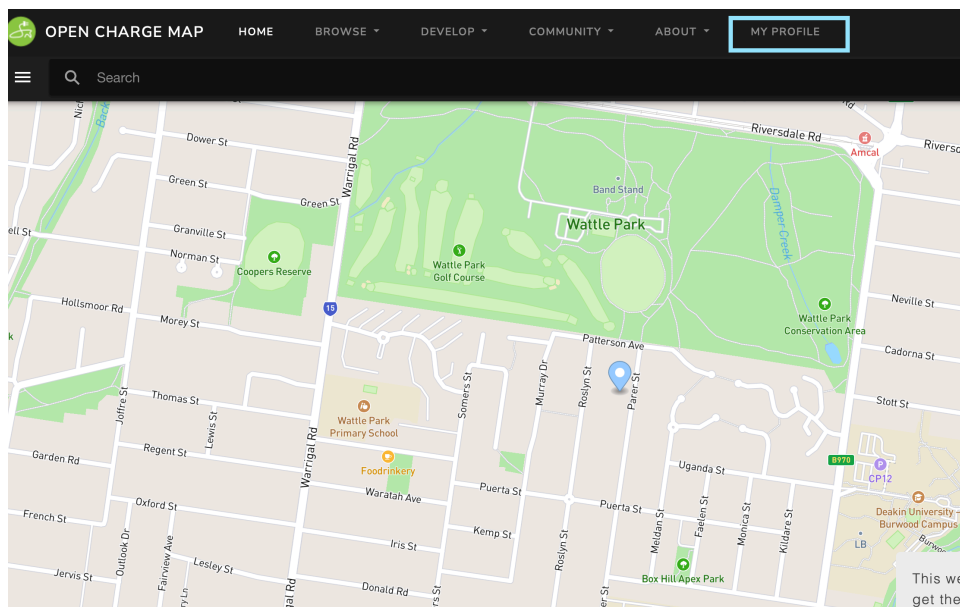


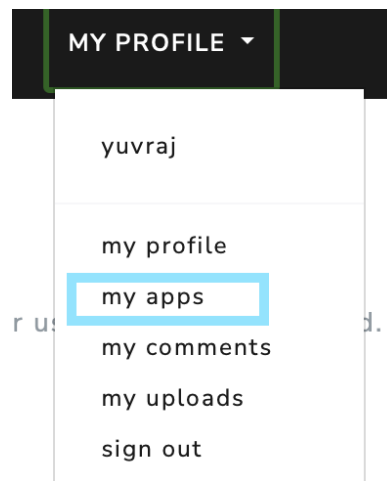
# EV Stations Data Collection Guide using Open Charge Map-

To access the Open Charge Map API and collect data, you'll need an API key. Follow these steps to obtain your API key:

- **Visit Open Charge Map:** Navigate to the Open Charge Map website at <https://openchargemap.org/>.
- **Create an Account:** If you don't have an account, sign up for free by clicking on the "Sign Up" button and filling out the required information.



- **Access Developer Dashboard:** Once logged in, access your account settings or navigate to the developer dashboard.



- **Register an application:** To get an API key you need to first register through this application first

without using your username and password.

There are no applications authorized to access your Open Charge Map account.

# MY API KEYS

If you are a software developer you may need an API key to use the Open Charge Map API. You can manage your API keys here.

REGISTER AN APPLICATION

APP	WEBSITE	CREATED	ENABLED	PUBLIC	API KEY
-----	---------	---------	---------	--------	---------

## CREATE NEW APPLICATION

Title

api key needed

Description

for my uni project

Website

List App in Public Showcase



SAVE



Open Charge Map is a non-commercial, non-profit service hosted and supported by a community of businesses, charities,

- **Generate API Key:** Look for the section related to API access or API keys. Generate a new API key if you don't have one already.

