

**Lab Manual- AKS Networking with Frontend and Backend (Lab1)**

**Prepared for**:

**Date:** 18th Dec 2023

**Prepared by:**

Document Name: Lab Manual **Document Number** AZLabn916

**Contributor:**

Contents

[1. Objective 2](#_Toc159225665)

[2. Dockerize Frontend and Backend 3](#_Toc159225666)

[1. Frontend Application 3](#_Toc159225667)

[2. Backend Application 4](#_Toc159225668)

[3. Create package.json for Backend 4](#_Toc159225669)

[4. Dockerize Frontend and Backend 4](#_Toc159225670)

[3. Build Docker images for both frontend and backend: 5](#_Toc159225671)

[4. Kubernetes YAML Configurations 9](#_Toc159225672)

# Objective

Here's a step-by-step guide to creating a sample frontend with a form that sends data to a backend database, and deploying both the frontend and backend applications in Kubernetes using ClusterIP for the backend and LoadBalancer for the frontend, along with necessary YAML configurations and a **package.json** file for the backend:

# Dockerize Frontend and Backend

## Frontend Application

Create an HTML form (index.html) for the frontend:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Sample Frontend</title>

</head>

<body>

    <h1>Submit Data</h1>

    <form id="data-form">

        <label for="name">Name:</label>

        <input type="text" id="name" name="name"><br><br>

        <label for="email">Email:</label>

        <input type="email" id="email" name="email"><br><br>

        <button type="submit">Submit</button>

    </form>

    <script>

        document.getElementById("data-form").addEventListener("submit", function(event) {

            event.preventDefault();

            const formData = new FormData(event.target);

            fetch("http://backend-service:3000/submit-data", {

                method: "POST",

                body: formData

            })

            .then(response => response.json())

            .then(data => {

                console.log("Data submitted:", data);

                alert("Data submitted successfully!");

            })

            .catch(error => {

                console.error("Error submitting data:", error);

                alert("Failed to submit data!");

            });

        });

    </script>

</body>

</html>

## Backend Application

Create a backend server using Node.js and Express to handle form submissions and store data in the database. Save this as **server.js**

const express = require("express");

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

const app = express();

const port = 3000;

app.use(bodyParser.urlencoded({ extended: true }));

app.post("/submit-data", (req, res) => {

    // Process form data and store it in the database

    const { name, email } = req.body;

    console.log("Received data:", name, email);

    // Here you would typically store the data in your database

    res.json({ message: "Data received and stored successfully!" });

});

app.listen(port, () => {

    console.log(`Backend server listening at http://localhost:${port}`);

});

## Create package.json for Backend

Create a **package.json** file for the backend:

{

    "name": "backend",

    "version": "1.0.0",

    "description": "Backend server for sample application",

    "main": "server.js",

    "dependencies": {

      "express": "^4.17.1"

    }

  }

## Dockerize Frontend and Backend

Create Dockerfiles for both the frontend and backend applications:

**Frontend Dockerfile** (**Dockerfile-frontend**):

FROM nginx:alpine

COPY index.html /usr/share/nginx/html/index.html

EXPOSE 80

**Backend Dockerfile** (**Dockerfile-backend**):

# docker build -t frontend-image -f Dockerfile-frontend .

# docker build -t backend-image -f Dockerfile-backend .

FROM node:alpine

WORKDIR /app

COPY package\*.json ./

RUN npm install

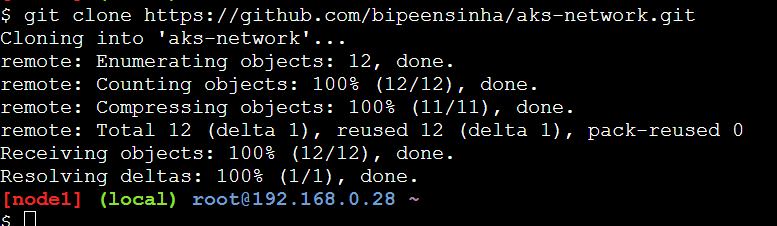
COPY . .

