

## DEVOPS ASSIGNMENT

### TEAM 8

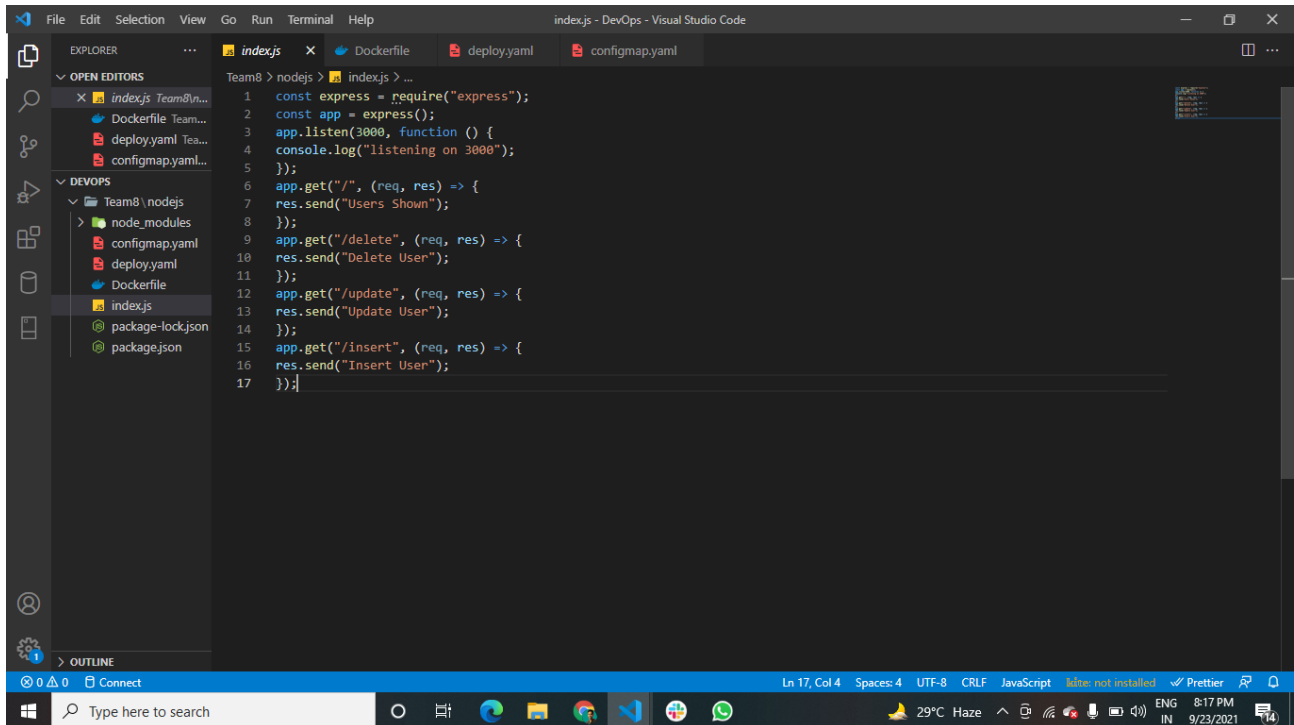
You can find the docker file here :

<https://hub.docker.com/r/deepanshusachdeva5/nodejs-starter>

Github link - [https://github.com/deepanshusachdeva5/CS457\\_DEVOPS/tree/main/Assignment2](https://github.com/deepanshusachdeva5/CS457_DEVOPS/tree/main/Assignment2)

