**Customized Docker / Kubernetes Workshop**

**AMD**

**Aug 2024**

## Download workshop material from:

<https://github.com/victor-tan-hk/DockerInDepth>

<https://github.com/victor-tan-hk/KubernetesBasics>

INSTALL MINIKUBE FROM

<https://minikube.sigs.k8s.io/docs/start/>

Install Postman (a REST API client)

<https://www.postman.com/downloads/>

To use, register a new account or use an existing GMail account

If you already have a favorite IDE of your own, use that, otherwise install VS Code

<https://code.visualstudio.com/download>

## Questions

CAn you bind mount to a single file in the container file system

<https://www.baeldung.com/ops/docker-mount-single-file-in-volume>

What are the different ways of filtering a list of pods that are running in a Kubernetes cluster ?

<https://www.baeldung.com/ops/kubernetes-list-every-pod-node#:~:text=Filtering>

<https://www.warp.dev/terminus/kubectl-list-pods>

<https://stackoverflow.com/questions/70282381/kubectl-get-pods-how-to-filter-pods-by-partial-name>

Solution for SSL Certificate error due to corporate firewall problem

<https://forums.docker.com/t/ssl-certificate-error-in-quickstart-behind-corporate-firewall/125858/2>

## **SOLUTION FOR SSL CERTIFICATE ERROR WHEN DOING NPM INSTALL**

Add into Dockerfile

RUN npm config set strict-ssl false

## **SOLUTION FOR SSL CERTIFICATE ERROR FIREWALL PROBLEM**

const express = require('express');

const bodyParser = require('body-parser');

const axios = require('axios').default;

const mongoose = require('mongoose');

const Favorite = require('./models/favorite');

const https = require('https')

const app = express();

const instance = axios.create({

httpsAgent: new https.Agent({ rejectUnauthorized: false })

});

app.use(bodyParser.json());

app.get('/favorites', async (req, res) => {

const favorites = await Favorite.find();

res.status(200).json({

favorites: favorites,

});

});

app.post('/favorites', async (req, res) => {

const favName = req.body.name;

const favType = req.body.type;

const favUrl = req.body.url;

try {

if (favType !== 'movie' && favType !== 'character') {

throw new Error('"type" should be "movie" or "character"!');

}

const existingFav = await Favorite.findOne({ name: favName });

if (existingFav) {

throw new Error('Favorite exists already!');

}

} catch (error) {

return res.status(500).json({ message: error.message });

}

const favorite = new Favorite({

name: favName,

type: favType,

url: favUrl,

});

try {

await favorite.save();

res

.status(201)

.json({ message: 'Favorite saved!', favorite: favorite.toObject() });

} catch (error) {

res.status(500).json({ message: 'Something went wrong.' });

}

});

app.get('/movies', async (req, res) => {

try {

const response = await instance.get('https://swapi.dev/api/films');

res.status(200).json({ movies: response.data });

} catch (error) {

res.status(500).json({ message: 'Something went wrong.' });

}

});

app.get('/people', async (req, res) => {

try {

const response = await instance.get('https://swapi.dev/api/people');

res.status(200).json({ people: response.data });

} catch (error) {

console.log(error);

res.status(500).json({ message: 'Something went wrong.' });

}

});