Background: This project focuses on analyzing 311 data in Los Angeles County for HackForLA. 311 is a local non-emergency service available for anyone to use. They deal with things such as mattresses on the side of the street or broken stoplights. Each 311 request should be allocated to a neighborhood council, subset of LA county, in which the issue is present.

Issue: In the 311 requests, there are certain rows of the dataset which are lacking a council name. Neighborhood councils are advisory bodies that make sure the government is acting on behalf of the communities. This project will explore the requests lacking council names.

In [ ]:
```
#Download ultilized libraries
import requests
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import folium
import datetime
import seaborn as sns
import matplotlib.ticker as mtick
```

In [7]:
```
#Define function to extract data from API
import argparse

REQUESTS_BATCH_SIZE = 10000


def get_311_request_data(start_date, end_date):
    """Fetches 311 requests from the 311 data server.
    Retreives 311 requests from the 311 data server for a given start_
    Args:
        start_date: The date from which the 311 request data have to b
        end_date: The date upto which the 311 request data have to be
    Return:
        Dataframe data_final is returned with 15 columns. The datafram
    """

    skip = 0
    all_requests = []
    while True:
        url = f'https://dev-api.311-data.org/requests?start_date={star
        response = requests.get(url)
        data = response.json()
        all_requests.extend(data)
        skip += REQUESTS_BATCH_SIZE
        if len(data) < REQUESTS_BATCH_SIZE:
            break
```

```
    data_final = pd.DataFrame(all_requests)
    data_final.sort_values(by='createdDate', inplace=True, ignore_inde
    return data_final


def main():
    """Prints out the preview of the dataframe data_final in the comma
    The result is written to a csv file and saved in the current worki
    """

    parser = argparse.ArgumentParser(
        description='Gets 311 request data from the server')
    parser.add_argument('start_date', type=str,
                        help='The start date that has to be entered')
    parser.add_argument('end_date', type=str,
                        help='The end data that has to be entered')
    args = parser.parse_args()
    start_date = args.start_date
    end_date = args.end_date
    data_final = get_311_request_data(start_date, end_date)
    data_final.to_csv('data_final.csv')
    print(data_final)


if __name__ == "__main__":
    main()
```

```
usage: ipykernel_launcher.py [-h] start_date end_date
ipykernel_launcher.py: error: the following arguments are required: e
nd_date

An exception has occurred, use %tb to see the full traceback.

SystemExit: 2



/Users/zak/.local/lib/python3.10/site-packages/IPython/core/interacti
veshell.py:3386: UserWarning: To exit: use 'exit', 'quit', or Ctrl-D.
  warn("To exit: use 'exit', 'quit', or Ctrl-D.", stacklevel=1)
```

In [8]:
```
#The following cells extract 311 request data for 2022
data = get_311_request_data(datetime.date(2022, 1, 1), datetime.date(2
```

In [9]:
```
df = get_311_request_data(datetime.date(2022, 2, 21), datetime.date(20
data = data.append(df)
```

```
/var/folders/2t/4c6543550d10qqbs4q41__600000gn/T/ipykernel_6850/40539
25863.py:2: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.concat in
stead.
  data = data.append(df)
```

In [10]:
```
df = get_311_request_data(datetime.date(2022, 4, 21), datetime.date(20
data = data.append(df)
```

```
/var/folders/2t/4c6543550d10qqbs4q41__600000gn/T/ipykernel_6850/10208
35562.py:2: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.concat in
stead.
  data = data.append(df)
```

In [11]:
```
df = get_311_request_data(datetime.date(2022, 6, 21), datetime.date(20
data = data.append(df)
```

```
/var/folders/2t/4c6543550d10qqbs4q41__600000gn/T/ipykernel_6850/29285
4740.py:2: FutureWarning: The frame.append method is deprecated and w
ill be removed from pandas in a future version. Use pandas.concat ins
tead.
  data = data.append(df)
```

In [12]:
```
df = get_311_request_data(datetime.date(2022, 8, 21), datetime.date(20
data = data.append(df)
df = get_311_request_data(datetime.date(2022, 10, 21), datetime.date(2
data = data.append(df)
```

```
/var/folders/2t/4c6543550d10qqbs4q41__600000gn/T/ipykernel_6850/13686
11529.py:2: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.concat in
stead.
  data = data.append(df)
/var/folders/2t/4c6543550d10qqbs4q41__600000gn/T/ipykernel_6850/13686
11529.py:4: FutureWarning: The frame.append method is deprecated and
will be removed from pandas in a future version. Use pandas.concat in
stead.
  data = data.append(df)
```

In [2]:
```
#Data is saved to a CSV for easy access
data.to_csv('/Users/zak/Downloads/311data2022.csv')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In [2], line 3
      1 ##response2022 = requests.get('https://dev-api.311-data.org/requests?start_date=2022-01-01&end_date=2022-12-31&limit=50000')
      2 #print(response2022)
----> 3 data.to_csv('/Users/zak/Downloads/311data2022.csv')

NameError: name 'data' is not defined
```

In [16]:
```
#Data is explored to ensure consistency
data = pd.read_csv('/Users/zak/Downloads/311data2022.csv')
data.tail()
```

Out[16]:

	Unnamed: 0	requestId	srnumber	councilId	councilName	typeId	typeName	ager
1080202	225162	11305203	1-3202588721	34	Harbor Gateway North	4	Bulky Items	
1080203	225163	11305204	1-3202596921	34	Harbor Gateway North	4	Bulky Items	
1080204	225164	11305205	1-3202601171	48	Mar Vista	2	Homeless Encampment	
1080205	225165	11305206	1-3202598501	96	Wilshire Center - Koreatown	4	Bulky Items	
1080206	225166	11305207	1-3202597761	27	Granada Hills North	4	Bulky Items	

In [17]:
```
#Subset data to only include request without councils
data2022nc = data[data['councilName'] == 'No council']
```

In [6]:
```
##Add map of neighborhood council downloaded from https://geohub.lacit
council = pd.read_csv('/Users/zak/Downloads/Neighborhood_Councils_(Cer
council.head()
```

Out[6]:

	OBJECTID	NAME	WADDRESS	DWEBSITE	
0	1	ARLETA NC	http://www.arletanc.org/	http://empowerla.org/ANC	ANC@En
1	2	ARROYO SECO NC	http://www.asnc.us/	http://empowerla.org/ASNC	ASNC@En
2	3	ARTS DISTRICT LITTLE TOKYO NC	http://www.hcncla.org/	http://empowerla.org/HCNC	HCNC@En
3	4	ATWATER VILLAGE NC	http://www.atwatervillage.org/	http://empowerla.org/AVNC	AVNC@En
4	5	BEL AIR-BEVERLY	http://bahana.org/	http://empowerla.org/BABCNC	BABCNC@Er

In [7]:
```
#load useful mapping library
import geopandas as gpd
```

In [8]:
```
##From neighborhood council map shape file, output the map
maps = gpd.GeoDataFrame.from_file('/Users/zak/Downloads/Neighborhood_C
```

