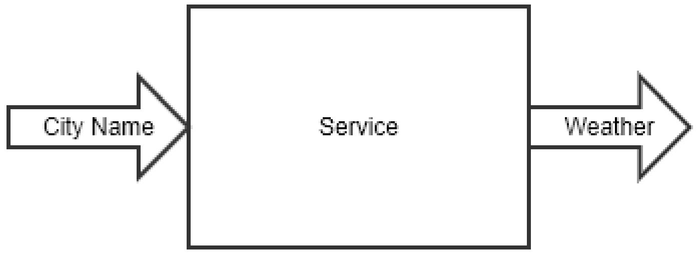
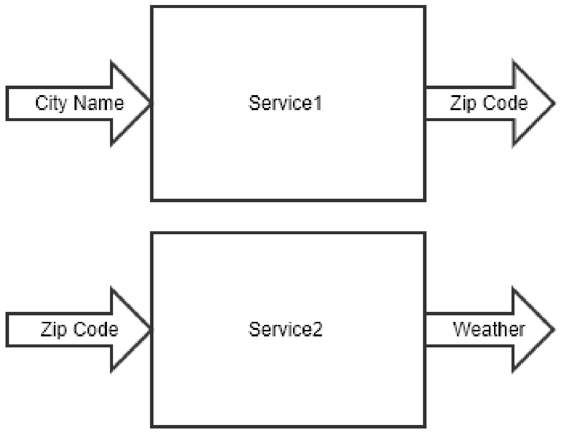
Problem:

Write a web application to find the *weather* of a given city.