### Q. Developing and deploying a Node.js app from Docker to Kubernetes

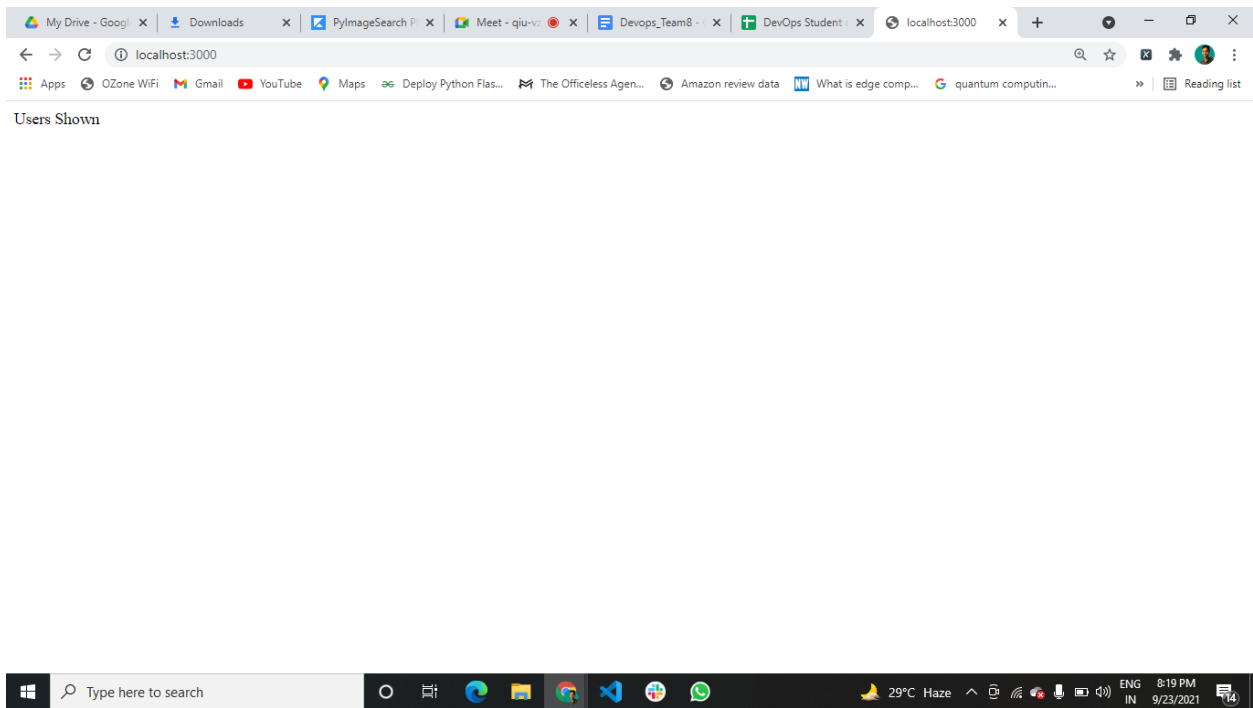
#### STEP1: NodeJs App



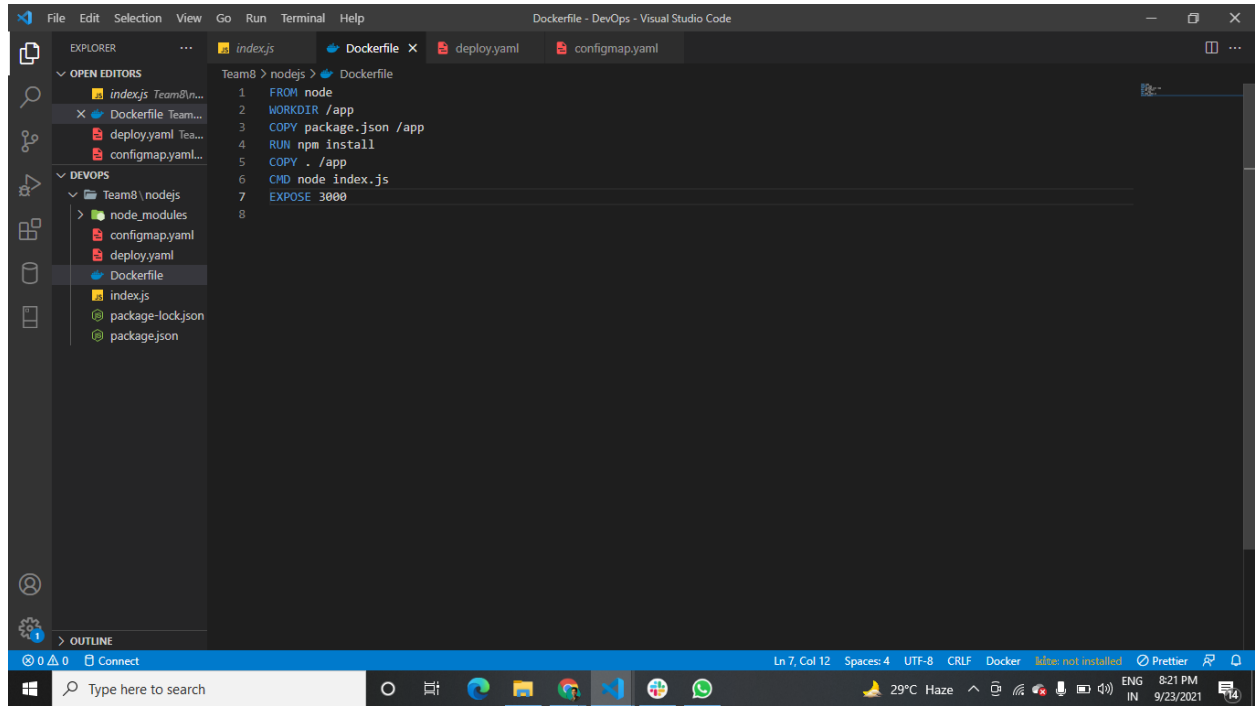
The screenshot shows the Visual Studio Code editor with a project named 'index.js - DevOps'. The Explorer sidebar on the left shows the file structure under 'Team8 > nodejs', including 'node\_modules', 'configmap.yaml', 'deploy.yaml', 'Dockerfile', 'index.js', 'package-lock.json', and 'package.json'. The main editor area displays the content of 'index.js', which is a Node.js application using Express.js. The code defines an Express app, listens on port 3000, and implements three routes: a root route that sends 'Users Shown', a delete route that sends 'Delete User', and an insert route that sends 'Insert User'. The status bar at the bottom indicates the current position is Line 17, Column 4, and shows various settings like 'Spaces: 4', 'UTF-8', 'CRLF', and 'JavaScript'.

```
1 const express = require("express");
2 const app = express();
3 app.listen(3000, function () {
4   console.log("listening on 3000");
5 });
6 app.get("/", (req, res) => {
7   res.send("Users Shown");
8 });
9 app.get("/delete", (req, res) => {
10  res.send("Delete User");
11 });
12 app.get("/update", (req, res) => {
13  res.send("Update User");
14 });
15 app.get("/insert", (req, res) => {
16  res.send("Insert User");
17 });
```

