Contents

1. *How to Run (Page2)*
1. *Lvl0_use_case_diagram (Page 3)*
2. *Lvl1_run_use_case_diagram (Page 4)*
3. *Lvl1_save_use_case_diagram (Page 5)*
4. *Lvl1_merge_data_diagram (Page 5)*
5. *Lvl2_.main_use_case_diagram (Page 6)*
6. *Objects (Page7)*
7. *Final Reports (Page8, Page9)Türkçe*

HOW TO RUN

dbname: tayfun_karakavuz
source .env/bin/activate

1. **With Docker**
   If you are not going to start the project via local, please set the mongodb uri to `client = MongoClient("mongodb://host.docker.internal:27017")` in save_data.py()

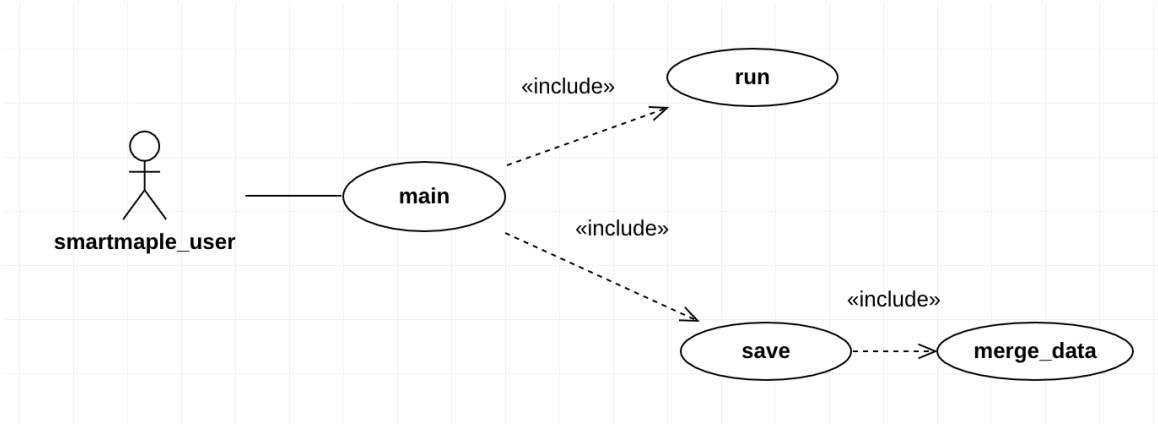
   Dockerfile and **docker-compose.yml** files have been created. You can run the project with the **docker-compose up** command.

```
CMD ["python3", "run.py"]
```

```
environment:
     - MONGODB_URI=mongodb://host.docker.internal:27017
```

2. **Local**
   If you are not going to start the project via the docker container, please set the mongodb uri to `client = MongoClient("mongodb://localhost:27017")` in save_data.py()
   Start the run.py with **% python3 run.py**

