

Homework-3: Microservice Orchestration

Deadline: October 5th, 11:59PM ET/8:59pm PT.

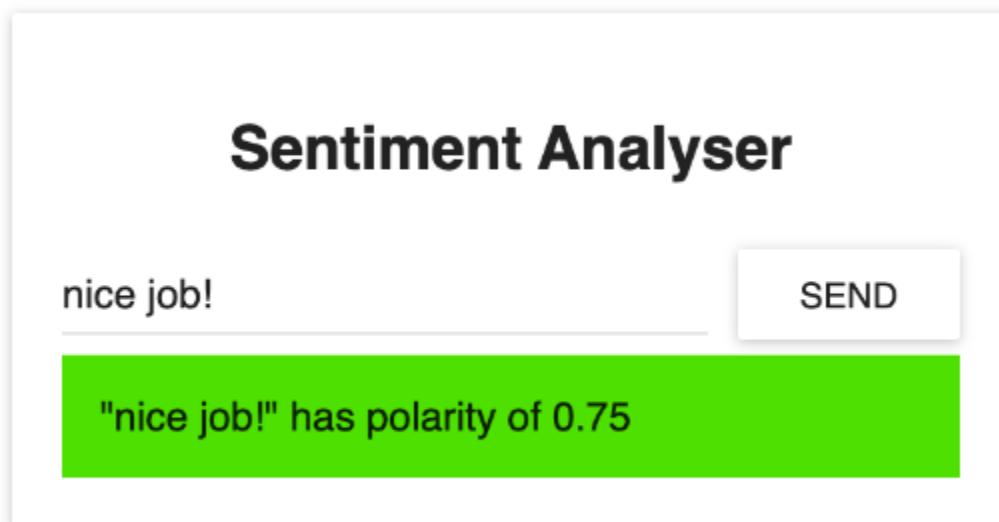
In this homework, you are expected to practice Microservice Orchestration on the Cloud and recording videos.

Use the following GitHub classroom to access the assignment and create your assignment repository: <https://classroom.github.com/a/PHZm4prC>

You should submit the URL for your GitHub repository on Canvas.

General Description:

In this homework, you will orchestrate a microservice-based sentimental analysis application and run it on Google Kubernetes Engine. This application accepts a sentence as input and it uses text analysis to calculate the emotion of the input sentence.



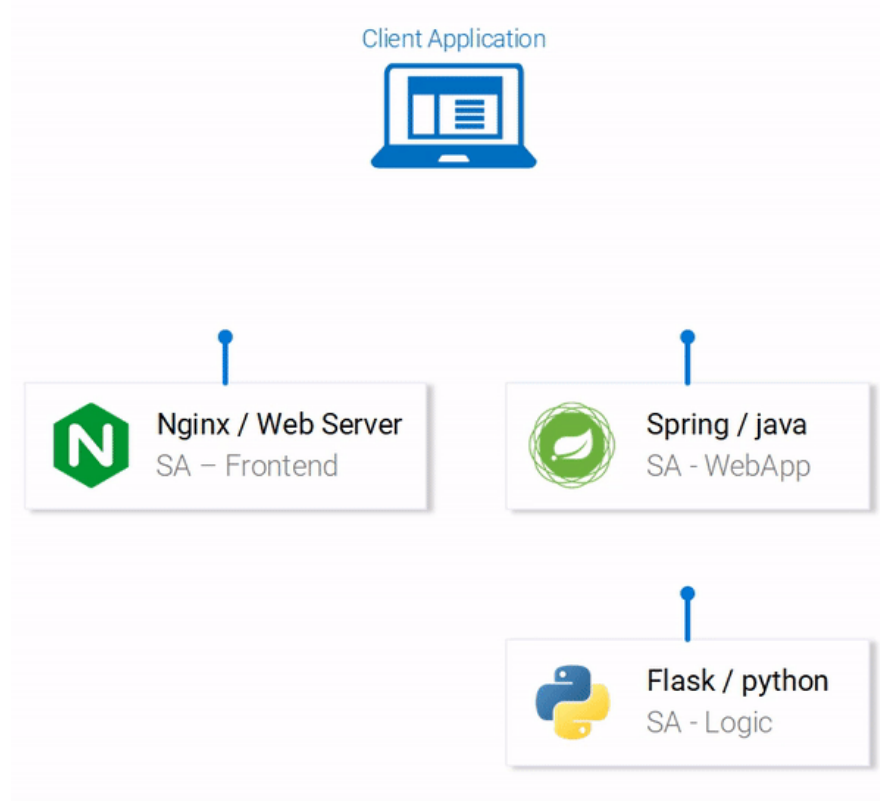
The screenshot shows a web application titled "Sentiment Analyser". It features a text input field containing the phrase "nice job!". To the right of the input field is a button labeled "SEND". Below the input field, there is a green rectangular box displaying the output: "\"nice job!\" has polarity of 0.75".

