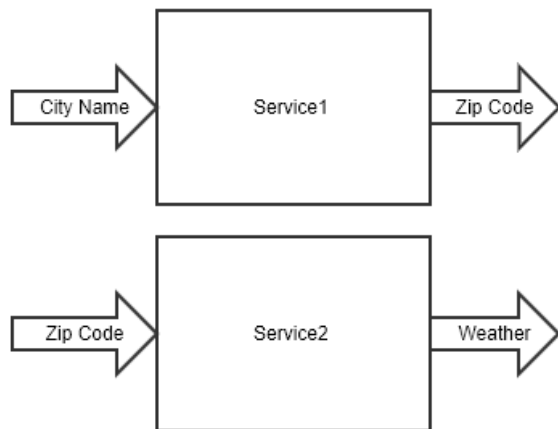


Problem:

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

The screenshot displays a development environment with three main components:

- Visual Studio Code (Left):** The Explorer sidebar shows a project structure with folders `app`, `assign`, `hw`, `city`, `weather`, and `temp`. The `city` folder contains `city.py` and `Dockerfile`. The `weather` folder contains `Dockerfile` and `weather.py`. The `temp` folder contains `Dockerfile` and `temp.py`. The `city` folder is selected, showing the `Dockerfile` in the editor. The Dockerfile content is:

```
1 FROM python:3.8-alpine
2
3 WORKDIR /app
4
5 COPY . .
6
7 RUN pip install --no-cache-dir flask
8
9 ENV FLASK_APP=city.py
10
11 CMD ["flask", "run", "--host=0.0.0.0"]
```
- Terminal (Bottom Left):** The terminal shows the command `docker run -p 5000:5000 city` being executed. The output indicates that the Flask app is serving on `http://127.0.0.1:5000` and `http://172.17.0.2:5000`. A warning message states: "WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead." The terminal also shows a GET request to `/zip_from_city?city_name=Acampo` returning a 200 status.
- Web Browser (Right):** The browser shows the URL `http://localhost:5000/zip_from_city?city_name=Acampo...`. The response is a JSON object: `{"zip code for Acampo":95220}`.

The bottom status bar of Visual Studio Code shows the file is at line 11, column 39, with 4 spaces, UTF-8 encoding, and CRLF line endings. The system tray at the bottom right shows the time as 4:15 PM on 2/2/2023.

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

The screenshot displays a development environment with Visual Studio Code on the left and a web browser on the right.

Visual Studio Code:

- EXPLORER:** Shows a project structure with folders `app`, `assign`, `hw`, `city`, `weather`, and `temp`. The `weather` folder is selected, showing `Dockerfile` and `weather.py`.
- Editor:** Displays the `weather.py` file. The code defines a Flask application with a dictionary of weather conditions for various zip codes and a route `/weather_from_zip` that returns the condition for a given zip code.
- TERMINAL:** Shows the command `docker run -p 5000 weather` being executed. The output indicates that the Flask app is serving on `http://127.0.0.1:5000` and `http://172.17.0.2:5000`.

Web Browser:

- The address bar shows `http://localhost:5000/weather_from_zip?zip_code=9...`.
- The response body shows a JSON object: `{"weather condition for zip(95220)": "Rainy"}`.