TECHDOME DEVOPS PROJECT

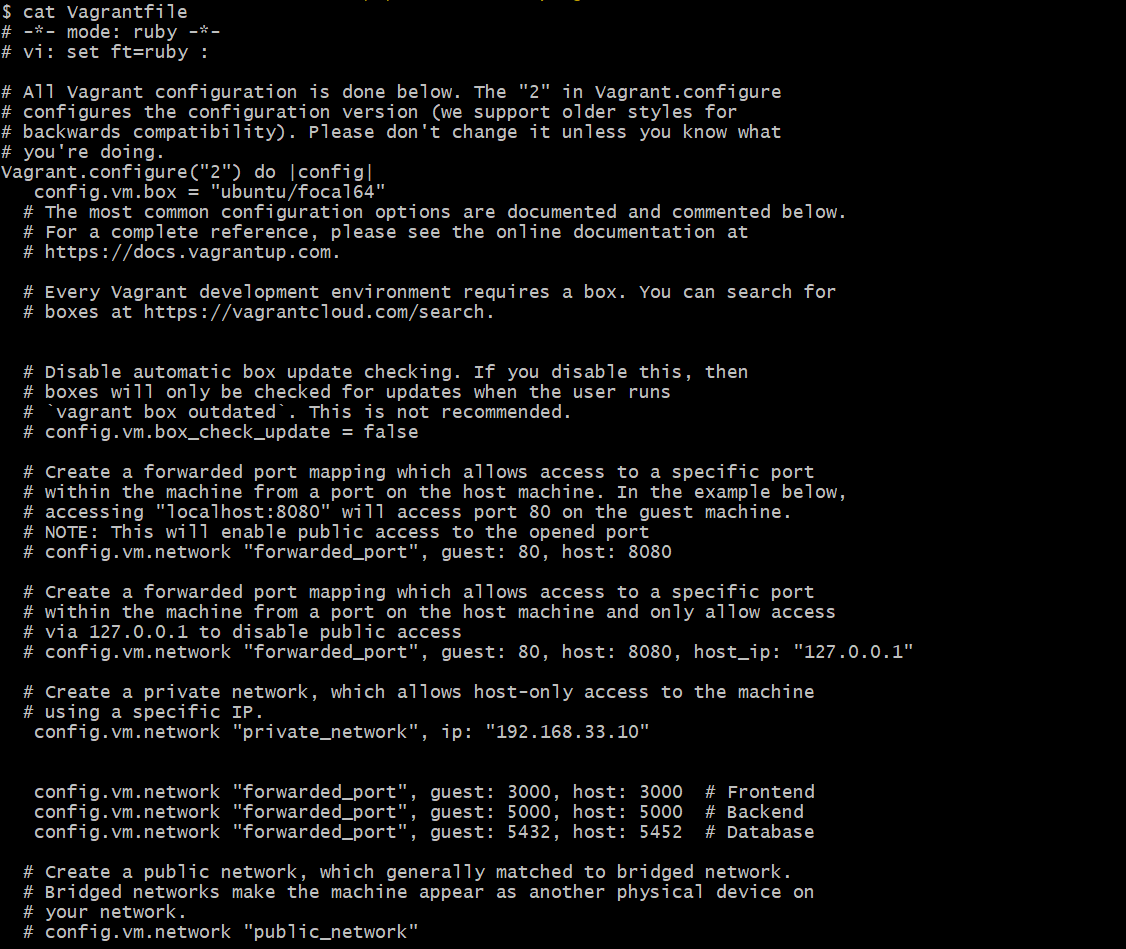
This project involves setting up a multi container architecture to run an application.

