2. Data sources and data cleaning:

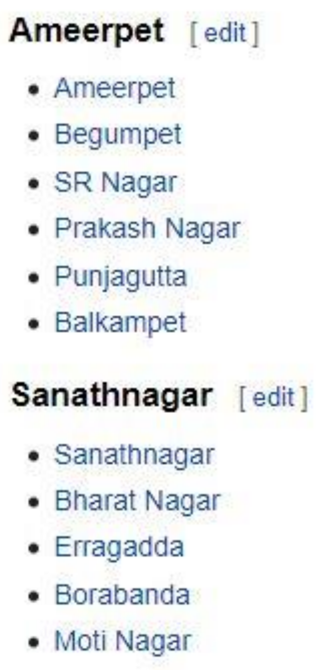
For this project, Hyderabad city neighborhood names, their respective latitude and longitude coordinates and the nearby venues for each neighborhood data are required. The following are the data sources for our project.

2.1 Get Neighborhood names:

After a quick google search, it was found that there was a Wikipedia page, that provides the information on neighbourhoods of Hyderabad city. Below is the link for the Wikipedia webpage.

Wikipedia-link:
https://en.wikipedia.org/wiki/List_of_neighbourhoods_in_Hyderabad

The data on the Wikipedia webpage was as shown below:



Ameerpet [edit]

- Ameerpet
- Begumpet
- SR Nagar
- Prakash Nagar
- Punjagutta
- Balkampet

Sanathnagar [edit]

- Sanathnagar
- Bharat Nagar
- Erragadda
- Borabanda
- Moti Nagar

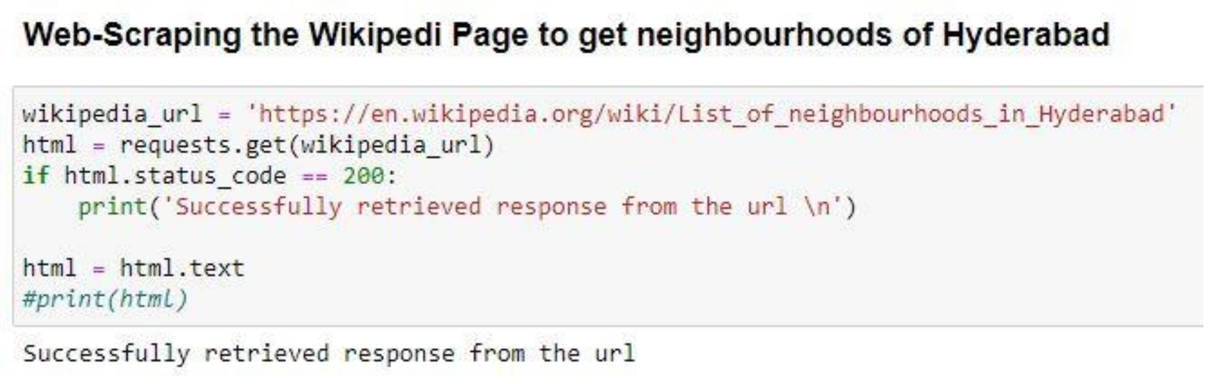
The neighborhood names are displayed row wise. Each neighborhood name is a link to its own Wikipedia webpage. We need to fetch the neighborhood names from the Wikipedia website.

To get data, let’s web-scrape this Wikipedia webpage and get the required data. For web-scraping, beautiful soup python package was used. On inspecting the webpage, our required data in the “<li>” tags.

Python requests library was used to fetch data from the URL. Beautiful soup object was used on the data from webpage. The fetched data is then filtered-out to obtain our required data. Finally, our required data is stored in a pandas data frame.

The web scraping was done as shown in the below figures.

Fig:



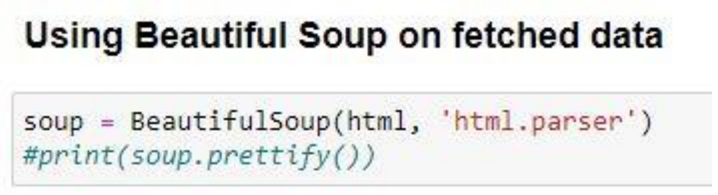
Web-Scraping the Wikipedi Page to get neighbourhoods of Hyderabad

```
wikipedia_url = 'https://en.wikipedia.org/wiki/List_of_neighbourhoods_in_Hyderabad'
html = requests.get(wikipedia_url)
if html.status_code == 200:
    print('Successfully retrieved response from the url \n')

html = html.text
#print(html)
```

```
Successfully retrieved response from the url
```