## STEP2: Testing NodeJs App using npm start



### STEP3: Writing Dockerfile



The screenshot shows the Visual Studio Code interface with the Dockerfile editor open. The Explorer sidebar on the left shows the file structure of the project, including the Dockerfile. The Dockerfile content is as follows:

```
1 FROM node
2 WORKDIR /app
3 COPY package.json /app
4 RUN npm install
5 COPY . /app
6 CMD node index.js
7 EXPOSE 3000
8
```

The status bar at the bottom indicates the current line and column (Ln 7, Col 12) and shows the Docker extension is not installed.

### STEP4: Building Docker Image using docker build -t node-starter

The screenshot shows the Visual Studio Code interface with a file explorer on the left showing 'index.js', 'Dockerfile', and 'deploy.yaml'. The main editor displays the 'deploy.yaml' file with the following content:

```
1 apiVersion: apps/v1 #1
2 kind: Deployment #2
3 metadata: #3
4   name: nodejs-deployment #4
```

The terminal window shows the output of the 'docker build' command. It starts with an error: 'failed to solve with frontend dockerfile.v0: failed to read dockerfile: open /var/lib/docker/tmp/buildkit-mount448437805/Dockerfile: no such file or directory'. After running 'cd .\nodejs\' and 'docker build -t node-server', the build process begins. It shows the resolution of the Dockerfile, the transfer of context, and the pulling of the 'node:latest' image from Docker Hub. The final output shows the image 'node-server' is built successfully with a size of 192.39MB.

STEP5: Running Docker Image

The screenshot shows the Visual Studio Code interface with the 'deploy.yaml' file open. The terminal window shows the output of the 'docker run' command. It starts with the command 'docker run -d --name nodongo -p 3000:3000 node-server'. The output shows the container 'nodongo' is created and started. It then shows the command 'docker login' being executed, followed by the command 'docker push deepanshusachdeva5/nodejs-starter:1.1'. The output shows the image is pushed successfully to Docker Hub.

STEP6: Pushing image to docker hub after tagging

The screenshot shows the Visual Studio Code interface with the 'deploy.yaml' file open in the editor. The file contains a Kubernetes deployment configuration for a Node.js application. The terminal window shows the execution of Docker commands to build and push the application image to Docker Hub.

```
1 apiVersion: apps/v1 #1
2 kind: Deployment #2
3 metadata: #3
4   name: nodejs-deployment #4
```