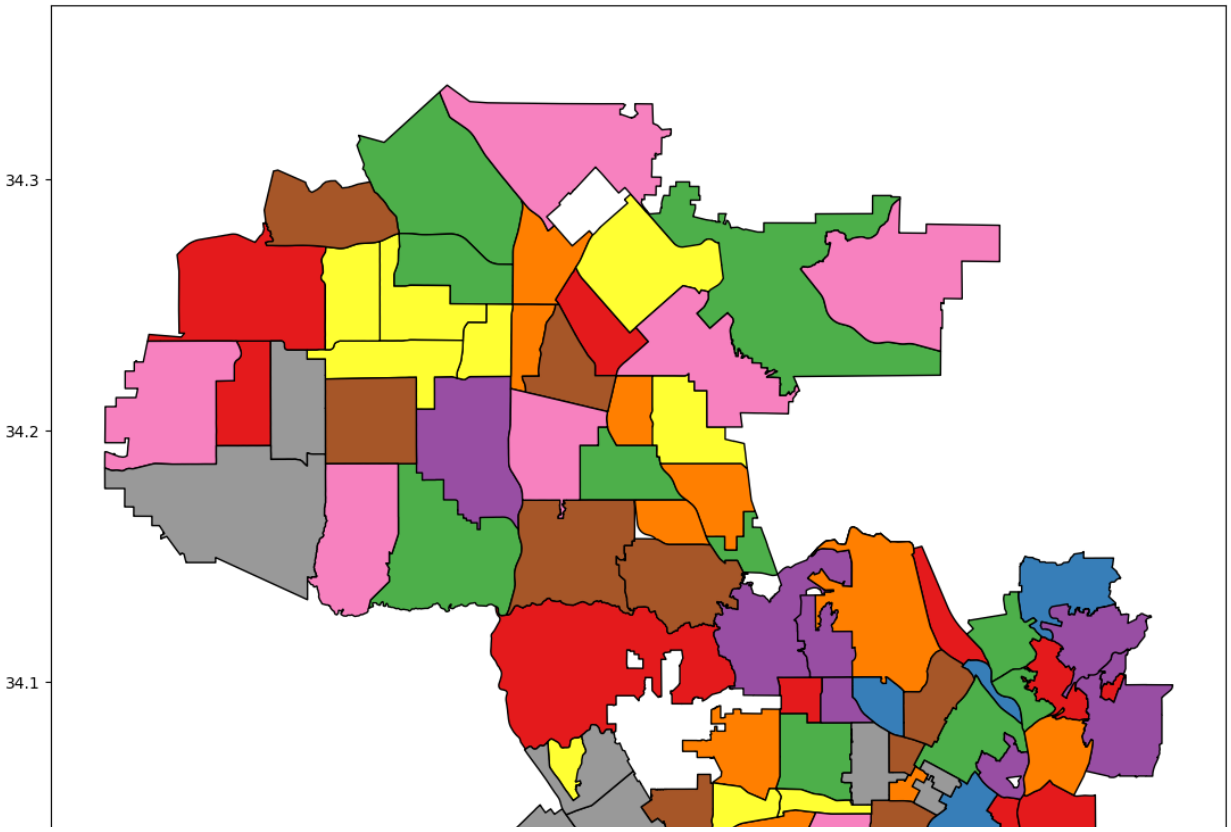
In [9]:
```
maps.head()
```

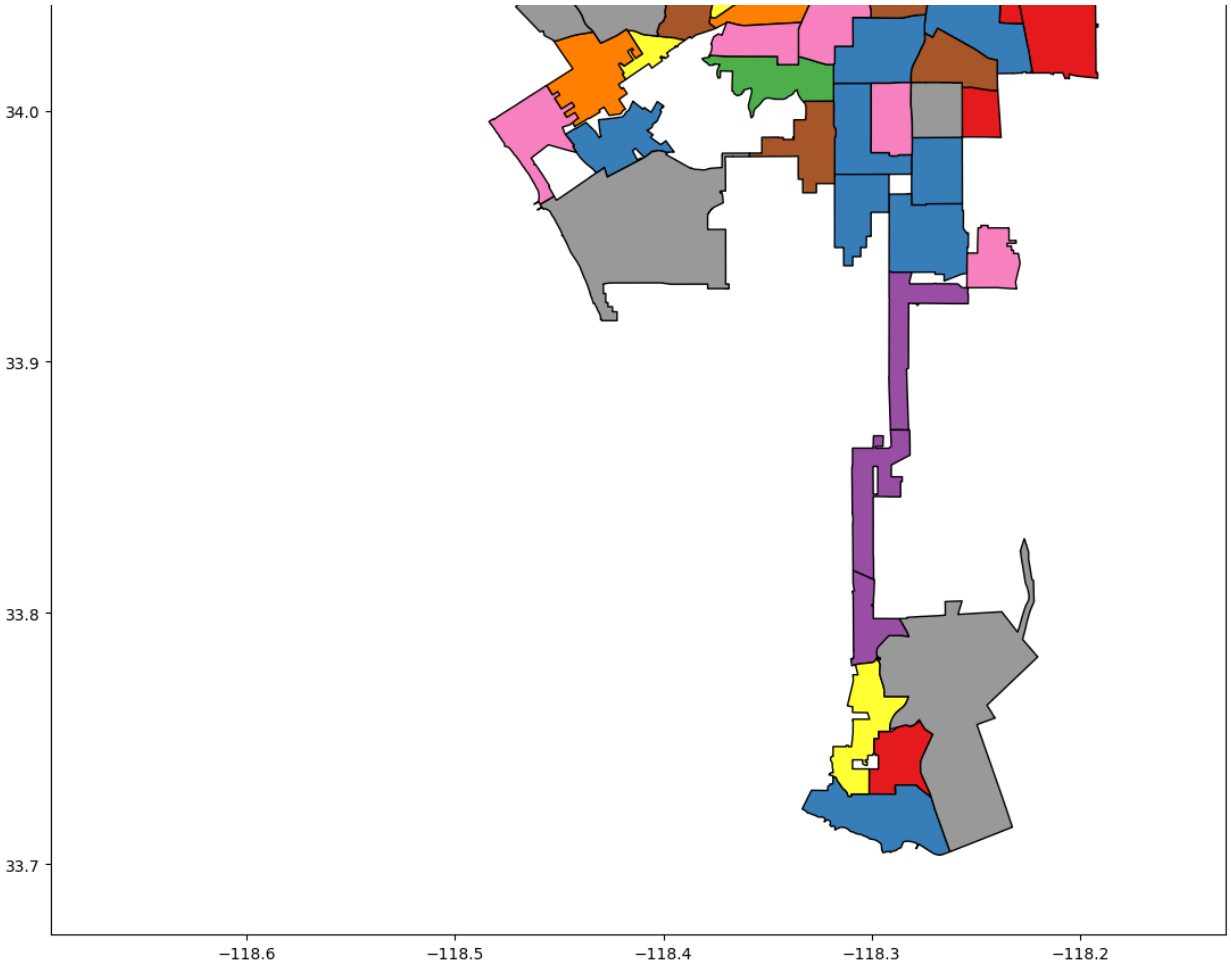
Out[9]:

	OBJECTID	NAME	WADDRESS	DWEBSITE	
0	1	ARLETA NC	http://www.arletanc.org/	http://empowerla.org/ANC	ANC@En
1	2	ARROYO SECO NC	http://www.asnc.us/	http://empowerla.org/ASNC	ASNC@En
2	3	ARTS DISTRICT LITTLE TOKYO NC	http://www.hcncla.org/	http://empowerla.org/HCNC	HCNC@En
3	4	ATWATER VILLAGE	http://www.atwatervillage.org/	http://empowerla.org/AVNC	AVNC@En

In [10]:
```
#Plot the map to make sure it works
maps.plot(color = 'white', edgecolor = 'black', cmap = 'Set1', figsize
```