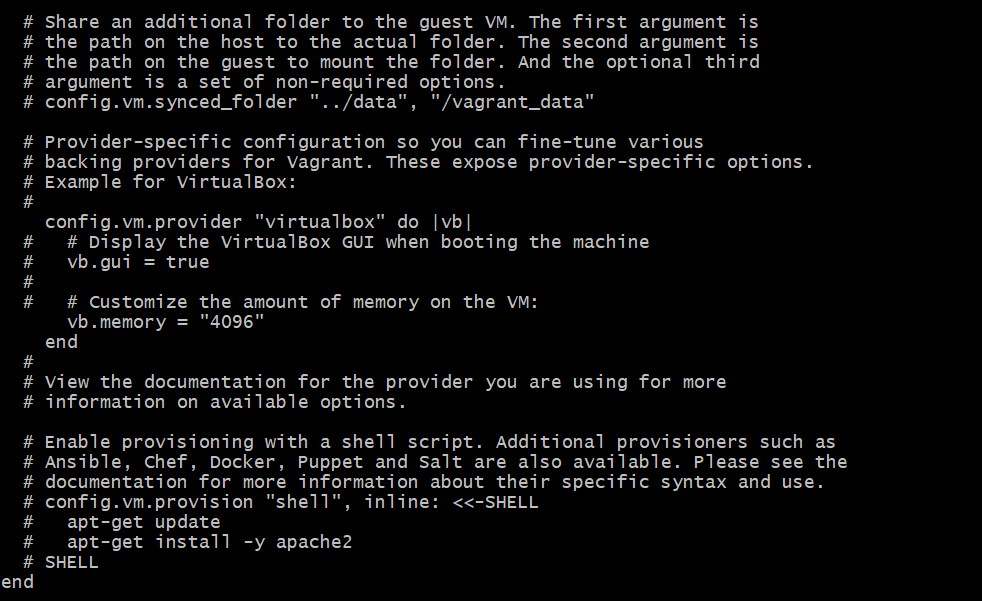
Tools Used: Docker, Docker compose, Kubernetes, Minikube

Step 1

Create a Virtual Machine which will be used to host the application

Vagrant file is as below



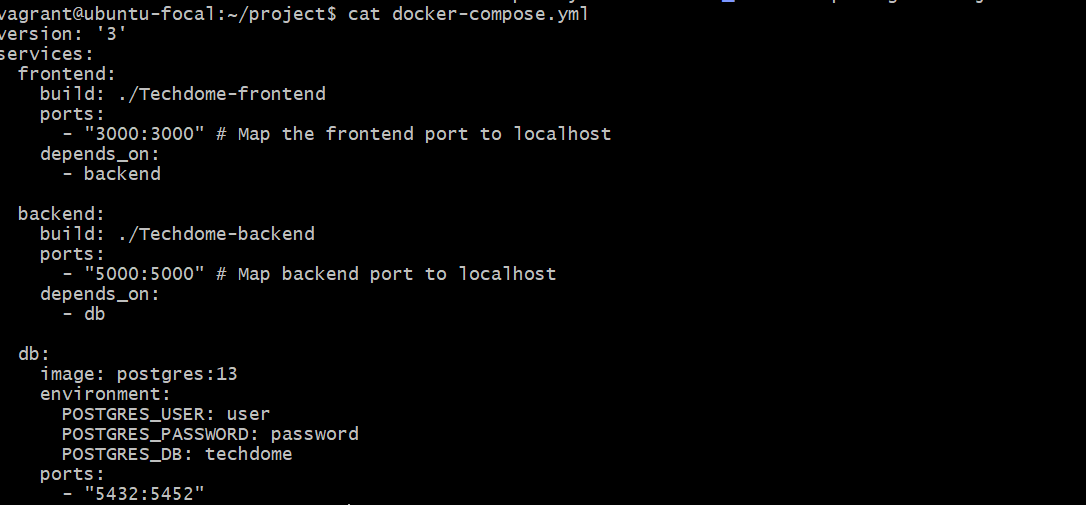


Used this Vagrant file to spin up the server which is an ubuntu server.

Step 2 : Created a directory named project and cloned both front-end and back-end repositories in it.