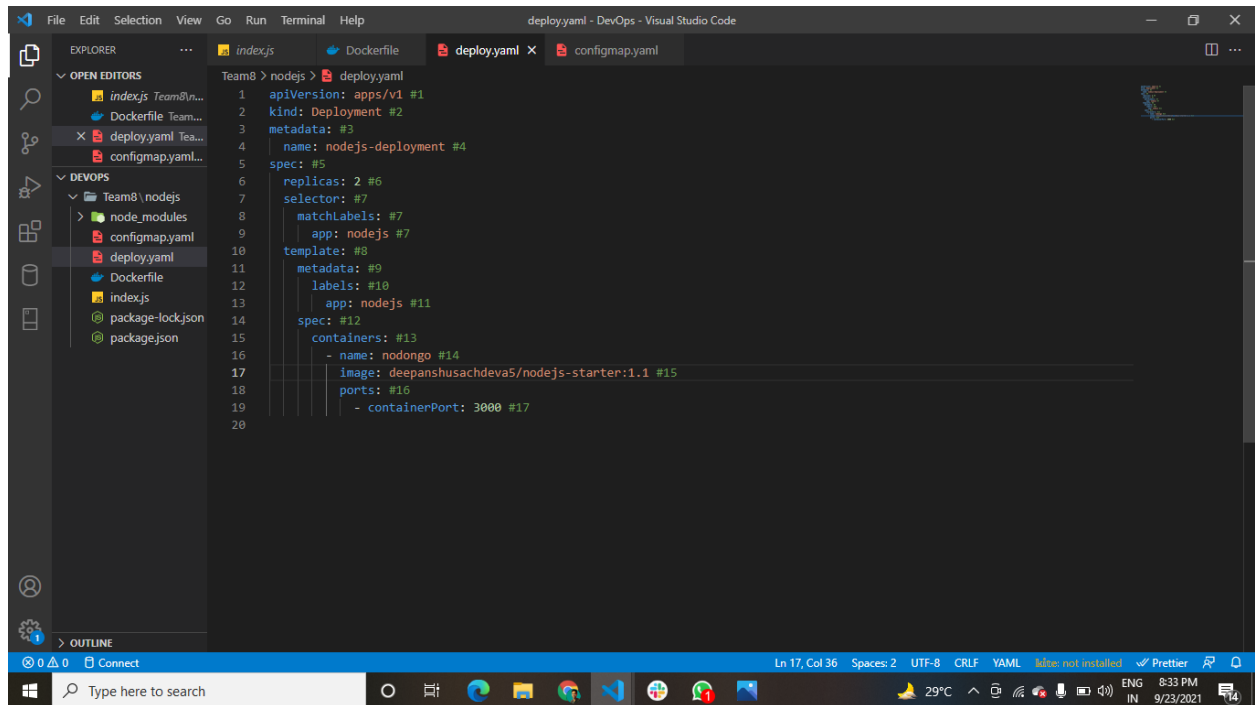
```
PS C:\Users\HP\Desktop\DevOps\Team8\nodejs> docker run -d --name nodongo -p 3000:3000 node-server
e8400dfb8bf99ac01abb077e8aa6e44e879e2568eb523fbf800358d087d356c5
Removing login credentials for https://index.docker.io/v1/
PS C:\Users\HP\Desktop\DevOps\Team8\nodejs> docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: deepanshusachdeva5
Password:
Login Succeeded
PS C:\Users\HP\Desktop\DevOps\Team8\nodejs> docker tag node-server deepanshusachdeva5/nodejs-starter
PS C:\Users\HP\Desktop\DevOps\Team8\nodejs> docker push deepanshusachdeva5/nodejs-starter:1.1
The push refers to repository [docker.io/deepanshusachdeva5/nodejs-starter]
tag does not exist: deepanshusachdeva5/nodejs-starter:1.1
PS C:\Users\HP\Desktop\DevOps\Team8\nodejs> docker push deepanshusachdeva5/nodejs-starter
Using default tag: latest
The push refers to repository [docker.io/deepanshusachdeva5/nodejs-starter]
c4c3fa1f2b7a: Pushed
55506e955bc5: Pushed
2b2948de99aa: Pushed
12ba348a2b30: Pushed
3ebf0ae2cdfb: Layer already exists
d95faf3ef1b0: Layer already exists
0e681c4be8c0: Layer already exists
ac5ca208761a: Layer already exists
6db3d1445d34: Layer already exists
1c721e0a3622: Layer already exists
1c721e0a3622: Layer already exists
6680d5caa699: Layer already exists
4dd0c5812fd4: Layer already exists
latest: digest: sha256:22608a234e5372daf16af360d1f74739379fa458c0c7ddb49200107b7e9bc1b size: 3049
```

## STEP7: Minikube Start

The screenshot shows the Visual Studio Code interface with the 'deploy.yaml' file open in the editor. The terminal window shows the execution of Minikube commands to start the cluster and deploy the application.