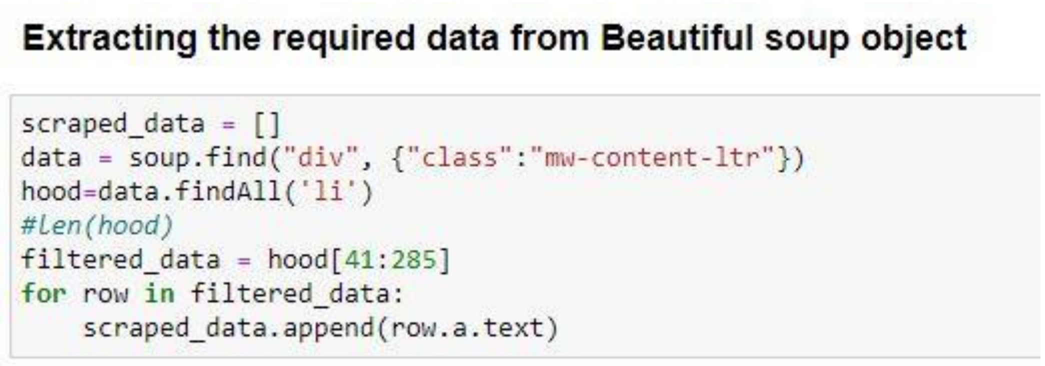
Fig:



Using Beautiful Soup on fetched data

```
soup = BeautifulSoup(html, 'html.parser')
#print(soup.prettify())
```

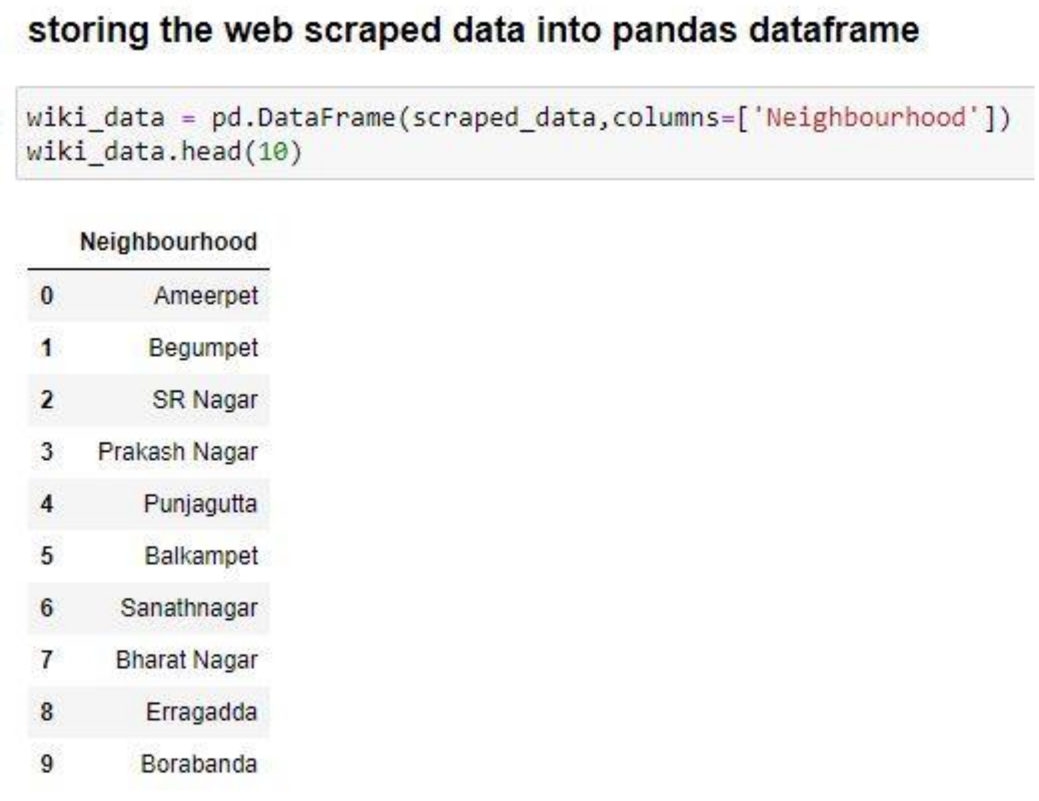
Fig:



Extracting the required data from Beautiful soup object

```
scraped_data = []
data = soup.find("div", {"class":"mw-content-ltr"})
hood=data.findAll('li')
#len(hood)
filtered_data = hood[41:285]
for row in filtered_data:
    scraped_data.append(row.a.text)
```

Fig:



storing the web scraped data into pandas dataframe

```
wiki_data = pd.DataFrame(scraped_data,columns=['Neighbourhood'])
wiki_data.head(10)
```

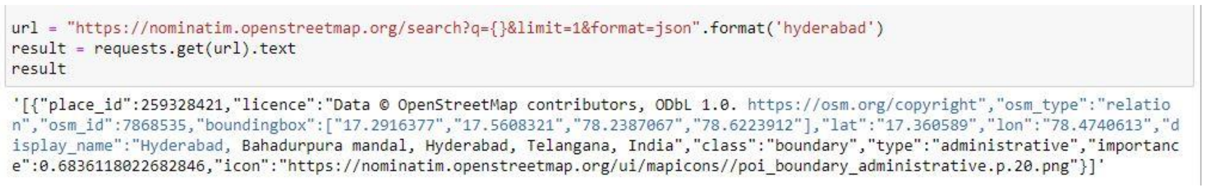
	Neighbourhood
0	Ameerpet
1	Begumpet
2	SR Nagar
3	Prakash Nagar
4	Punjagutta
5	Balkampet
6	Sanathnagar
7	Bharat Nagar
8	Erragadda
9	Borabanda

2.2 Get coordinates for neighborhoods:

We got the neighborhood names of Hyderabad city by web scraping the Wikipedia webpage. The latitude and longitude coordinates of the neighborhoods were not available in the Wikipedia page. To obtain the coordinates, the openstreetmap.org website's nominatim API was used. An API request should be sent to the API to get data. The following is the URL format.

URL: "https://nominatim.openstreetmap.org/search?q={}&limit=1&format=json"

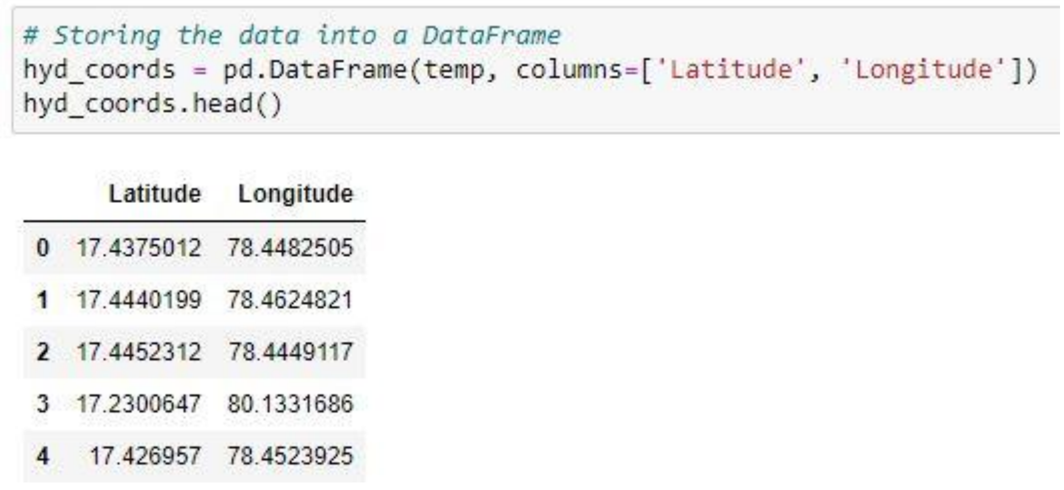
The response we get, contains the coordinates for a requested address in json format. By using this API, coordinates for all the neighborhoods of Hyderabad city are obtained. This is shown below.



```
url = "https://nominatim.openstreetmap.org/search?q={}&limit=1&format=json".format('hyderabad')
result = requests.get(url).text
result
```

```
'[{"place_id":259328421,"licence":"Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright","osm_type":"relatio
n","osm_id":7868535,"boundingbox":["17.2916377","17.5608321","78.2387067","78.6223912"],"lat":"17.360589","lon":"78.4740613","d
isplay_name":"Hyderabad, Bahadurpura mandal, Hyderabad, Telangana, India","class":"boundary","type":"administrative","importanc
e":0.6836118022682846,"icon":"https://nominatim.openstreetmap.org/ui/mapicons//poi_boundary_administrative.p.20.png"}]'
```

