**Pseudo Code for various Methods and Functions in this Project**

**-> (#) - Special character Indicating important comments (Some are kept bold and some or not on purpose).**

**-> A,B,C.. and 1,2,3.. Kept for Categories and sub-categories in a sequential order.**

**1) Define API key for Google Places API**

**# 1 Function get\_restaurants\_in\_location(location):**

Construct API URL using the provided location and API key

Send a GET request to the constructed URL

If the response status code is 200 (OK):

Extract restaurant data from the JSON response

Return a list of restaurant results

Otherwise, return an empty list

**#2 Function display\_restaurant\_details(restaurant):**

Display the name, address, and rating of the given restaurant

**# 3 Fetch restaurant data for a specific location:**

Call get\_restaurants\_in\_location function with the desired location (e.g., Dallas)

Store the returned restaurant data in a variable

**# 4 Create an empty DataFrame to store the fetched restaurant data**

For each restaurant in the fetched data:

Create a dictionary containing restaurant details (name, address, rating)

Append the dictionary to the DataFrame

**# 5 Display restaurant details for each entry in the DataFrame:**

For each row (restaurant) in the DataFrame:

Call display\_restaurant\_details function with the restaurant details

**2) Define API key and calling a function to get Place ID for a given place name**

**# 1 Define API key**

api\_key = "YOUR\_API\_KEY"

**# A) Function to get Place ID for a given place name**

function get\_place\_id(place\_name):

url = construct\_url\_for\_place\_search(place\_name, api\_key)

response = make\_api\_request(url)

if response.status\_code == 200:

results = response.json().get('results', [])

if results:

place\_id = results[0].get('place\_id')

return place\_id

return None

**# B) Function to fetch reviews for a place using its Place ID**

function get\_reviews\_for\_place(place\_id):

url = construct\_url\_for\_place\_details(place\_id, api\_key)

response = make\_api\_request(url)

if response.status\_code == 200:

result = response.json().get('result', {})

reviews = result.get('reviews', [])

return reviews

return []

**3) Location Based Data collection**

**# 1 Search for 'Rj Mexican Cuisine' or any and get its Place ID**

place\_name = 'Rj Mexican Cuisine'

place\_id\_rj\_mexican = get\_place\_id(place\_name)

**# 2 If Place ID exists**

if place\_id\_rj\_mexican is not None:

# Get reviews for 'Rj Mexican Cuisine' using its Place ID

reviews\_rj\_mexican = get\_reviews\_for\_place(place\_id\_rj\_mexican)

**# 3 Save reviews to a CSV file**

review\_texts = [review.get('text') for review in reviews\_rj\_mexican]

data = {'Review Text': review\_texts}

reviews\_df = create\_dataframe(data)

save\_to\_csv(reviews\_df, 'rj\_mexican\_reviews.csv')

display\_message("Reviews saved to 'rj\_mexican\_reviews.csv'")

else:

display\_message("No Place ID found for 'Rj Mexican Cuisine'")

**4) Data visualizations: Bar Graph**

**# 1. Fetch data of restaurants in Dallas from an API using Google Places API.**

**# 2. Store the fetched data in a variable, maybe a list of dictionaries containing restaurant information (name, address, rating).**

**# 3. Extract the restaurant names and their corresponding ratings from the fetched data.**

**# 4. Create two lists: one for restaurant names (restaurants) and another for their ratings (ratings).**

**# 5. Create a horizontal bar graph to display restaurant ratings.**

a. Import **matplotlib.pyplot** as plt.

b. Use **plt.figure()** to define the figure size.

c. Use **plt.barh()** to create a horizontal bar graph.

d. Provide restaurant names as y-axis and ratings as x-axis.

e. Set labels for **x-axis** and **y-axis**.

f. Set a **title** for the graph.

g. Show the graph using **plt.show()**.