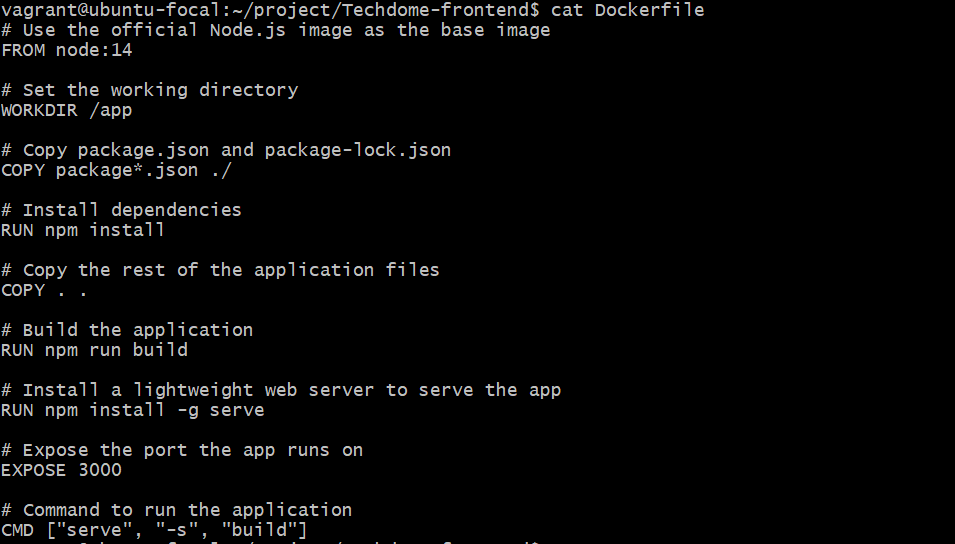
Step 3 : Created a docker compose file for running multi-container app



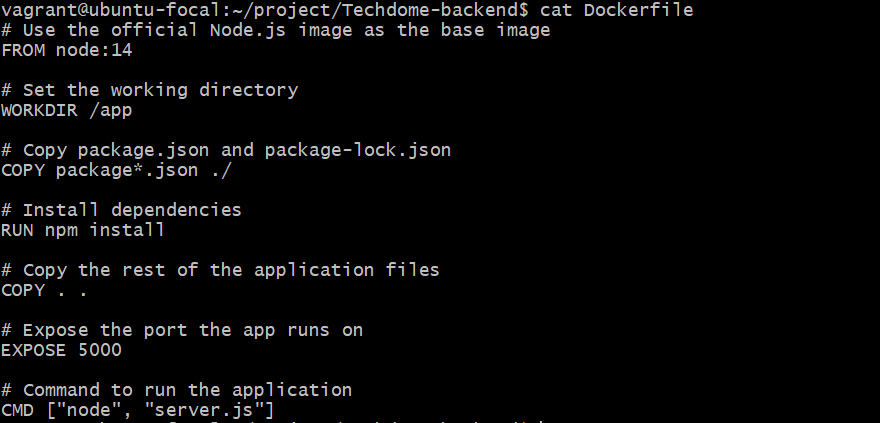
This is the docker compose file used for running the containers

