**Nodejs project deployment**

**Step 1 : Fork the details into your git**

Use fork button on the public git

**Step 2: set up locally**

Commands used on command prompt

Git clone <https://github.com/Shivani-siri/skillupnodejs.git>

**Step 3: Initialize package.json**

Command: npm init -y

**Step 4: Install Dependencies**

Install the necessary dependencies for the project:

Command : npm install express body-parser mysql

**Step 5: Add the Required Code**

Make sure index.js contains the logic to:

Set up Express routes.

Parse incoming requests.

Connect to a MySQL database.

**Step 5: Set Up the Database**

Start your MySQL server or you do the same using the command line.

Create a database and any required tables.

I used the command line.

Command to login: mysql -u root -p

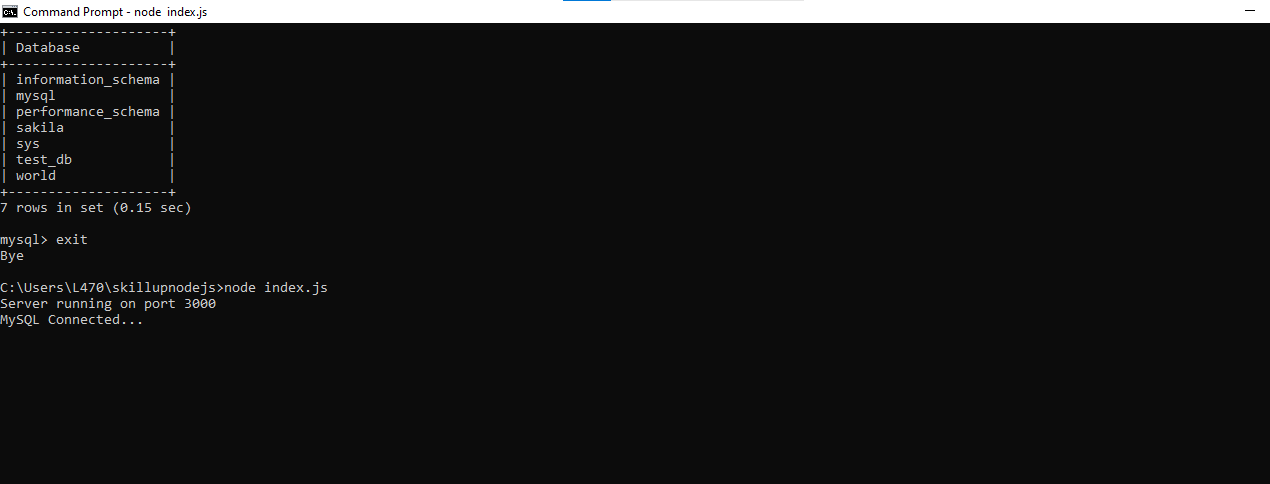
CREATE DATABASE test\_db;

CREATE TABLE example\_table (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100)

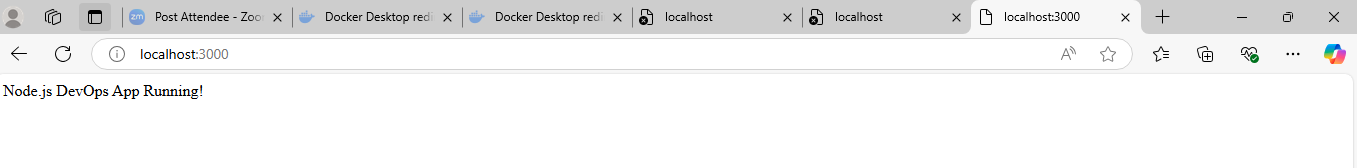
);



Sql is successfully connected

**Step 6: Run the Application**

Command : Start the Node.js server:



Or you can use docker

Docker script:

FROM node:18

WORKDIR /usr/src/app

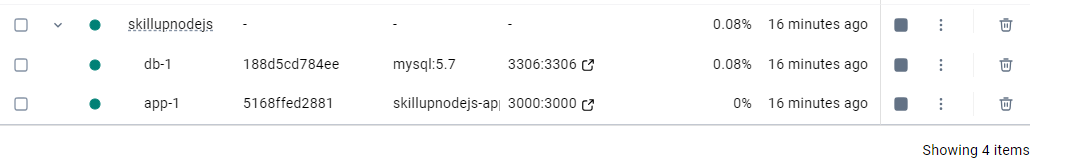
COPY package.json ./

RUN npm install

COPY . .