Out[10]: <AxesSubplot: >




34.0
33.9
33.8
33.7
−118.6
−118.5
−118.4
−118.3
−118.2

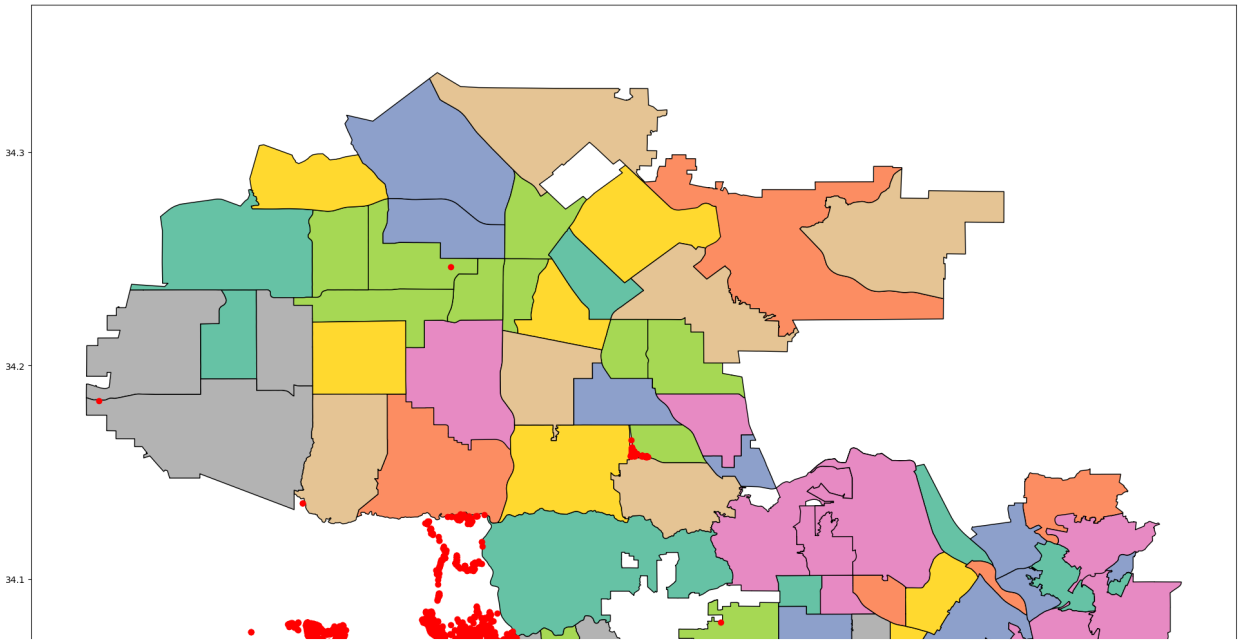
In [11]:
```
#Convert request longitude and latitude into points for plotting
data_points = gpd.GeoDataFrame(
    data2022nc, geometry=gpd.points_from_xy(data2022nc.longitude, data
data_points.head()
```

Out[11]:

	Unnamed: 0	requestId	srnumber	councilId	councilName	typeId	typeName	agencyId
137	137	9728261	1-2156024681	0	No council	4	Bulky Items	3
160	160	9269717	1-2156309901	0	No council	2	Homeless Encampment	3
167	167	10613137	1-2156371221	0	No council	8	Single Streetlight	1
170	170	9442913	1-2156385411	0	No council	8	Single Streetlight	1
195	195	11315108	1-2156515861	0	No council	1	Graffiti	4

In [12]:
```
#Add data points to map shape file
NC_plot = maps.plot(color = 'white', edgecolor = 'black', cmap = 'Set2
data_points.plot(ax = NC_plot, color = 'red', marker = 'o')
plt.show
```

Out[12]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



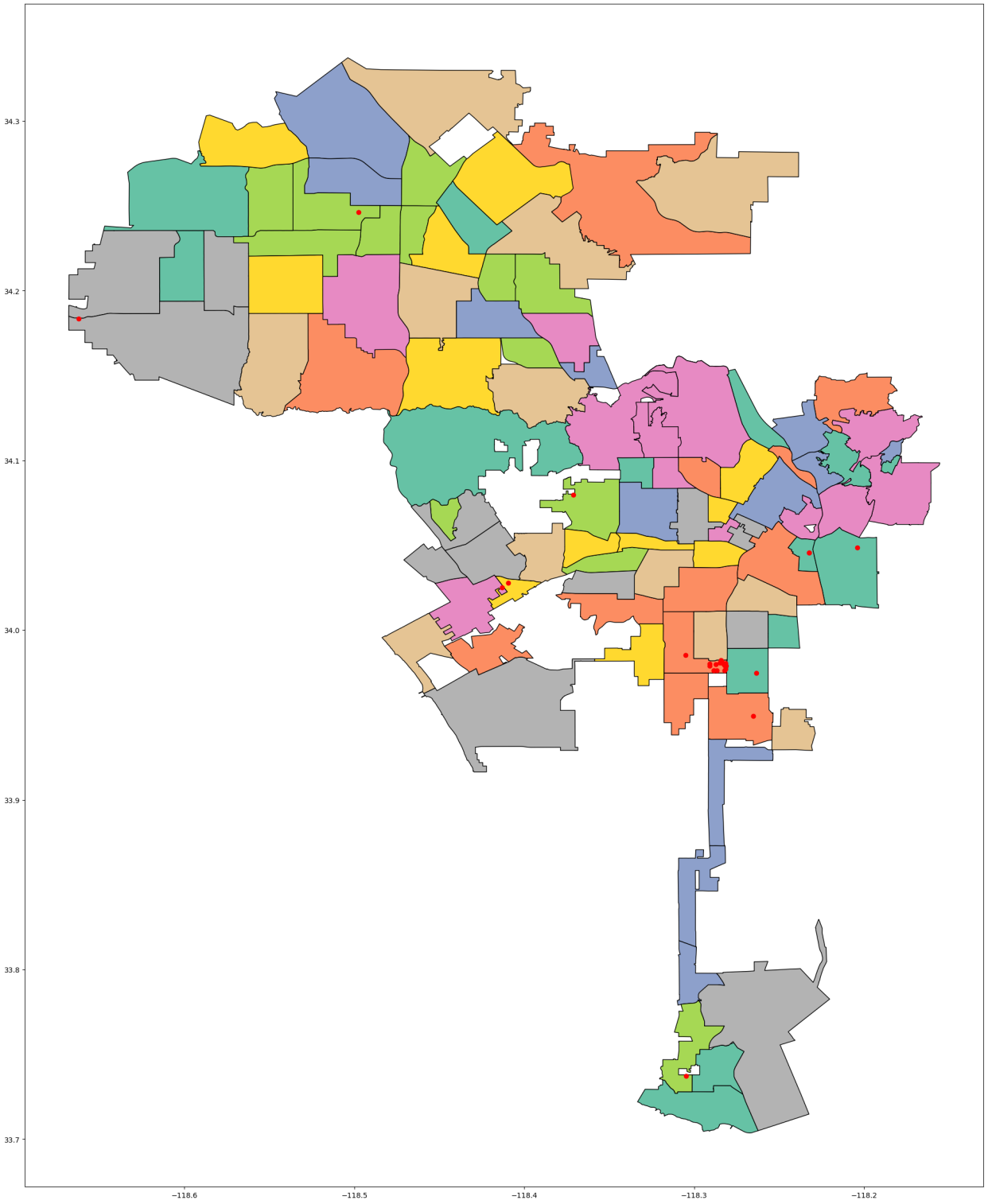
From the map above it seems as though there are some 311 request that are not within the LA County Neighborhood Councils.