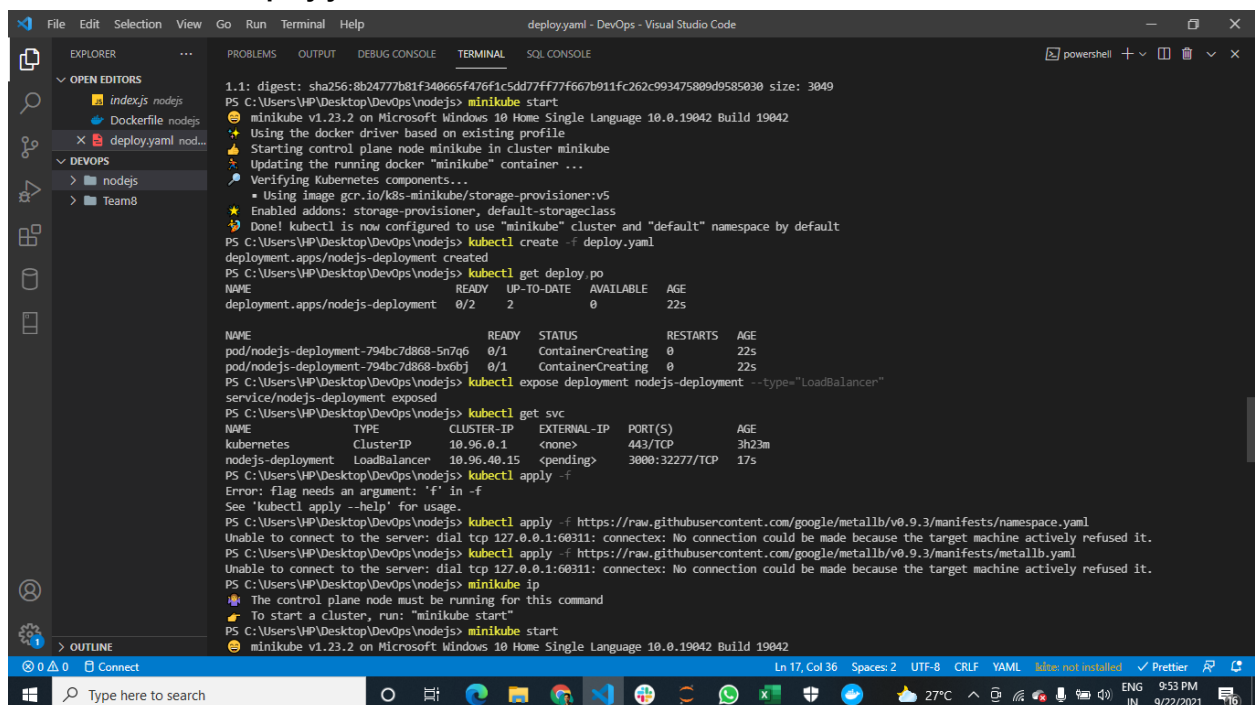
```
PS C:\Users\HP\Desktop\DevOps\nodejs> docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: deepanshusachdeva5
Password:
Login Succeeded
The push refers to repository [docker.io/deepanshusachdeva5/nodejs-starter]
c6037137d5b6: Pushed
174a12ba5978: Pushed
6e0023cb31fe: Pushed
36bb431f0787: Pushed
3ebf0ae2cdfb: Mounted from library/node
d95faf3ef1b0: Mounted from library/node
0e681c4be8c0: Mounted from library/node
ac5ca208761a: Mounted from library/node
6db3d1445d34: Mounted from library/node
1c721e0a3622: Mounted from library/node
4dd0c5812fd4: Mounted from library/node
1.1: digest: sha256:8b24777b81f340665f476f1c5dd77ff77f667b911fc262c993475809d9585030 size: 3049
PS C:\Users\HP\Desktop\DevOps\nodejs> minikube start
minikube v1.23.2 on Microsoft Windows 10 Home Single Language 10.0.19042 Build 19042
* Using the docker driver based on existing profile
* Starting control plane node minikube in cluster minikube
* Updating the running docker "minikube" container ...
* Verifying Kubernetes components...
  * Using image gcr.io/k8s-minikube/storage-provisioner:v5
  * Enabled addons: storage-provisioner, default-storageclass
  * Don't kubectl is now configured to use "minikube" cluster and "default" namespace by default
PS C:\Users\HP\Desktop\DevOps\nodejs> kubectl create -f deploy.yaml
deployment.apps/nodejs-deployment created
PS C:\Users\HP\Desktop\DevOps\nodejs> kubectl get deploy po
NAME READY UP-TO-DATE AVAILABLE AGE
deployment.apps/nodejs-deployment 0/2 2 0 22s
NAME READY STATUS RESTARTS AGE
pod/nodejs-deployment-794bc7d868-5n7q6 0/1 ContainerCreating 0 22s
pod/nodejs-deployment-794bc7d868-bx6bj 0/1 ContainerCreating 0 22s
PS C:\Users\HP\Desktop\DevOps\nodejs> kubectl expose deployment nodejs-deployment --type=LoadBalancer
```