APP	WEBSITE	CREATED	ENABLED	PUBLIC	API KEY	
mr		4/10/2024 8:27:59 AM	True	True	82aaef2d-e9b8-4caa-8c0f-e7869973a241 Generate New API Key	Edit

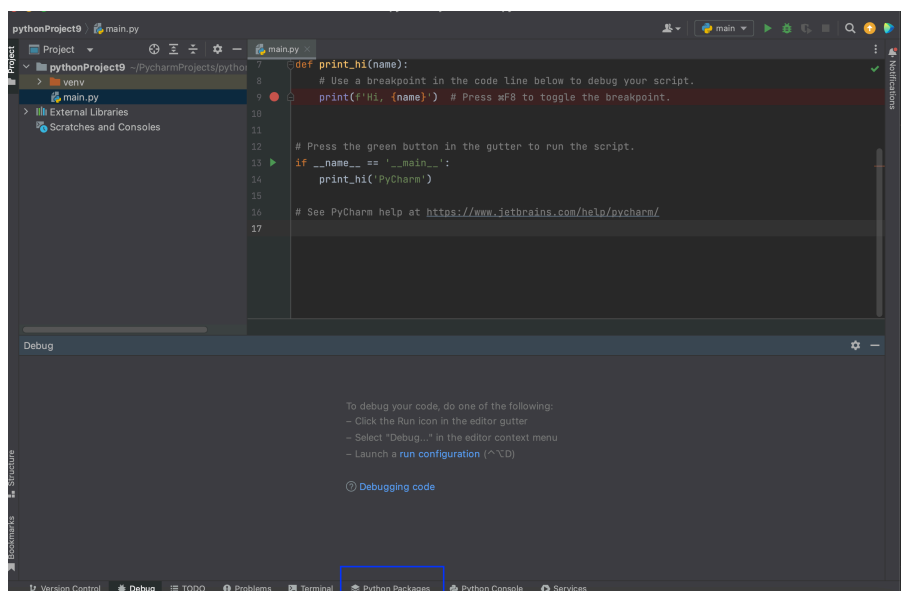
- **Copy API Key:** Once generated, copy the API key provided. This key will be used to authenticate your requests to the Open Charge Map API.

## Generalized Code to Get Data from the API:

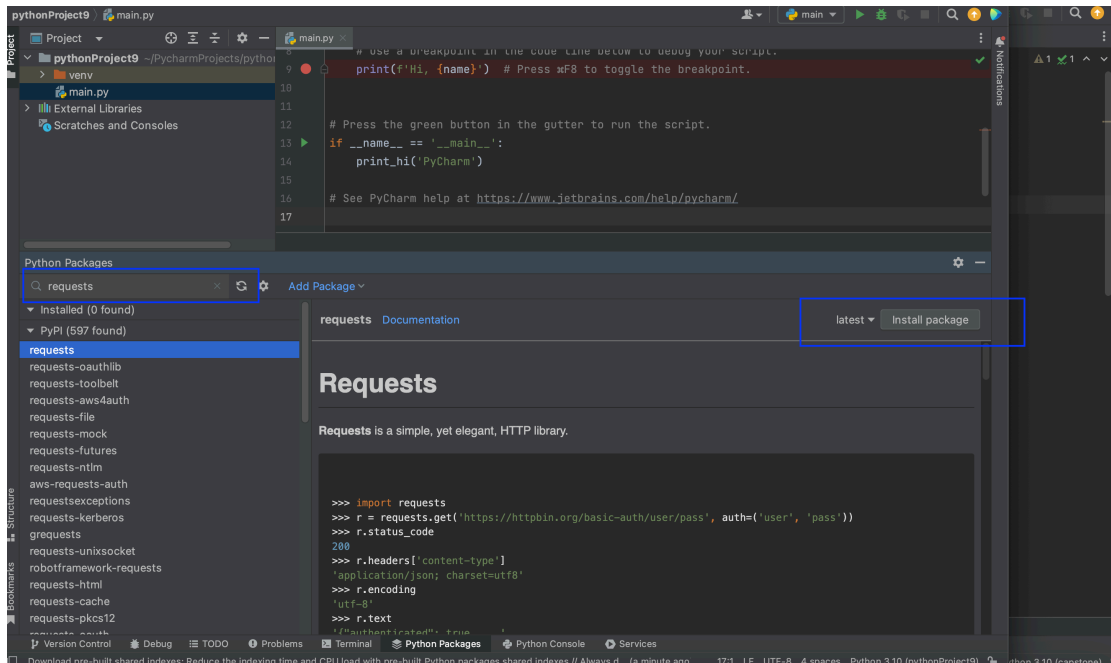
-First, we need to install the requests library using - pip install requests

```
base) yuvrajsinghsekhon@yuvrajs-MacBook-Air ~ % pip install requests
Requirement already satisfied: requests in /opt/anaconda3/lib/python3.11/site-packages (2.31.0)
Requirement already satisfied: charset-normalizer<4,>=2.5 in /opt/anaconda3/lib/python3.11/site-packages (from requests) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /opt/anaconda3/lib/python3.11/site-packages (from requests) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/anaconda3/lib/python3.11/site-packages (from requests) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /opt/anaconda3/lib/python3.11/site-packages (from requests) (2024.2.2)
```

- The next step is to add this library to the file we are writing code in
- I am using pycharm to write code so first make a new file and then go to the bottom and find python packages.