The application's source code is already developed but you will have to update some Dockerfile settings and/or create Kubernetes configuration files

<https://github.com/rinormaloku/k8s-mastery>

This application consists of the following microservices:

- SA-Frontend: a Nginx web server that serves our ReactJS static files.
- SA-WebApp: a Java Web Application that handles requests from the frontend.
- SA-Logic: a python application that performs Sentiment Analysis.



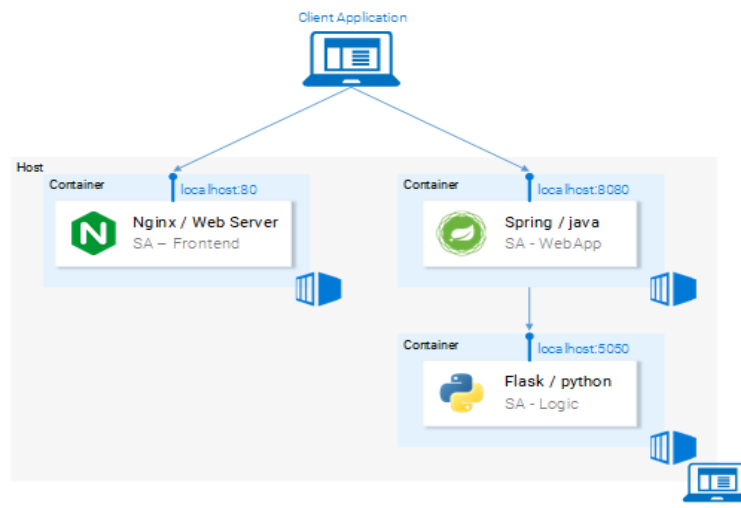
This interaction is best illustrated by showing how the data flows between them:

- A client application requests the index.html (which in turn requests bundled scripts of ReactJS application)
- The user interacting with the application triggers requests to the Spring WebApp.
- Spring WebApp forwards the requests for sentiment analysis to the Python app.
- Python Application calculates the sentiment and returns the result as a response.

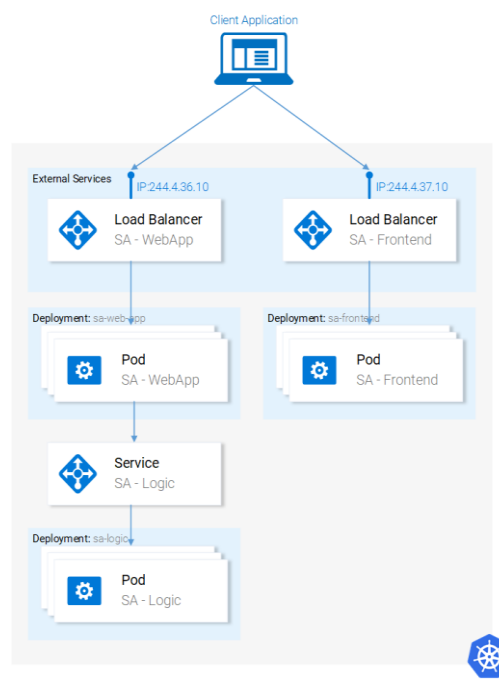
- The Spring WebApp returns the response to the React app. (Which then represents the information to the user.)

To complete this homework, you need to complete the following steps:

Step-I: Build/update a container image for each service and push it to Docker Hub. The repository contains some Dockerfiles so feel free to reuse any of them.



Step-II: Orchestrate Sentiment Analyzer's containers and run them on GKE by creating/updating necessary YAML files. There are some YAML files that can be reused under resource-manifests folder.



Step-III: Record two videos:

- First video where you demo the application working on the Cloud. Input a sentence to the application and check the polarity score (as shown on the screenshot in the first page)
- Second video where you explain the changes that you applied on the current codebase. You don't need to explain any code that you didn't develop.

Submission Guidelines:

- Your GitHub repository should contain the following:
 1. (40%) Your ReadMe file listing all the steps to get the application to work on Google Kubernetes Engine, starting from the GitHub repository code here: <https://github.com/rinormaloku/k8s-mastery>.
 2. (20%) List of all URLs for the Docker Hub images that were used (If you created docker images, make sure they are publicly available).
 3. (40%) Two video recordings showing 1) the application functionality demo on the cloud and 2) code walkthrough for your code changes.

Don't forget to disable billing after you finish using GCP

Grading Nodes:

- Docker Hub images are not public (You will receive up to 75% of the maximum grade)

Reference:

<https://www.freecodecamp.org/news/learn-kubernetes-in-under-3-hours-a-detailed-guide-to-orchestrating-containers-114ff420e882>