In [13]:
```
#Extract only requests that should have a Neighborhood Council, based
mismarked_requests = data_points.sjoin(maps, how="inner", predicate='i
```

```
/Users/zak/opt/anaconda3/envs/geo_env/lib/python3.10/site-packages/ge
opandas/geodataframe.py:2090: UserWarning: CRS mismatch between the C
RS of left geometries and the CRS of right geometries.
Use `to_crs()` to reproject one of the input geometries to match the
CRS of the other.

Left CRS: None
Right CRS: EPSG:4326

  return geopandas.sjoin(left_df=self, right_df=df, *args, **kwargs)
```

In [29]:
```
#Plot the requests that are within a council but don't have one in the
NC_plot = maps.plot(color = 'white', edgecolor = 'black', cmap = 'Set2
mismarked_requests.plot(ax = NC_plot, color = 'red', marker = 'o')
plt.show
```

Out[29]: <function matplotlib.pyplot.show(close=None, block=None)>



In [14]:
```
#Explore these "mismarked requests"
mismarked_requests
```

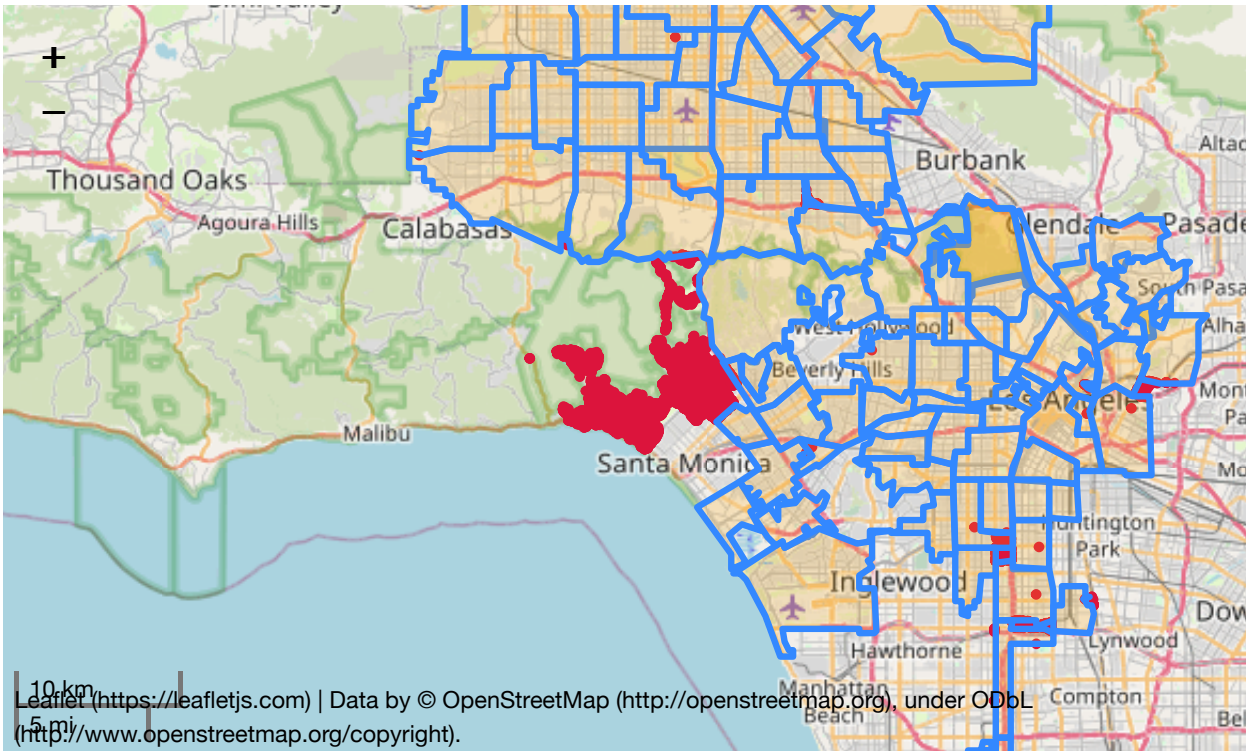
Out[14]:

	Unnamed: 0	requestId	srnumber	councilId	councilName	typeId	typeName	a
137	137	9728261	1-2156024681	0	No council	4	Bulky Items	
167	167	10613137	1-2156371221	0	No council	8	Single Streetlight	
804	804	9725120	1-2161105331	0	No council	4	Bulky Items	
1305	1305	9638271	1-2163604821	0	No council	1	Graffiti	

In [57]:
```
#The following cell plot the requests using Folium (to better control
map = folium.Map(location=[34, -118.4], zoom_start=10, control_scale=T
```

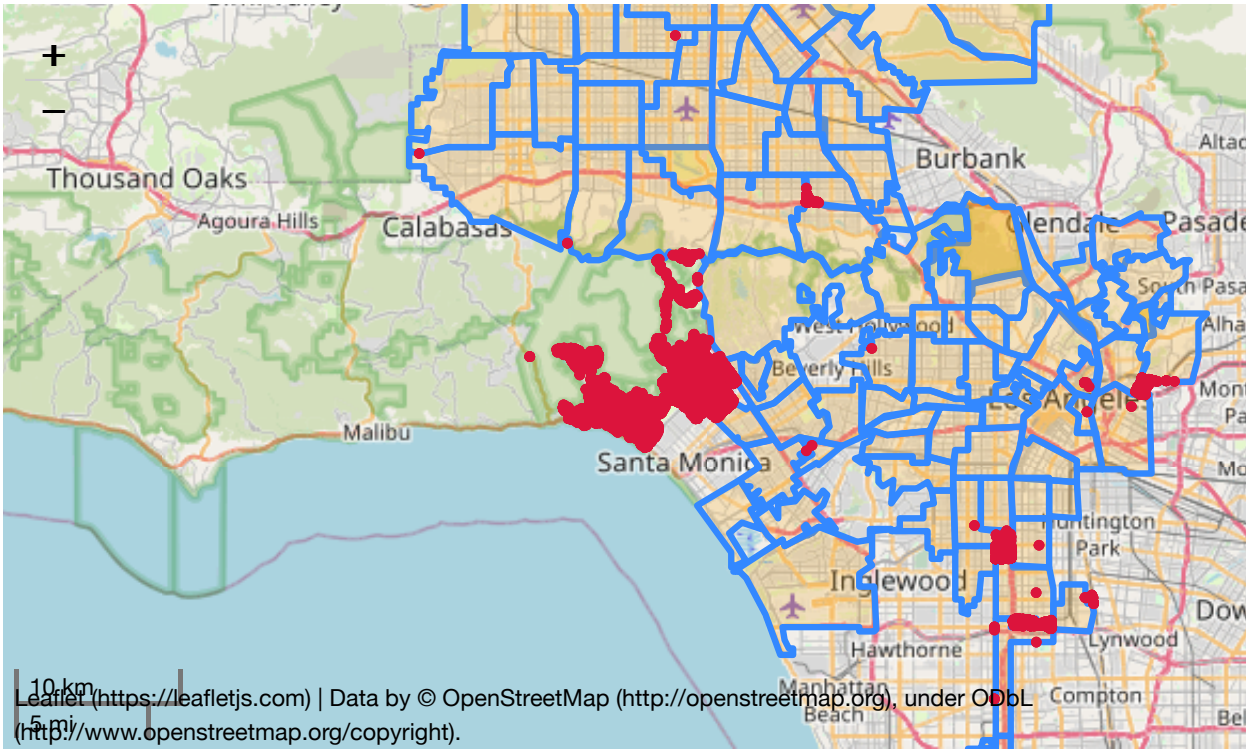
In [25]:
```
##output Folium map
for _, r in maps.iterrows():
    # Without simplifying the representation of each borough,
    # the map might not be displayed
    sim_geo = gpd.GeoSeries(r['geometry']).simplify(tolerance=0.001)
    geo_j = sim_geo.to_json()
    geo_j = folium.GeoJson(data=geo_j,
                           style_function=lambda x: {'fillColor': 'ora
    folium.Popup(r['SERVICE_RE']).add_to(geo_j)
    geo_j.add_to(map)
map
```

Out[25]:



