Final Project Submission

Project code

Project code is at the github repository: https://github.com/Austinwww/SI507-Final-Project.git

Package

requests: a package to making HTTP requests
re: working with regular expression
beautifulSoup: to scrape information from web pages

Data source

1. News website

In the first proposal, I planned to use Web crawler to get the information from News (Best U.S. cities to visit). https://travel.usnews.com/rankings/best-cities-usa/
However, I found that this website may have some protection. When I used API to obtain the data, the jupyter lab will crash. Finally, I switch the website below to show the city list.
https://www.worldatlas.com/cities/america-s-10-most-visited-cities.html

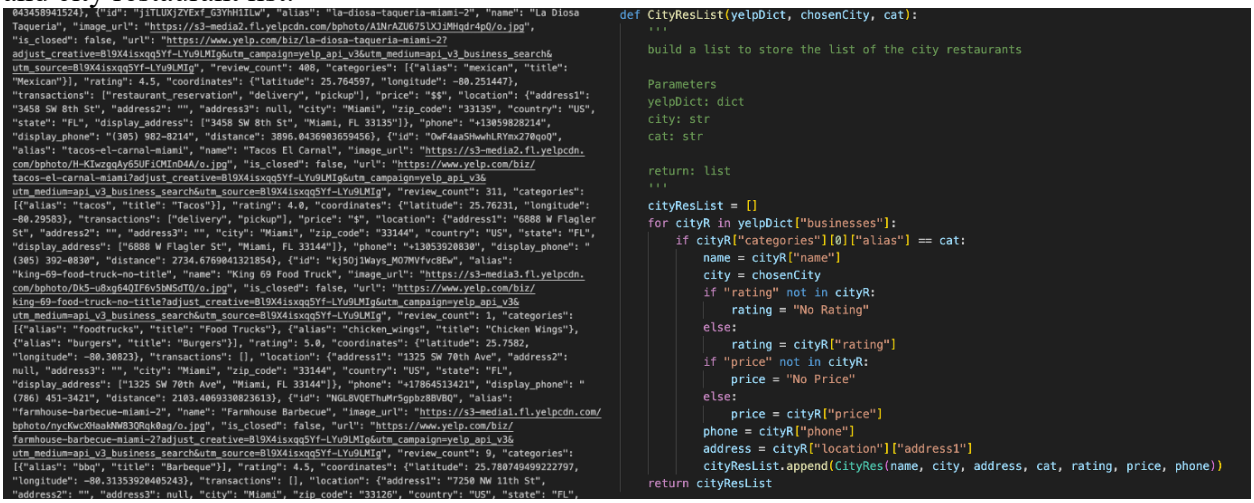
BeautifulSoup is used to crawl the information of cities from the web site. City class is used created to organize the information from the website. City class has the city information (rank, name, visitation). A city list is used to store city class. The number of cities is 10. Their ranking, visitation are obtained, and they are stored the in the cache file. Two images shown below are the cache file and City list.

2. Yelp fusion API

Yelp Fusion API was used to collect all the information of the restaurants of cities. A private key authentication was used to authenticate all endpoint from Yelp Fusion API.
https://api.yelp.com/v3/businesses/search

CityRes class is created to store the information of restaurant in the chosen city (name, city, address, categories, rating, price, phone). A city restaurant list will be used to store the city restaurants. For each request, the yelp fusion API can provide 50 restaurants at maximum. The recodes for ten cities will be 500. The information of restaurants (name, city, address, categories,

