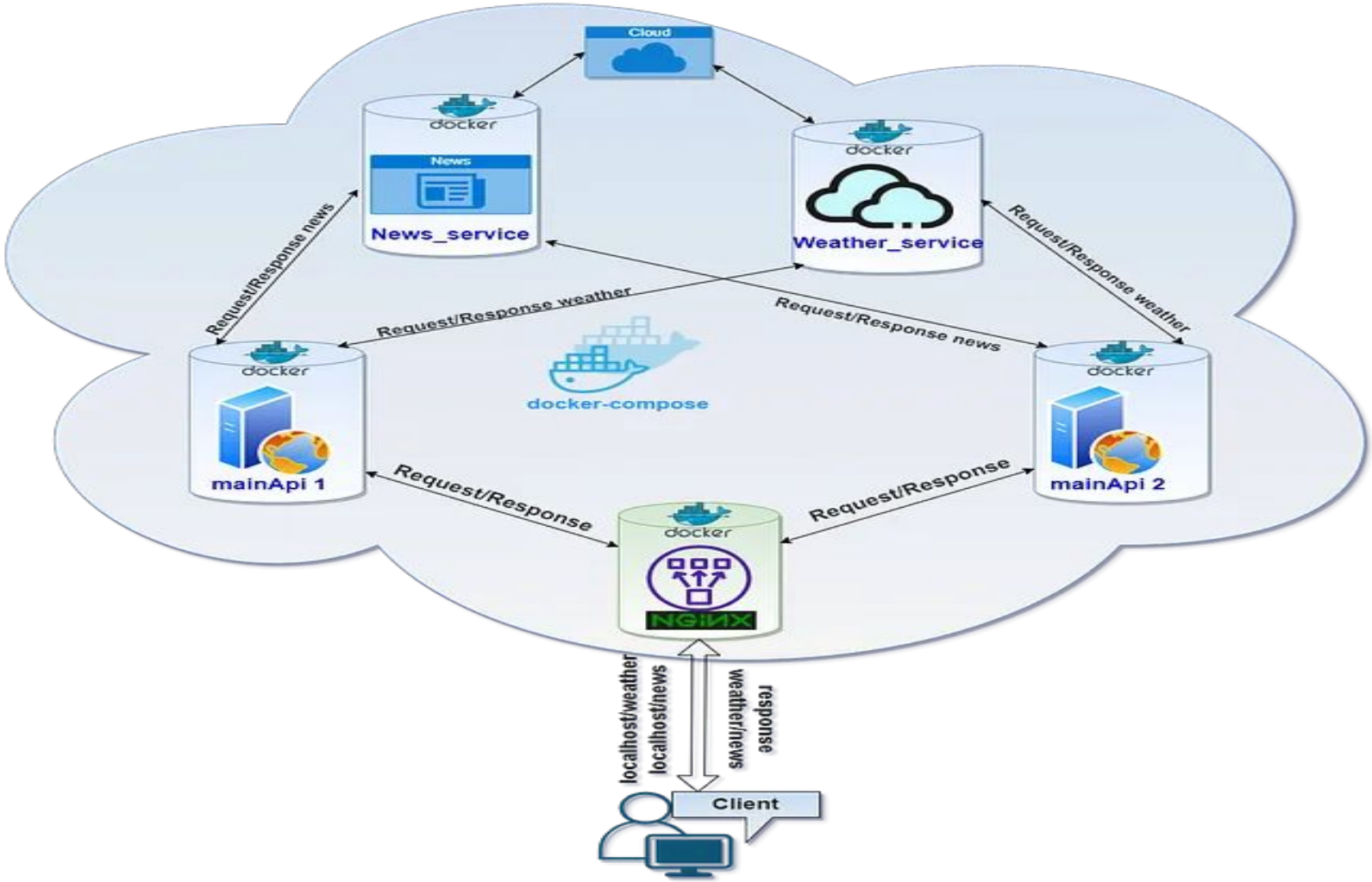
exemple of web application with Python's Flask web framework using Docker and docker-compose

what is this project about ?

The main goal of this project is to demonstrate how to build, test, and deploy microservices-based web applications using containerization tools like Docker and Docker Compose.

The project consists of two microservices: news and weather. The news service provides an API endpoint to fetch the latest news headlines from various news sources using the NewsAPI. The weather service provides an API endpoint to fetch the current weather information for a given city using the OpenWeatherMap API. + image


Cloud
docker
News
News_service
docker
Weather_service
Request/Response news
Request/Response weather
Request/Response weather
Request/Response news
docker
docker-compose
mainApi 1
mainApi 2
Request/Response
Request/Response
docker
NGINX
localhost/weather
localhost/news
weather/news
response
Client

Tools used

Flask

is a lightweight and flexible web framework for Python that is used to build web applications. It provides a simple and easy-to-use interface for handling web requests and generating responses. Flask is often used to build RESTful APIs and web services.

Docker

is a containerization platform that allows developers to package their applications and dependencies into portable containers. Containers provide a way to run applications consistently across different environments, which helps to ensure that the application behaves the same way in development, testing, and production. Docker containers can be easily deployed to cloud platforms like AWS, Azure, and Google Cloud.

Docker-compose

is used to define and run multi-container Docker applications. The docker-compose.yml file defines the services that make up the application, including the News and Weather services, their respective databases, and the NGINX reverse proxy.

Postman

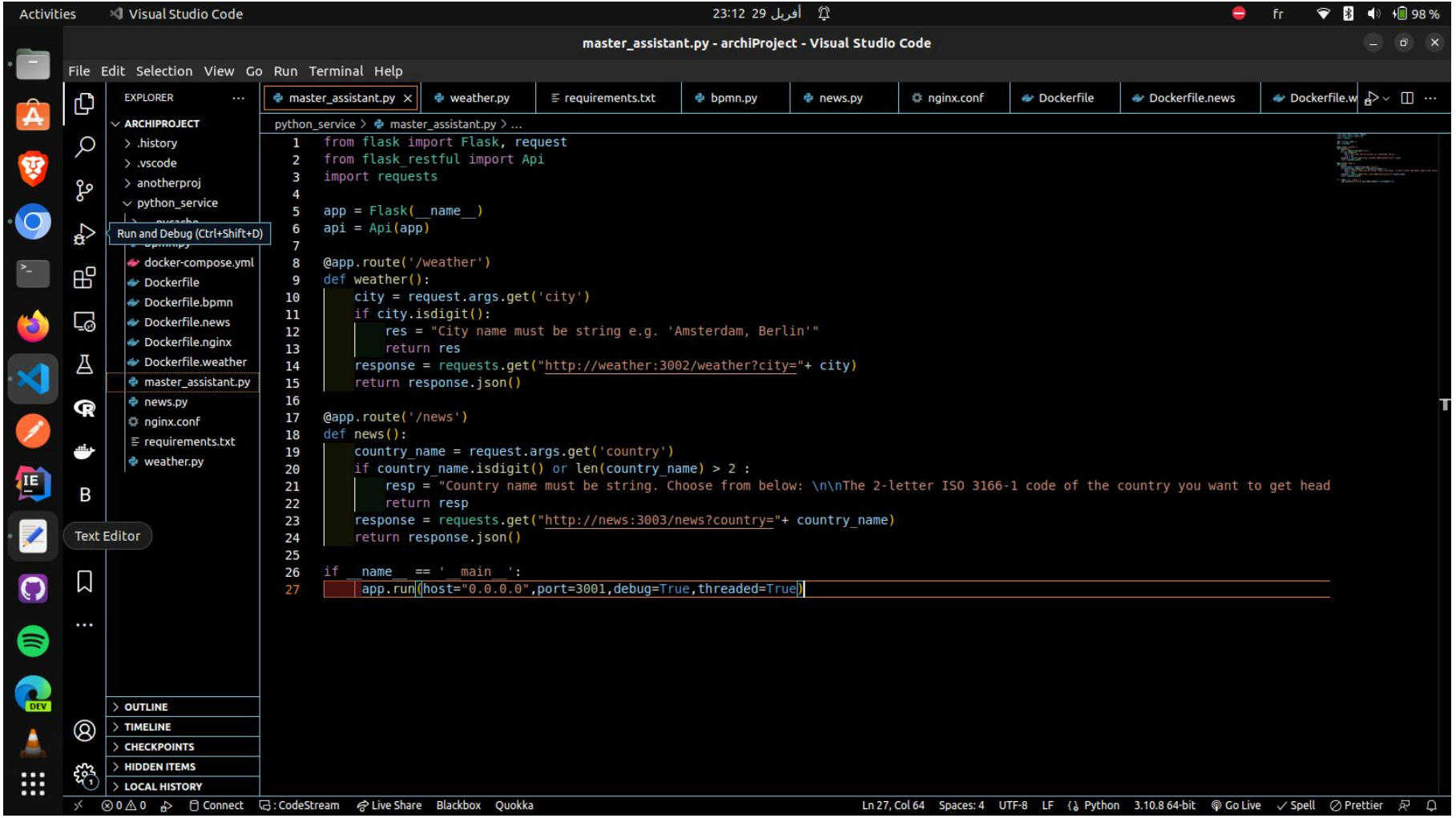
is a popular API development tool that allows developers to create, test, and document APIs easily.

