6.3 One Pager

Import dependencies and build query URL

```
# Dependencies
import requests
import json

# Google developer API key
from config import gkey

# Target city
target_city = "Boise, Idaho"

# Build the endpoint URL
target_url = ('https://maps.googleapis.com/maps/api/geocode/json?'
    'address={0}&key={1}').format(target_city, gkey)
```

```
# Run a request to endpoint and convert result to json
geo_data = requests.get(target_url).json()

# Print the json
print(geo_data)
```

Print json response

```
# Print the json (pretty printed)
print(json.dumps(geo_data, indent=4, sort_keys=True))
```

Extract from json

```
lat = geo_data["results"][0]["geometry"]["location"]["lat"]
lng = geo_data["results"][0]["geometry"]["location"]["lng"]
```

Import Google developer API key

```
from config import gkey
```

Structure request and GET

```
# geocoordinates
target_coordinates = "43.6187102, -116.2146068"
target_search = "Chinese"
target_radius = 8000
target_type = "restaurant"

# set up a parameters dictionary
params = {
    "location": target_coordinates,
    "keyword": target_search,
    "radius": target_radius,
    "type": target_type,
    "key": gkey
}

# base url
base_url = "https://maps.googleapis.com/maps/api/place/nearbysearch/json"

# run a request using our params dictionary
response = requests.get(base_url, params=params)
```

Convert response to json

```
places_data = response.json()
```

Structure request with dataframe

```
# find the closest restaurant of each type to coordinates

base_url = "https://maps.googleapis.com/maps/api/place/nearbysearch/json"
params = {
    "location": "39.952583,-75.16522",  # philadelphia coords
    "rankby": "distance",
    "type": "restaurant",
    "key": gkey,
}
# use iterrows to iterate through pandas dataframe
for index, row in types_df.iterrows():

    # get restaurant type from df
    restr_type = row['ethnicity']

    # add keyword to params dict
    params['keyword'] = restr_type

    # assemble url and make API request
    print(f"Retrieving Results for Index {index}: {restr_type}.")
    response = requests.get(base_url, params=params).json()

    # extract results
    results = response['results']

    try:
        print(f"Closest {restr_type} restaurant is {results[0]['name']}.")

        types_df.loc[index, 'name'] = results[0]['name']
        types_df.loc[index, 'address'] = results[0]['vicinity']
        types_df.loc[index, 'price_level'] = results[0]['price_level']
        types_df.loc[index, 'rating'] = results[0]['rating']

    except (KeyError, IndexError):
        print("Missing field/result... skipping.")
```