EXPOSE 3000

CMD ["node", "server.js"]

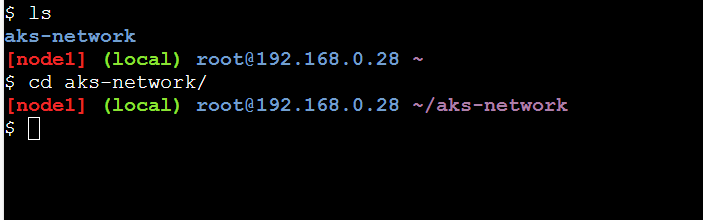
# Build Docker images for both frontend and backend:

git clone https://github.com/bipeensinha/aks-network.git

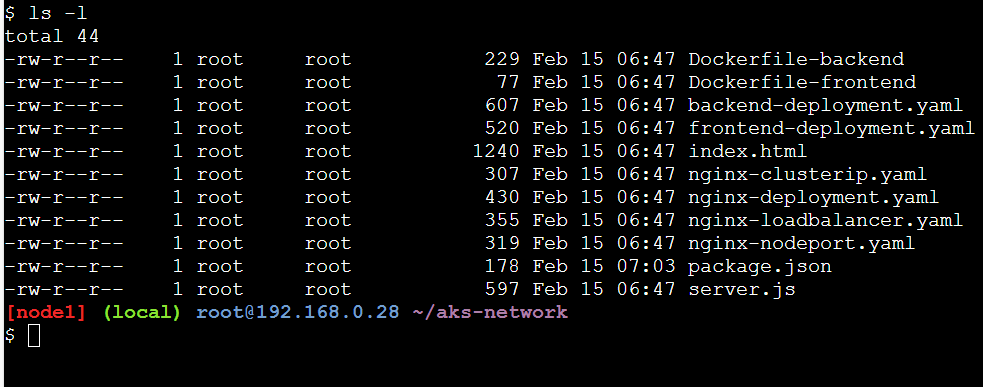


ls

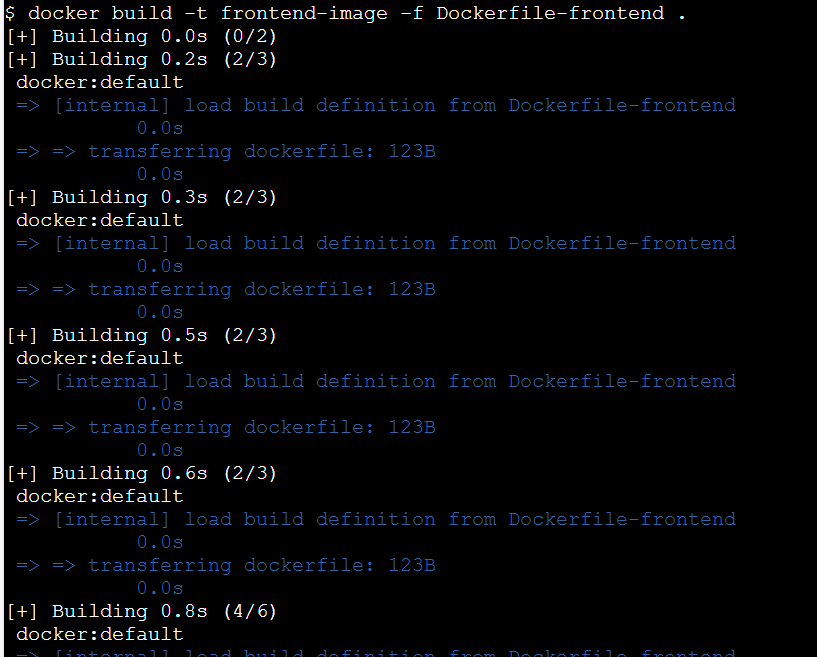