## STEP8: Creating deploy.yaml file



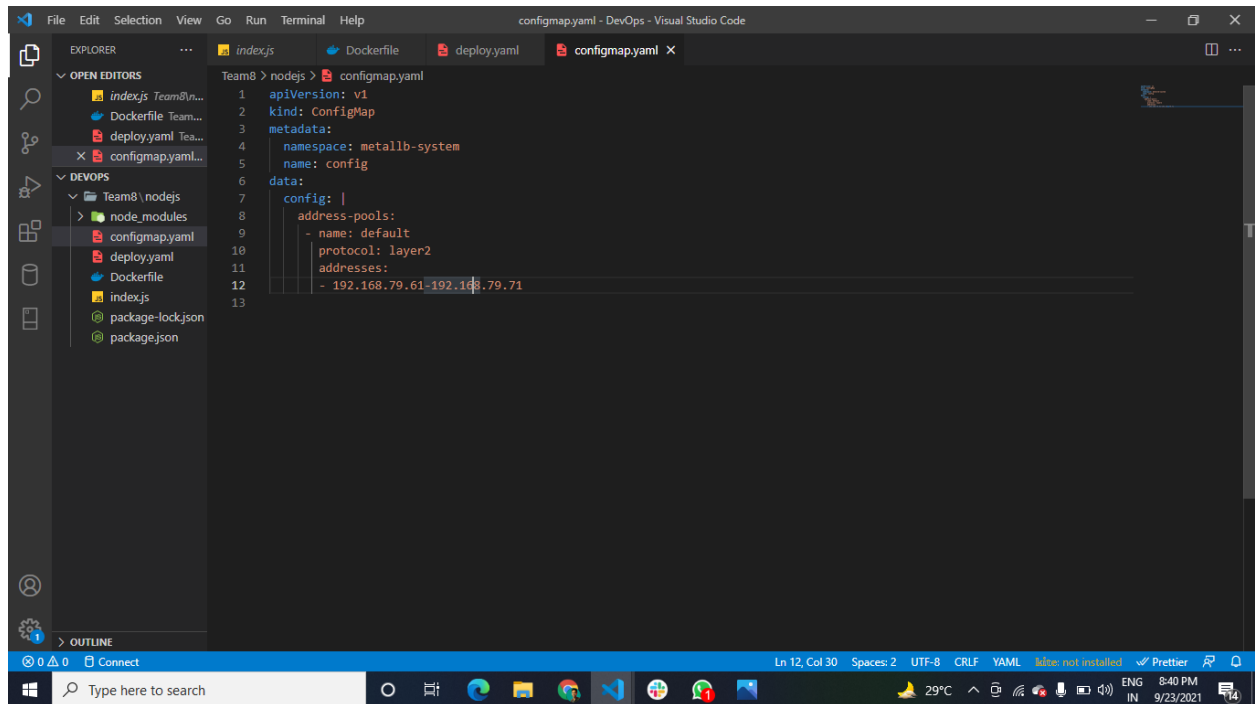
```
1 apiVersion: apps/v1 #1
2 kind: Deployment #2
3 metadata: #3
4   name: nodejs-deployment #4
5 spec: #5
6   replicas: 2 #6
7   selector: #7
8     matchLabels: #7
9     app: nodejs #7
10  template: #8
11    metadata: #9
12      labels: #10
13        app: nodejs #11
14    spec: #12
15      containers: #13
16        - name: nodongo #14
17          image: deepanshusachdeva5/nodejs-starter:1.1 #15
18          ports: #16
19            - containerPort: 3000 #17
```