Steps followed

1)Creating flask services

master_assistant.py

The "/weather" route expects a query parameter called "city", retrieves it from the request object using request.args.get('city'), and sends a GET request to a separate service (which must be running on the network at address "weather:3002") to retrieve the current weather for the specified city. The response from the weather service is returned to the client in JSON format.
The "/news" route expects a query parameter called "country", retrieves it from the request object using request.args.get('country'), and sends a GET request to another separate service (which must be running on the network at address "news:3003") to retrieve the latest news headlines for the specified country. The response from the news service is also returned to the client in JSON format.
If the "city" or "country" query parameters are not in the expected format (either an invalid string or a number), an error message is returned to the client.


Activities
Visual Studio Code
23:12 29 أفريل
fr
98 %
master_assistant.py - archiProject - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ARCHIPROJECT
.history
.vscode
anotherproj
python_service
Run and Debug (Ctrl+Shift+D)
docker-compose.yml
Dockerfile
Dockerfile.bpmn
Dockerfile.news
Dockerfile.nginx
Dockerfile.weather
master_assistant.py
news.py
nginx.conf
requirements.txt
weather.py
Text Editor
OUTLINE
TIMELINE
CHECKPOINTS
HIDDEN ITEMS
LOCAL HISTORY
master_assistant.py
weather.py
requirements.txt
bpmn.py
news.py
nginx.conf
Dockerfile
Dockerfile.news
Dockerfile.w
python_service > master_assistant.py > ...
1 from flask import Flask, request
2 from flask_restful import Api
3 import requests
4
5 app = Flask(__name__)
6 api = Api(app)
7
8 @app.route('/weather')
9 def weather():
10 city = request.args.get('city')
11 if city.isdigit():
12 res = "City name must be string e.g. 'Amsterdam, Berlin'"
13 return res
14 response = requests.get("http://weather:3002/weather?city="+ city)
15 return response.json()
16
17 @app.route('/news')
18 def news():
19 country_name = request.args.get('country')
20 if country_name.isdigit() or len(country_name) > 2 :
21 resp = "Country name must be string. Choose from below: \n\nThe 2-letter ISO 3166-1 code of the country you want to get head
22 return resp
23 response = requests.get("http://news:3003/news?country="+ country_name)
24 return response.json()
25
26 if __name__ == '__main__':
27 app.run(host="0.0.0.0",port=3001,debug=True,threaded=True)
Connect
: CodeStream
Live Share
Blackbox
Quokka
Ln 27, Col 64
Spaces: 4
UTF-8
LF
Python
3.10.8 64-bit
Go Live
Spell
Prettier

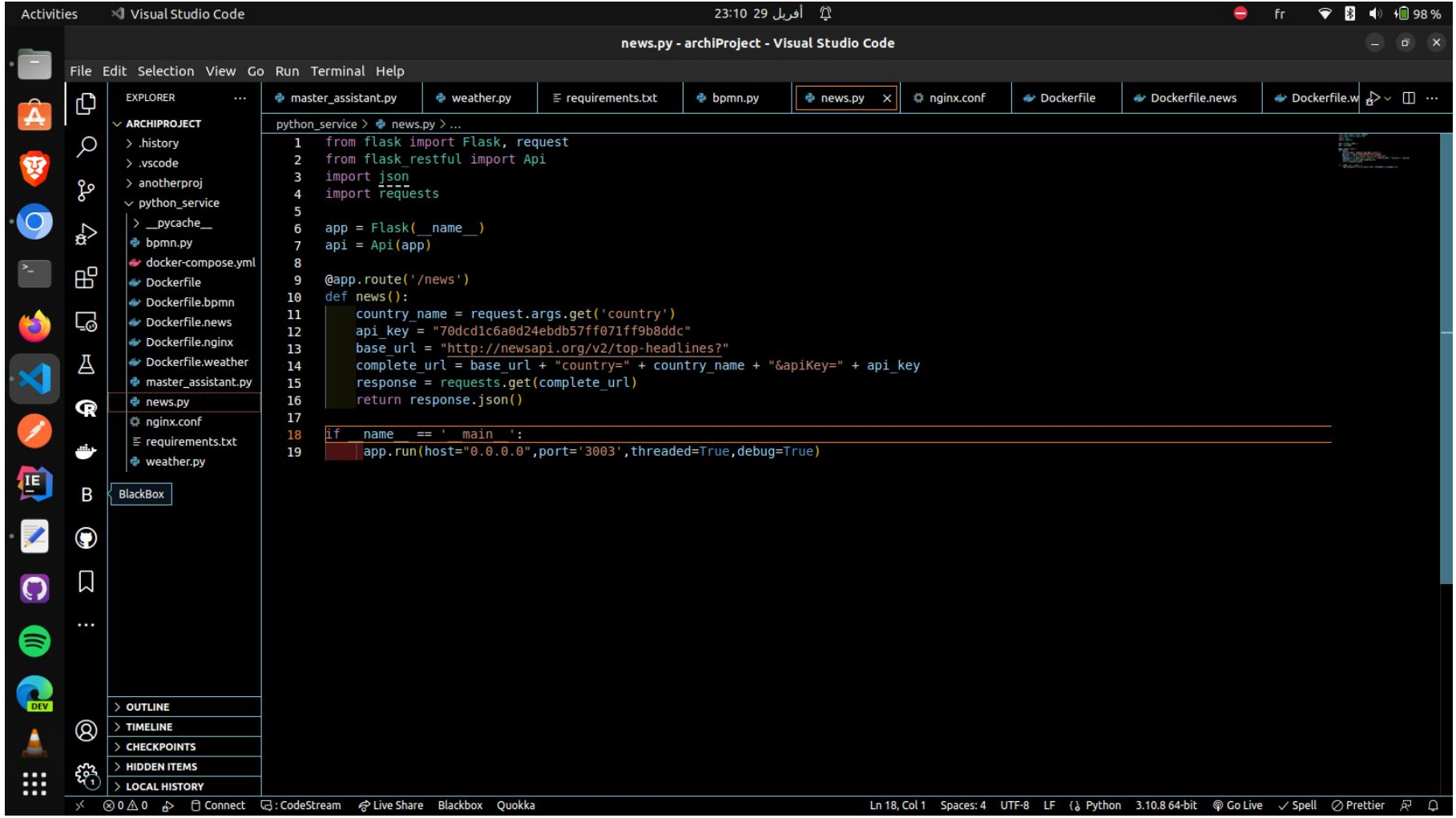
1)Creating flask services

news.py

The code creates an instance of the Flask application and an instance of the API using the Flask instance. It then defines a route '/news' that accepts GET requests.

Inside the news function, the code gets the country name from the query string using the request.args.get() method. It then constructs a URL to request news articles from the NewsAPI service using the country name and the NewsAPI API key.