**5) Data collection single entity**

**# 1 Define API key for accessing Google Places API**

api\_key = "YOUR\_API\_KEY"

**# 2 Define function to retrieve reviews for a place using its Place ID**

function get\_reviews\_for\_place(place\_id):

construct URL using the place\_id and API key

send GET request to the constructed URL

if response status code is 200:

extract reviews data from the response JSON

return the reviews data

else:

return an empty list

**# 3 Search for the Place ID of 'Rj Mexican Cuisine'**

place\_name = 'Rj Mexican Cuisine'

place\_id\_rj\_mexican = get\_place\_id(place\_name)

if place\_id\_rj\_mexican exists:

print the Place ID for 'Rj Mexican Cuisine'

**# 4 Get reviews for 'Rj Mexican Cuisine' using its Place ID**

reviews\_rj\_mexican = get\_reviews\_for\_place(place\_id\_rj\_mexican)

if reviews\_rj\_mexican are retrieved:

for each review in reviews\_rj\_mexican:

print the review text

print a separator

**# 5 Save the reviews to a CSV file**

extract review texts from reviews\_rj\_mexican

create a DataFrame with review texts

save the DataFrame to a CSV file named 'rj\_mexican\_reviews.csv'

print a message indicating that reviews are saved to the CSV file

else:

print a message indicating that no reviews were found for 'Rj Mexican Cuisine'

else:

print a message indicating that no Place ID was found for 'Rj Mexican Cuisine'

**6) Define API key and calling a function to get Place ID for a given place name**

**# 1. Define the API key required for accessing the Google Places API.**

**# 2. Create a function, get\_reviews\_for\_place, that takes a place ID as input and retrieves reviews for that place using the Google Places API.**

a. Construct the URL with the place ID and necessary fields for reviews.

b. Send a GET request to the constructed URL using requests library.

c. If the response status is 200 (success), extract the reviews from the response JSON.

d. Return the reviews if available, otherwise return an empty list.

**# 3. Create another function, get\_all\_reviews\_in\_dallas, to fetch reviews for all restaurants in Dallas.**

a. Use a function, **get\_restaurants\_in\_location,** to fetch restaurant information for Dallas.

b. Iterate through each restaurant retrieved in Dallas:

- Extract the **place ID** for the restaurant.

- Call **get\_reviews\_for\_place** with the place ID to get reviews for that restaurant.

- Extend a list (**all\_reviews**) with the retrieved reviews.

# 4. Fetch all reviews for restaurants in Dallas by invoking the **get\_all\_reviews\_in\_dallas** function**.**

a. Print the total number of reviews fetched.

# 5. Print a sample of review texts (e.g., first 5 lines) from the fetched reviews in Dallas.

**6) Data collection for all restaurants in Dallas**

**# 1 Define API key for accessing Google Places API**

api\_key = "YOUR\_API\_KEY"

**# 2 Define function to get reviews for a specific place using its Place ID**

function **get\_reviews\_for\_place(place\_id)**:

construct URL using place\_id and API key for Google Places API

send a GET request to the constructed URL

if **response** status code is **200**:

extract review data from the response **JSON** (if available)

return the review data, otherwise return an empty list

else:

return an empty list

**# 3 Define function to get all reviews for restaurants in Dallas**

function **get\_all\_reviews\_in\_dallas()**:

dallas\_restaurants = **get\_restaurants\_in\_location**("Dallas")

create an empty list to store all reviews

for each restaurant in **dallas\_restaurants:**

get the **place\_id** for the restaurant

fetch reviews for the restaurant using its **place\_id**

append the fetched reviews to the list of all reviews

return the list containing all reviews for Dallas restaurants

**# 4 Get all reviews for restaurants in Dallas using the defined functions**

all\_reviews\_in\_dallas = **get\_all\_reviews\_in\_dallas(**)

print the total number of reviews fetched

**# 5 Display sample review text (first 5 lines)**

for each review in the first 5 reviews in **all\_reviews\_in\_dallas**:

print the review text

**7) Sentiment Analysis for all restaurants in Dallas**

# 1 Initialize **NLTK libraries** (download required resources)

#2 Define a list of sample reviews

#3 Define a function **clean\_text(text)** to preprocess and clean the reviews:

**- Convert text to lowercase**

**- Remove punctuation**

**- Remove numbers**

**- Tokenize the text**

**- Remove stopwords**

**- Join the cleaned tokens back into a single string**

#4 Apply the **clean\_text** function to each review in the sample reviews list

#5 Create a **Pandas DataFrame** containing the cleaned reviews

#6 Save the DataFrame as a CSV file named **'cleaned\_restaurant\_reviews\_dallas.csv'**

#7 Display a message indicating the **CSV file's** successful creation

#8 Display both the original and cleaned reviews within the DataFrame