Deploying a 12-Factor App with Docker and Kubernetes

This guide provides a step-by-step walkthrough for deploying a containerised application that follows the 12-Factor methodology. We will use Docker to build the container image and a local Kubernetes cluster, managed by Minikube, for deployment.

What is a 12-Factor App?

The 12-Factor App is a set of principles for building software-as-a-service (SaaS) applications that are robust, scalable, and portable. Key factors we'll focus on in this guide include:

- **Factor III: Config** - Storing configuration in the environment.
- **Factor VII: Port Binding** - Exporting services via a port.

Prerequisites

Before we begin, ensure you have the following tools installed on your system:

- **Docker:** Used to build and run the container image.
- **Minikube:** A tool that runs a single-node Kubernetes cluster on your local machine.
- **kubectl:** The command-line tool for interacting with your Kubernetes cluster.

Step 1: Create the Sample App

This simple Python Flask application adheres to the 12-Factor methodology by reading its configuration, `MESSAGE`, from an environment variable.

main.py

```
import os
from flask import Flask

app = Flask(__name__)

# Factor III: Config - Get configuration from the environment
MESSAGE = os.environ.get('MESSAGE', 'Hello, World!')

@app.route('/')
def hello():
    return f"<h1>{MESSAGE}</h1>"

if __name__ == '__main__':
    # Factor VII: Port Binding - Export service via port binding
    port = int(os.environ.get('PORT', 5000))
    app.run(host='0.0.0.0', port=port)
```

requirements.txt

```
flask
```

Step 2: Containerize the App with Docker

This `Dockerfile` packages the application and its dependencies into a single, portable container image.

Dockerfile

```
# Use an official Python runtime as a parent image
FROM python:3.9-slim

# Set the working directory in the container
WORKDIR /app
```

```
# Copy the dependencies file to the working directory
COPY requirements.txt .

# Install any needed packages specified in requirements.txt
RUN pip install --no-cache-dir -r requirements.txt

# Copy the rest of the application code
COPY . .

# Expose the port the app runs on
EXPOSE 5000

# Run the application
CMD ["python", "main.py"]
```

Step 3: Create Kubernetes Manifests

This `kubernetes-manifest.yaml` file defines a Kubernetes Deployment to manage the application's Pods and a Service to expose the application to the network. The Deployment uses a `ConfigMap` to inject the `MESSAGE` environment variable, demonstrating the "Config" factor.

kubernetes-manifest.yaml

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config
data:
  message: "Greetings from a Kubernetes Pod!"
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-12-factor-app-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-12-factor-app
  template:
    metadata:
      labels:
        app: my-12-factor-app
    spec:
      containers:
      - name: my-12-factor-app-container
        image: my-12-factor-app-image:v1
        ports:
        - containerPort: 5000
        env:
        # Load environment variables from the ConfigMap
        - name: MESSAGE
          valueFrom:
            configMapKeyRef:
              name: app-config
              key: message
---
apiVersion: v1
kind: Service
```

```
metadata:
  name: my-12-factor-app-service
spec:
  selector:
    app: my-12-factor-app
  ports:
  - protocol: TCP
    port: 80
    targetPort: 5000
  type: NodePort
```

Instructions for Deployment

Start Minikube and Build the Docker Image: Open your terminal and navigate to the directory where you saved the files. The `minikube -p minikube docker-env` command points your Docker daemon to the Minikube cluster, ensuring the image is available to Kubernetes.

```
minikube start --driver=docker
eval $(minikube -p minikube docker-env)
docker build -t my-12-factor-app-image:v1 .
```

Apply the Kubernetes Manifest:

```
kubectl apply -f kubernetes-manifest.yaml
```

Check the Deployment

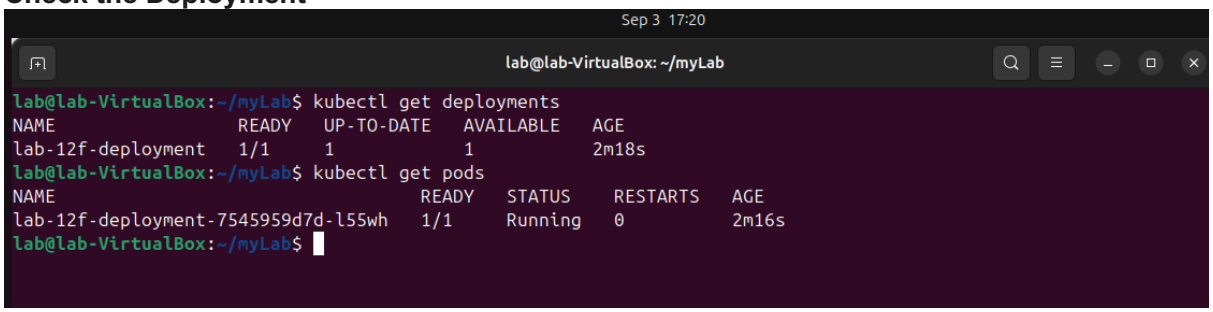

Figure 1: Pods and Deployments in Kubernetes

Access the Application:

```
minikube service my-12-factor-app-service --url
```

This command will provide the URL to access your application in a web browser.

Figure 2: Access the Application

Attachment: logging.txt – this file shows the deployment