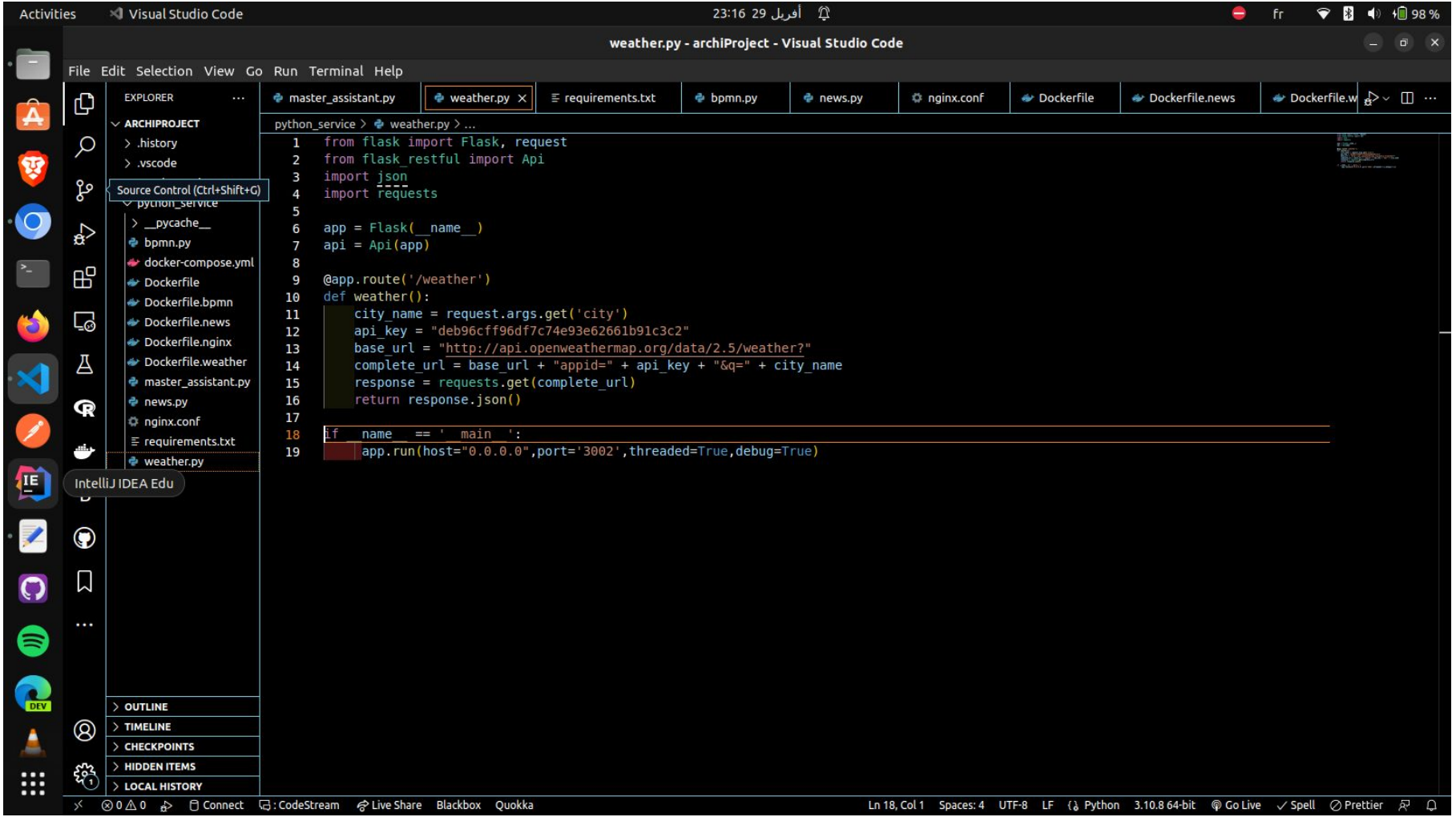
After that, it sends an HTTP GET request to the constructed URL using the requests module, and returns the JSON response from the NewsAPI service using response.json().


Activities
Visual Studio Code
23:10 29 أفريل
fr
98 %
news.py - archiProject - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ARCHIPROJECT
.history
.vscode
anotherproj
python_service
__pycache__
bpmn.py
docker-compose.yml
Dockerfile
Dockerfile.bpmn
Dockerfile.news
Dockerfile.nginx
Dockerfile.weather
master_assistant.py
news.py
nginx.conf
requirements.txt
weather.py
BlackBox
OUTLINE
TIMELINE
CHECKPOINTS
HIDDEN ITEMS
LOCAL HISTORY
master_assistant.py
weather.py
requirements.txt
bpmn.py
news.py
nginx.conf
Dockerfile
Dockerfile.news
Dockerfile.w
python_service > news.py > ...
1 from flask import Flask, request
2 from flask_restful import Api
3 import json
4 import requests
5
6 app = Flask(__name__)
7 api = Api(app)
8
9 @app.route('/news')
10 def news():
11 country_name = request.args.get('country')
12 api_key = "70dcd1c6a0d24ebdb57ff071ff9b8ddc"
13 base_url = "http://newsapi.org/v2/top-headlines?"
14 complete_url = base_url + "country=" + country_name + "&apiKey=" + api_key
15 response = requests.get(complete_url)
16 return response.json()
17
18 if __name__ == '__main__':
19 app.run(host="0.0.0.0",port='3003',threaded=True,debug=True)
Connect
: CodeStream
Live Share
Blackbox
Quokka
Ln 18, Col 1
Spaces: 4
UTF-8
LF
Python
3.10.8 64-bit
Go Live
Spell
Prettier

1)Creating flask services

weather.py

single route '/weather' that retrieves the current weather data for a specified city using the OpenWeatherMap API. It uses the request module to get the city name from the query parameters of the request URL. The OpenWeatherMap API key is stored in a variable called api_key and the base URL for the API is stored in base_url. These values are used to construct a complete URL for the API request which is then made using the requests.get() method. The JSON response from the API is returned as the response from the route. Finally, the __name__ variable is used to check if the script is being run directly and the Flask app is run on the local host at port 3002 with debug mode enabled.


Activities
Visual Studio Code
23:16 29 أفريل
fr
98 %
weather.py - archiProject - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ARCHIPROJECT
.history
.vscode
python_service
__pycache__
bpmn.py
docker-compose.yml
Dockerfile
Dockerfile.bpmn
Dockerfile.news
Dockerfile.nginx
Dockerfile.weather
master_assistant.py
news.py
nginx.conf
requirements.txt
weather.py
Source Control (Ctrl+Shift+G)
IntelliJ IDEA Edu
master_assistant.py
weather.py
requirements.txt
bpmn.py
news.py
nginx.conf
Dockerfile
Dockerfile.news
python_service > weather.py > ...
1 from flask import Flask, request
2 from flask_restful import Api
3 import json
4 import requests
5
6 app = Flask(__name__)
7 api = Api(app)
8
9 @app.route('/weather')
10 def weather():
11 city_name = request.args.get('city')
12 api_key = "deb96cff96df7c74e93e62661b91c3c2"
13 base_url = "http://api.openweathermap.org/data/2.5/weather?"
14 complete_url = base_url + "appid=" + api_key + "&q=" + city_name
15 response = requests.get(complete_url)
16 return response.json()
17
18 if __name__ == '__main__':
19 app.run(host="0.0.0.0",port='3002',threaded=True,debug=True)
OUTLINE
TIMELINE
CHECKPOINTS
HIDDEN ITEMS
LOCAL HISTORY
Connect
: CodeStream
Live Share
Blackbox
Quokka
Ln 18, Col 1
Spaces: 4
UTF-8
LF
Python
3.10.8 64-bit
Go Live
Spell
Prettier

what is RESTful API?

RESTful API (Representational State Transfer) is a popular architectural style used for designing web services that communicate over HTTP. It defines a set of constraints and principles that developers should follow when creating APIs to enable client-server communication. RESTful APIs use HTTP methods like GET, POST, PUT, and DELETE to perform CRUD (Create, Read, Update, Delete) operations on resources, which can be represented in various formats such as JSON, XML, or plain text.

2)Building Nginx server

setting up a load-balanced web server to handle incoming traffic and distribute it to multiple Python services running on different servers.

2)Dockerize everything

We will create the Dockerfiles for our services.
create Docker images for the News and Weather services as well as an Nginx reverse proxy.

Dockerfile.news:

It starts with a Python 3.8 slim-buster base image.
Copies the requirements.txt file to install the necessary dependencies for the News service.
Installs the dependencies using pip.
Exposes port 3003.
Copies the news.py file.
Sets the command to run the news.py file.

Dockerfile.weather:

It starts with a Python 3.8 slim-buster base image.
Copies the requirements.txt file to install the necessary dependencies for the Weather service.
Installs the dependencies using pip.
Exposes port 3002.
Copies the weather.py file.
Sets the command to run the weather.py file.

