Matrixian Group

Home Assignment:

**Steps:**

1. **Extract all the postal codes:** Find the links to all the areas in Amsterdam

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

To verify that this is the exact range of **valid** postal codes in Amsterdam available in Thuisbezorgt, I checked for all the postal codes in Amsterdam via <https://postcodebijadres.nl/amsterdam>. Then, I entered the values which are not shown in the list but shown on the postcodebijadres website (for instance: 1000, 1010, 1109). That way I **verified** that the postal codes shown on the Thuisbezorgt page.

I also found out that the valid URL could be written in two ways, either using the city + postal code 4 digits only:



Or using the full form, as taken from the home page:



Using my Python program, I stored all these links in a txt file (to fetch it easily without the need to find those links every single time) 🡪 *postal\_codes\_amsterdam.txt*

* Only afterwards, I realized that what I needed was **only** the **postal code itself**, to make the **api** call.
* It was important to **remove duplicates** (there are some areas with the same postal codes, but with a different name. However, they result in the same feed)

1. For each area, get the links to all the restaurants in the area

When I tried to scrape on the feed of a postal code (e.g., 1011), the driver was redirected to this page:

A screenshot of a computer

Description automatically generated

Which means – this website is protected by CloudFlare.

**A screenshot of a computer

Description automatically generated**

I found the XHR that renders the feed in Amsterdam-1011:

<https://cw-api.takeaway.com/api/v33/restaurants?deliveryAreaId=935430&postalCode=1011&lat=52.37275186895077&lng=4.905706126949892&limit=0&isAccurate=true&filterShowTestRestaurants=false>

It will be easier to just send the same api request by filtering based on the **postal code** (the 4 digit) **only**!

[https://cw-api.takeaway.com/api/v33/restaurants? postalCode=1011&limit=0&isAccurate=true&filterShowTestRestaurants=false](https://cw-api.takeaway.com/api/v33/restaurants?deliveryAreaId=935430&postalCode=1011&lat=52.37275186895077&lng=4.905706126949892&limit=0&isAccurate=true&filterShowTestRestaurants=false)

I used Fiddler to copy the session headers, and use it in cloudscraper:

A screenshot of a computer

Description automatically generated

I wanted to learn how the JSON response is constructed, and fetch the data accordingly. Therefore, I saved the response into a json file called ***amsterdam-1011.json.***

Then, using UltraEdit, I inspected the JSON better, and found out some of the requested details.

That way, I knew what kind of values I should get from this response, and which I must take from within the restaurant-page request itself.

A screen shot of a computer

Description automatically generated

The details I took from this response for building the initial restaurants JSON:

**Restaurant name** -> [‘brand’][‘name’]

**Restaurant Slug** -> [‘primarySlug’]

**Restaurant price range** -> [‘priceRange’]

**Restaurant popularity, rating value and rating count** -> [‘popularity’] & [‘rating’][‘score’] & [‘rating’][‘votes’]

**Restaurant cuisine types** -> [‘cuisineTypes’]

**Minimum order amount** (bonus: minimum order amount for each delivery area)

[‘shippingInfo’][‘delivery’][‘minOrdervalue’]  
I could have added the minOrderValue per postal code by adding an inner dictionary with the postal\_codes as keys, and minOrderValue as the value.  
However, I missed this assignment while reading, therefore did not implement it yet ☹

**Delivery area / list of postal** codes -> by adding the postal code to the list of postal codes of the entry in the dictionary

I have built a json that contains the required data of each restaurants 🡪 ***amsterdam\_restaurants.json***

1. I followed the same procedure with the restaurants details themselves. I first inspected the response of a specific restaurant (***thai-street-bar.json***), by saving it as a JSON and using UE.

A screenshot of a computer

Description automatically generated

The details given within this response:

**Restaurant address** (bonus: store street name and house number separately) -> [‘location’][‘streetName’] & [‘location’][‘streetNumber’]

**Restaurant logo link and website** -> [‘brand’][‘logoUrl’] & [‘minisiteUrl’]

**Restaurant delivery times** -> [‘delivery’][‘times’][DAY(from 0 to 6)][‘formattedStart’] to [‘delivery’][‘times’][DAY(from 0 to 6)][‘formattedEnd’]

**Restaurant item categories** -> [‘menu’][‘categories’][(INDEX 0-6)][‘name’]

I loaded the json taken from ***amsterdam\_restaurants.json***, and then added data to each restaurant object. Now each restaurant (only the first 50, due to rate limit issues) contains the full details (+ some bonus data).

Then, I saved all the updated objects, and built a json that contains all the required data of each restaurants 🡪***restaurants\_amsterdam\_full\_details.json***.

1. I then loaded this json and stored each of the restaurants as a document in a **MongoDB collection**.