The API response is in json format, here our required data is latitudes and longitude coordinates. The coordinates are filtered-out from the response and stored in a pandas data frame as shown below.



```
# Storing the data into a DataFrame
hyd_coords = pd.DataFrame(temp, columns=['Latitude', 'Longitude'])
hyd_coords.head()
```

```
     Latitude   Longitude
0  17.4375012  78.4482505
1  17.4440199  78.4624821
2  17.4452312  78.4449117
3  17.2300647  80.1331686
4   17.426957  78.4523925
```

2.3 Data Cleaning: The challenge with coordinates data

By using the nominatim API, coordinates for 14 neighborhoods are not obtained. This is mainly because,

1. For some neighbourhoods, the API doesn’t provide coordinates due to unknown reason.
2. Few neighborhood names in the Wikipedia webpage are misspelt. This was found by manually searching for coordinates on google website.

This challenge has overcome by searching coordinates for remaining neighborhoods, using the below mentioned website.

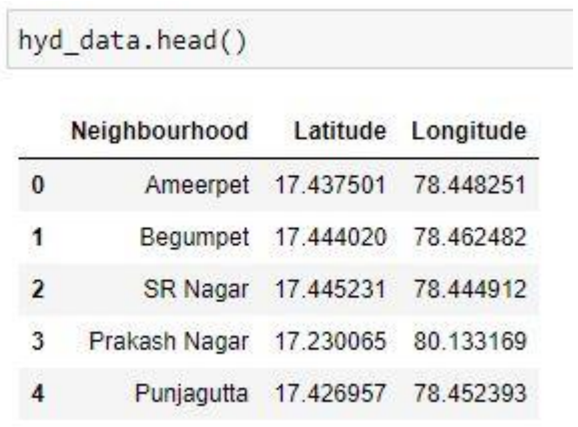
Website: https://www.latlong.net/

This website provides latitude and longitude coordinates for a requested address or a place. This is a free website; it allows only a limited number of searches in a day.

Finally, after getting coordinates for all the neighborhoods, the data is stored in a pandas data frame. Now our required data is stored in two data frames. One for neighborhood names and the other for coordinates. These two data frames are merged into a single data frame which is required data for our project.

After merging the neighborhoods data and the coordinates data,

The final data frame obtained was as shown below.



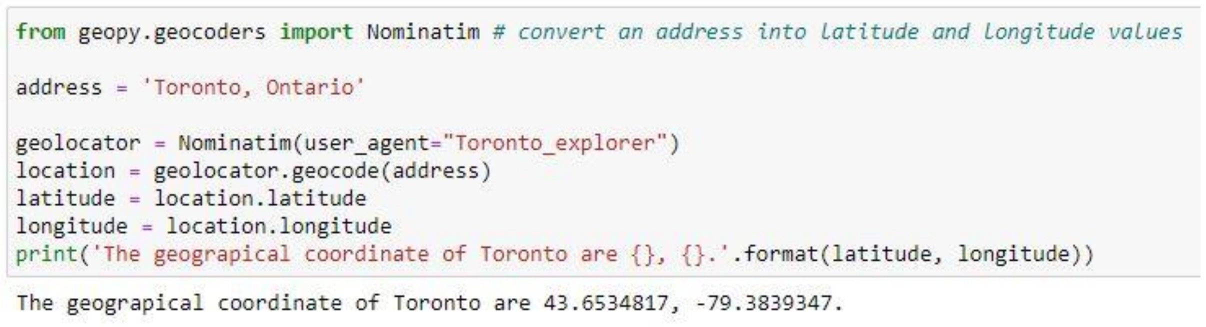
```
hyd_data.head()
```

	Neighbourhood	Latitude	Longitude
0	Ameerpet	17.437501	78.448251
1	Begumpet	17.444020	78.462482
2	SR Nagar	17.445231	78.444912
3	Prakash Nagar	17.230065	80.133169
4	Punjagutta	17.426957	78.452393

2.4 Geopy library to get coordinates of Hyderabad city:

Geopy is a python library that can be used to fetch coordinates of an address. This library was used to get the coordinates of Hyderabad city. The Hyderabad city coordinates will used to plot the map of Hyderabad city using python's folium visualization library. Below is an example.

