
           'issue_number': 475},
          '_type': 'github repo issues'}
```

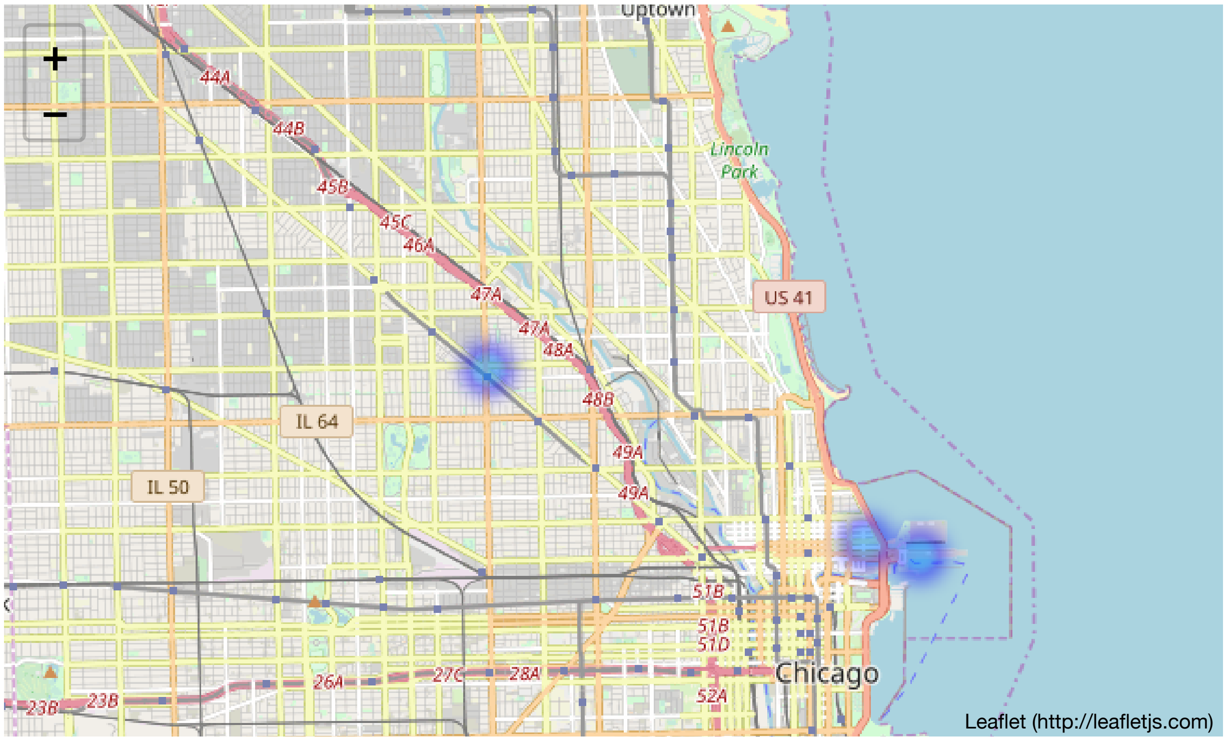
```
In [33]: output2_4Lat_Long = []
         exists = False
         for i in (output2_4['hits']['hits']):
             temp = []
             try:
                 temp.append(float(i['_source']['Latitude']))
                 temp.append(float(i['_source']['Longitude']))
                 output2_4Lat_Long.append(temp)
             except:
                 pass
```

```
In [34]: output2_4Lat_Long
```

```
Out[34]: [[41.853136, -87.63316],
          [41.89323, -87.617419],
          [41.917164, -87.686965],
          [41.891551, -87.607375]]
```

```
In [35]: output2_4HeatMap = folium.Map([41.891551, -87.607375], zoom_start=16)
         output2_4HeatMap.add_child(plugins.HeatMap(output2_4Lat_Long, radius = 15))
```

Out[35]:



In [36]:
```
query2_5 = { #Query structure referenced from tutorial provided
    'aggs' : {
        'selected_data' : {
            'terms' : {
                'field' : 'Latitude.keyword', #Search for issues with same Lati
tude and Longitude
                'field' : 'Longitude.keyword',
                'min_doc_count' : 5, #Set minimum hit count as 5
                'size' : 500
            },
            'aggs' : {
                'accurate_hits' : {
                    'top_hits' : {
                        'size' : 10
                    }
                }
            }
        }
    }
}
```

In [37]:
```
output2_5 = es.search(index = 'issues_fin', body = query2_5, scroll = '1h')
```

In [38]:
```
print(output2_5.keys()) #Prints all the keys in the output1 dictionary
print(output2_5['aggregations'].keys())
print(output2_5['aggregations']['selected_data'].keys())
#print(output2_5['aggregations']['selected_data']['buckets']) #The data is insi
de dictionary 'accurate_hits'
output2_5['hits']['hits'][0]
#Similar output for rest of the issues.
```