USE CASE DIAGRAMS



lvl0_use_case_diagram

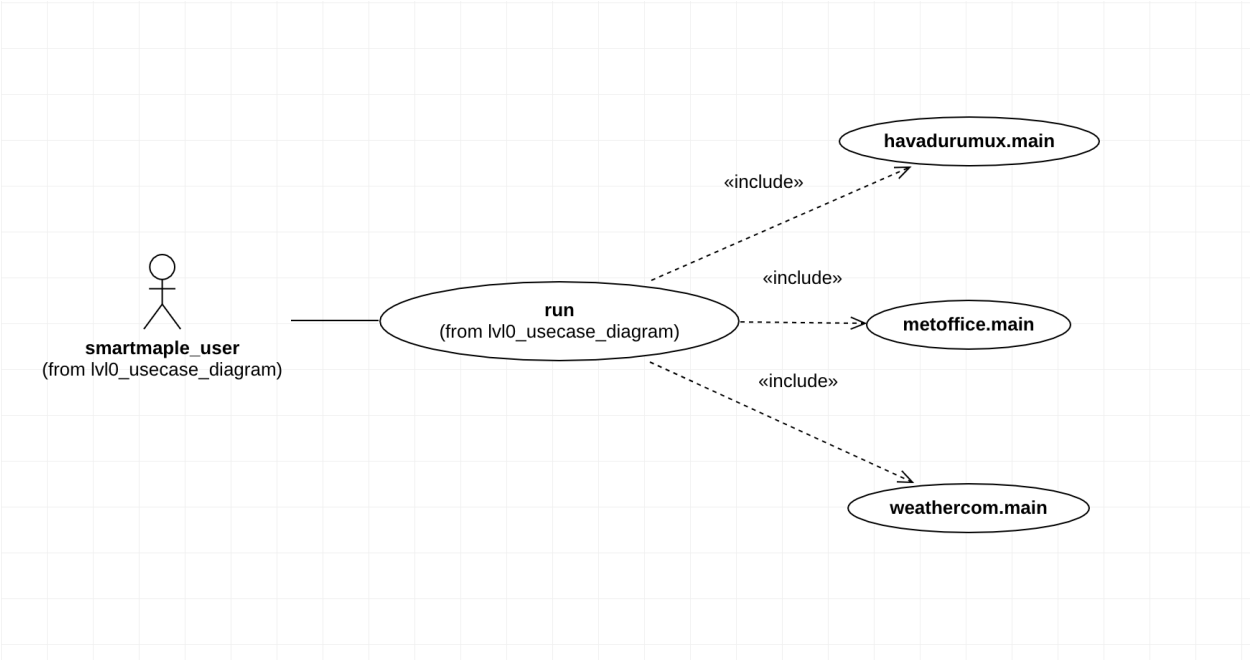
Description: This use case represents retrieving weather data.

Actors: smartmaple_user

Flow of Events:

The **user** initiates the script execution. (1)
run() retrieves weather data from multiple sources. (2)
merge_data() After fetching data from different sources, the script merges the data. (3)
save() The merged data is then saved to a MongoDB database. (4)
The script calculates the elapsed time during its execution. (5)
The script displays the total elapsed time in seconds. (6)

Postconditions: Weather data is fetched from various sources, merged, and saved to db.



lvl1_run_use_case_diagram

Description: This use case represents fetching weather data from multiple resources
Actors: smartmaple_user

Flow of Events:

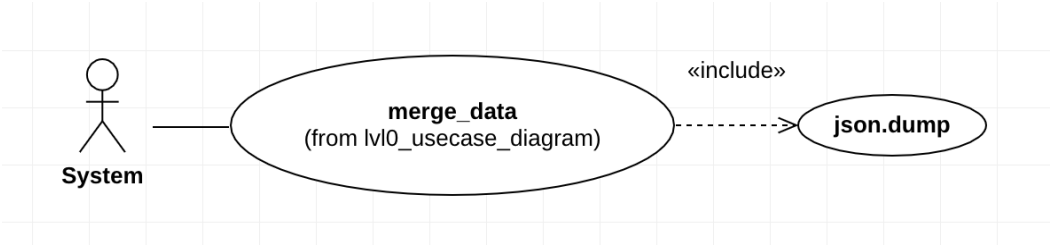
The **user** initiates the script execution. (1)
run() creates asynchronous HTML session (asession) and fetches weather data from havadurumux, metoffice, and weathercom. (2)

Postconditions: Weather data is fetched from various sources.

lvl1_save_use_case_diagram

Description: This is the primary use case that represents saving data to a MongoDB database.

