

**CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENERIAS**

**Aldo Guillermo Román Del Muro**

**INCO**

**217554557**

**Kubernetes.**

**Computación Tolerante a fallas.**

**DR. MICHEL EMANUEL LOPEZ FRANCO.**

Como toda estructura básica, comenzamos con la estructura de nuestro archivo Python, como lo hemos hecho las otras veces.

from flask import Flask

app = Flask(\_\_name\_\_)

@app.route("/")

def hello():

    return "Hello, World!"

if \_\_name\_\_ == "\_\_main\_\_":

    app.run(host="0.0.0.0", port=5000)

Ahora tenemos que definir un archivo de tipo txt donde indicaremos los requisitos de nuestro programa.

Flask==2.0.1

Unicamente indicaremos la versión de Flask que queremos utilizar.

Ahora indicamos en un archivo docker lo que necesitaremos.

# Usa una imagen base de Python

FROM python:3.8-slim

# Establece el directorio de trabajo

WORKDIR /app

# Copia los archivos necesarios

COPY . .

# Instala las dependencias

RUN pip install --no-cache-dir -r requirements.txt

# Expone el puerto en el que la aplicación Flask está ejecutándose

EXPOSE 5000

# Comando para ejecutar la aplicación

CMD ["python", "app/main.py"]

Ahora en un archivo .yaml indicaremos más información para nuestro programa. Básicamente aquí son las réplicas de la aplicación.

apiVersion: apps/v1

kind: Deployment

metadata:

  name: my-python-app

spec:

  replicas: 2

  selector:

    matchLabels:

      app: my-python-app

  template:

    metadata:

      labels:

        app: my-python-app

    spec:

      containers:

      - name: my-python-app

        image: your-docker-username/my-python-app:latest

        ports:

        - containerPort: 5000

Creamos otro de los servicios. Aquí es donde asignamos un puerto en especifico

apiVersion: v1

kind: Service

metadata:

  name: my-python-app

spec:

  selector:

    app: my-python-app

  ports:

    - protocol: TCP

      port: 80

      targetPort: 5000

  type: LoadBalancer

Ahora uno de ingreso. Aquí creamos un ingress para configurar el enrutamiento de tráfico de un dominio a el servicio.

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

  name: my-python-app-ingress

spec:

  rules:

  - host: your-domain.com

    http:

      paths:

      - path: /

        pathType: Prefix

        backend:

          service:

            name: my-python-app

            port:

              number: 80

Básicamente este es un ejemplo muy sencillo de una aplicación web que únicamente muestra un saludo. La aplicación es empaquetada dentro de un contenedor usando Docker y esta es desplegada en un Kubernetes.

Con el siguiente resultado.