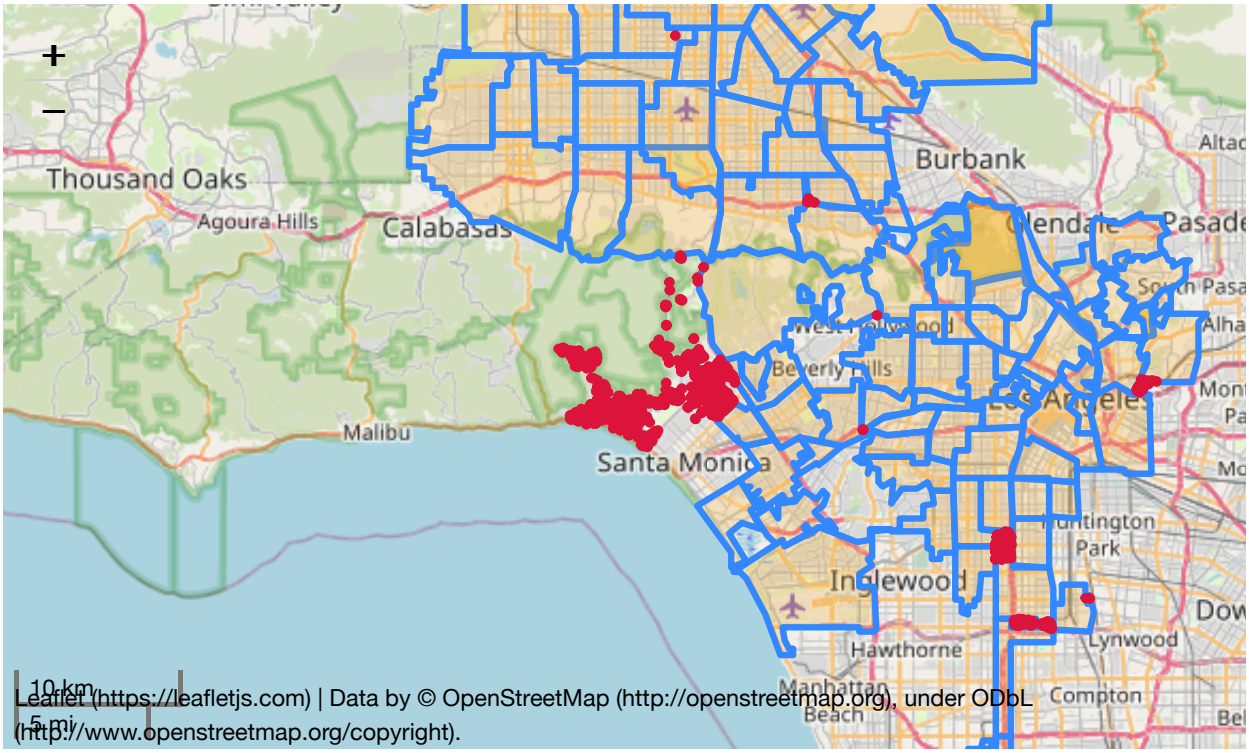
In [28]:
```
#Add points(requests) to Folium map
for i in range(0,len(data2022nc)):

    folium.Circle(
        radius=50,
        location=[data2022nc.iloc[i]['latitude'], data2022nc.iloc[i]['
        popup=data2022nc.iloc[i]['typeName'],
        color=["crimson"],
        fill=True,
    ).add_to(map)
map
```

Out[28]:



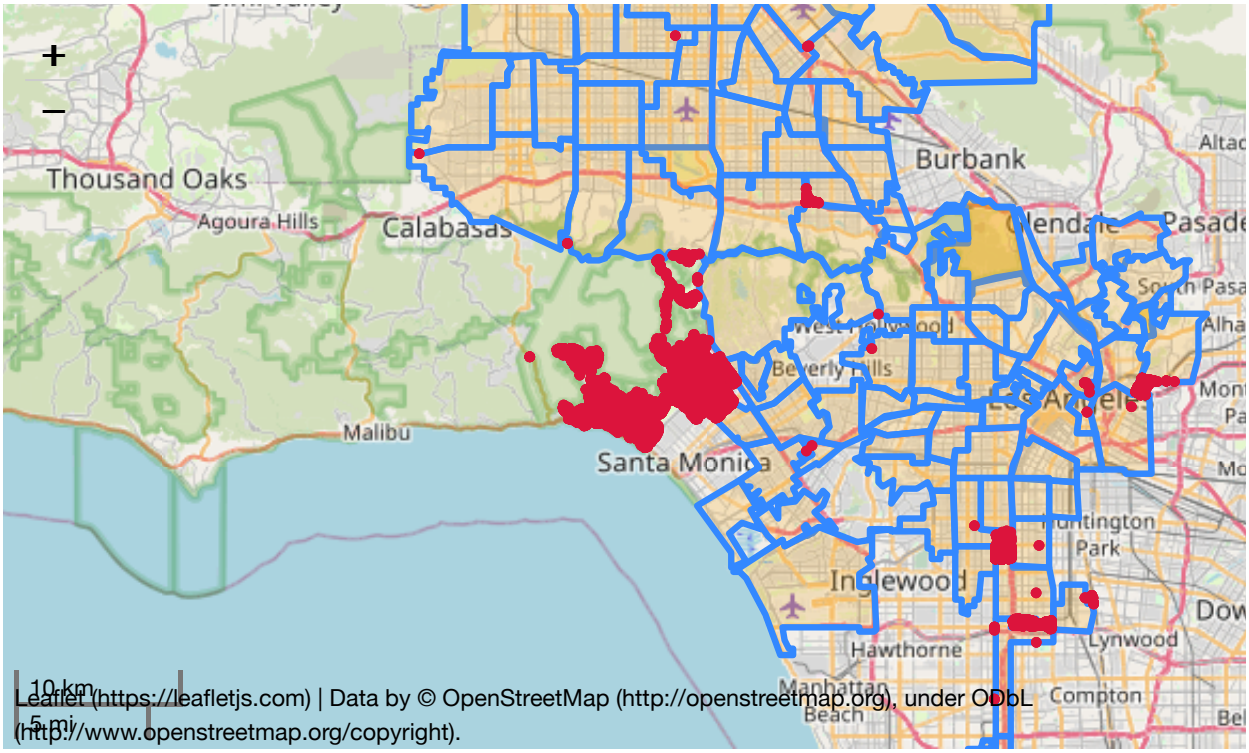
In [29]:
```
#repeat for alternative years (2020)
response2020 = requests.get('https://dev-api.311-data.org/requests?sta
data2020 = pd.json_normalize(response2020.json())
data2020 = data2020[data2020['councilName'] == 'No council']
map2 = folium.Map(location=[34, -118.4], zoom_start=10, control_scale=
for _, r in maps.iterrows():
    # Without simplifying the representation of each borough,
    # the map might not be displayed
    sim_geo = gpd.GeoSeries(r['geometry']).simplify(tolerance=0.001)
    geo_j = sim_geo.to_json()
    geo_j = folium.GeoJson(data=geo_j,
                           style_function=lambda x: {'fillColor': 'ora
    folium.Popup(r['SERVICE_RE']).add_to(geo_j)
    geo_j.add_to(map2)
for i in range(0,len(data2020)):

    folium.Circle(
        radius=50,
        location=[data2020.iloc[i]['latitude'], data2020.iloc[i]['long
        popup=data2020.iloc[i]['typeName'],
        color="crimson",
        fill=True,
    ).add_to(map)
map
```

Out[29]:



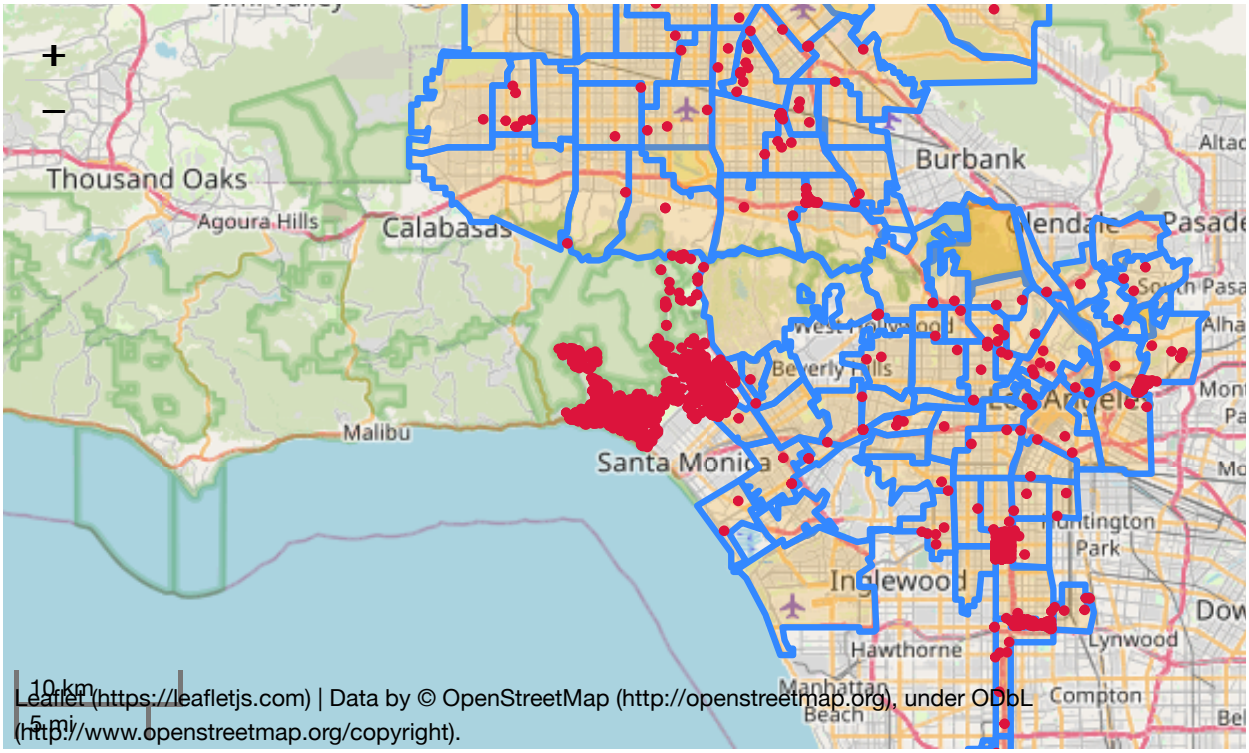
In [30]:
```
##(2018)
response2018 = requests.get('https://dev-api.311-data.org/requests?sta
data2018 = pd.json_normalize(response2018.json())
data2018 = data2018[data2018['councilName'] == 'No council']
map2 = folium.Map(location=[34, -118.4], zoom_start=10, control_scale=
for _, r in maps.iterrows():
    # Without simplifying the representation of each borough,
    # the map might not be displayed
    sim_geo = gpd.GeoSeries(r['geometry']).simplify(tolerance=0.001)
    geo_j = sim_geo.to_json()
    geo_j = folium.GeoJson(data=geo_j,
                           style_function=lambda x: {'fillColor': 'ora
    folium.Popup(r['SERVICE_RE']).add_to(geo_j)
    geo_j.add_to(map2)
for i in range(0,len(data2018)):

    folium.Circle(
        radius=50,
        location=[data2018.iloc[i]['latitude'], data2018.iloc[i]['long
        popup=data2018.iloc[i]['typeName'],
        color="crimson",
        fill=True,
    ).add_to(map)
map
```

Out[30]:



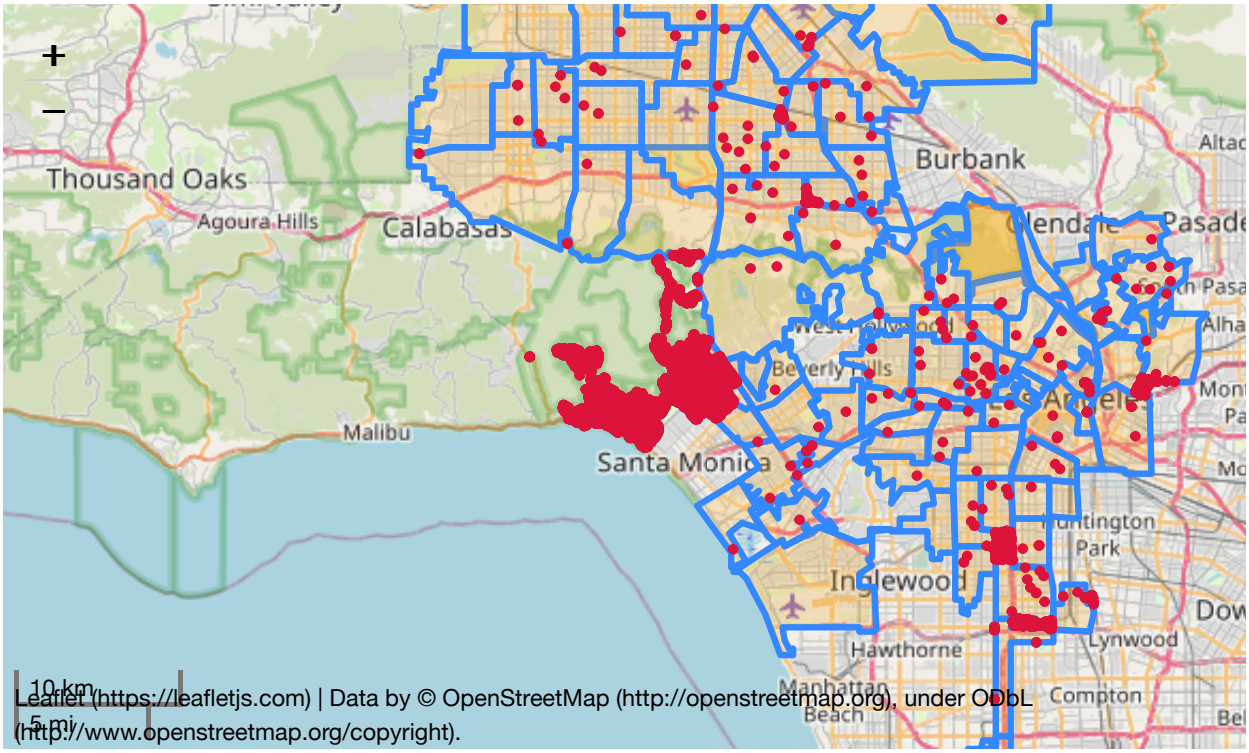
In [31]:
```
##(2016)
response2016 = requests.get('https://dev-api.311-data.org/requests?sta
data2016 = pd.json_normalize(response2016.json())
data2016 = data2016[data2016['councilName'] == 'No council']
map2 = folium.Map(location=[34, -118.4], zoom_start=10, control_scale=
for _, r in maps.iterrows():
    # Without simplifying the representation of each borough,
    # the map might not be displayed
    sim_geo = gpd.GeoSeries(r['geometry']).simplify(tolerance=0.001)
    geo_j = sim_geo.to_json()
    geo_j = folium.GeoJson(data=geo_j,
                           style_function=lambda x: {'fillColor': 'ora
    folium.Popup(r['SERVICE_RE']).add_to(geo_j)
    geo_j.add_to(map2)
for i in range(0,len(data2016)):

    folium.Circle(
        radius=50,
        location=[data2016.iloc[i]['latitude'], data2016.iloc[i]['long
        popup=data2016.iloc[i]['typeName'],
        color="crimson",
        fill=True,
    ).add_to(map)
map
```

Out[31]:



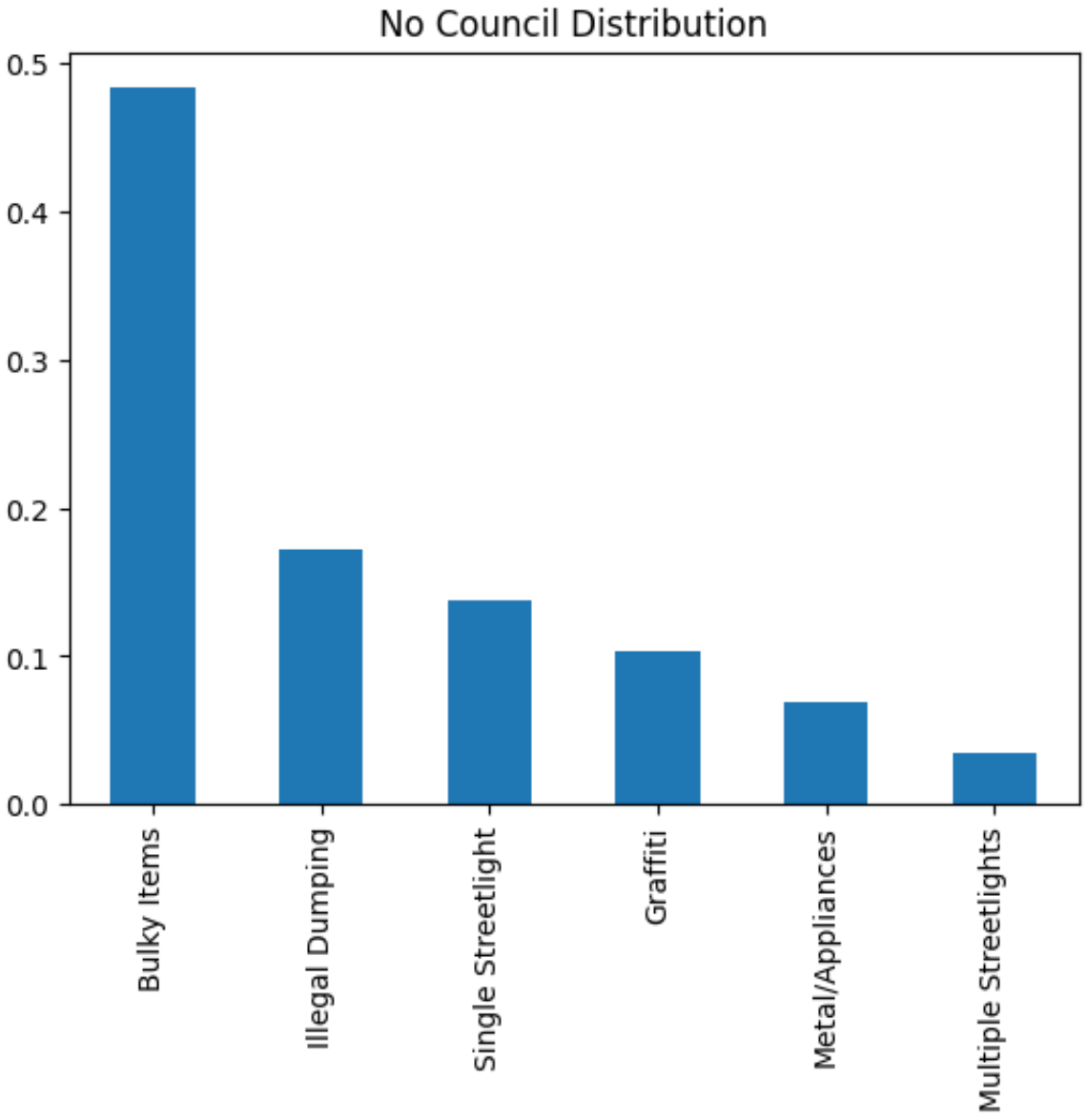
In [31]:
```
##(2015)
response2015 = requests.get('https://dev-api.311-data.org/requests?sta
data2015 = pd.json_normalize(response2015.json())
data2015 = data2015[data2015['councilName'] == 'No council']
map2 = folium.Map(location=[34, -118.4], zoom_start=10, control_scale=
for _, r in maps.iterrows():
    # Without simplifying the representation of each borough,
    # the map might not be displayed
    sim_geo = gpd.GeoSeries(r['geometry']).simplify(tolerance=0.001)
    geo_j = sim_geo.to_json()
    geo_j = folium.GeoJson(data=geo_j,
                           style_function=lambda x: {'fillColor': 'ora
    folium.Popup(r['SERVICE_RE']).add_to(geo_j)
    geo_j.add_to(map2)
for i in range(0,len(data2015)):

    folium.Circle(
        radius=50,
        location=[data2015.iloc[i]['latitude'], data2015.iloc[i]['long
        popup=data2015.iloc[i]['typeName'],
        color="crimson",
        fill=True,
    ).add_to(map)
map
```

Out[31]:



In [32]:
```
count_nc_types = mismarked_requests['typeName'].value_counts()
total_nc_types = sum(count_nc_types)
percent_nc_types = count_nc_types/total_nc_types
percent_nc_types.plot(kind = 'bar')
plt.xticks(rotation = 90)
plt.title('No Council Distribution')
```

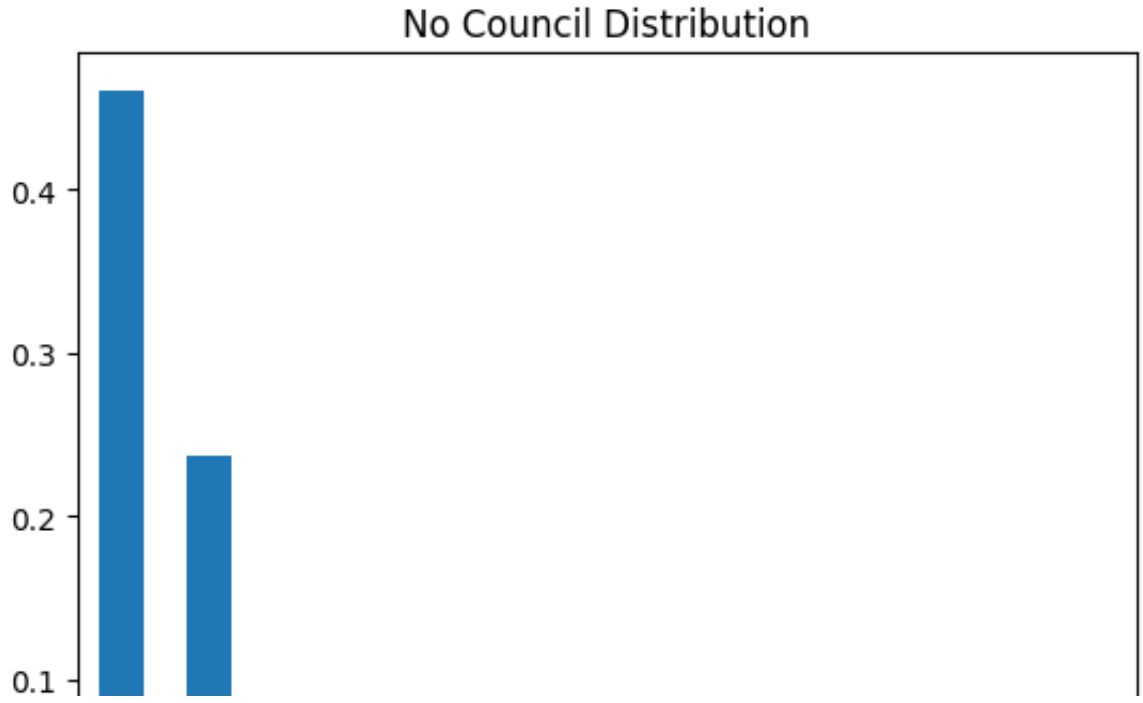
Out[32]: Text(0.5, 1.0, 'No Council Distribution')