Includes sub-actions:
Load final JSON file (1)
Connect to MongoDB (2)
Save Data to MongoDB (3)
Close MongoDB Client (4)

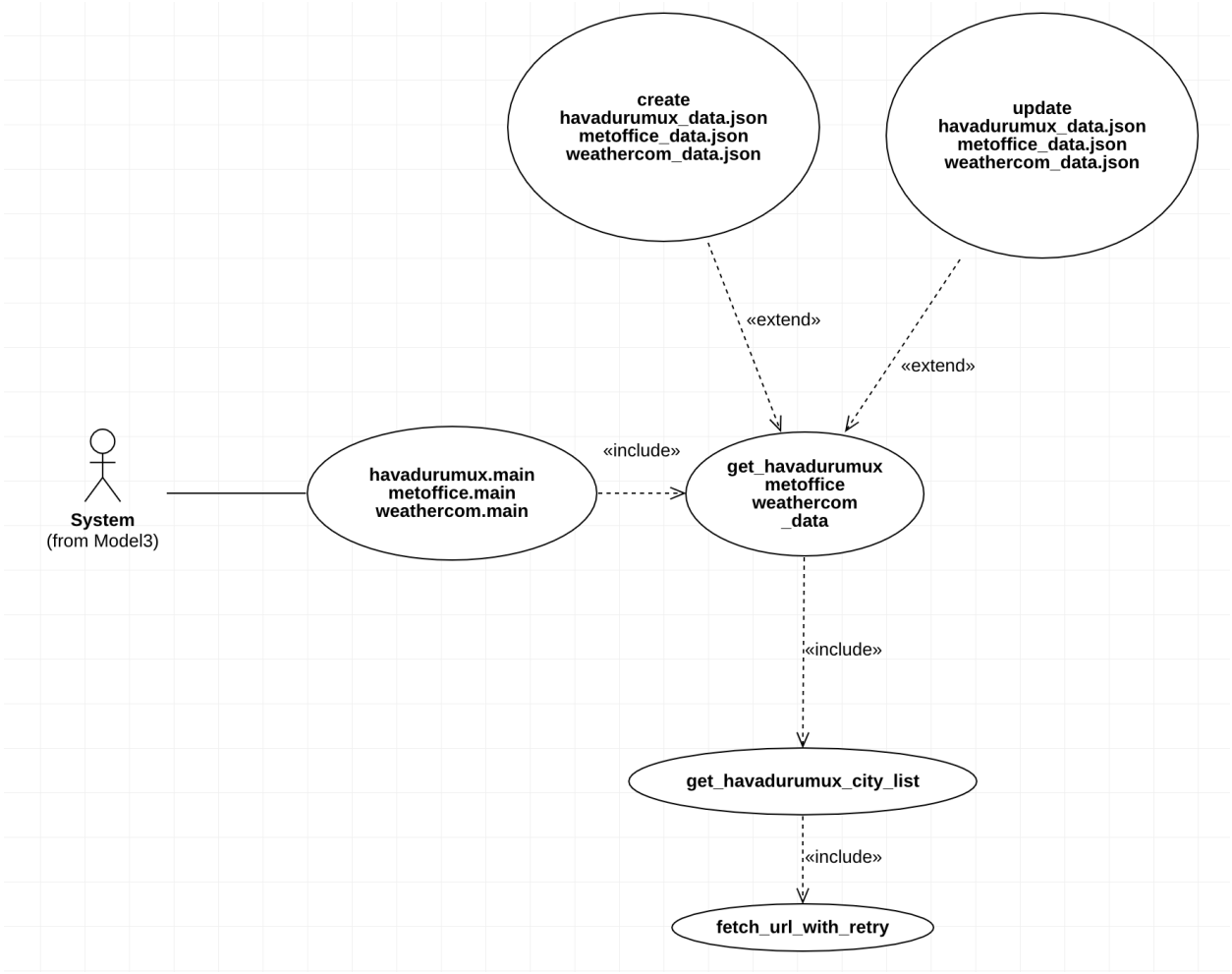


Lvl1_merge_data_diagram

Description: This primary use case represents merging weather data from local sources and saving it as final data.

Includes sub-actions:
Load metoffice Data (1)
Load havadurumux Data (2)
Load weathercom Data (3)
Merge and filter Data (4)
Save Final Data (5)

Actor: System
The system itself executes the script and performs the specified actions.