- Click on python packages search for the requests library and download it



- Now below is a generalised code, use the following Python code to retrieve data from the Open Charge Map API and process it:

```
import requests

# Define the API endpoint and parameters
api_url = "https://api.openchargemap.io/v3/poi/"
params = {
    "output": "json",
    "countrycode": "AU",
    "maxresults": 500, # Increase max results to ensure all
stations are retrieved
    "latitude": -37.8136, # Latitude of Melbourne
    "longitude": 144.9631, # Longitude of Melbourne
```

```
    "distance": 50, # Search radius in kilometers

    "key": "82aaef2d-e9b8-4caa-8c0f-e7869973a241" # Get your
API key from Open Charge Map
}
```

```
# Make the API request
response = requests.get(api_url, params=params)
```

```
# Check if the request was successful
if response.status_code == 200:
    data = response.json()
    # Process the data as needed
    for station in data:
        title = station["AddressInfo"]["Title"]
        latitude = station["AddressInfo"]["Latitude"]
        longitude = station["AddressInfo"]["Longitude"]
```

```
        # Check if "UsageType" exists and is not None
        if "UsageType" in station and station["UsageType"] is
not None:
            parking_status = station["UsageType"]["Title"]
        else:
            parking_status = "N/A"
```

```
        # Check if "UsageCost" exists and is not None
        if "UsageCost" in station and station["UsageCost"] is
not None:
            usage_cost = station["UsageCost"]
```

```
else:
    usage_cost = "N/A"
```

```
print(
    f"Title: {title}, Latitude: {latitude}, Longitude: {longitude}, Parking Status: {parking_status}, Usage Cost: {usage_cost}")
else:
    print("Failed to fetch data. Status code:", response.status_code)
```