cd aks-network/



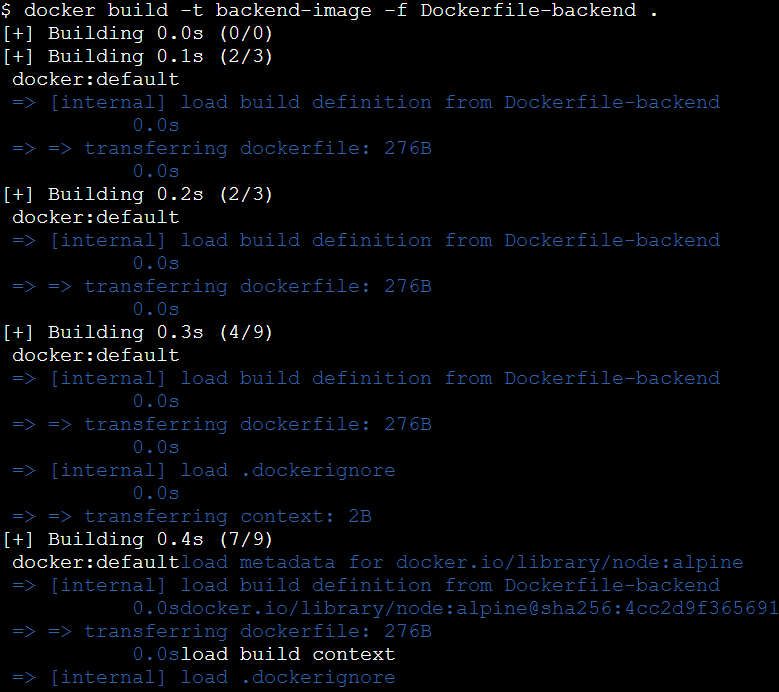
Ls -l



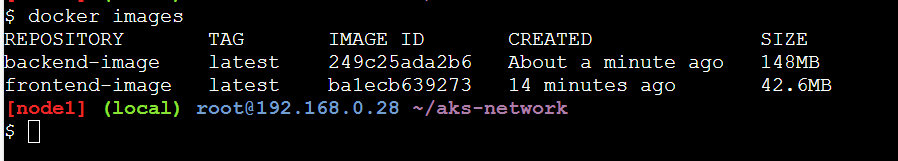
docker build -t frontend-image -f Dockerfile-frontend .



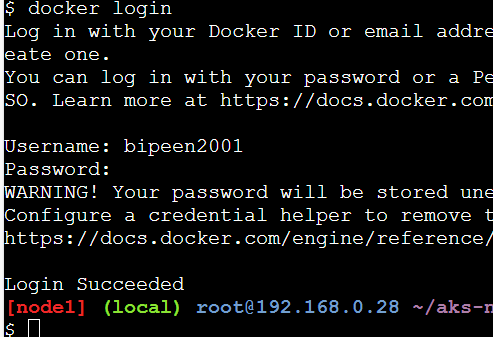
docker build -t backend-image -f Dockerfile-backend .