Lvl2_.main_use_case_diagram

Description: This is the primary use case that represents the process of creating or updating Havadurumux, metoffice, and weathercom weather data.

I**ncludes sub-actions:**

Fetch havadurumux, metoffice, and weathercom* city urls (1)
Create city urls JSON File (2)
Fetch havadurumux, metoffice, and weathercom Data with City List JSON File(3)
create or update JSON File (4)**
Filter Cities and update JSON File (5)

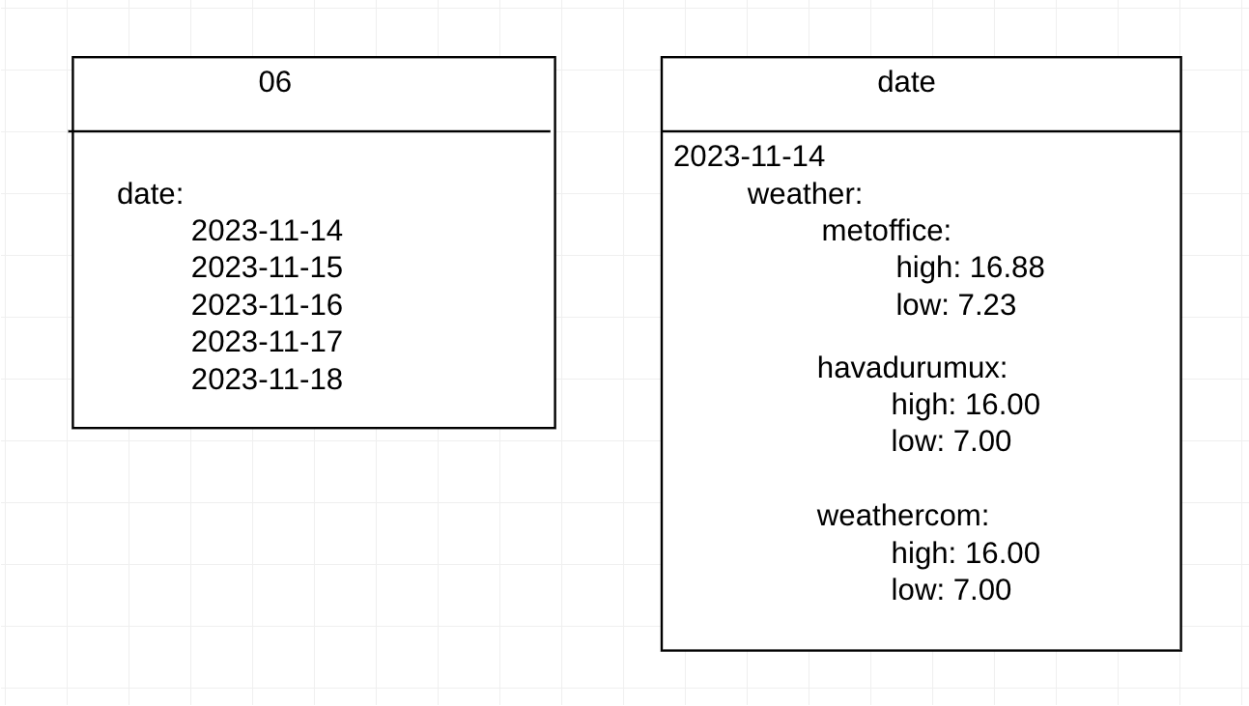
Actor: System

The system itself, which executes the script and performs the specified actions.