The result I got is like this-

```
capstone main.py
main.py 1.py
if response.status_code == 200 for station in data if "UsageCost" in station and s...
Run: main
Title: Mornington Super Charger, Latitude: -38.2408896222658556, Longitude: 145.07444843528583, Parking Status: Public - Membership Required, Usage Cost: N/A
Title: North Bellarine Aquatic Centre, Latitude: -38.188088367705056, Longitude: 144.55520001587047, Parking Status: Public - Membership Required, Usage Cost: $0.45/kWh
Title: Victoria Street, Latitude: -37.42305178823792, Longitude: 144.562626054862, Parking Status: Public - Membership Required, Usage Cost: $0.45 per kWh
Title: 50 Learmonth Street, Latitude: -38.26770792670353, Longitude: 144.6589785355144, Parking Status: Public - Membership Required, Usage Cost: $0.45 per kWh
Title: Kilmore Library, Latitude: -37.294362, Longitude: 144.952084, Parking Status: Public, Usage Cost: $0.40 per kWh
Title: Geelong Supercharger, Latitude: -38.064796504586276, Longitude: 144.38238251338635, Parking Status: Public - Membership Required, Usage Cost: $0.78/kWh
Title: Hastings, Latitude: -38.30823, Longitude: 145.191932, Parking Status: Public, Usage Cost: Payment Required $0.45 per kWh
Title: Kingston Village Square, Latitude: -38.24661134219002, Longitude: 144.54005635897056, Parking Status: Public - Membership Required, Usage Cost: Payment Required $0.25 per kWh (at all times)
Title: Sorrento , Latitude: -38.33874299773124, Longitude: 144.73713563519414, Parking Status: Public - Membership Required, Usage Cost: 0.40/kWh + While parked, not charging: 4.00/hr after 60 mins
Title: Rosebud RSL, Latitude: -38.368575, Longitude: 144.905885, Parking Status: Public, Usage Cost:
Title: bp pulse Rye, Latitude: -38.37130601612926, Longitude: 144.8277248633259, Parking Status: Public - Membership Required, Usage Cost: 55c/kWh
Title: High St Parking, Latitude: -37.35582370879025, Longitude: 144.52881756976018, Parking Status: Public - Membership Required, Usage Cost: $0.40 per kWh. Parking overstay charges may apply
Title: Bru4U Geelong , Latitude: -38.1722195376013, Longitude: 144.36787923669385, Parking Status: Private - For Staff, Visitors or Customers, Usage Cost: free
Title: Broadford Library, Latitude: -37.20295, Longitude: 145.048954, Parking Status: Public, Usage Cost: $0.40 per kWh
Title: RACV Solar Grovedale, Latitude: -38.197594122360314, Longitude: 144.34487185372615, Parking Status: Public - Membership Required, Usage Cost: $0.25 per kWh
Title: Blue Tongue Industries, Latitude: -38.197502248120415, Longitude: 144.34395122104382, Parking Status: Public - Membership Required, Usage Cost: $0.30 per kWh
Title: Armstrong Creek East Community Hub, Latitude: -38.231759419255376, Longitude: 144.36927996323692, Parking Status: Public - Membership Required, Usage Cost: Free
Title: Leisurelink Aquatic & Recreation Centre, Latitude: -38.19749674302702, Longitude: 144.3210373885172, Parking Status: Public - Membership Required, Usage Cost: $0.45 per kWh
Title: Armstrong Creek Shopping Centre, Latitude: -38.22890721146294, Longitude: 144.3389741604681, Parking Status: Public, Usage Cost: Free
Title: Deakin University Waurr Ponds, Latitude: -38.19671590570452, Longitude: 144.29935439201148, Parking Status: Public - Membership Required, Usage Cost: Free
Title: Evie Flinders, Latitude: -38.474526184383976, Longitude: 145.01862334358879, Parking Status: Public - Membership Required, Usage Cost: $0.45 per kWh
Title: Kyneton, Latitude: -37.246931, Longitude: 144.452169, Parking Status: Public - Membership Required, Usage Cost: Payment Required $0.40 per kWh parking overstay charges may apply.
Title: Evie Meredith, Latitude: -37.84437816290739, Longitude: 144.0768346824019, Parking Status: Public - Membership Required, Usage Cost: $0.45 per kWh
Title: Yea Supercharger, Latitude: -37.21006161015167, Longitude: 145.4261473877039, Parking Status: Public - Membership Required, Usage Cost: $0.78/kWh
Title: Penguin Parade Car Park, Latitude: -38.505470002342136, Longitude: 145.14849508723148, Parking Status: Public - Membership Required, Usage Cost: AUD 0.37 per kWh
```

Consequently, in order to convert this file to a CSV file, I changed my code to convert the data I am receiving directly to a CSV file.

Below is the code used to turn the data into a CSV file.

```
import csv
import os
import requests
```

```
# Define the API endpoint and parameters
api_url = "https://api.openchargemap.io/v3/poi/"
params = {
    "output": "json",
    "countrycode": "AU",
    "maxresults": 500, # Increase max results to ensure all
stations are retrieved
    "latitude": -37.8136, # Latitude of Melbourne
    "longitude": 144.9631, # Longitude of Melbourne
    "distance": 75, # Search radius in kilometers
    "key": "82aaef2d-e9b8-4caa-8c0f-e7869973a241" # Get your
API key from Open Charge Map
}
```

```
# Make the API request
response = requests.get(api_url, params=params)
```

```
# Check if the request was successful
if response.status_code == 200:
    data = response.json()
```

```
# Specify CSV file path on the desktop

desktop_path = os.path.join(os.path.expanduser("~/Desktop"), "capstoneB")

os.makedirs(desktop_path, exist_ok=True) # Create the folder if it doesn't exist

csv_file_path = os.path.join(desktop_path, "ev_stations.csv")
```

```
try:

    # Open CSV file in write mode

    with open(csv_file_path, mode="w", newline="", encoding="utf-8") as csv_file:

        # Create a CSV writer object

        csv_writer = csv.writer(csv_file)
```

```
    # Write header row

    csv_writer.writerow(["Title", "Latitude", "Longitude", "Parking Status", "Usage Cost"])
```

```
    # Process the data and write rows to CSV

    for station in data:

        title = station.get("AddressInfo", {}).get("Title", "N/A")

        latitude = station.get("AddressInfo", {}).get("Latitude", "N/A")

        longitude = station.get("AddressInfo", {}).get("Longitude", "N/A")
```



```
        # Check if "UsageType" exists and is not None
        if "UsageType" in station and
station["UsageType"] is not None:
            parking_status = station["UsageType"]
["Title"]
        else:
            parking_status = "Information not
available"
```

```
        # Check if "UsageCost" exists and is not None
        if "UsageCost" in station and
station["UsageCost"] is not None:
            usage_cost = station["UsageCost"]
        else:
            usage_cost = "Cost information not
available"
```

```
        # Write row to CSV
        csv_writer.writerow([title, latitude,
longitude, parking_status, usage_cost])
```