## STEP9: Kubectl -f deploy.yaml

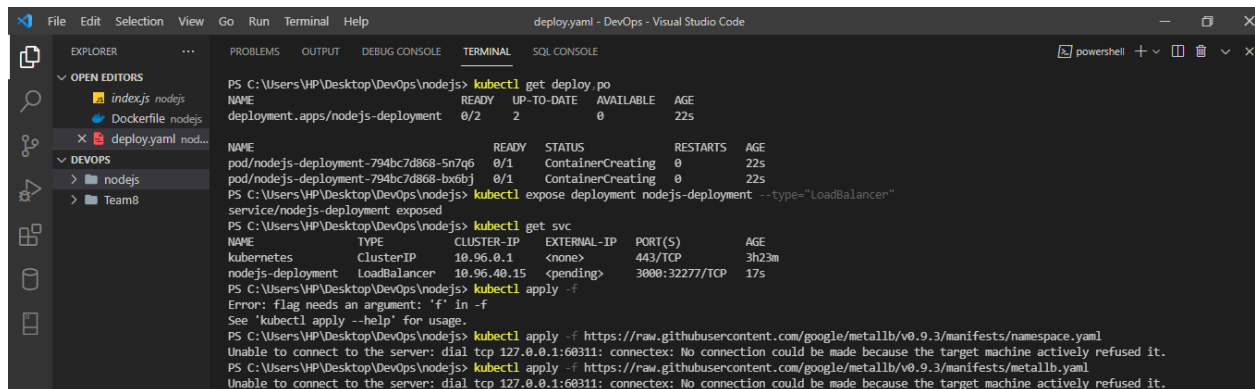


```
1.1: digest: sha256:8b24777b81f348665f476f1c5dd7ff77f667b911fc262c993475809d9585030 size: 3049
PS C:\Users\HP\Desktop\DevOps\nodejs> minikube start
minikube v1.23.2 on Microsoft Windows 10 Home Single Language 10.0.19042 Build 19042
* Using the docker driver based on existing profile
* Starting control plane node minikube in cluster minikube
* Updating the running docker "minikube" container ...
* Verifying Kubernetes components...
  * Using image gcr.io/k8s-minikube/storage-provisioner:v5
  * Enabled addons: storage-provisioner, default-storageclass
  * Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
PS C:\Users\HP\Desktop\DevOps\nodejs> kubectl create -f deploy.yaml
deployment.apps/nodejs-deployment created
PS C:\Users\HP\Desktop\DevOps\nodejs> kubectl get deploy po
NAME READY UP-TO-DATE AVAILABLE AGE
deployment.apps/nodejs-deployment 0/2 2 0 22s
NAME READY STATUS RESTARTS AGE
pod/nodejs-deployment-794bc7d868-5n7q6 0/1 ContainerCreating 0 22s
pod/nodejs-deployment-794bc7d868-bx6bj 0/1 ContainerCreating 0 22s
PS C:\Users\HP\Desktop\DevOps\nodejs> kubectl expose deployment nodejs-deployment --type=LoadBalancer
service/nodejs-deployment exposed
PS C:\Users\HP\Desktop\DevOps\nodejs> kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 3h23m
nodejs-deployment LoadBalancer 10.96.40.15 <pending> 3000:32277/TCP 17s
PS C:\Users\HP\Desktop\DevOps\nodejs> kubectl apply -f
Error: flag needs an argument: 'f' in -f
See 'kubectl apply --help' for usage.
PS C:\Users\HP\Desktop\DevOps\nodejs> kubectl apply -f https://raw.githubusercontent.com/google/metallb/v0.9.3/manifests/namespace.yaml
Unable to connect to the server: dial tcp 127.0.0.1:60311: connect: No connection could be made because the target machine actively refused it.
PS C:\Users\HP\Desktop\DevOps\nodejs> kubectl apply -f https://raw.githubusercontent.com/google/metallb/v0.9.3/manifests/metallb.yaml
Unable to connect to the server: dial tcp 127.0.0.1:60311: connect: No connection could be made because the target machine actively refused it.
PS C:\Users\HP\Desktop\DevOps\nodejs> minikube ip
minikube v1.23.2 on Microsoft Windows 10 Home Single Language 10.0.19042 Build 19042
The control plane node must be running for this command
To start a cluster, run: "minikube start"
PS C:\Users\HP\Desktop\DevOps\nodejs> minikube start
minikube v1.23.2 on Microsoft Windows 10 Home Single Language 10.0.19042 Build 19042
```