** The URLs to be retrieved from Weather.com for 81 provinces are provided with selenium automation. This automation was used because weather.com encodes the links of these provinces and does not provide an HTML page containing all the cities. This script will be sent to you, but it will not run in the main function.*

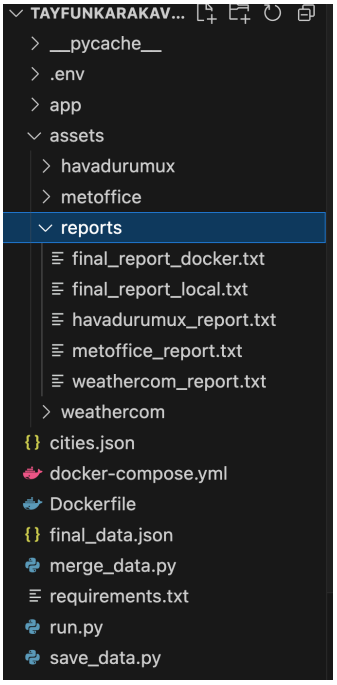
*** metoffice provides data for only 77 regions of Turkey. 38 of these regions include provinces. Others include districts, transportation zones, or cultural districts. These data are not saved in the database. This filter was made by mapping 81 city objects in the previously created cities.json file.*

OBJECTS

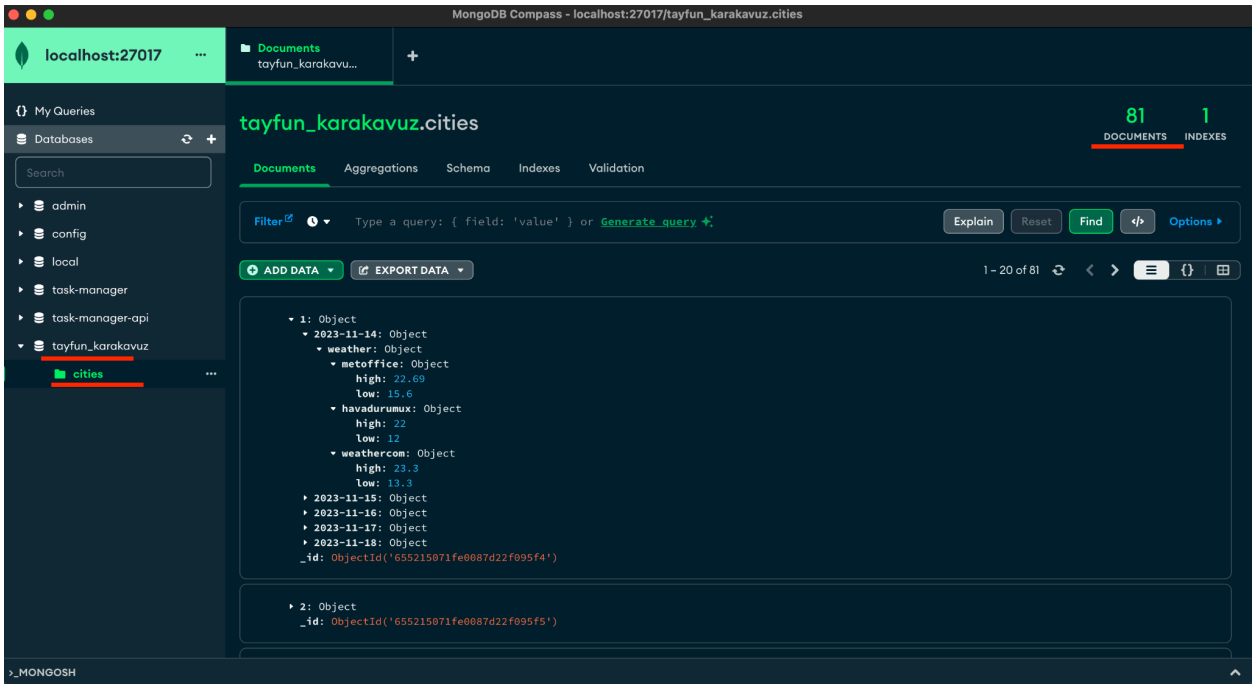


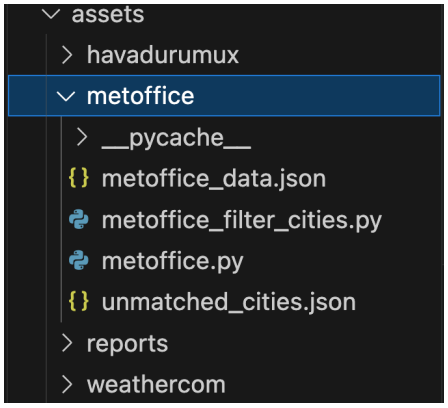
Object

Final Reports

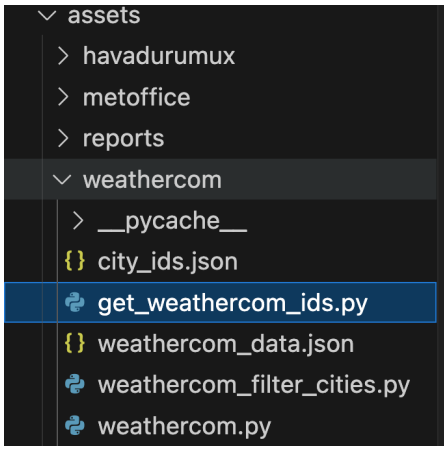


- Proje hem localde hem de docker üzerinden çalıştırılabiliyor. Kod raporlar reports klasöründe paylaşılmıştır.
- Proje **venv** üzerinden çalışıyor.
- Çıktıların tutarlılığı mongodb yöntemleri ile test edilmiştir.
- Projenin tamamlanma süresi kodun bitişinde yazdırılmaktadır. Benim sistemimle projenin çalışma süresi yaklaşık 2 dakikadır.
  Total elapsed time: 102.08 seconds
- Projeyi modüllere ayırarak tamamlamaya çalıştım. Toplamda **4 modül**den oluşan bu proje ana modül ile başlatılıyor, ardından verileri kazıdığım websitelerinin modülleri başlatılıyor. Son olarak veri tabanına kayıt işlemini gerçekleştiren modül başlatılıyor.