docker images

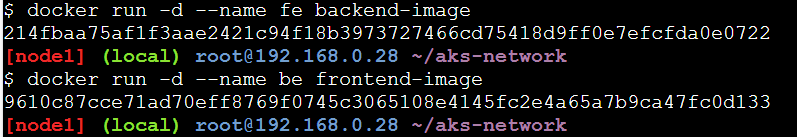


docker login



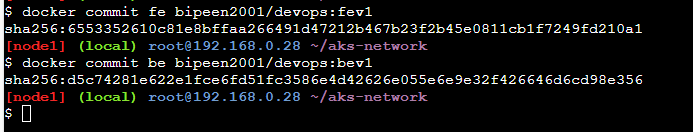
docker run -d --name fe frontend-image

docker run -d --name be backend-image

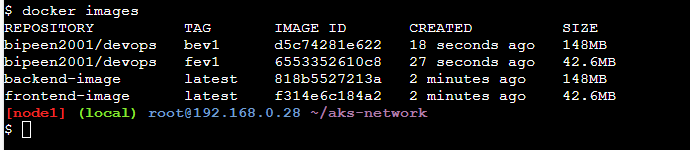


docker commit fe bipeen2001/devops:fev1

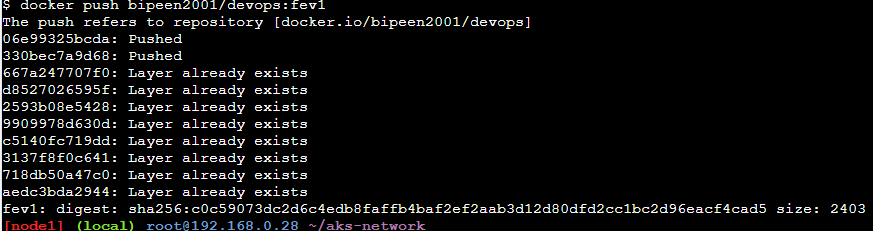
docker commit be bipeen2001/devops:bev1



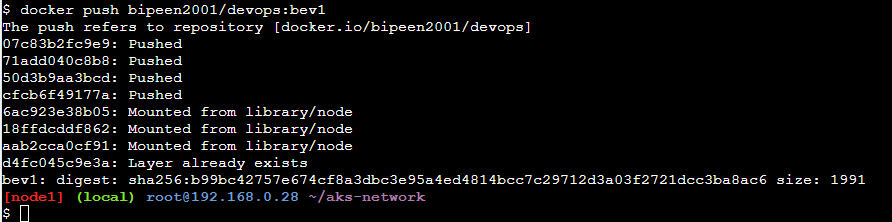
docker images

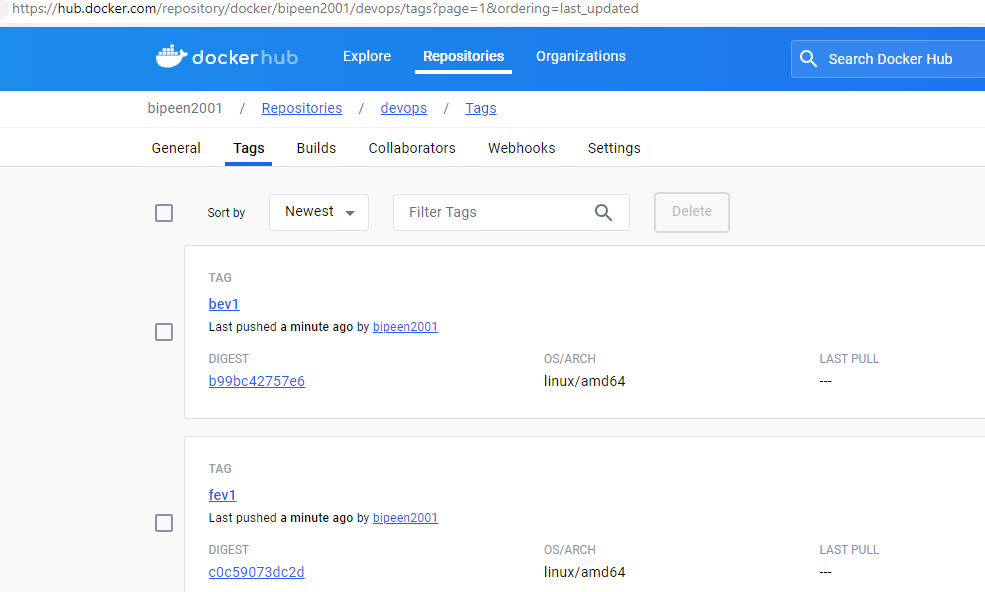


docker push bipeen2001/devops:fev1



docker push bipeen2001/devops:bev1





# Kubernetes YAML Configurations

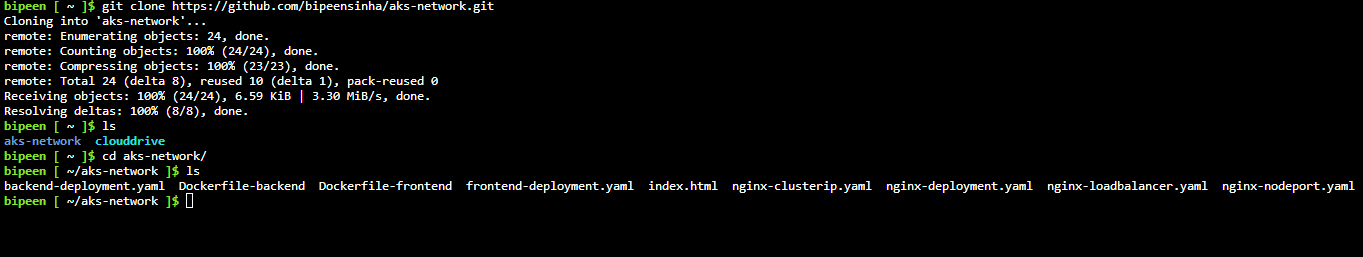
Create Kubernetes YAML configurations for frontend, backend, and ClusterIP services:

**frontend-deployment.yaml**:

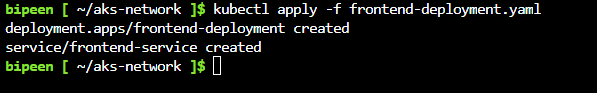
git clone https://github.com/bipeensinha/aks-network.git

ls

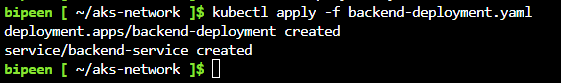
cd aks-network/



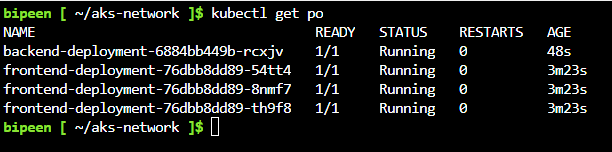
kubectl apply -f frontend-deployment.yaml



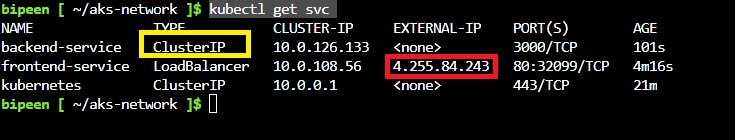
kubectl apply -f backend-deployment.yaml



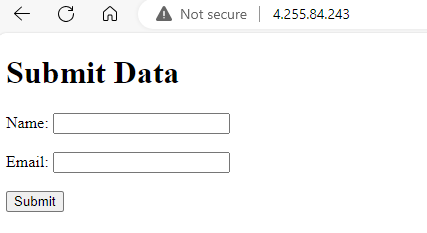
kubectl get po

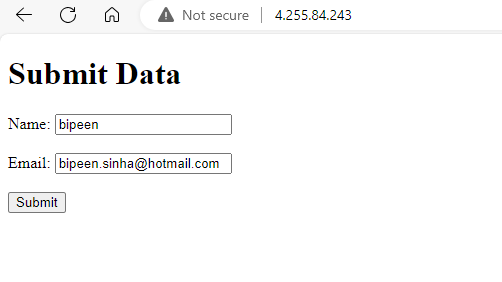


kubectl get svc



<http://4.255.84.243/>





kubectl run -it --rm --restart=Never frontend-test --image=bipeen2001/devops:fev1 sh

kubectl run -it --rm --restart=Never frontend-test --image=bipeen2001/devops:bev1 sh