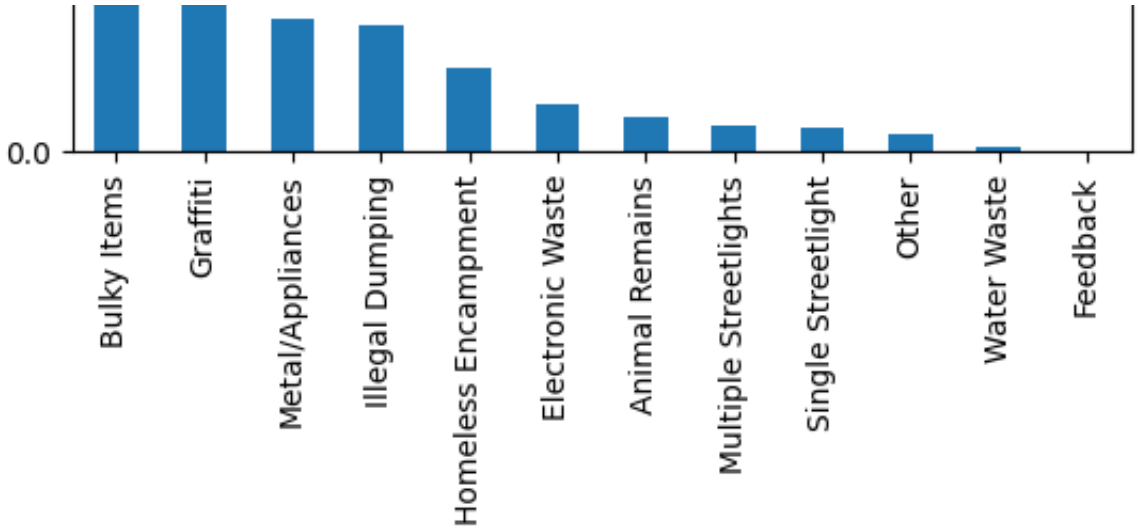


In [49]:
```

```

```
#data2022c = data[data['requestId'] not in mismarked_requests['request
outer = data.merge(mismarked_requests, how='outer', indicator=True, on

#perform anti-join
data2022c = outer[(outer._merge=='left_only')].drop('_merge', axis=1)



count_c_types = data2022c['typeName_x'].value_counts()
total_c_types = sum(count_c_types)
percent_c_types = count_c_types/total_c_types
percent_c_types.plot(kind = 'bar')
plt.xticks(rotation = 90)
plt.title('No Council Distribution')
type(percent_c_types)
percent_c_types
```

Out[49]:
```
Bulky Items              0.460572
Graffiti                 0.236290
Metal/Appliances         0.081138
Illegal Dumping          0.076739
Homeless Encampment      0.051484
Electronic Waste         0.028584
Animal Remains           0.021134
Multiple Streetlights    0.015948
Single Streetlight       0.014131
Other                    0.010346
Water Waste              0.003212
Feedback                 0.000421
Name: typeName_x, dtype: float64
```





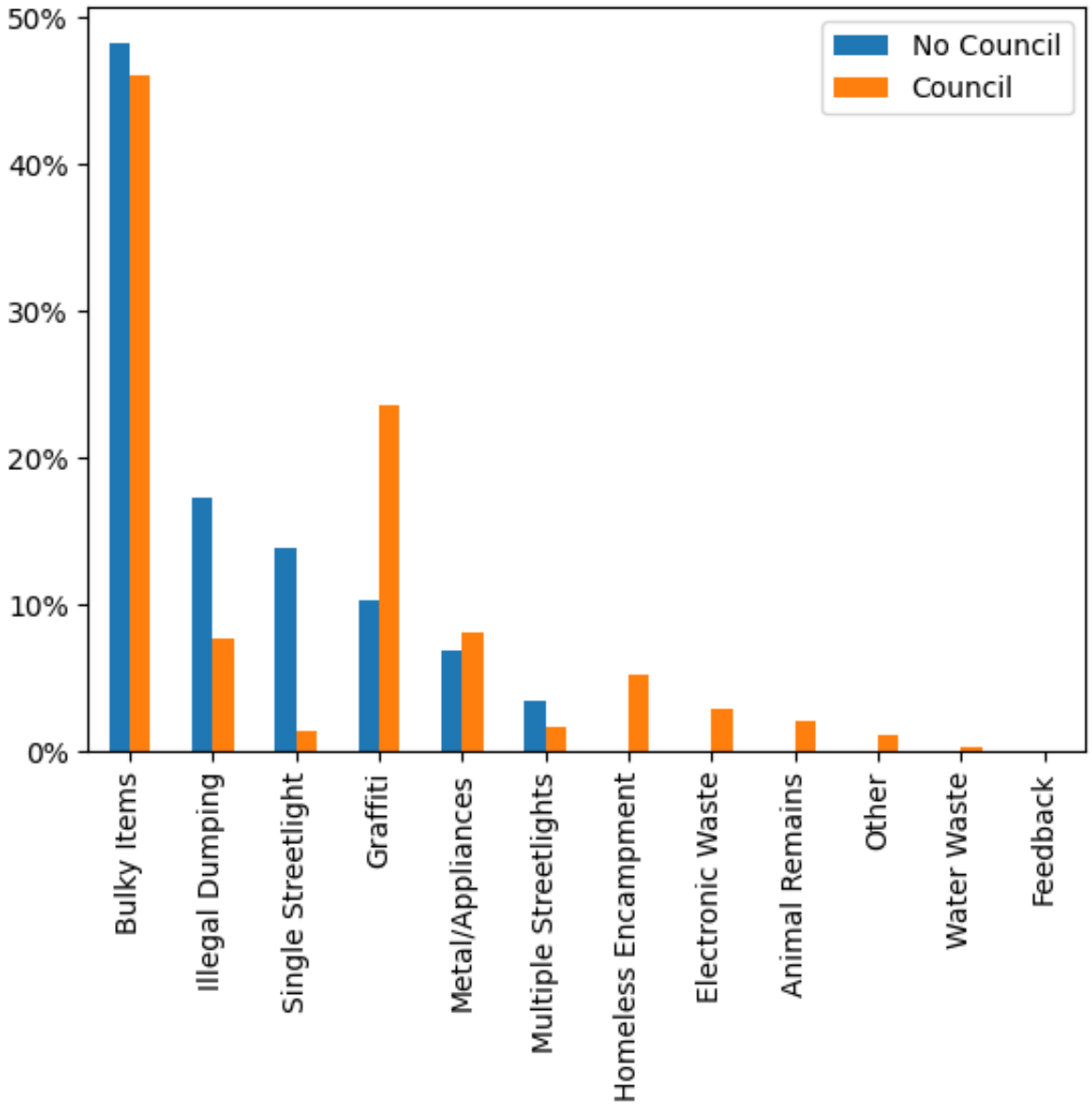
In [48]:
```
data2022c.head()
```

Out[48]:

	requestId	srnumber_x	councilId_x	councilName_x	typeId_x	typeName_x	agencyId_x	age
0	10094555	1-2155026001	27	Granada Hills North	9	Multiple Streetlights	1	St
1	9621995	1-2155027291	77	Sun Valley Area	6	Illegal Dumping	3	
2	9269764	1-2155043921	48	Mar Vista	2	Homeless Encampment	3	
3	9304705	1-2155054991	14	Downtown Los Angeles	9	Multiple Streetlights	1	St
4	9273972	1-2155055281	14	Downtown Los Angeles	9	Multiple Streetlights	1	St

5 rows × 42 columns

In [50]:
```
#compare no council requests to requests with a council to see if ther
df = pd.concat([percent_nc_types, percent_c_types], axis=1)
df.columns=['No Council', 'Council']
ax = df.plot(kind = 'bar')
ax.yaxis.set_major_formatter(mtick.PercentFormatter(1.0))
```



Conclusion: There does not seem to be any relation between 311 requests with a council and requests without a council. Therefore, it is recommended to assign data without councils to councils based on longitude and latitude data.

In [ ]: