Date: 01.07.2025



Copyright @ 2024 PibyThree.com All Rights Reserved

Web Application in Kubernetes

Contents

[1. Script to install and setup k8s requirements in EC2 Instance with Linux AMI 2](#_Toc202278133)

[2. Creating image for a simple web application 2](#_Toc202278134)

[a. Final Directory Structure 2](#_Toc202278135)

[b. Code 2](#_Toc202278136)

[c. Next steps 4](#_Toc202278137)

[3. Initializing k8s cluster and accessing that web app from browser 4](#_Toc202278138)

# 1. Script to install and setup k8s requirements in EC2 Instance with Linux AMI

#!/usr/bin/env bash

set -e

sudo yum update -y

sudo yum install docker -y

sudo systemctl start docker

sudo usermod -aG docker $USER

curl -LO https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64

sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64

curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"

sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

newgrp docker

# 2. Creating image for a simple web application

## Final Directory Structure

jswa/

|-- .dockerignore

|-- Dockerfile

|-- node\_modules/

|-- package-lock.json

|-- package.json

|-- public/

|-- app.js

|-- index.html

|-- style.css

|-- server.js

## Code

|  |  |  |
| --- | --- | --- |
| package.json | | style.css |
| {  "name": "jswa",  "version": "1.0.0",  "description": "Simple web app in JavaScript",  "main": "server.js",  "scripts": {  "start": "node server.js",  "test": "echo \"Error: no test specified\" && exit 1"  },  "author": "aryan-fafo",  "license": "ISC",  "dependencies": {  "express" : "^5.1.0"  }  } | | body {  height: 100vh;  width: 50%;  margin: auto;  }  section {  display: flex;  }  section, div {  align-items: center;  justify-content: center;  }  input, button{  padding: 10px;  margin: 2px;  } |
| index.html | | |
| <!DOCTYPE html>  <html>  <head>  <title>Login Demo</title>  <link rel="stylesheet" href="style.css">  </head>  <body>  <h1>Login</h1>  <hr>  <section><div>  <input type="text" id="username" placeholder="Username"><br>  <input type="password" id="password" placeholder="Password"><br>  <section><div>  <button onclick="login()">Login</button></div></section>  <p id="message"></p>  </div></section>  <hr>  <script src="app.js"></script>  </body>  </html> | | |
| app.js | | |
| function login() {  const user = document.getElementById("username").value;  const pass = document.getElementById("password").value;  if (user === "admin" && pass === "admin@123")  { document.body.innerHTML = `<hr><h2>Welcome ${user}!</h2><hr>` }  else  { document.getElementById("message").textContent = "Wrong username or password!" }  } | | |
| server.js | | |
| const express = require('express')  const app = express()  const PORT = 5000  app.use(express.static('public'));  app.listen(PORT, () => { console.log(`App running on http://localhost:${PORT}`); }); | | |
| Dockerfile | .dockerignore | |
| FROM node:alpine  WORKDIR /app  COPY package.json ./  RUN npm install  COPY . .  EXPOSE 5000  CMD ["npm", "start"] | node\_modules/ | |

## Next steps

1. `npm install` to install required dependencies
2. `sudo docker login -u aryanfafo docker.io` to login into repository
3. `sudo docker build -t aryanfafo/jswa:latest` to build the image
4. (optional) `sudo docker build -d -p 5000:5000 aryanfafo/jswa:latest` to verify image
5. `sudo docker push aryanfafo/jswa:latest` to push the image to dockerhub

# 3. Initializing k8s cluster and accessing that web app from browser

1. `minikube start`
2. `kubectl create deployment jswa aryanfafo/jswa:latest`
3. `kubectl expose deployment jswa –type=LoadBalancer –port=5000`
4. Open a new terminal with SSH connection to the instance and run `minikube tunnel`
5. Switch to the previous terminal and run `kubectl get svc jswa` to get external IP
6. Install socat - `sudo yum install socat -y`
7. Forward EC2 public port 5000 to minikube internal IP - `sudo socat TCP4-LISTEN:5000, fork TCP4:<EXTERNAL\_IP>:5000`
8. Access the web app with `http://<EC2\_IP>:5000` from browser