Step 4 : Created docker file for each back-end and front-end projects

Docker file for front-end project

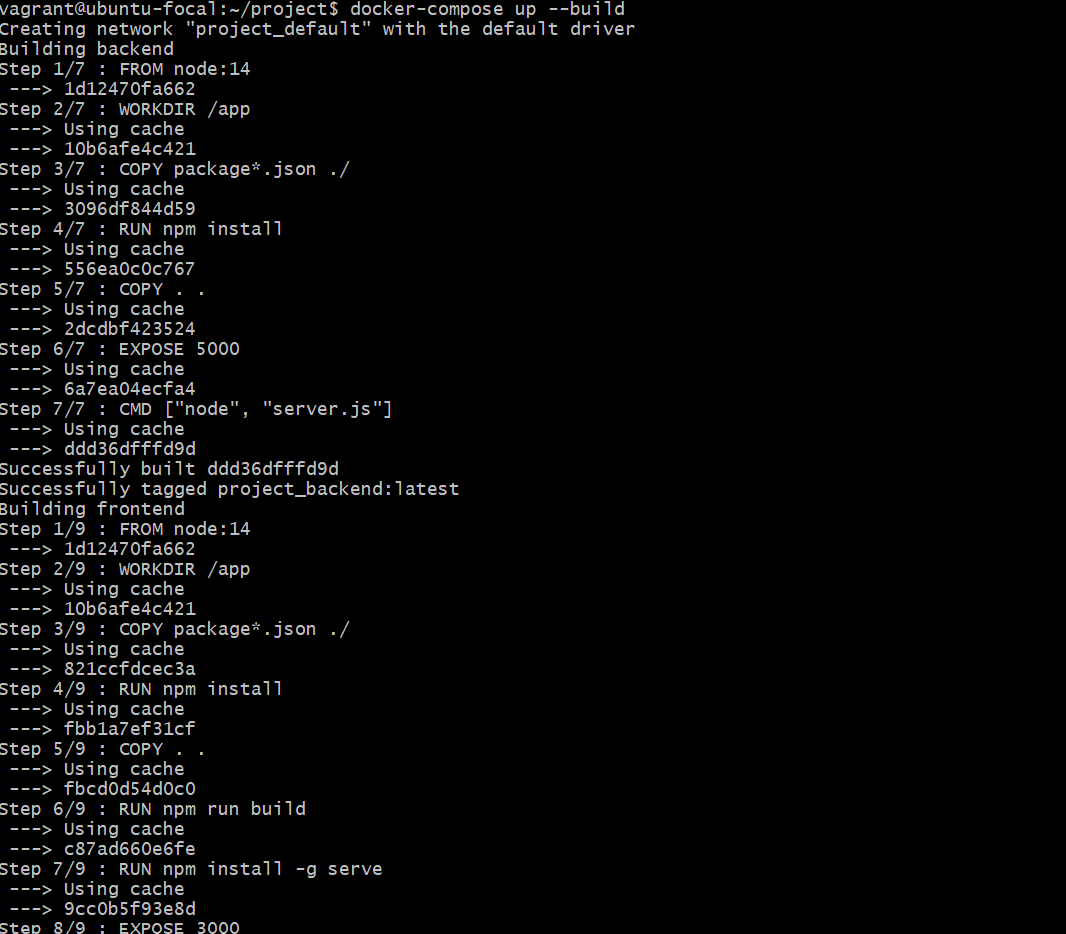


Dockerfile for back-end project

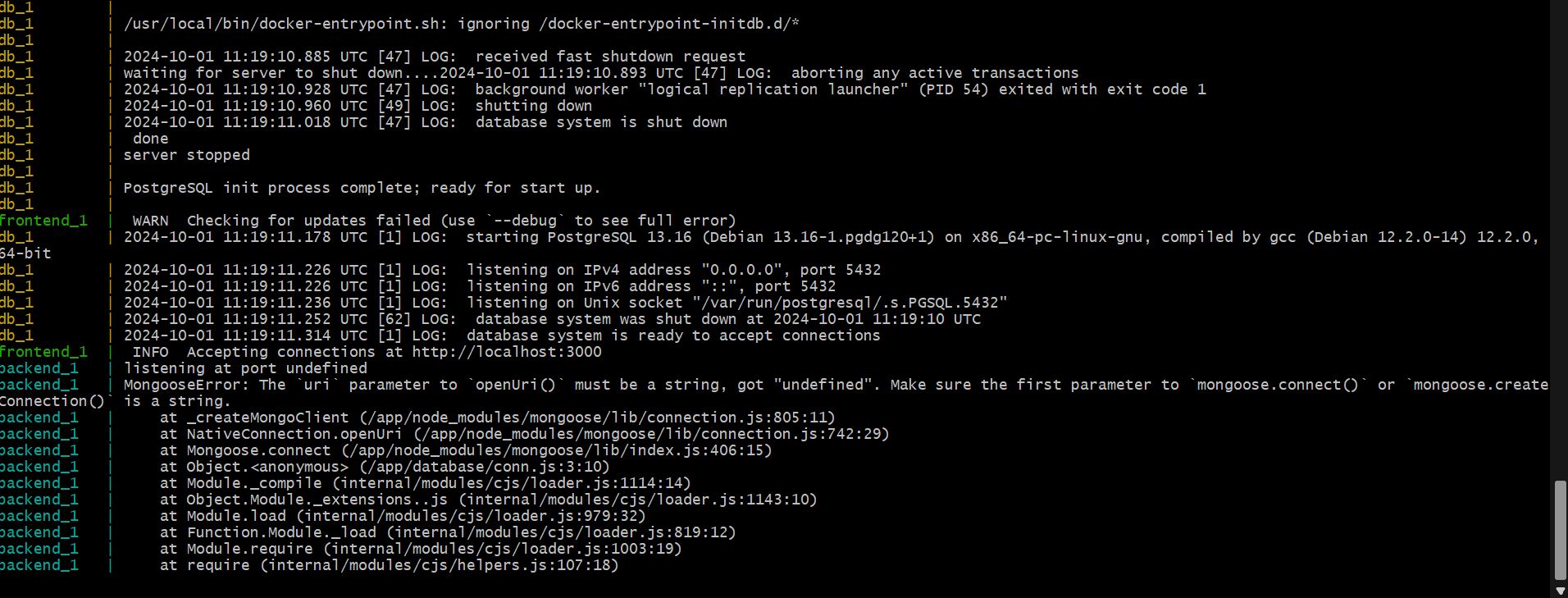


Step 5: Built the Docker-containers

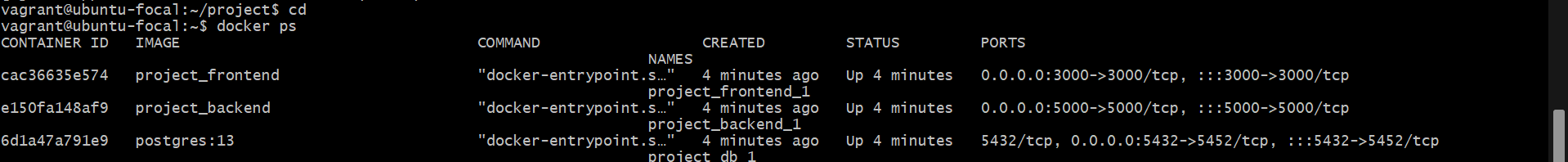
Executed docker-compose up --build command and the result were as below







After the execution three containers got created and running.



Back-end, front-end and database is successfully running.

Step 6: Installing minikube

Commands

curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

sudo install minikube-linux-amd64 /usr/local/bin/minikube

curl -LO https://storage.googleapis.com/kubernetes-release/release/`curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt`/bin/linux/amd64/kubectl (add https)