Dockerfile.nginx:

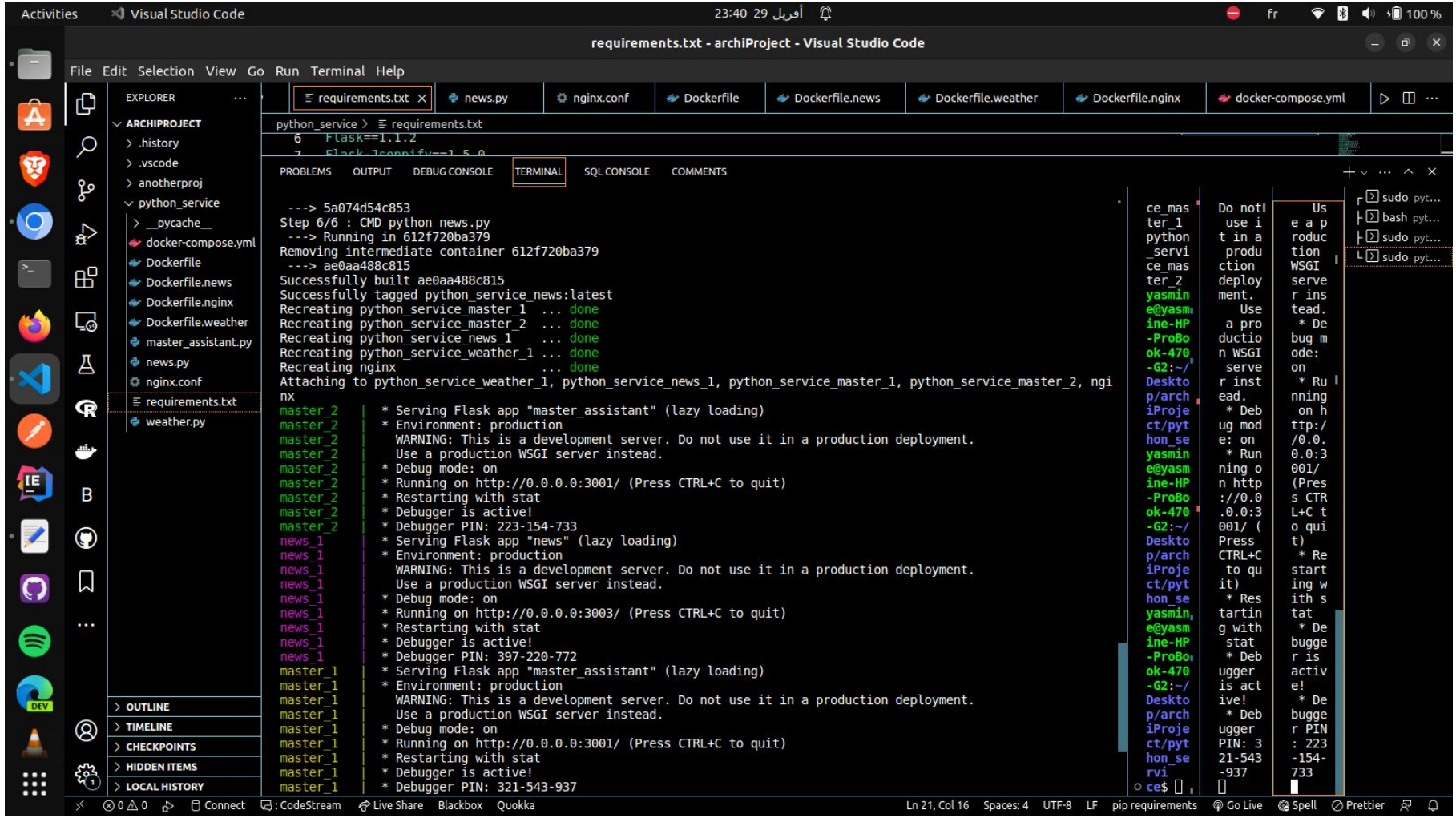
It starts with an Nginx base image.
Deletes the default Nginx configuration file.
Copies the custom Nginx configuration file named "nginx.conf" to the image.

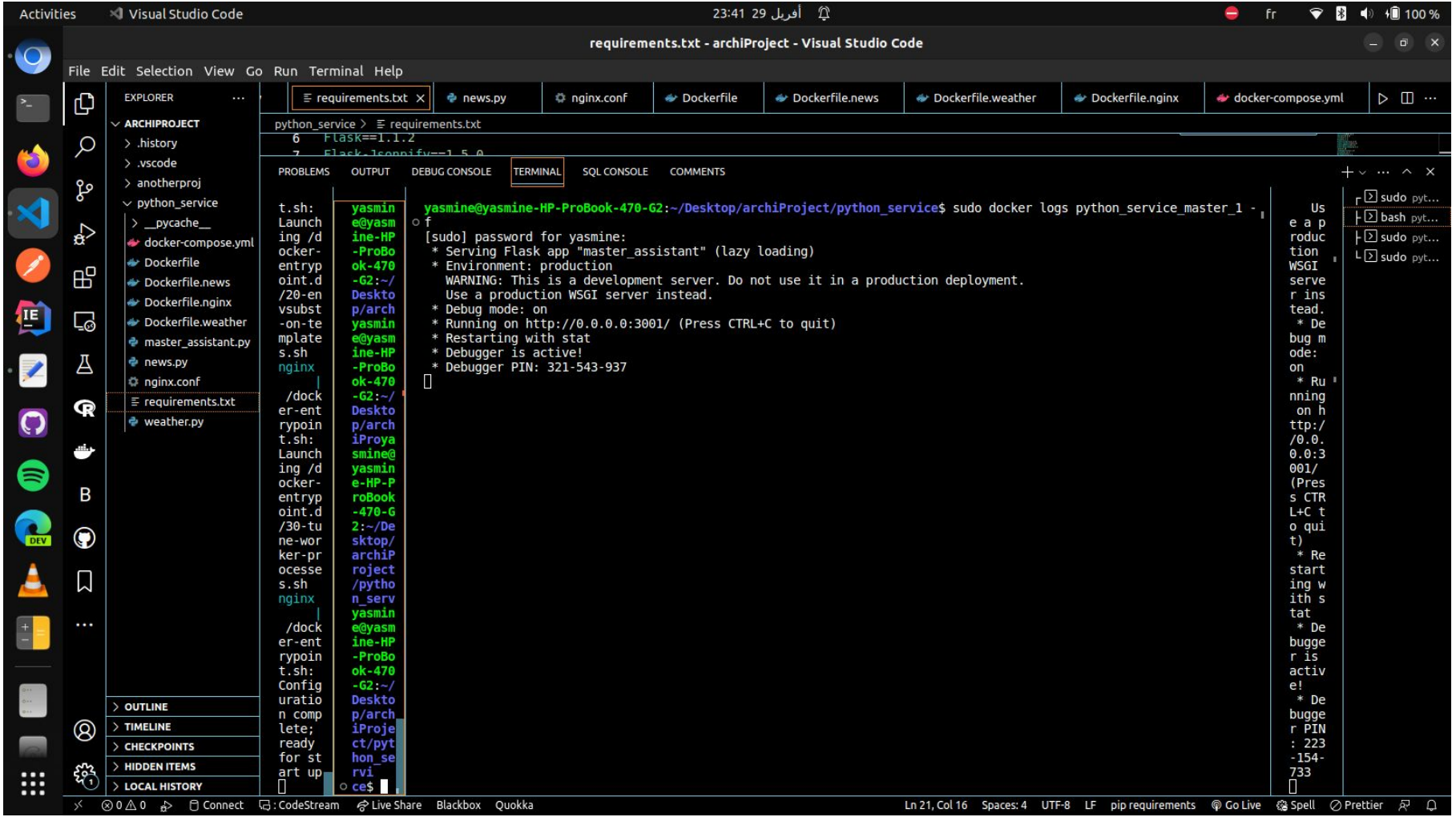
what is image?