EXPOSE 3000

CMD ["npm", "index.js"]



**Step 7. Set Up CI/CD with Jenkins**

* Create a Jenkinsfile to automate builds.
* Use GitHub Actions or AWS CodePipeline for deployment.

1. **Basic script I used in the Jenkinsfile given below:**

pipeline {

agent any

stages {

stage('Build') {

steps {

echo 'Building...'

}

}

stage('Test') {

steps {

echo 'Testing...'

}

}

stage('Deploy') {

steps {

echo 'Deploying...'

}

}

}

}

**B. Set Up a New Jenkins Pipeline Job**

Open Jenkins Dashboard.

Click **New Item > Pipeline**.

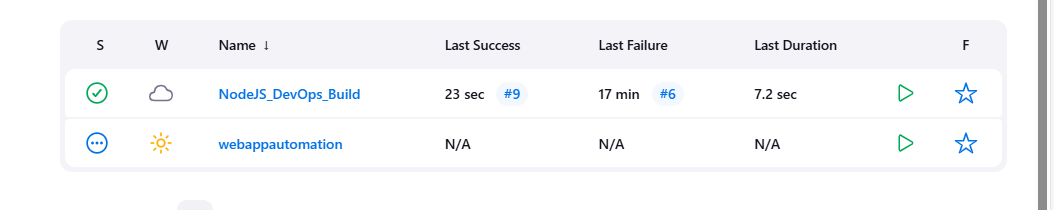
Name it (e.g., NodeJS\_DevOps\_Build).

Under **Pipeline**, select **Pipeline script from SCM**.

Set **SCM** to **Git** and add your GitHub repo URL.

In the **Branch Specifier**, enter \*/main.

Save and trigger a build.



**Build is successful.**

**Step 8: Orchestrating using minikube.**

1. Start minikube

Command: minikube start

**B) Write Kubernetes files**

You'll need below YAML files:

**Deployment** (to manage app instances)

**Service** (to expose the app)

3. **Ingress (optional)** (for friendly URLs)

**Deployment.yaml file**

apiVersion: apps/v1

kind: Deployment

metadata:

name: skillupnodejs

spec:

replicas: 2

selector:

matchLabels:

app: skillupnodejs

template:

metadata:

labels:

app: skillupnodejs

spec:

containers:

- name: skillupnodejs

image: your-dockerhub-username/skillupnodejs:latest # Replace with your actual image

ports:

- containerPort: 3000

env:

- name: NODE\_ENV

value: "production"

**Service.yaml file**

apiVersion: v1

kind: Service

metadata:

name: skillupnodejs-service

spec:

type: NodePort

selector:

app: skillupnodejs

ports:

- protocol: TCP

port: 3000

targetPort: 3000

nodePort: 30080 # Exposes the app on Minikube's IP at port 30080

**Ingress.yaml file**

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

name: skillupnodejs-ingress

spec:

rules:

- host: skillupnodejs.local # Custom domain for local access

http:

paths:

- path: /

pathType: Prefix

backend:

service:

name: skillupnodejs-service

port:

number: 3000

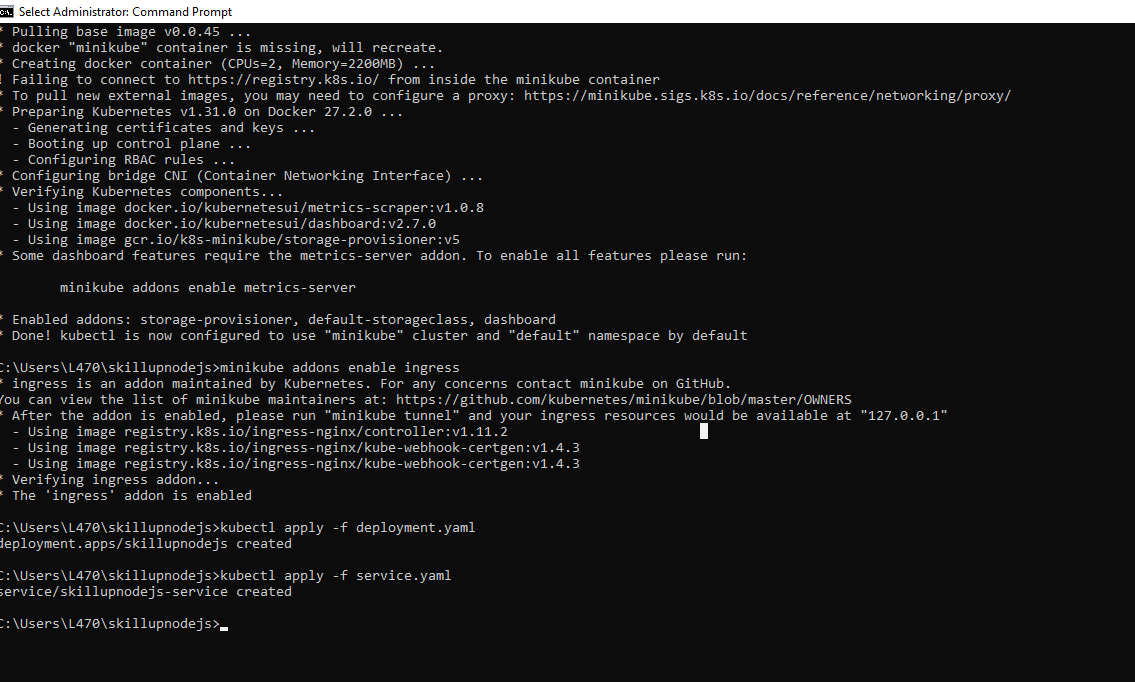
**To use Ingress, enable it first:**

**minikube addons enable ingress**

**Apply the Deployment & Service**

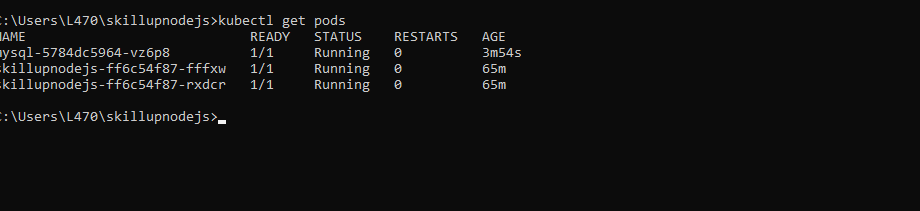
kubectl apply -f deployment.yaml

kubectl apply -f service.yaml



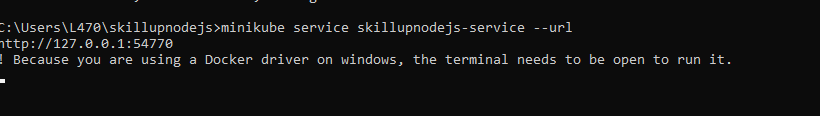
**Check if pods are running**

kubectl get pods



**Check service details**

minikube service skillupnodejs-service --url



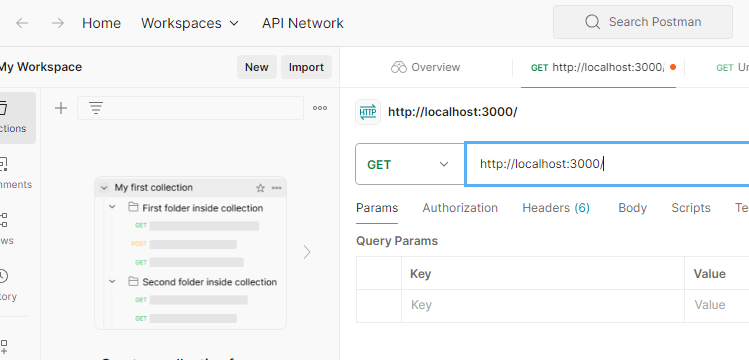
We see that pods are running successfully and service details are available. This means that nodejs application is successfully deployed.

