Assignment 3 – Microservices based IoT platform

Introduction

This assignment involves the development of a web-based REST application and its deployment as Web Roles and Worker Roles on Docker containers.

The web application is an IoT system – sensor nodes (containing sensors of various types) send sensor data to the Web Role continuously, which then offloads (asynchronously) the background tasks to the Worker Role. For the sake of simplicity, the background task is the calculation of the rolling average of the sensor values and an instance each of Web Role and Worker Role is sufficient. Data visualizations are hosted on the Web Role and can be viewed on the client's browser. A sensor node simulator that sends random values to the web role can be created to test the API's working.

The web application's code pertaining to the web role and the worker role is fed separately to the Docker builder which creates a web role and a worker role image respectively. One instance of each image is created as a container. On startup, these containers will run the web role and the worker role code that was provided during Docker build, and interact with each other to carry out the web application's functionality.

Python is the recommended choice of the programming language, although other options such as NodeJS/PHP can be used as per one's level of comfort. However, the setup instructions and the examples provided for reference will be in Python3 (online resources can be easily found along similar lines for other languages). For Python3, the Flask web framework is most suitable for creating web applications and will be used in all the setup guides and examples. For this assignment, data can be locally stored as files instead of using databases.

Learning Outcome

This assignment is aimed at familiarizing you with the usage of Docker containers and the deployment of web applications on them. You will also learn about the various tasks of web roles and worker roles and how they interact. It is recommended that you read about the Docker technology [1] and Containers[2] before continuing further. Refer to [3] to learn the difference between Containers and VMs. You can read more about it at [4].

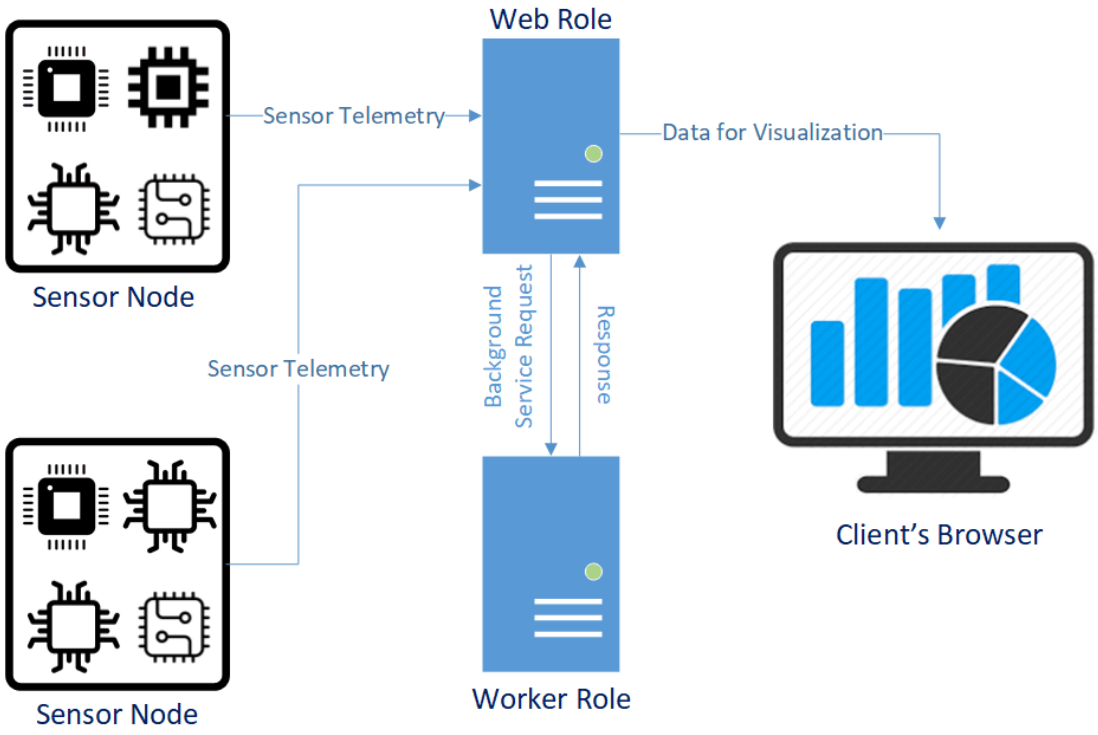

Figure 1. An overview of the IoT use case setup

Deliverables

REST APIs for the web application (similar to those of Assignment 1) have already been provided. In the folder 'Code' you can find the APIs for the Web Role and the Worker Role in their respective directories. The Requirements file and the Dockerfile which is used for the setup of Python3 Flask containers for Web Role and the Worker Role have also been provided. The assignment involves the implementation of these REST APIs in the respective .py files, resulting in a working IoT Web Application, deployed on the Web Role and Worker Role Docker containers.

A zip file containing the code along with any additional files that have been used, to be submitted in the same file structure that has been provided is expected to be submitted. If languages other than Python have been used, mention so in the Google Form during submission.

Setup

The following steps are guidelines to be used to setup Docker CE on an Ubuntu machine and create Python – Flask containers for Web Role and Worker Role.

IMPORTANT - EXECUTE THE FOLLOWING STEPS ONLY ON A VM. DO NOT SET IT UP ON YOUR OWN MACHINE.

- Deploy a VM with Ubuntu 16.04 LTS as the OS.
- Copy the Assignment folder to the VM.
- Navigate to the folder as superuser from a terminal
- Install Docker CE on Ubuntu. Refer to [5].
- Setup a user-defined Docker bridge network (Subnet 10.10.10.0/24). Refer to [6].

- Build the Docker images from the python flask app, requirements.txt and the Dockerfile for web role and worker role - refer to [7]. A sample Dockerfile and Requirements file for Python3 has already been provided. You may need to add external dependencies that you use to the Requirements file.
- Start a container each for web role and worker role on static IP addresses and ports as follows: (the web role must be aware of the worker role's IP address and port to communicate) 10.10.10.2 and 10.10.10.3 on ports 9000:5000 and 9001:5000 respectively. Make sure that the containers are on the same network that was created in earlier steps.

Mandatorily use the following configurations for the containers:

- Web role:
  - IP address - 10.10.10.2
  - Port mapping - 9000:5000
- Worker role:
  - IP address - 10.10.10.3
  - Port mapping - 9001:5000

Reference Links

1. https://opensource.com/resources/what-docker
2. https://www.docker.com/what-container
3. https://blog.netapp.com/blogs/containers-vs-vms/
4. https://stackoverflow.com/questions/16047306/how-is-docker-different-from-a-normal-virtual-machine
5. https://docs.docker.com/engine/installation/linux/docker-ce/ubuntu/#install-docker-ce
6. https://docs.docker.com/engine/userguide/networking/work-with-networks/#basic-container-networking-example
7. http://containertutorials.com/docker-compose/flask-simple-app.html