hello-world-go-webserver

Overview

This project demonstrates a self-contained deployment of a simple Go web server that responds with "Hello World" to HTTP requests. The solution uses Docker for containerization and adheres to infrastructure-as-code principles, ensuring the deployment is reusable and consistent across different environments. Additionally, the project includes Kubernetes configuration files for deploying the web server in a Kubernetes cluster and accessing it outside the cluster.

Project Structure

```
hello-world-webserver-main/
├── Dockerfile                                # Dockerfile for building the
Docker image
├── hello-world-deployment-improvements       # Files for improving a Kubernetes
deployment
├── README.md                                 # Documentation file
├── README.pdf                                # Documentation file in pdf
├── app/
│   ├── main.go                               # Go source code for the web server
│   ├── go.mod                                # The Go module file that manages
dependencies
├── k8s/
│   ├── deployment.yaml                       # Kubernetes Deployment file
│   └── service.yaml                          # Kubernetes Service file
├── installAndDeploy.sh                       # Script to install K3s and deploy
the application
```

Prerequisites

To build and run this project, you need the following software installed on your Unix-based system:

- **Docker**: Ensure Docker is installed and running on your machine. You can download it from Docker's official website.

Setup

1. Clone the repository or copy the files into a directory on your local machine.
2. Navigate to the project directory.

Build the Docker Image

Build the Docker image using the following command:

```
docker build -t hello-world-go .
```

Run the Docker Container

Run the Docker container using the following command:

```
docker run -p 8080:8080 hello-world-go
```

This command maps port 8080 on your host machine to port 8080 in the container, making the web server accessible from your browser or any HTTP client.

Access the Web Server

Once the container is running, access the web server by navigating to the following URL in your web browser:

```
http://localhost:8080
```

You should see the message "Hello World" displayed.

Option: Use Pre-built Image from Docker Hub

Alternatively, you can use the pre-built image that has already been pushed to my registry. Follow these commands:

To pull the image from my Docker Hub registry, use the following command:

```
docker pull bhavinprajapti/hello-world-go:1.0
```

Run the Docker container using the image pulled from the registry with the following command:

```
docker run -p 8080:8080 bhavinprajapti/hello-world-go:1.0
```

Additional Deployment with Kubernetes

In addition to running the web server in Docker, you can deploy it on a Kubernetes cluster. Below are the steps to do so.

Prerequisites for Kubernetes Deployment

- Kubernetes cluster set up (local or cloud-based).
- `kubectl` configured to interact with your Kubernetes cluster.

Push Docker Image to Docker Hub

Ensure the Docker image is pushed to Docker Hub so that it can be pulled by Kubernetes nodes.

```
docker tag hello-world:latest <your-docker-repo>/hello-world-go:1.0
docker push <your-docker-repo>/hello-world-go:1.0
```

Apply Kubernetes Configurations

Navigate to the `k8s` directory:

```
cd k8s
```

Apply the deployment and service configurations:

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

Access the Web Server

Once the service is created, you can get the NodePort assigned to the service using:

```
kubectl get services
```

Look for the `PORT(S)` column for the `hello-world-service` service. It will display something like `80:<NodePort>`. Find the IP address of one of your nodes (you can use `kubectl get nodes -o wide`), then open your web browser and navigate to:

```
http://<Node_IP>:<NodePort>
```

You should see "Hello World" displayed.

k3s Installation and Deploy application

This section guides you through installing K3s, a lightweight Kubernetes distribution, and deploying the application on it.

Prerequisites

- A Unix-based system.
- Sudo privileges.

Setup

1. copy the `installAndDeploy.sh` into a directory on your local machine.

2. Navigate to the project directory.

Running the Script

1. Make the script executable:

```
chmod +x installAndDeploy.sh
```

2. Run the script:

```
./installAndDeploy.sh
```

You should see "Hello World" displayed.