- Modülleri **async** tasarlamaya çalıştım. Kazıma işlemleri aynı loop üzerinden aynı anda gerçekleşiyor. Bu durum projenin tamamlanma süresini kısaltıyor.





- metoffice.py modülünde unmatched_cities.json dosyası yukarıdaki kullanım durumlarında belirttiğim özel bölgeleri içeriyor. Bu bölgeler veri tabanına dahil edilmemiştir. Filtreleme için metoffice_filter_cities.py scripti çalıştırılıyor.



- weather.com şehirleri listeleyecek bir sayfa sunmadığı için şehirlerin linklerini almak için diğer modüllerde olmayan bir fonksiyon çalıştırdım. Bu fonksiyon projeye dahil edilmemiştir. Bu fonskiyon için selenium import edilmiş, bir otomasyon oluşturulmuş, bu otomasyon ile arama kutusuna 81 ilin isimleri yazılmış ve list-box' tan tüm şehirlerin linkleri çekilmiştir. Bu fonskiyon city_ids.json dosyasını oluşturuyor. Baes_url ile city_id birleştirilerek her şehirin sayfası ziyaret edilebiliyor. Bu fonskiyonu çalıştırmak isterseniz ayrıca **chromedriver** bulundurmanız gerekiyor. Bu dosyayı da sizinle paylaşacağım. Bu driver chrome 123 versiyonu ile stable olmadığı için bu adımı projeye dahil etmedim.
- Kazıma işlemi yapılan websitelerden kazıma sonucu assets klasörüne json dosyası olarak kaydediliyor. (Örn. havadurumux_data.json). Ardından belirttiğiniz yapıya uygun filtreleme ve manipülasyon işlemleri gerçekleştiriliyor ve bu json dosyası güncelleniyor. Son olarak her websitesinden elde edilen json dosyaları merge ediliyor ve final_data.json dosyası oluşturuluyor. Bu dosya save_data modülü ile açılıyor ve veriler mongodb/tayfun_karakavuz veritabanına kayıt ediliyor.