```
dict_keys(['_scroll_id', 'took', 'timed_out', '_shards', 'hits', 'aggregation
s'])
dict_keys(['selected_data'])
dict_keys(['doc_count_error_upper_bound', 'sum_other_doc_count', 'buckets'])
```

Out[38]:
```
{'_id': '470',
 '_index': 'issues_fin',
 '_score': 1.0,
 '_source': {'Author': 'RSP18SCM19N',
  'Category': 'Enhancement',
  'DetectionPhase': 'Testing',
  'OriginationPhase': 'Design',
  'Priority': 'Major',
  'Status': 'Completed',
  'issue_number': 470},
 '_type': 'github repo issues'}
```

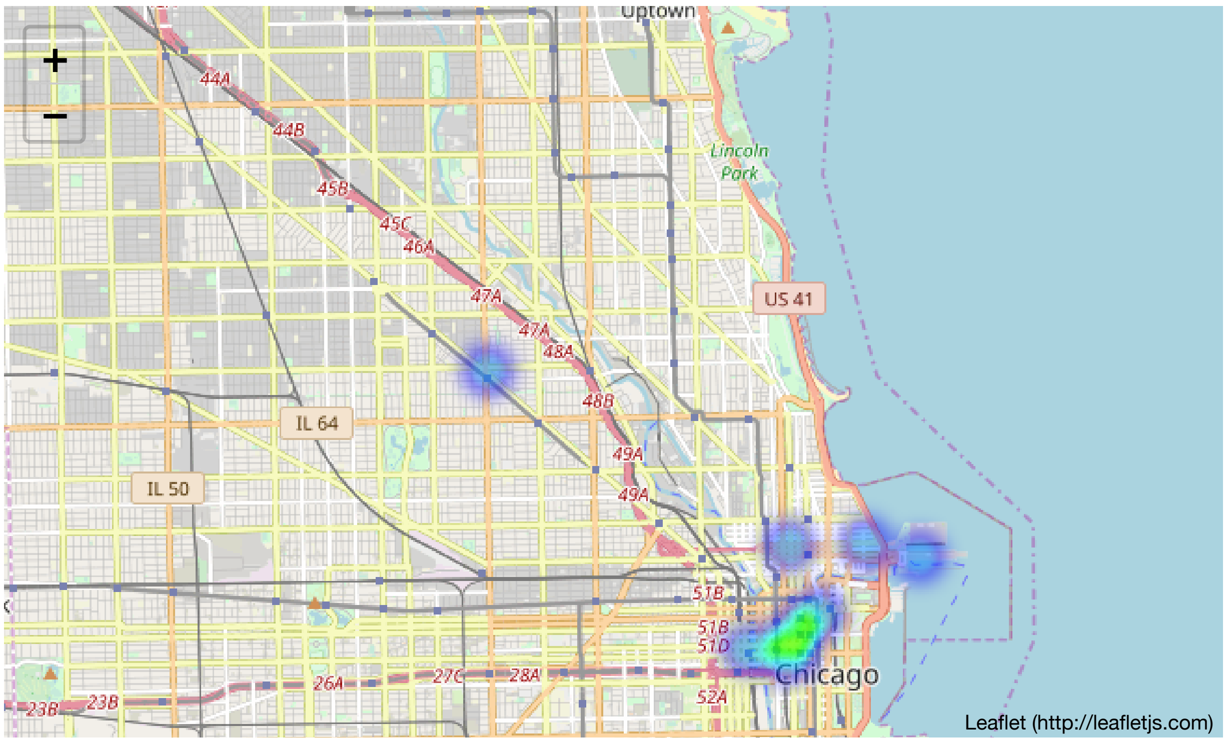
In [39]:
```
output2_5Lat_Long = []
for i in (output2_5['aggregations']['selected_data']['buckets']): #Structure of
 elastic search dictionary, this is the key
    try:
        for j in i['accurate_hits']['hits']['hits'] : #this the value to access
            temp1 = []
            temp1.append(float(j['_source']['Latitude']))
            temp1.append(float(j['_source']['Longitude']))
        output2_5Lat_Long.append(temp1)
    except:
        pass
```

```
In [40]: output2_5Lat_Long

Out[40]: [[41.847095, -87.616767],
          [41.877817, -87.631247],
          [41.883772, -87.625962],
          [41.852623, -87.611958],
          [41.89323, -87.617419],
          [41.880982, -87.630553],
          [41.877846, -87.631296],
          [41.891551, -87.607375],
          [41.89302, -87.631556],
          [41.853136, -87.63316],
          [41.917164, -87.686965],
          [41.878693, -87.638924],
          [41.838897, -87.646804],
          [40.170101, -92.177847]]
```

```
In [41]: output2_5HeatMap = folium.Map([41.891551, -87.607375], zoom_start=16)
         output2_5HeatMap.add_child(plugins.HeatMap(output2_5Lat_Long, radius = 15))

Out[41]:
```



Uptown
Lincoln Park
US 41
IL 64
IL 50
Chicago
44A
44B
45B
45C
46A
47A
48A
48B
49A
51B
51D
52A
23B
26A
27C
28A
Leaflet (http://leafletjs.com)