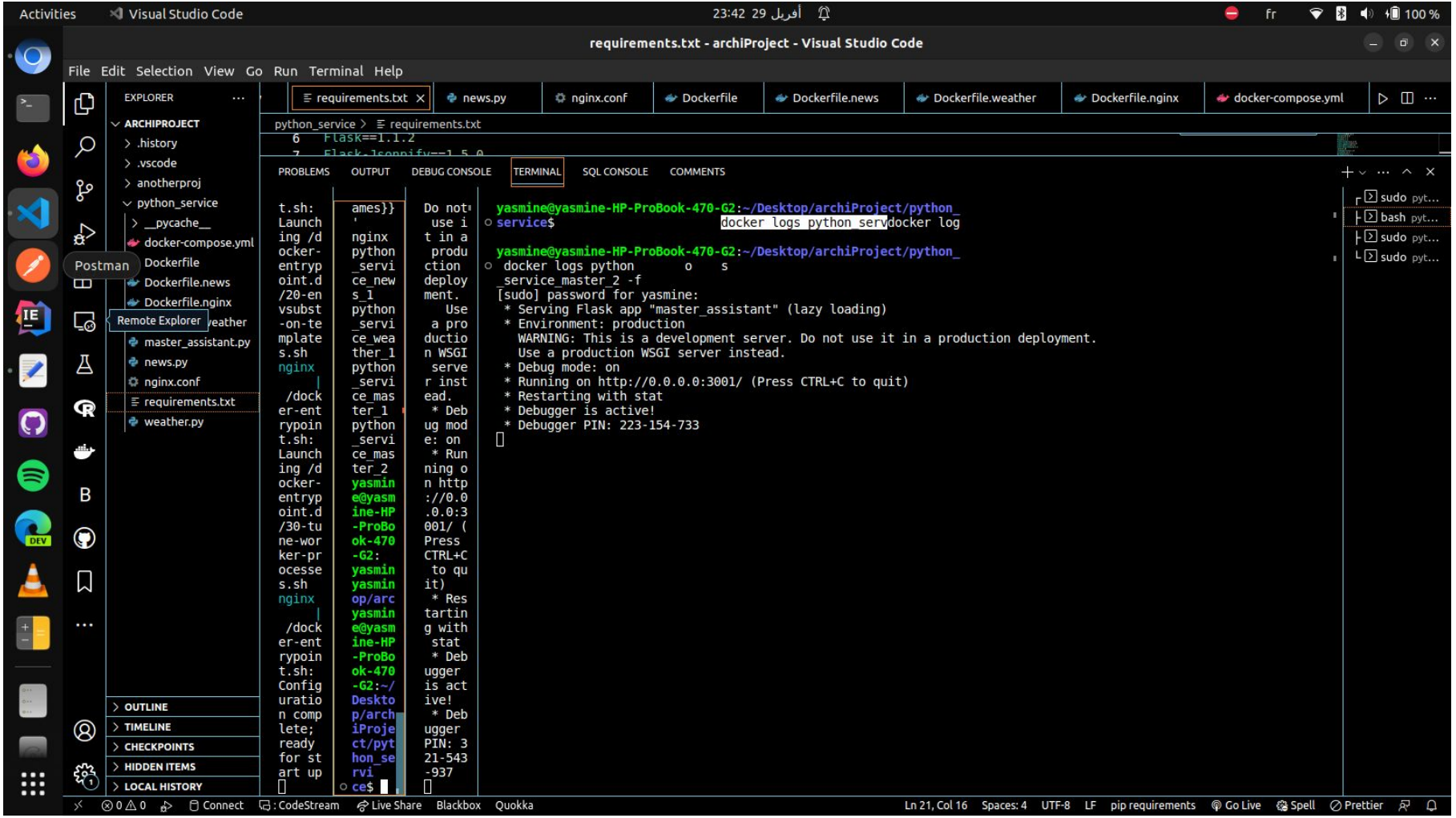
In Docker, an image is a lightweight, standalone, and executable package that includes everything needed to run a piece of software, including the code, runtime, libraries, environment variables, and system tools.
Docker images are built from a set of instructions called a Dockerfile, which specifies the application's dependencies, configuration, and other details.