chmod +x kubectl

sudo mv kubectl /usr/local/bin/

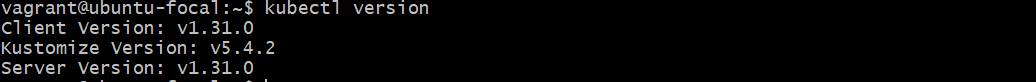
minikube start --driver=docker

kubectl get nodes

These commands are used to install minikube and it successfully running

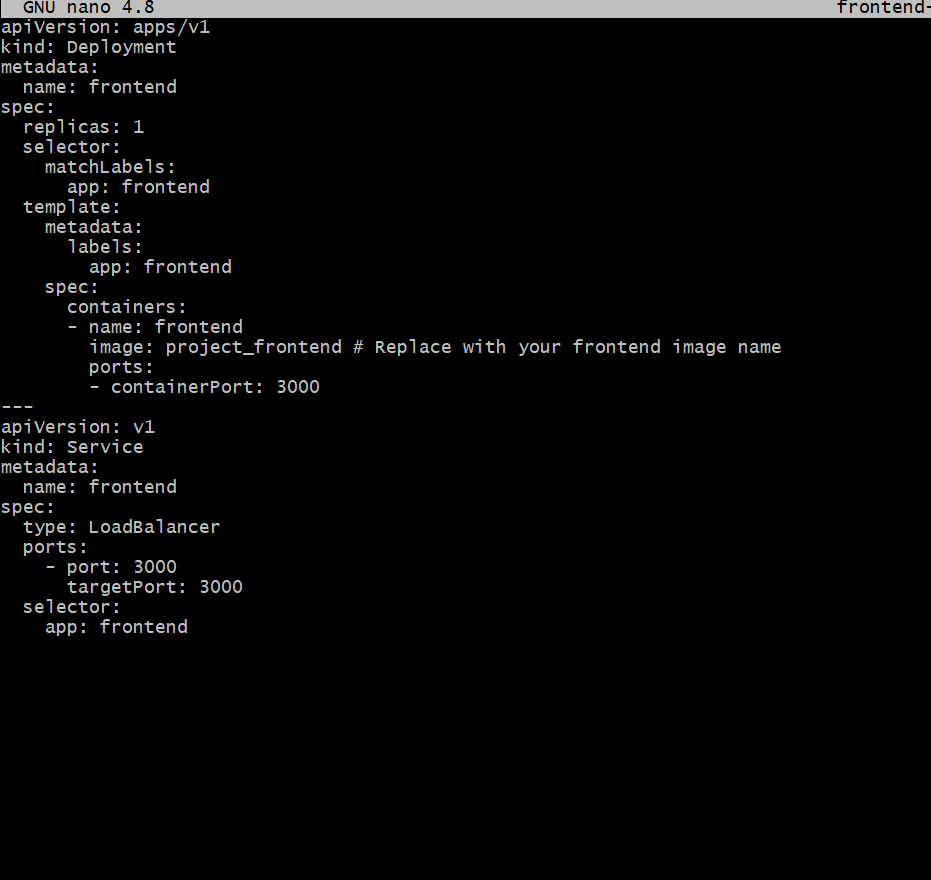


Step 7: Kubectl install

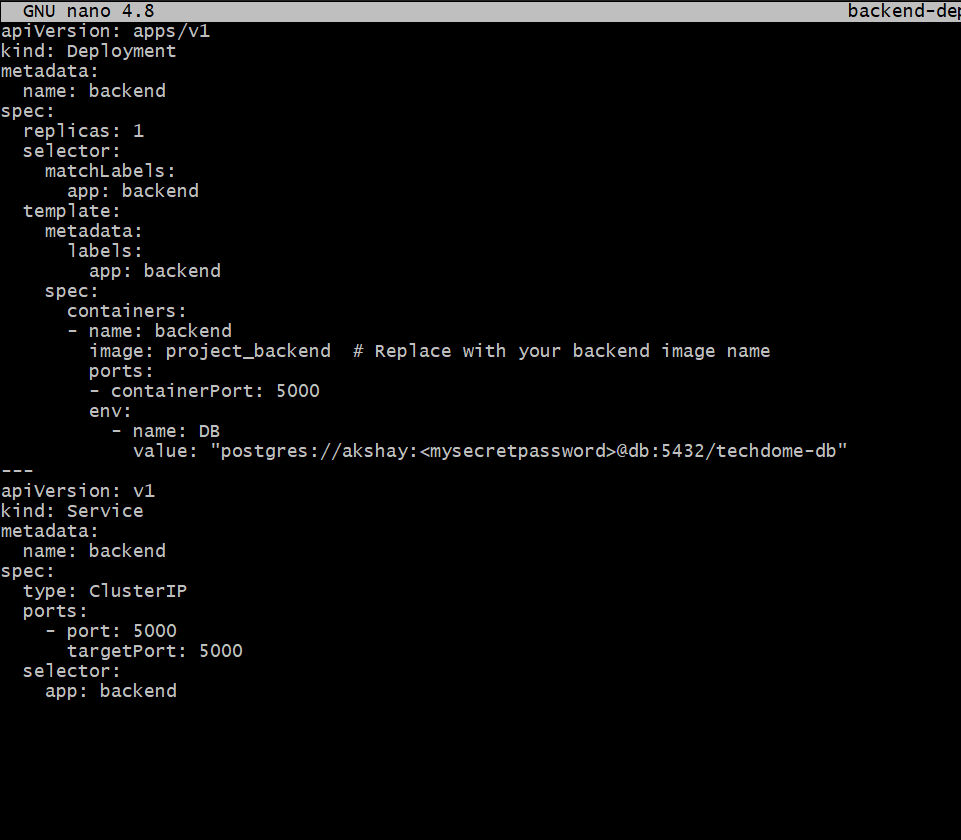


Step 8: Created a directory called K8s in he project folder and stored yaml file for front-end , back-end and database deployments