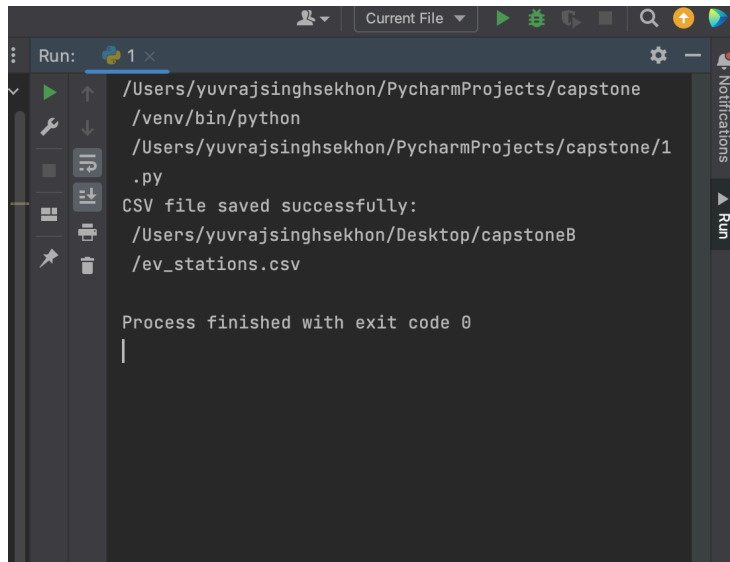
```
        print(f"CSV file saved successfully:
{csv_file_path}")
```

```
    except Exception as e:
        print("Error occurred while creating the CSV file:",
e)
```

```
else:
```

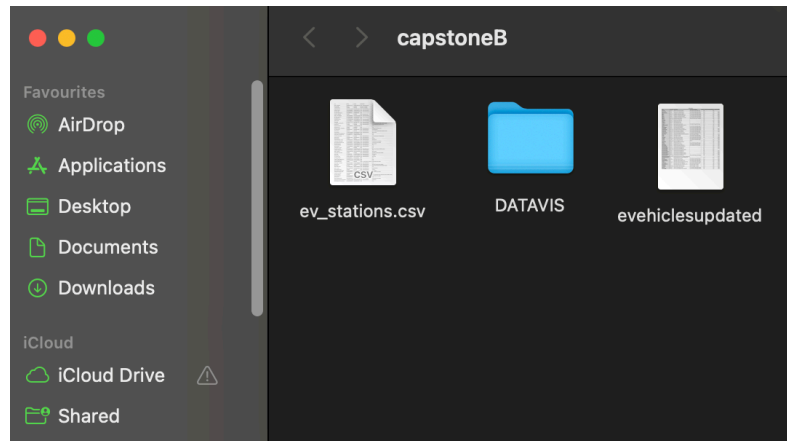
```
print("Failed to fetch data. Status code:",  
response.status_code)
```

I successfully converted all of the data to a CSV file in my desired directory, as evidenced by the output I saw on the console.



The screenshot shows a terminal window with the following output:

```
Run: 1 x  
/Users/yuvrajsinghsekhon/PycharmProjects/capstone  
/venv/bin/python  
/Users/yuvrajsinghsekhon/PycharmProjects/capstone/1  
.py  
CSV file saved successfully:  
/Users/yuvrajsinghsekhon/Desktop/capstoneB  
/ev_stations.csv  
  
Process finished with exit code 0  
|
```



With the provided steps and customizable code, you can seamlessly extract comprehensive information about EV stations using the Open Charge Map API. Whether you need data on charging costs, parking availability, or any other details, this solution offers flexibility and efficiency. Feel empowered to tailor the code to your specific requirements, ensuring that you can gather the precise insights needed for your projects.

Happy data extraction!!!