Script 1 : import time

import googlemaps

import pandas as pd

def miles\_to\_meters(miles):

    try:

        return miles \* 1\_609.344

    except:

        return 0

API\_KEY = 'xxxx'

map\_client = googlemaps.Client(API\_KEY)

address = '333 Market St, San Francisco, CA'

geocode = map\_client.geocode(address=address)

(lat, lng) = map(geocode[0]['geometry']['location'].get, ('lat', 'lng'))

search\_string = 'ramen'

distance = miles\_to\_meters(2)

business\_list = []

response = map\_client.places\_nearby(

    location=(lat, lng),

    keyword=search\_string,

    radius=distance

)

business\_list.extend(response.get('results'))

next\_page\_token = response.get('next\_page\_token')

while next\_page\_token:

    time.sleep(2)

    response = map\_client.places\_nearby(

        location=(lat, lng),

        keyword=search\_string,

        radius=distance,

        page\_token=next\_page\_token

    )

    business\_list.extend(response.get('results'))

    next\_page\_token = response.get('next\_page\_token')

df = pd.DataFrame(business\_list)

df['url'] = 'https://www.google.com/maps/place/?q=place\_id:' + df['place\_id']

df.to\_excel('{0}.xlsx'.format(search\_string), index=False)

Script 2 : import requests

def fetch\_businesses(api\_key, location, radius, keyword):

    endpoint = 'https://maps.googleapis.com/maps/api/place/nearbysearch/json'

    params = {

        'key': api\_key,

        'location': location,

        'radius': radius,  # Radius in meters

        'keyword': keyword  # Keyword to filter businesses

    }

    # Making the HTTP GET request

    response = requests.get(endpoint, params=params)

    if response.status\_code == 200:

        data = response.json()

        return data.get('results', [])

    else:

        print('Error:', response.status\_code)

        return []

def main():

    api\_key = 'xxxx'

    # Defining the locations and radius for VA and TX

    locations = {

        'VA': '38.0034,-79.4588',

        'TX': '31.9686,-99.9018'

    }

    # Defining the radius for the Nearby Search

    radius = 5000

    # Defining the keyword to filter businesses

    keyword = 'electricians'

    # Iterate over the locations and fetch businesses

    for state, location in locations.items():

        businesses = fetch\_businesses(api\_key, location, radius, keyword)

        print(f'Small businesses in {state}:')

        for business in businesses:

            print(business['name'])

if \_\_name\_\_ == '\_\_main\_\_':

    main()

Script 3 : import googlemaps

def get\_plumbing\_companies(api\_key, latitude=None, longitude=None):

    gmaps = googlemaps.Client(key=api\_key)

    if latitude is not None and longitude is not None:

        location = f"{latitude},{longitude}"

    else:

        location = "Virginia"  # Default location

    places = gmaps.places(query="roofing companies", location=location)

    plumbing\_results = []

    for place in places['results']:

        place\_id = place['place\_id']

        details = gmaps.place(place\_id=place\_id, fields=['name', 'formatted\_address', 'formatted\_phone\_number'])

        name = details['result']['name']

        address = details['result']['formatted\_address']

        phone\_number = details['result'].get('formatted\_phone\_number', 'Phone number not available')

        plumbing\_results.append({'name': name, 'address': address, 'phone\_number': phone\_number})

    return plumbing\_results

def main():

    api\_key = "xxxx"

    # Example latitude and longitude coordinates for Virginia

    virginia\_latitude = 39.0438,

    virginia\_longitude = 77.4874

    plumbing\_companies = get\_plumbing\_companies(api\_key, virginia\_latitude, virginia\_longitude)

    # Printing details of plumbing companies

    for idx, company in enumerate(plumbing\_companies, start=1):

        print(f"Auto Mechanic{idx}:")

        print(f"Name: {company['name']}")

        print(f"Address: {company['address']}")

        print(f"Phone Number: {company['phone\_number']}")

        print()

if \_\_name\_\_ == "\_\_main\_\_":

    main()