1. Design the service using two microservices:

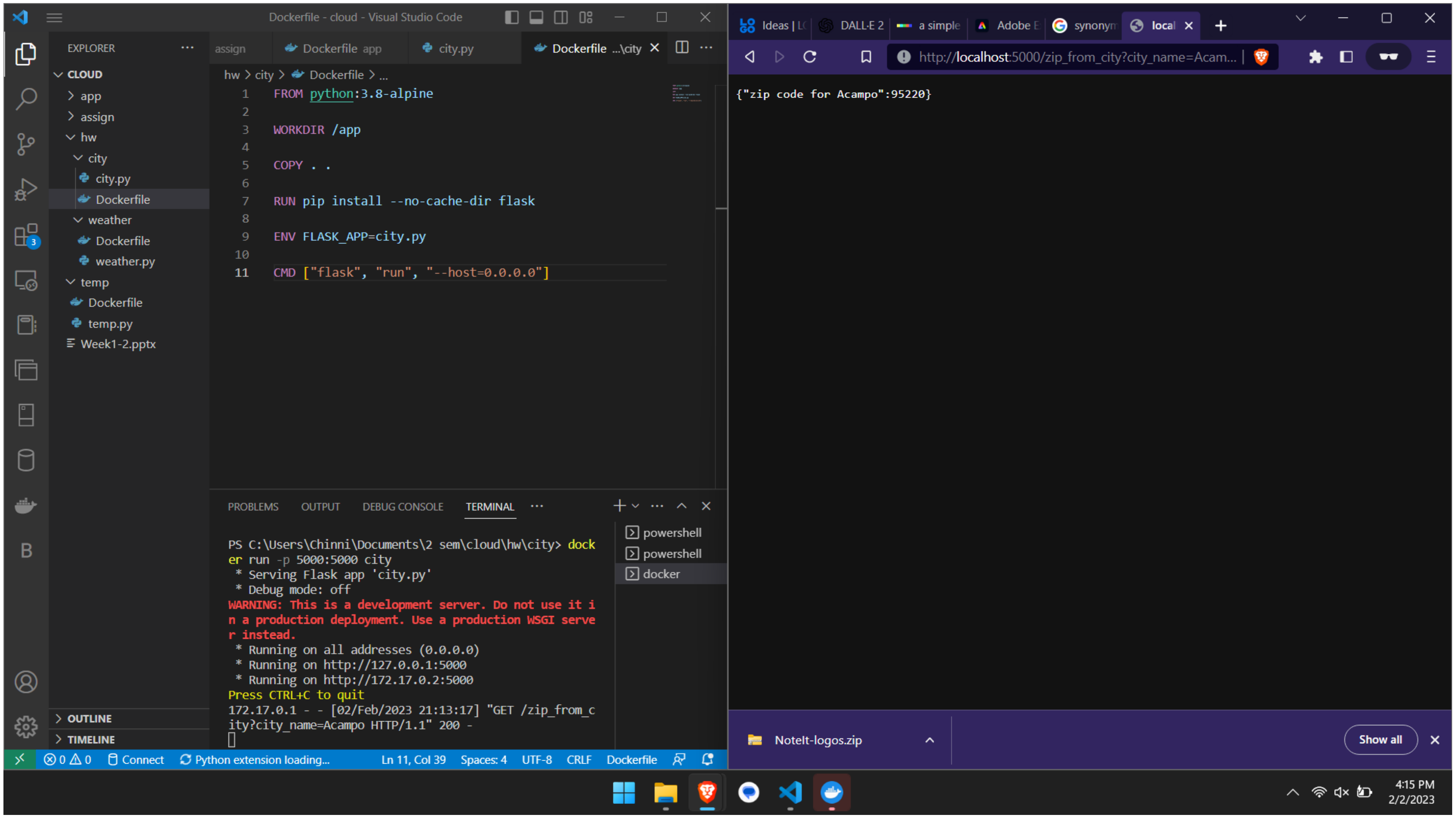


2. Implement these two independent microservices and then test them using either the browser client or the curl client.

Solution:

- First I created python programs and Dockerfile(s) for two services
- Used the command in the terminal to create an image that runs with help of the docker desktop. That is `docker build -t weather .` Here the weather is the name of the image I gave
- Then to run the image and containerize it, we use the command `docker run -p 5000:5000 weather` where 5000 is the port
- Now I tested these services as shown below in screenshots.

1st service which gave Zip code from City Name



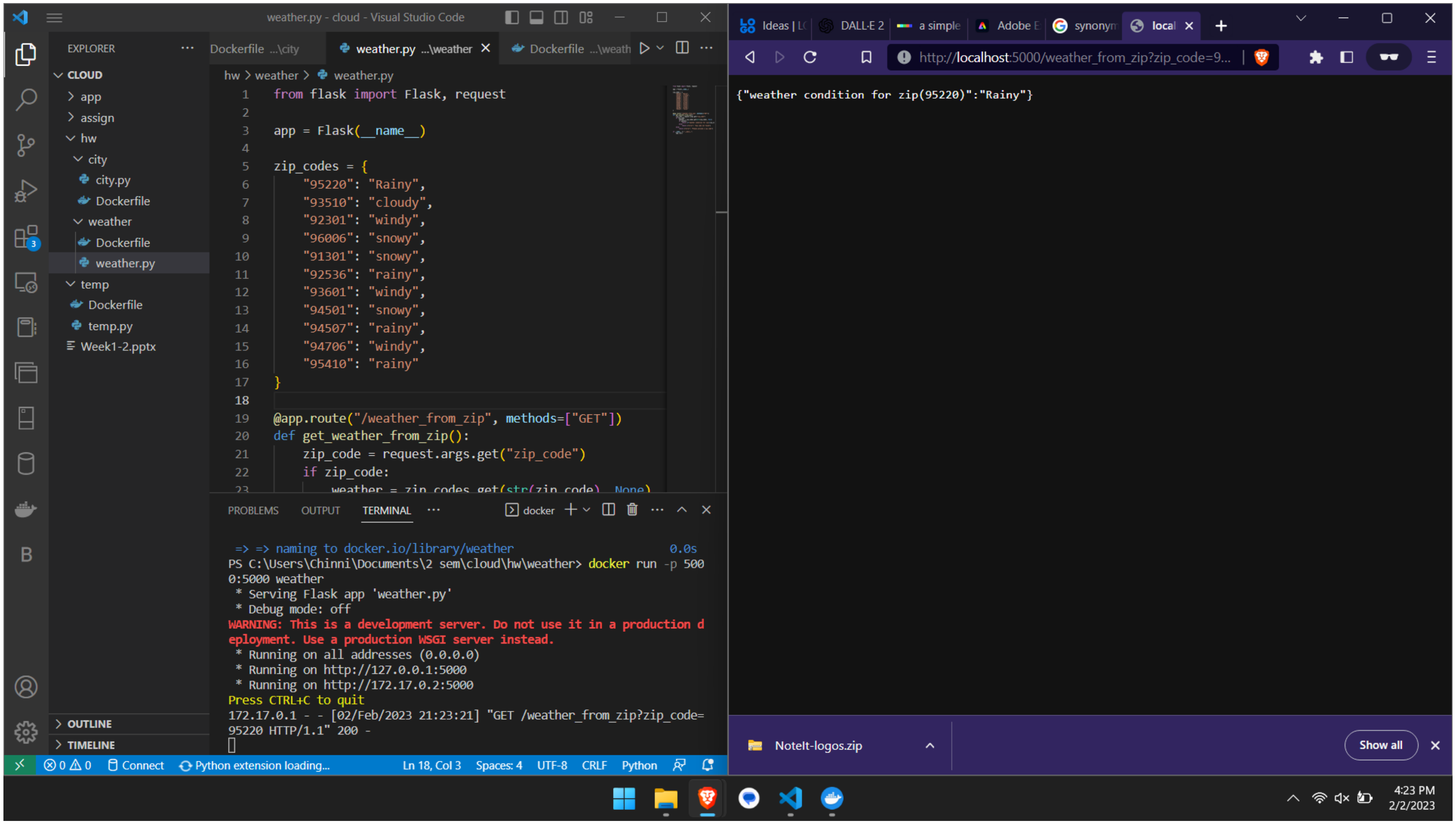
```
FROM python:3.8-alpine

WORKDIR /app

COPY . .

RUN pip install --no-cache-dir flask

ENV FLASK_APP=city.py

CMD ["flask", "run", "--host=0.0.0.0"]
```

```
PS C:\Users\Chinni\Documents\2 sem\cloud\hw\city> docker run -p 5000:5000 city
 * Serving Flask app 'city.py'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.17.0.2:5000
Press CTRL+C to quit
172.17.0.1 - - [02/Feb/2023 21:13:17] "GET /zip_from_city?city_name=Acampo HTTP/1.1" 200 -
```

http://localhost:5000/zip_from_city?city_name=Acam...

```
{"zip code for Acampo":95220}
```

2nd service takes the zip code and gives the weather details:



```python
from flask import Flask, request

app = Flask(__name__)

zip_codes = {
    "95220": "Rainy",
    "93510": "cloudy",
    "92301": "windy",
    "96006": "snowy",
    "91301": "snowy",
    "92536": "rainy",
    "93601": "windy",
    "94501": "snowy",
    "94507": "rainy",
    "94706": "windy",
    "95410": "rainy"
}

@app.route("/weather_from_zip", methods=["GET"])
def get_weather_from_zip():
    zip_code = request.args.get("zip_code")
    if zip_code:
        weather = zip_codes.get(str(zip_code), None)
```

```
 => => naming to docker.io/library/weather                          0.0s
PS C:\Users\Chinni\Documents\2 sem\cloud\hw\weather> docker run -p 500
0:5000 weather
 * Serving Flask app 'weather.py'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production d
eployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.17.0.2:5000
Press CTRL+C to quit
172.17.0.1 - - [02/Feb/2023 21:23:21] "GET /weather_from_zip?zip_code=
95220 HTTP/1.1" 200 -
```

http://localhost:5000/weather_from_zip?zip_code=9...

```
{"weather condition for zip(95220)":"Rainy"}
```