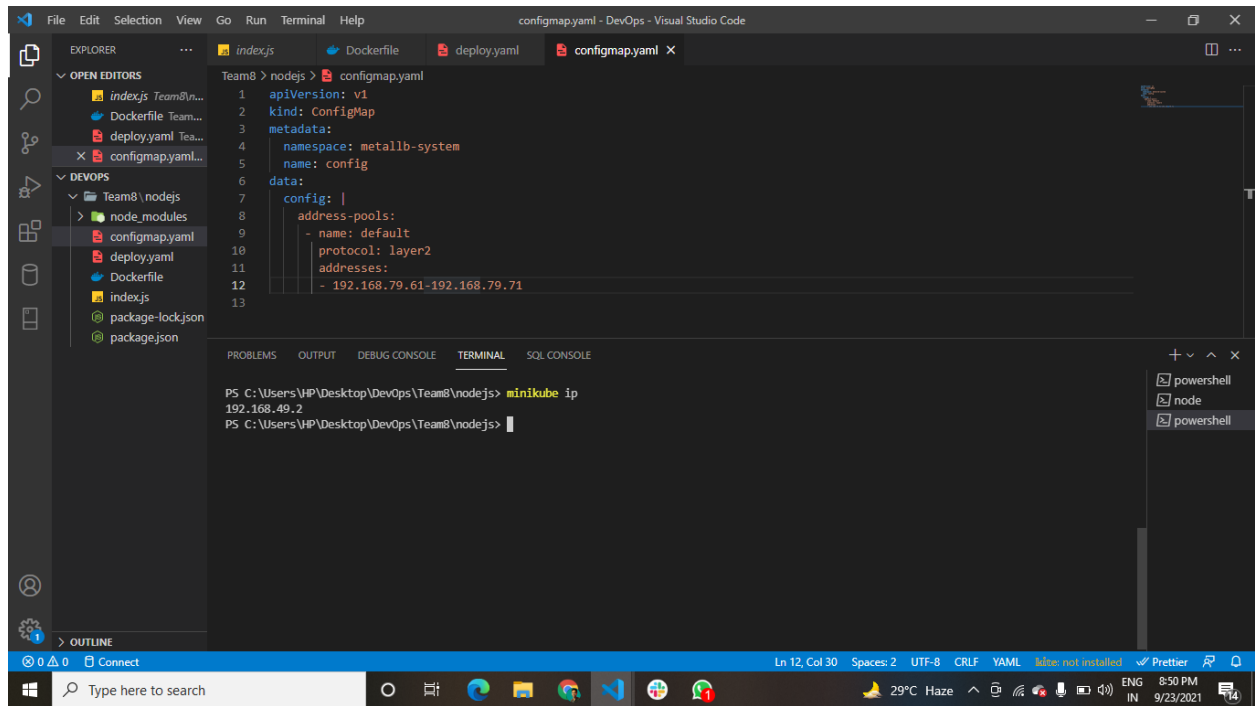
## STEP10: configmap.yaml



## STEP11: Deploying on kubernetes using minikube



## STEP12: Checking minikube ip



### STEP13: Verifying external IP



File Edit Selection View Go Run Terminal Help deploy.yaml - Team8 - Visual Studio Code

EXPLORER

OPEN EDITORS

TEAM8

nodejs

node\_modules

configmap.yaml

deploy.yaml

Dockerfile

index.js

package-lock.json

package.json

nodejs > deploy.yaml

```
1 apiVersion: apps/v1 #1
2 kind: Deployment #2
3 metadata: #3
4   name: nodejs-deployment #4
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL SQL CONSOLE

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell <https://aka.ms/pscore6>

```
PS C:\Users\HP\Desktop\DevOps\Team8> kubectl version
Client Version: version.Info{Major:"1", Minor:"22", GitVersion:"v1.22.2", GitCommit:"8b5a19147530eaac9476b0ab82980b4088bbc1b2", GitTreeState:"clean", BuildDate:"2021-09-15T21:38:50Z", GoVersion:"go1.16.8", Compiler:"gc", Platform:"windows/amd64"}
Server Version: version.Info{Major:"1", Minor:"22", GitVersion:"v1.22.2", GitCommit:"8b5a19147530eaac9476b0ab82980b4088bbc1b2", GitTreeState:"clean", BuildDate:"2021-09-15T21:32:41Z", GoVersion:"go1.16.8", Compiler:"gc", Platform:"linux/amd64"}
PS C:\Users\HP\Desktop\DevOps\Team8> kubectl get namespace
NAME          STATUS   AGE
default       Active  11h
kube-node-lease Active  11h
kube-public   Active  11h
kube-system   Active  11h
metallb-system Active  11h
PS C:\Users\HP\Desktop\DevOps\Team8> kubectl get service
NAME          TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
kubernetes    ClusterIP   10.96.0.1     <none>         443/TCP     11h
nodejs-deployment LoadBalancer 10.102.240.100 192.168.79.61 3000:31642/TCP 11h
PS C:\Users\HP\Desktop\DevOps\Team8>
```

Ln 19, Col 38 Spaces: 2 UTF-8 CRLF YAML kube: not installed ✓ Prettier

Type here to search

31°C 12:20 PM 9/23/2021