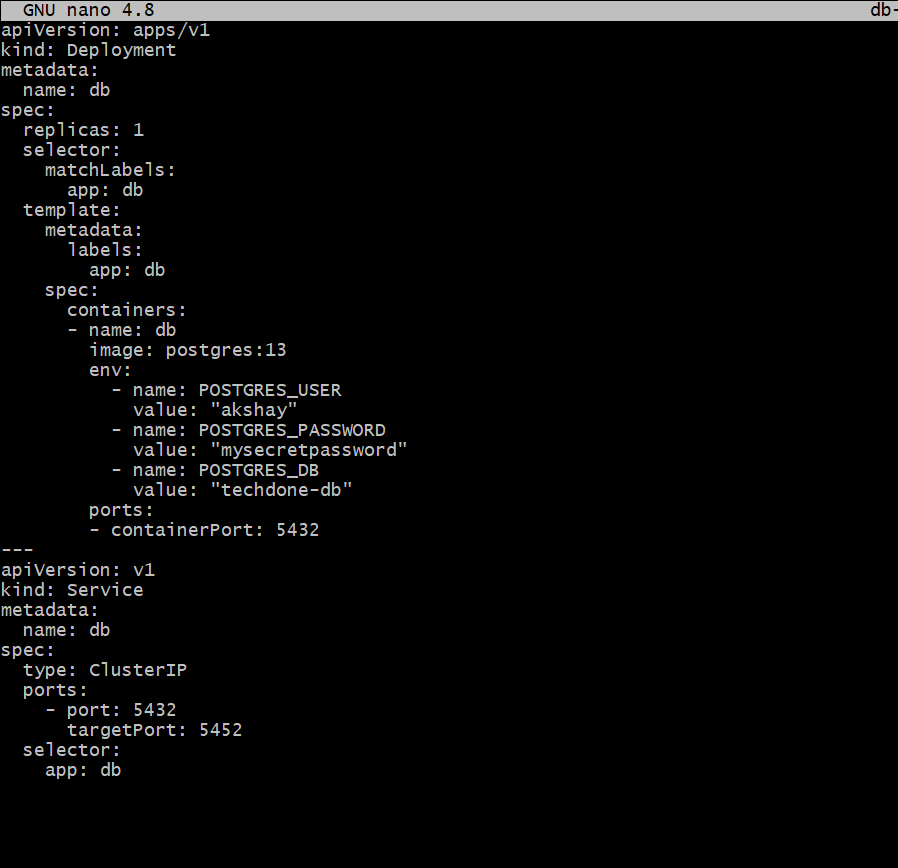
Yaml file for front-end deployment



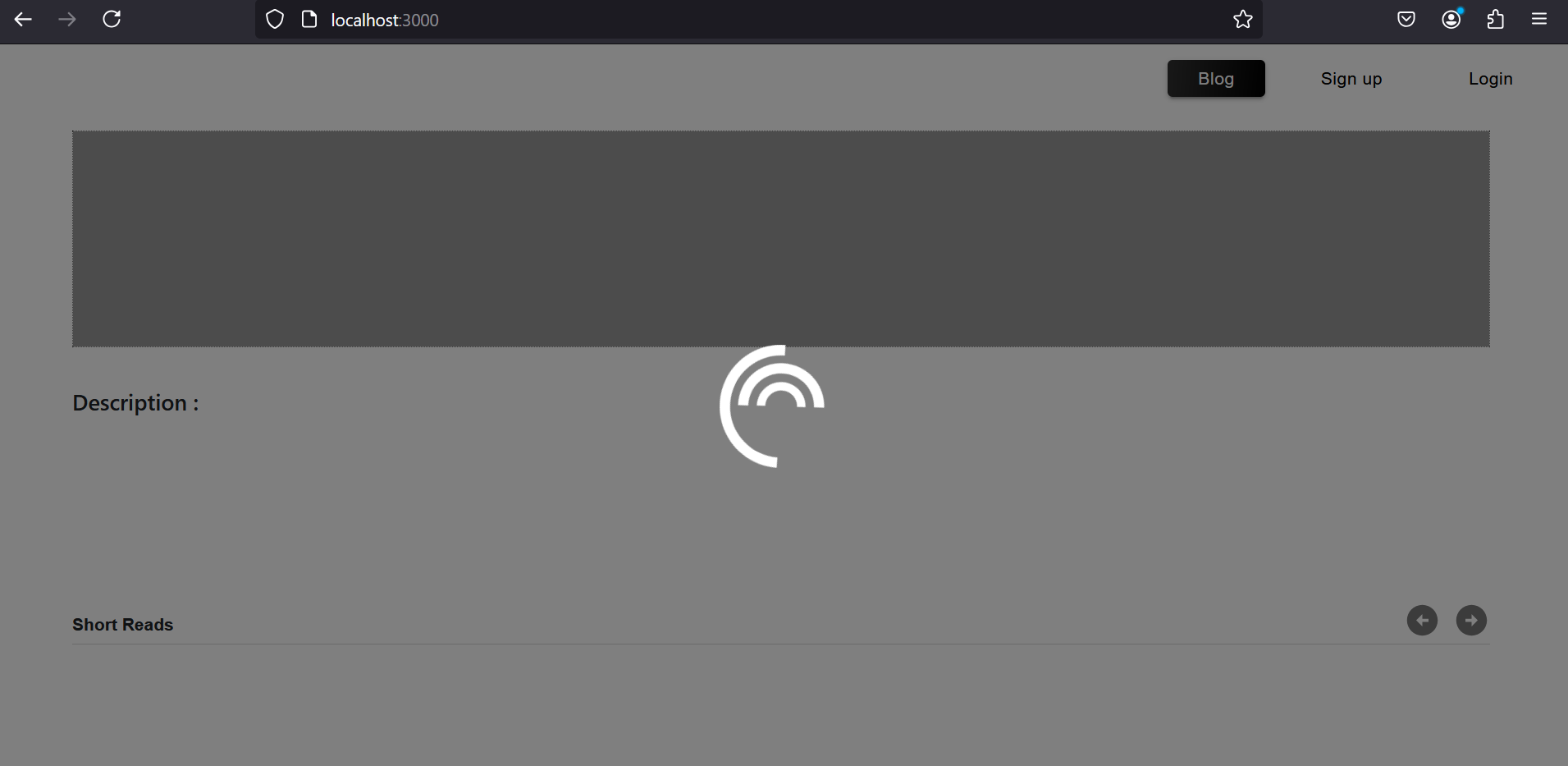
Yaml file for back-end deployment



Yaml file for database deployment



Step 9: Deployed the application into Kubernetes

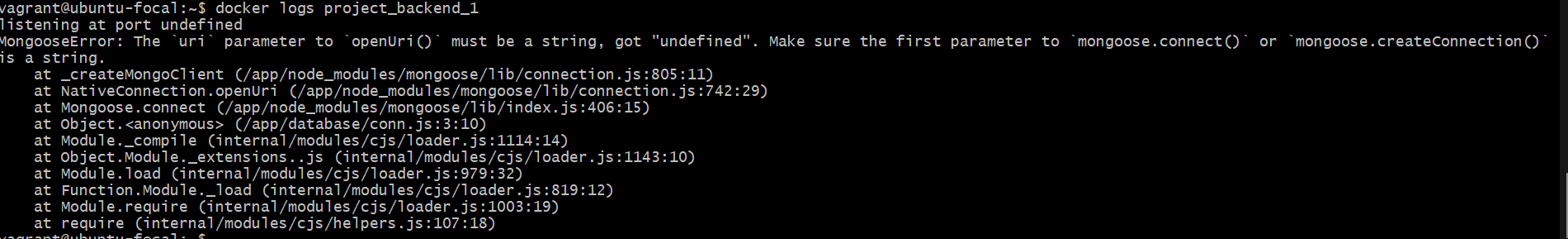


Able to access the application through <https://localhost:3000>

Note: Instead of using MongoDB as the database, I have used Postgres for this process.

MongoDB is a non-relational database, and postgres is a relational database.

Since I used postgres as the database, the application is buffering a lot.



As seem here, the error logs coming for the backend project is because I have used Postgres instead of MongoDB.

Thanks & Regards

Akshay Sahadevan