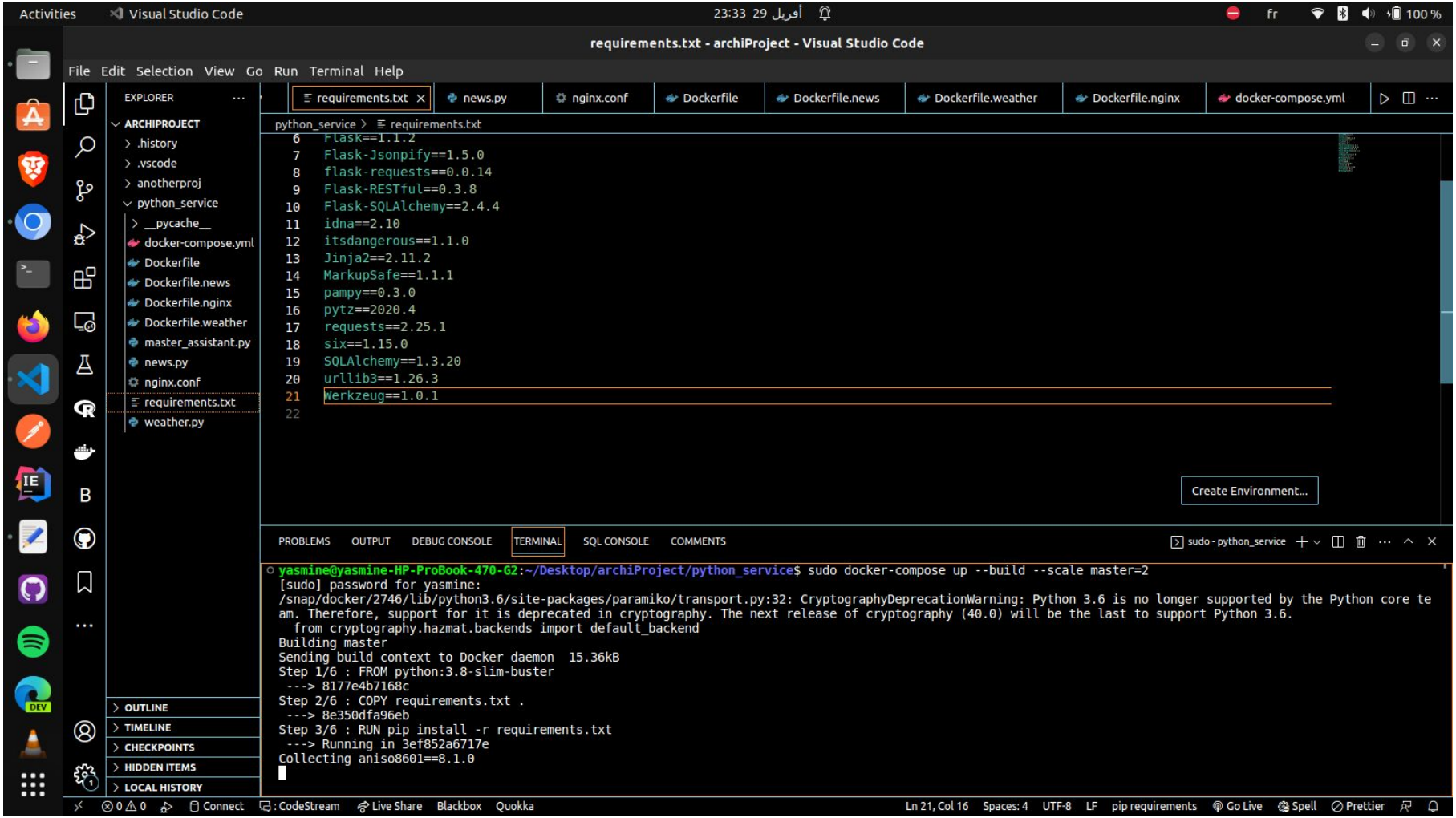
4)Docker-compose

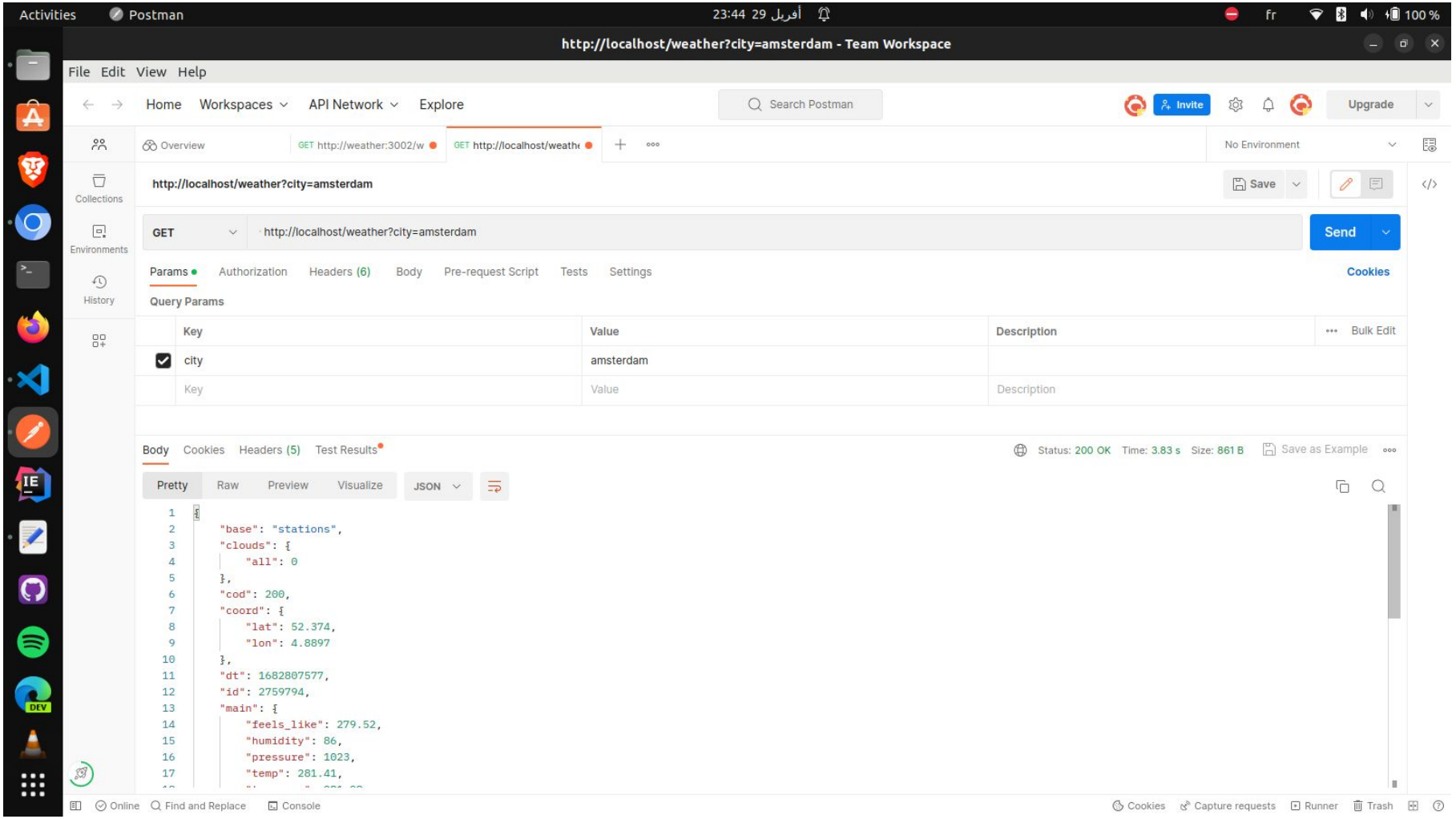
Docker-compose creates a container with our Nginx server, a container for each of the news and weather services and two instances of our main Api master(server).


Activities
Visual Studio Code
23:40 29 أفريل
fr
100 %
requirements.txt - archiProject - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ARCHIPROJECT
.history
.vscode
anotherproj
python_service
__pycache__
docker-compose.yml
Dockerfile
Dockerfile.news
Dockerfile.nginx
Dockerfile.weather
master_assistant.py
news.py
nginx.conf
requirements.txt
weather.py
OUTLINE
TIMELINE
CHECKPOINTS
HIDDEN ITEMS
LOCAL HISTORY
requirements.txt
news.py
nginx.conf
Dockerfile
Dockerfile.news
Dockerfile.weather
Dockerfile.nginx
docker-compose.yml
python_service > requirements.txt
6 Flask==1.1.2
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
SQL CONSOLE
COMMENTS
---> 5a074d54c853
Step 6/6 : CMD python news.py
---> Running in 612f720ba379
Removing intermediate container 612f720ba379
---> ae0aa488c815
Successfully built ae0aa488c815
Successfully tagged python_service_news:latest
Recreating python_service_master_1 ... done
Recreating python_service_master_2 ... done
Recreating python_service_news_1 ... done
Recreating python_service_weather_1 ... done
Recreating nginx ... done
Attaching to python_service_weather_1, python_service_news_1, python_service_master_1, python_service_master_2, nginx
master_2 | * Serving Flask app "master_assistant" (lazy loading)
master_2 | * Environment: production
master_2 | WARNING: This is a development server. Do not use it in a production deployment.
master_2 | Use a production WSGI server instead.
master_2 | * Debug mode: on
master_2 | * Running on http://0.0.0.0:3001/ (Press CTRL+C to quit)
master_2 | * Restarting with stat
master_2 | * Debugger is active!
master_2 | * Debugger PIN: 223-154-733
news_1 | * Serving Flask app "news" (lazy loading)
news_1 | * Environment: production
news_1 | WARNING: This is a development server. Do not use it in a production deployment.
news_1 | Use a production WSGI server instead.
news_1 | * Debug mode: on
news_1 | * Running on http://0.0.0.0:3003/ (Press CTRL+C to quit)
news_1 | * Restarting with stat
news_1 | * Debugger is active!
news_1 | * Debugger PIN: 397-220-772
master_1 | * Serving Flask app "master_assistant" (lazy loading)
master_1 | * Environment: production
master_1 | WARNING: This is a development server. Do not use it in a production deployment.
master_1 | Use a production WSGI server instead.
master_1 | * Debug mode: on
master_1 | * Running on http://0.0.0.0:3001/ (Press CTRL+C to quit)
master_1 | * Restarting with stat
master_1 | * Debugger is active!
master_1 | * Debugger PIN: 321-543-937
sudo pyt...
bash pyt...
sudo pyt...
sudo pyt...
Ln 21, Col 16
Spaces: 4
UTF-8
LF
pip requirements
Go Live
Spell
Prettier


Activities
Visual Studio Code
23:41 29 أفريل
fr
100 %
requirements.txt - archiProject - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ARCHIPROJECT
.history
.vscode
anotherproj
python_service
__pycache__
docker-compose.yml
Dockerfile
Dockerfile.news
Dockerfile.nginx
Dockerfile.weather
master_assistant.py
news.py
nginx.conf
requirements.txt
weather.py
requirements.txt
news.py
nginx.conf
Dockerfile
Dockerfile.news
Dockerfile.weather
Dockerfile.nginx
docker-compose.yml
python_service > requirements.txt
6 Flask==1.1.2
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
SQL CONSOLE
COMMENTS
yasmine@yasmine-HP-ProBook-470-G2:~/Desktop/archiProject/python_service$ sudo docker logs python_service_master_1 -f
[sudo] password for yasmine:
* Serving Flask app "master_assistant" (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: on
* Running on http://0.0.0.0:3001/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 321-543-937
sudo pyt...
bash pyt...
sudo pyt...
sudo pyt...
OUTLINE
TIMELINE
CHECKPOINTS
HIDDEN ITEMS
LOCAL HISTORY
Ln 21, Col 16
Spaces: 4
UTF-8
LF
pip requirements
Go Live
Spell
Prettier