Fig:



```
from geopy.geocoders import Nominatim # convert an address into latitude and longitude values

address = 'Toronto, Ontario'

geolocator = Nominatim(user_agent="Toronto_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Toronto are {}, {}.'.format(latitude, longitude))
```

```
The geograpical coordinate of Toronto are 43.6534817, -79.3839347.
```

2.5 Neighborhood location data using Foursquare API:

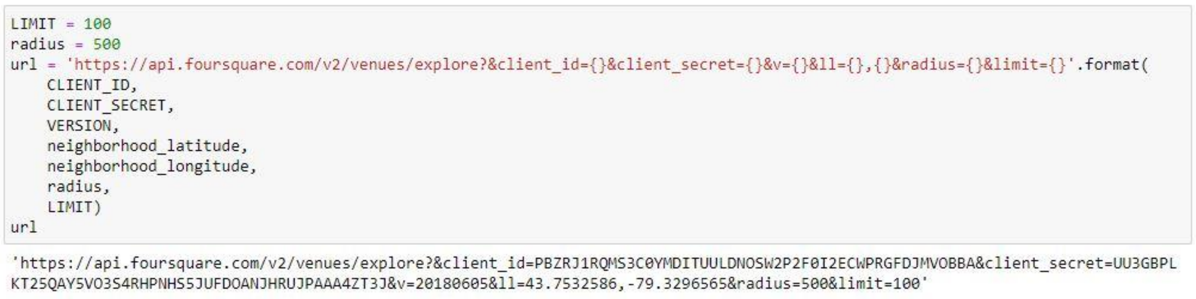
To get the nearby venues of a neighborhood, Foursquare API was used. Foursquare API provides location data of an address. It provides diverse information about venues, users, photos, check-in's, geo-tagging…etc.

This API was used in this project to get nearby venue details of a neighborhood. To get data from the API, a search query is to be sent to the foursquare API.

The response from the API contains the requested data in json format. The required data from Foursquare API was venues of a neighborhood.

The API requires credentials like CLIENT_ID, CLIENT_SECRET and VERSION to get data. These can be obtained by creating an account on Foursquare website. The response from API is in json format. The response is filtered-out to get our required data. Finally, the data is stored in a data frame.

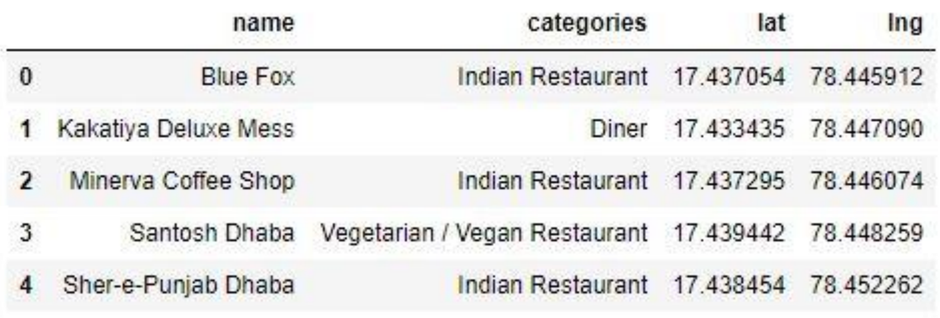
Fig: Request to Foursquare API:



```
LIMIT = 100
radius = 500
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
    neighborhood_longitude,
    radius,
    LIMIT)
url
```

```
'https://api.foursquare.com/v2/venues/explore?&client_id=PBZRJ1RQMS3C0YMDITUULDNOSW2P2F0I2ECWPRGFDJMVOBBA&client_secret=UU3GBPL
KT25QAY5VO3S4RHPNHS5JUFDOANJHRUJPAAA4ZT3J&v=20180605&ll=43.7532586,-79.3296565&radius=500&limit=100'
```

Fig: The filtered json response stored in a data frame:



	name	categories	lat	lng
0	Blue Fox	Indian Restaurant	17.437054	78.445912
1	Kakatiya Deluxe Mess	Diner	17.433435	78.447090
2	Minerva Coffee Shop	Indian Restaurant	17.437295	78.446074
3	Santosh Dhaba	Vegetarian / Vegan Restaurant	17.439442	78.448259
4	Sher-e-Punjab Dhaba	Indian Restaurant	17.438454	78.452262

2.6 Exploratory Data Analysis:

Exploratory data analysis is not required for our project. As our project requires detailed analysis of location data such as venues of neighborhoods and neighborhood coordinates. This data was obtained from the data sources as explained above. The location data will suffice for making a decision on where to open the restaurant.