rating, price, phone) is saved as JSON cache file. Two images shown below are the cache file and city restaurant list.



```
043458941524}, {"id": "jiTLUKjZVExf_G3YhH1ILw", "alias": "la-diosa-taqueria-miami-2", "name": "La Diosa
Taqueria", "image_url": "https://s3-media2.fl.yelpcdn.com/bphoto/AINrAZU675lXJiMHqdr4pQ/o.jpg",
"is_closed": false, "url": "https://www.yelp.com/biz/la-diosa-taqueria-miami-2?
adjust_creative=Bl9X4isxqq5Yf-LYu9LMIg&utm_campaign=yelp_api_v3&utm_medium=api_v3_business_search&
utm_source=Bl9X4isxqq5Yf-LYu9LMIg", "review_count": 408, "categories": [{"alias": "mexican", "title":
"Mexican"}], "rating": 4.5, "coordinates": {"latitude": 25.764597, "longitude": -80.251447},
"transactions": ["restaurant_reservation", "delivery", "pickup"], "price": "$$", "location": {"address1":
"3458 SW 8th St", "address2": "", "address3": null, "city": "Miami", "zip_code": "33135", "country": "US",
"state": "FL", "display_address": ["3458 SW 8th St", "Miami, FL 33135"]}, "phone": "+13059828214",
"display_phone": "(305) 982-8214", "distance": 3896.0436903659456}, {"id": "OwF4aaSHwwhLRYmx270qoQ",
"alias": "tacos-el-carnal-miami", "name": "Tacos El Carnal", "image_url": "https://s3-media2.fl.yelpcdn.
com/bphoto/H-KIwzgqAy65UFiCMInD4A/o.jpg", "is_closed": false, "url": "https://www.yelp.com/biz/
tacos-el-carnal-miami?adjust_creative=Bl9X4isxqq5Yf-LYu9LMIg&utm_campaign=yelp_api_v3&
utm_medium=api_v3_business_search&utm_source=Bl9X4isxqq5Yf-LYu9LMIg", "review_count": 311, "categories":
[{"alias": "tacos", "title": "Tacos"}], "rating": 4.0, "coordinates": {"latitude": 25.76231, "longitude":
-80.29583}, "transactions": ["delivery", "pickup"], "price": "$", "location": {"address1": "6888 W Flagler
St", "address2": "", "address3": "", "city": "Miami", "zip_code": "33144", "country": "US", "state": "FL",
"display_address": ["6888 W Flagler St", "Miami, FL 33144"]}, "phone": "+13053920830", "display_phone": "
(305) 392-0830", "distance": 2734.6769041321854}, {"id": "kj5Oj1Ways_MO7MVfvc8Ew", "alias":
"king-69-food-truck-no-title", "name": "King 69 Food Truck", "image_url": "https://s3-media3.fl.yelpcdn.
com/bphoto/Dk5-uBxp64QIF6v5bN5dTQ/o.jpg", "is_closed": false, "url": "https://www.yelp.com/biz/
king-69-food-truck-no-title?adjust_creative=Bl9X4isxqq5Yf-LYu9LMIg&utm_campaign=yelp_api_v3&
utm_medium=api_v3_business_search&utm_source=Bl9X4isxqq5Yf-LYu9LMIg", "review_count": 1, "categories":
[{"alias": "foodtrucks", "title": "Food Trucks"}, {"alias": "chicken_wings", "title": "Chicken Wings"},
{"alias": "burgers", "title": "Burgers"}], "rating": 5.0, "coordinates": {"latitude": 25.7582,
"longitude": -80.30823}, "transactions": [], "location": {"address1": "1325 SW 70th Ave", "address2":
null, "address3": "", "city": "Miami", "zip_code": "33144", "country": "US", "state": "FL",
"display_address": ["1325 SW 70th Ave", "Miami, FL 33144"]}, "phone": "+17864513421", "display_phone": "
(786) 451-3421", "distance": 2103.4069330823613}, {"id": "NGL8VQEThuMr5gpbzBBVBQ", "alias":
"farmhouse-barbecue-miami-2", "name": "Farmhouse Barbecue", "image_url": "https://s3-media1.fl.yelpcdn.com/
bphoto/nycKwcXHaaNW83QRqk0ag/o.jpg", "is_closed": false, "url": "https://www.yelp.com/biz/
farmhouse-barbecue-miami-2?adjust_creative=Bl9X4isxqq5Yf-LYu9LMIg&utm_campaign=yelp_api_v3&
utm_medium=api_v3_business_search&utm_source=Bl9X4isxqq5Yf-LYu9LMIg", "review_count": 9, "categories":
[{"alias": "bbq", "title": "Barbeque"}], "rating": 4.5, "coordinates": {"latitude": 25.780749499222797,
"longitude": -80.3135392040524}, "transactions": [], "location": {"address1": "7250 NW 11th St",
"address2": "", "address3": null, "city": "Miami", "zip_code": "33126", "country": "US", "state": "FL",
```