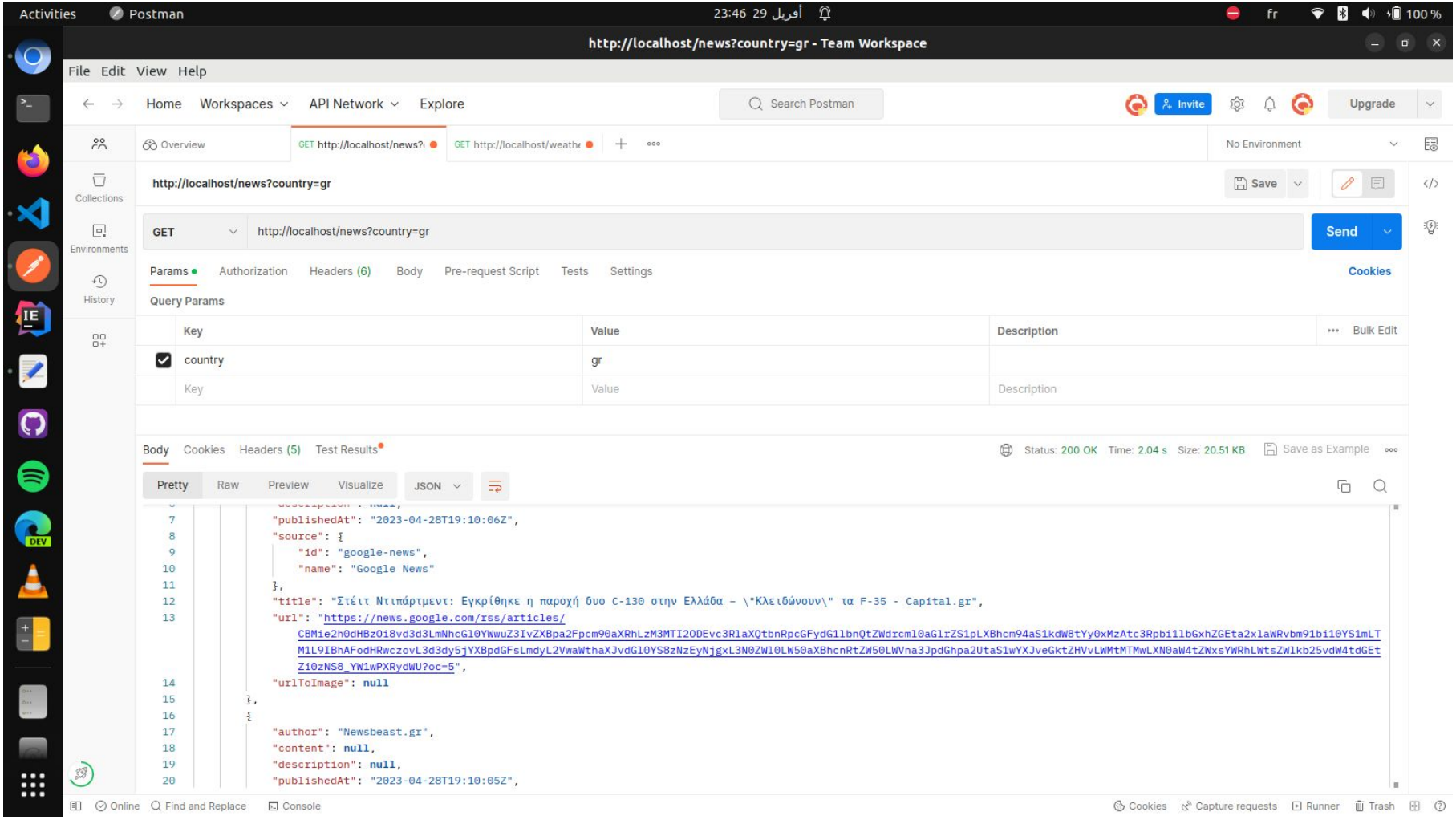

Activities
Visual Studio Code
23:42 29 أفريل
fr
100 %
requirements.txt - archiProject - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ARCHIPROJECT
.history
.vscode
anotherproj
python_service
__pycache__
docker-compose.yml
Dockerfile
Dockerfile.news
Dockerfile.nginx
master_assistant.py
news.py
nginx.conf
requirements.txt
weather.py
Postman
Remote Explorer
requirements.txt
news.py
nginx.conf
Dockerfile
Dockerfile.news
Dockerfile.weather
Dockerfile.nginx
docker-compose.yml
python_service > requirements.txt
6 Flask==1.1.2
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
SQL CONSOLE
COMMENTS
yasmine@yasmine-HP-ProBook-470-G2:~/Desktop/archiProject/python_service$ docker logs python_service_master_2 -f
[sudo] password for yasmine:
* Serving Flask app "master_assistant" (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: on
* Running on http://0.0.0.0:3001/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 223-154-733
sudo pyt...
bash pyt...
sudo pyt...
sudo pyt...
OUTLINE
TIMELINE
CHECKPOINTS
HIDDEN ITEMS
LOCAL HISTORY
Ln 21, Col 16
Spaces: 4
UTF-8
LF
pip requirements
Go Live
Spell
Prettier


Activities
Visual Studio Code
23:33 29 أفريل
fr
100 %
requirements.txt - archiProject - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ARCHIPROJECT
.history
.vscode
anotherproj
python_service
__pycache__
docker-compose.yml
Dockerfile
Dockerfile.news
Dockerfile.nginx
Dockerfile.weather
master_assistant.py
news.py
nginx.conf
requirements.txt
weather.py
OUTLINE
TIMELINE
CHECKPOINTS
HIDDEN ITEMS
LOCAL HISTORY
requirements.txt
news.py
nginx.conf
Dockerfile
Dockerfile.news
Dockerfile.weather
Dockerfile.nginx
docker-compose.yml
python_service > requirements.txt
6 Flask==1.1.2
7 Flask-Jsonpify==1.5.0
8 flask-requests==0.0.14
9 Flask-RESTful==0.3.8
10 Flask-SQLAlchemy==2.4.4
11 idna==2.10
12 itsdangerous==1.1.0
13 Jinja2==2.11.2
14 MarkupSafe==1.1.1
15 pampy==0.3.0
16 pytz==2020.4
17 requests==2.25.1
18 six==1.15.0
19 SQLAlchemy==1.3.20
20 urllib3==1.26.3
21 Werkzeug==1.0.1
22
Create Environment...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL SQL CONSOLE COMMENTS
sudo - python_service
yasmine@yasmine-HP-ProBook-470-G2:~/Desktop/archiProject/python_service$ sudo docker-compose up --build --scale master=2
[sudo] password for yasmine:
/snap/docker/2746/lib/python3.6/site-packages/paramiko/transport.py:32: CryptographyDeprecationWarning: Python 3.6 is no longer supported by the Python core team. Therefore, support for it is deprecated in cryptography. The next release of cryptography (40.0) will be the last to support Python 3.6.
from cryptography.hazmat.backends import default_backend
Building master
Sending build context to Docker daemon 15.36kB
Step 1/6 : FROM python:3.8-slim-buster
---> 8177e4b7168c
Step 2/6 : COPY requirements.txt .
---> 8e350dfa96eb
Step 3/6 : RUN pip install -r requirements.txt
---> Running in 3ef852a6717e
Collecting aniso8601==8.1.0
Connect
: CodeStream
Live Share
Blackbox
Quokka
Ln 21, Col 16
Spaces: 4
UTF-8
LF
pip requirements
Go Live
Spell
Prettier


Activities
Postman
23:44 29 أفريل
fr
100 %
http://localhost/weather?city=amsterdam - Team Workspace
File Edit View Help
Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
Overview
GET http://weather:3002/w
GET http://localhost/weathe
No Environment
Collections
Environments
History
http://localhost/weather?city=amsterdam
Save
GET
http://localhost/weather?city=amsterdam
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
city
amsterdam
Key
Value
Description
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 3.83 s
Size: 861 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
{
"base": "stations",
"clouds": {
"all": 0
},
"cod": 200,
"coord": {
"lat": 52.374,
"lon": 4.8897
},
"dt": 1682807577,
"id": 2759794,
"main": {
"feels_like": 279.52,
"humidity": 86,
"pressure": 1023,
"temp": 281.41,
Online
Find and Replace
Console
Cookies
Capture requests
Runner
Trash

what does that json file mean?

"base": "stations" - This specifies the source of the weather data.
"clouds": {"all": 20} - This gives information about the cloud cover. In this case, there are "few clouds" with a coverage of 20%.
"cod": 200 - This is the HTTP status code returned by the API, indicating a successful response.
"coord": {"lat": 52.374, "lon": 4.8897} - This provides the latitude and longitude of Amsterdam.
"dt": 1682782385 - This is the time of data calculation in Unix timestamp format.
"id": 2759794 - This is the ID of the city in OpenWeatherMap's database.
"main": {...} - This contains various temperature and pressure data. The "temp" field gives the current temperature in Kelvin, which is approximately 15 degrees Celsius.
"name": "Amsterdam" - This is the name of the city.
"sys": {...} - This contains various information about the location and time zone of Amsterdam.
"timezone": 7200 - This is the time zone offset in seconds from UTC.
"visibility": 10000 - This gives the visibility distance in meters.
"weather": [{...}] - This gives a description of the weather conditions in Amsterdam. In this case, there are "few clouds".
"wind": {...} - This gives information about the wind speed and direction in Amsterdam.