**Step 9 : Monitoring**

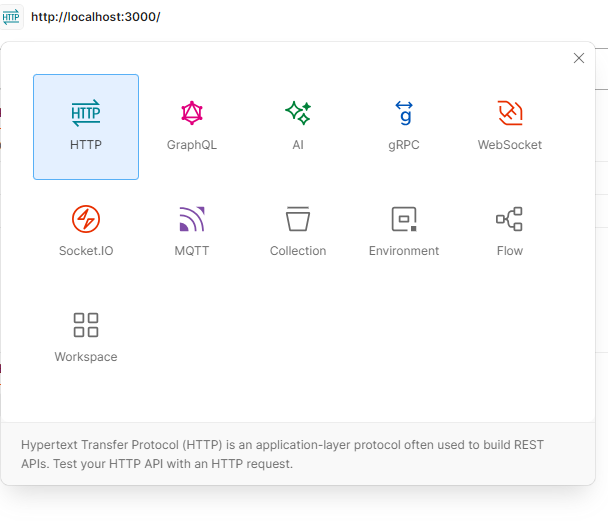
I used Postman to monitor the URL

Install Postman and follow the steps below to start the monitoring:

1. Click on New

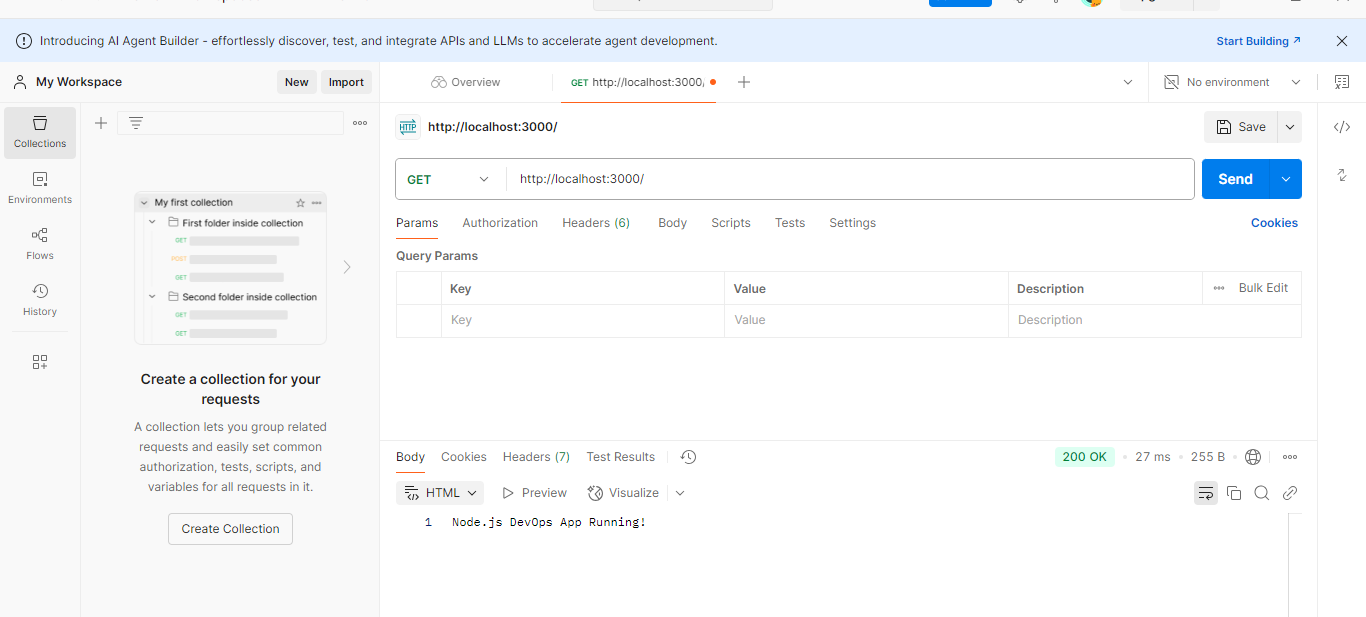


2) Now, we get a small window like this, select HTTP



3) Give the URL of the application that you wish to monitor, and click on send:

http://localhost:3000/



You can now have complete control on your application.

Project by

**Shivani Siripurapu**