MSA Tutorial

Steven Zambrano[1]

School of Mathematical and Computational Science

Universidad Yachay Tech, Urcuquí - Ecuador

July 14, 2025.

General Description

This document describes the steps required to successfully execute the developed microservices architecture, starting from the GitHub repository, which contains all the files needed to build the Docker images and Kubernetes YAML manifests.

Prerequisites

- Docker installed and running.
- Minikube installed and running.
- kubectl configured to use the Minikube cluster.
- Access to the GitHub repository with:
  - Dockerfiles for each microservice
  - YAML files for deployment and services
  - Optional test or initialization scripts

1 MSA Image Classification

1.1 Clone the Repository

```
git clone https://github.com/StevenZ-G/Tesis.git
cd image-classificator
```

1.2 Building and Publishing Docker Images

```
cd data-collector
# Building the image
docker build -t user/data-handler .
# Pushing the image
docker push user/data-handler
```

Repeat for all microservices: *data-handler, extract-features, predict, orchestrator*. Additionally, it is important that (user) refers to your DockerHub username.

1.3 Minikube Start

```
    minikube start
```

1.4 Creating Persistent Volume

```
cd volume
kubectl apply - f persistent-volume.yaml
kubectl apply - f persistent-volume-claim.yaml
```

1.5 Deploy Microservices

I recommend this order.

```
kubectl apply -f data-collector-deployment.yaml
kubectl apply -f data-handler-deployment.yaml
kubectl apply -f feature-extraction-deployment.yaml
kubectl apply -f predict-deployment.yaml
kubectl apply -f orchestrator-deployment.yaml
```

And the services

```
kubectl apply -f data-collector-service.yaml
kubectl apply -f data-handler-service.yaml
kubectl apply -f feature-extraction-service.yaml
kubectl apply -f predict-service.yaml
kubectl apply -f orchestrator-service.yaml
```

Verifying

```
kubectl get pods
kubectl get services
```

Also you have to upload the pretrained model

```
kubectl cp image_recognition_model <POD-NAME>:/mnt/data/image_recognition_model
```

1.6 Access the System

```
kubectl port-forward service/data-collector-service 5000:5000
kubectl port-forward service/data-handler-service 5001:5001
kubectl port-forward service/feature-extraction-service 5002:5002
kubectl port-forward service/predict-service 5003:5003
```

With NodePort:

```
minikube service orchestrator-service
```

Each service in a different cmd

1.7 Deploy Frontend

The next step is run the Frontend. (Optionality you can create a docker image of it)

```
cd frontend
python app.py
```

You can upload an image in it. You also can test the architecture using CURL:

```
curl -X POST -F "image=@ruta/a/la/imagen.png" http://localhost:5004/orchestrate
```

2 MSA LLM Songs Recommendation

2.1 Clone the Repository

Assuming that the repository have already clone we only need to enter in the directory.

```
cd LLM
```

2.2 Building and Publishing Docker Images

```
cd emotions
# Building the image
docker build -t user/emotion-service .
# Pushing the image
docker push user/emotion-service
```

Repeat for all microservices: *emotions-songs, history, recommendations, songs, user-interface and orchestrator*. Additionally, it is important that (user) refers to your DockerHub username.

2.3 Minikube Start

```
minikube start
```

2.4 Creating Persistent Volume

```
cd kubernetes
cd volume
kubectl apply - f persistent-volume.yaml
kubectl apply - f persistent-volume-claim.yaml
```

2.5 Deploy Microservices

First

```
cd kebernetes
cd ds
```

I recommend this order.

```
kubectl apply -f user-interface-deployment.yaml
kubectl apply -f emotion-deployment.yaml
kubectl apply -f songs-deployment.yaml
kubectl apply -f emotion-songs-deployment.yaml
kubectl apply -f recommendations-deployment.yaml
kubectl apply -f history-deployment.yaml
kubectl apply -f orchestrator-deployment.yaml
```

And the services

```
kubectl apply -f user-interface-service.yaml
kubectl apply -f emotion-service.yaml
kubectl apply -f songs-service.yaml
kubectl apply -f emotion-songs-service.yaml
kubectl apply -f recommendations-service.yaml
kubectl apply -f history-service.yaml
kubectl apply -f orchestrator-service.yaml
```

And Ollama services

```
cd ollama
kubectl apply -f ollama-deployment.yaml
kubectl apply -f ollama-service.yaml
```

Verifying

```
kubectl get pods
kubectl get services
```

Also you have to upload the database

```
kubectl cp system.db emotions-deployment-<POD NAME>:/llm/data
```

2.6 Access the System

Each service in a different cmd

```
kubectl port-forward service/user-interface-service 5010:5010
kubectl port-forward service/emotions-service 5011:5011
kubectl port-forward service/songs-service 5012:5012
kubectl port-forward service/emotions-songs-service 5013:5013
kubectl port-forward service/recommendations-service 5014:5014
kubectl port-forward service/history-service 5015:5015
kubectl port-forward service/llmorchestrator-service 5016:5016
kubectl port-forward service/ollama-service 11434:11434
```

2.7 Deploy Frontend

The next step is run the Frontend. (Optionality you can create a docker image of it)

```
cd frontend
python app.py
```