Activities
Postman
23:46 29 أفريل
fr
100 %
http://localhost/news?country=gr - Team Workspace
File Edit View Help
Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
Overview
GET http://localhost/news?
GET http://localhost/weathe
No Environment
Collections
Environments
History
http://localhost/news?country=gr
Save
GET
http://localhost/news?country=gr
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
country
gr
Key
Value
Description
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 2.04 s
Size: 20.51 KB
Save as Example
Pretty
Raw
Preview
Visualize
JSON
"publishedAt": "2023-04-28T19:10:06Z",
"source": {
"id": "google-news",
"name": "Google News"
},
"title": "Στέιτ Ντιπάρτμεντ: Εγκρίθηκε η παροχή δυο C-130 στην Ελλάδα – \"Κλειδώνουν\" τα F-35 - Capital.gr",
"url": "https://news.google.com/rss/articles/
CBMie2h0dHBzOi8vd3d3LmNhcGl0YWwuZ3IvZXBpa2Fpcm90aXRhLzM3MTI2ODEvc3R1aXQtbnRpcGFydG11bnQtZWdrcml0aGlrZS1pLXBhcm94aS1kdW8tYy0xMzAtc3Rpbi1lbGxhZGEta2xlaWRvbm91bi10YS1mLT
M1L9IBhAFodHRwczovL3d3dy5jYXBpdGFsLmdyL2VwaWthaXJvdGl0YS8zNzEyNjgxL3N0ZW10LW50aXBhcnRtZW50LWVna3JpdGhpa2UtaS1wYXJveGktZHVvLWMtMTMwLXN0aW4tZWxsYWRhLWtsZW1kb25vdW4tdGEt
Zi0zNS8_YW1wPXRydWU?oc=5",
"urlToImage": null
},
{
"author": "Newsbeast.gr",
"content": null,
"description": null,
"publishedAt": "2023-04-28T19:10:05Z",
Online
Find and Replace
Console
Cookies
Capture requests
Runner
Trash

Στέιτ Ντιπάρτμεντ: Εγκρίθηκε η παροχή δυο C-130 στην Ελλάδα – "Κλειδώνουν"... - Chromium

Στέιτ Ντιπάρτμεντ: Εγκρί

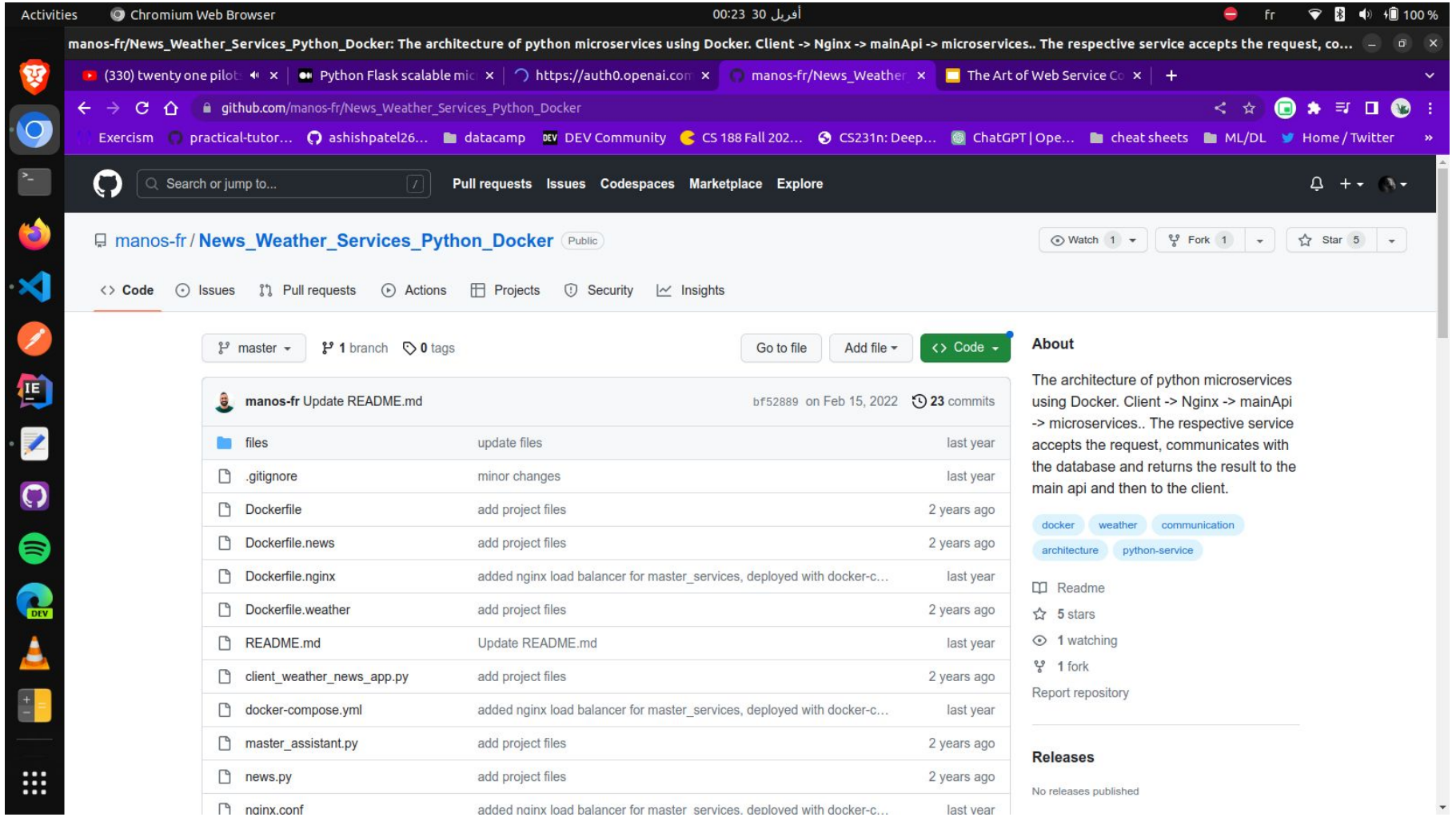
capital.gr/epikairotita/3712681/steit-ntipartment-egkrithike-i-paroxi-duo-c-130-stin-ellada-kleidonoun-ta-f-35/

Exercism practical-tutor... ashishpatel26... datacamp DEV Community CS 188 Fall 202... CS231n: Deep... ChatGPT | Ope... cheat sheets ML/DL Home / Twitter

CS 188 Fall 2022 | Introduction to Artificial Intelligence at UC Berkeley
https://inst.eecs.berkeley.edu/~cs188/fa22/

Παρασκευή, 28-Απρ-2023 22:10

Στέιτ Ντιπάρτμεντ: Εγκρίθηκε η παροχή δυο C-130 στην Ελλάδα – "Κλειδώνουν" τα F-35


Activities
Chromium Web Browser
00:23 30 أفريل
fr
100 %
manos-fr/News_Weather_Services_Python_Docker: The architecture of python microservices using Docker. Client -> Nginx -> mainApi -> microservices.. The respective service accepts the request, co...
(330) twenty one pilots
Python Flask scalable mic
https://auth0.openai.com
manos-fr/News_Weather
The Art of Web Service Co
github.com/manos-fr/News_Weather_Services_Python_Docker
Exercism
practical-tutor...
ashishpatel26...
datacamp
DEV Community
CS 188 Fall 202...
CS231n: Deep...
ChatGPT | Ope...
cheat sheets
ML/DL
Home / Twitter
Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
manos-fr / News_Weather_Services_Python_Docker
Public
Watch 1
Fork 1
Star 5
Code
Issues
Pull requests
Actions
Projects
Security
Insights
master
1 branch
0 tags
Go to file
Add file
Code
manos-fr Update README.md
bf52889 on Feb 15, 2022
23 commits
files update files last year
.gitignore minor changes last year
Dockerfile add project files 2 years ago
Dockerfile.news add project files 2 years ago
Dockerfile.nginx added nginx load balancer for master_services, deployed with docker-c... last year
Dockerfile.weather add project files 2 years ago
README.md Update README.md last year
client_weather_news_app.py add project files 2 years ago
docker-compose.yml added nginx load balancer for master_services, deployed with docker-c... last year
master_assistant.py add project files 2 years ago
news.py add project files 2 years ago
nginx.conf added nginx load balancer for master_services, deployed with docker-c... last year
About
The architecture of python microservices using Docker. Client -> Nginx -> mainApi -> microservices.. The respective service accepts the request, communicates with the database and returns the result to the main api and then to the client.
docker
weather
communication
architecture
python-service
Readme
5 stars
1 watching
1 fork
Report repository
Releases
No releases published