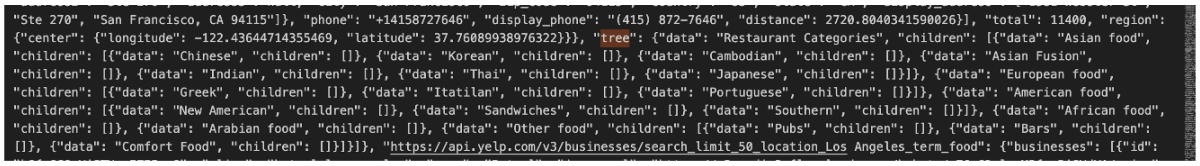
```python
def CityResList(yelpDict, chosenCity, cat):
    '''
    build a list to store the list of the city restaurants

    Parameters
    yelpDict: dict
    city: str
    cat: str

    return: list
    '''
    cityResList = []
    for cityR in yelpDict["businesses"]:
        if cityR["categories"][0]["alias"] == cat:
            name = cityR["name"]
            city = chosenCity
            if "rating" not in cityR:
                rating = "No Rating"
            else:
                rating = cityR["rating"]
            if "price" not in cityR:
                price = "No Price"
            else:
                price = cityR["price"]
            phone = cityR["phone"]
            address = cityR["location"]["address1"]
            cityResList.append(CityRes(name, city, address, cat, rating, price, phone))
    return cityResList
```

Data structure

After the cities information were obtained from the website, the restaurants will be shown based on cities. For each city, in order to well categorized the restaurants, the Tree structure are used to make the interactive platform looking good. After displaying these categories, the users can choose the category they want to search for the restaurants.

1. TreeNode were built to store the information of each node in the tree, which contains the data (food category), parent (bigger category), children (smaller category).
2. Search the json file whether the tree was stored. If so, load the tree. If not, use the buildTree function to build the tree.
3. Function toDict() and dictToTree were built to convert the format between TreeNode and json.

A stand only python file (readTree.py) that reads the json of the tree.
As pictures showed below. The top picture shows how the trees are stored in the json file. The bottom left figure shows how the tree was constructed. The bottom right picture shows how the tree structure looks like at the interactive platform.




```python
class TreeNode()
    def __init__(self.data):
        self.data = data
        self.children = []
        self.parent = None

    def addChild(self, child):
        child.parent = self
        self.children.append(child)


def buildTree():
    root = TreeNode("Recommended Restaurants")

    asian = TreeNode("Asian food")
    asian.addChild(TreeNode("Chinese"))
    asian.addChild(TreeNode("Korean"))

    euro = TreeNode("European food")
    euro.addChild(TreeNode("Greek"))
    euro.addChild(TreeNode("Itatilan"))

    american = TreeNode("American food")
    american.addChild(TreeNode("New American"))
    american.addChild(TreeNode("Sandwiches"))

    african = TreeNode("African food")

    arabian = TreeNode("Arabian food")

    other = TreeNode("Other food")
```

```
************************************
Restaurant Categories
   Asian food
      Chinese
      Korean
      Cambodian
      Asian Fusion
      Indian
      Thai
      Japanese
   European food
      Greek
      Itatilan
      Portuguese
   American food
      New American
      Sandwiches
      Southern
   African food
   Arabian food
   Other food
      Pubs
      Bars
      Comfort Food
************************************
```

Interaction and Presentation plans

The interaction and presentation technology used in this program is command line prompts.

1. In the beginning, the list of the America's most visited cities with their visitation in 2022 will be shown. The users can choose the cities which attracts them most or the city they plan to go.
2. After the city was chosen, the program will list main categories and subcategories stored in the database. The users can choose the category they are most interested. Then, the restaurants in this category will be listed on screen. All the information (name, city, address, categories, rating, price, phone) about these restaurants will be provided. If the users cannot find the category they want, the program will ask the user to input the category of restaurant they want to go. The program will search such category in the database. If this category exists, the results will be printed.
3. After the first search is done, if the users are not satisfied with such category of restaurant, they can go back to find another category or search the restaurants in another city.

Demo link

https://drive.google.com/file/d/1iN37f1ScbabSAqhNjITFImX3FCoYssW9/view?usp=sharing