// Source code for CI/CD PIPELINE To host the application on AWS EC2..

App.js

const express = require('express');

const app = express();

const port = 3000;

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

  res.send('<h1>Express Demo App</h1> <h4>Message: Success</h4> <p>Version 1.1</p>');

})

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

  res.send([

    {

      productId: '101',

      price: 100

    },

    {

      productId: '102',

      price: 150

    }

  ])

})

app.listen(port, ()=> {

  console.log(`Demo app is up and listening to port: ${port}`);

})

// Appspec.yml

version: 0.0

os: linux

files:

  - source: /

    destination: /home/ec2-user/express-app

hooks:

  ApplicationStop:

    - location: scripts/application\_stop.sh

      timeout: 300

      runas: ec2-user

  BeforeInstall:

    - location: scripts/before\_install.sh

      timeout: 300

      runas: ec2-user

  ApplicationStart:

    - location: scripts/application\_start.sh

      timeout: 300

      runas: ec2-user

// package.json

{

  "name": "nodejs-express-on-aws-ec2",

  "version": "0.0.1",

  "dependencies": {

    "express": "4.17.1"

  }

}

// README.md

**# nodejs-express-on-aws-ec2**

This repo hosts the source code for my YouTube tutorial on CI/CD from Github to an AWS EC2 instance via CodePipeline and CodeDeploy (https://www.youtube.com/watch?v=Buh3GjHPmjo). This tutorial uses a node.js express app as an example for the demo.

I also created a video to talk about how to fix some of the common CodeDeploy failures I have run into (https://www.youtube.com/watch?v=sXZVkOH6hrA). Below are a couple of examples:

```

ApplicationStop failed with exit code 1

```

```

The overall deployment failed because too many individual instances failed deployment, too few healthy instances are available for deployment, or some instances in your deployment group are experiencing problems.

```

**===========================**

EC2 script on creation to install the CodeDeploy Agent:

```

#!/bin/bash

sudo yum -y update

sudo yum -y install ruby

sudo yum -y install wget

cd /home/ec2-user

wget https://aws-codedeploy-us-east-1.s3.amazonaws.com/latest/install

sudo chmod +x ./install

sudo ./install auto

```

Check if CodeDeploy agent is running:

```

sudo service codedeploy-agent status

```

Location for CodeDeploy logs:

```

/opt/codedeploy-agent/deployment-root/deployment-logs/codedeploy-agent-deployments.log

```

Uninstall CodeDeploy Agent:

```

sudo yum erase codedeploy-agent

```

// scripts

$ application\_start.sh

#!/bin/bash

#give permission for everything in the express-app directory

sudo chmod -R 777 /home/ec2-user/express-app

#navigate into our working directory where we have all our github files

cd /home/ec2-user/express-app

#add npm and node to path

export NVM\_DIR="$HOME/.nvm"

[ -s "$NVM\_DIR/nvm.sh" ] && \. "$NVM\_DIR/nvm.sh"  # loads nvm

[ -s "$NVM\_DIR/bash\_completion" ] && \. "$NVM\_DIR/bash\_completion"  # loads nvm bash\_completion (node is in path now)

#install node modules

npm install

#start our node app in the background

node app.js > app.out.log 2> app.err.log < /dev/null &

$ application\_stop.sh

#!/bin/bash

#Stopping existing node servers

echo "Stopping any existing node servers"

pkill node

$ before\_install.sh

#!/bin/bash

#download node and npm

curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.34.0/install.sh | bash

. ~/.nvm/nvm.sh

nvm install node

#create our working directory if it doesnt exist

DIR="/home/ec2-user/express-app"

if [ -d "$DIR" ]; then

  echo "${DIR} exists"

else

  echo "Creating ${DIR} directory"

  mkdir ${DIR}

fi