



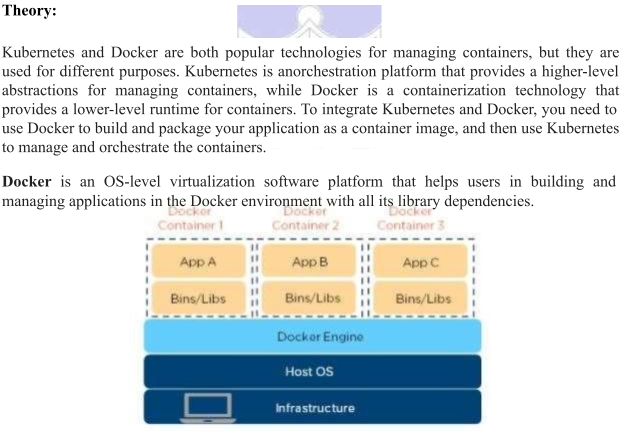
|  |  |  |
| --- | --- | --- |
| **Batch:B1** | **Roll No.:16010421119** | **Experiment No.:**05 |

**Aim:** To create Docker image using Dockerfile for NodeJS application, run and Push it on DockerHub.



**Resources needed:** Docker Hub Account, Kubernetes





**Docker image** is a read-only, inert template that comes with instructions. In Docker, everything basically revolves around images. An image consists of a collection of files (or layers) that pack together all the necessities—such as dependencies, source code, and libraries—needed to set up a completely functional container environment. Images are stored on a Docker registry, such as the [Docker Hub,](https://hub.docker.com/) or on a local registry.

[**Docker Container**](https://www.simplilearn.com/tutorials/docker-tutorial/what-is-docker-container) is a lightweight software package that includes all the dependencies (frameworks, libraries, etc.) required to execute an application. Running image is the container.

**Dockerfile** is a simple text file that consists of instructions to build Docker images. Dockerfile is used to create Docker images. Dockerfile is nothing but a blueprint for the image. All the Docker images have Dockerfile. Dockerfile consists of specific [commands](https://www.simplilearn.com/tutorials/docker-tutorial/docker-commands) that guide you on how to build a specific Docker image.

The specific commands you can use in a dockerfile are:   
FROM, PULL, RUN, and CMD   
 FROM - Creates a layer from the ubuntu:18.04   
 PULL - Adds files from your Docker repository

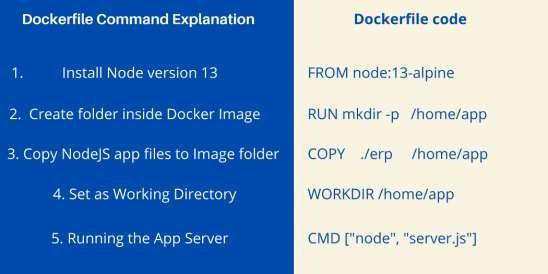


RUN - Builds your container   
CMD - Specifies what command to run within the container

**Create a Docker image using two methods:**   
 1. Develop all the code locally(on your local computer). And package all the code as Docker image using Dockerfile.

2. Run Docker container. Then place all your code inside the Docker container(Enter into docker container using *docker exec* command). And build an image using the running Docker container.

The below diagram is a **Dockerfile example** for the NodeJS application:





**Procedure:**

1. Create Account on Docker Hub.

2. Install Docker and explore Docker commands.

3. Write a NodeJS Application.

4. Write a Dockerfile for NodeJS Application.

5. Create/build image and check for its availability in local computer. 6. Run the created image.

7. Push the image on Docker Hub.

8. Use Kubernetes to deploy the Docker image to a cluster a. Create a deployment   
b. Create a service   
9. Modify the code, create/build and run the updated image

**Results: (Document with screenshots)**

1. NodeJS Application code   
2. Dockerfile content.

3. Commands with output



a. Create image

b. Check for its availability in local computer.



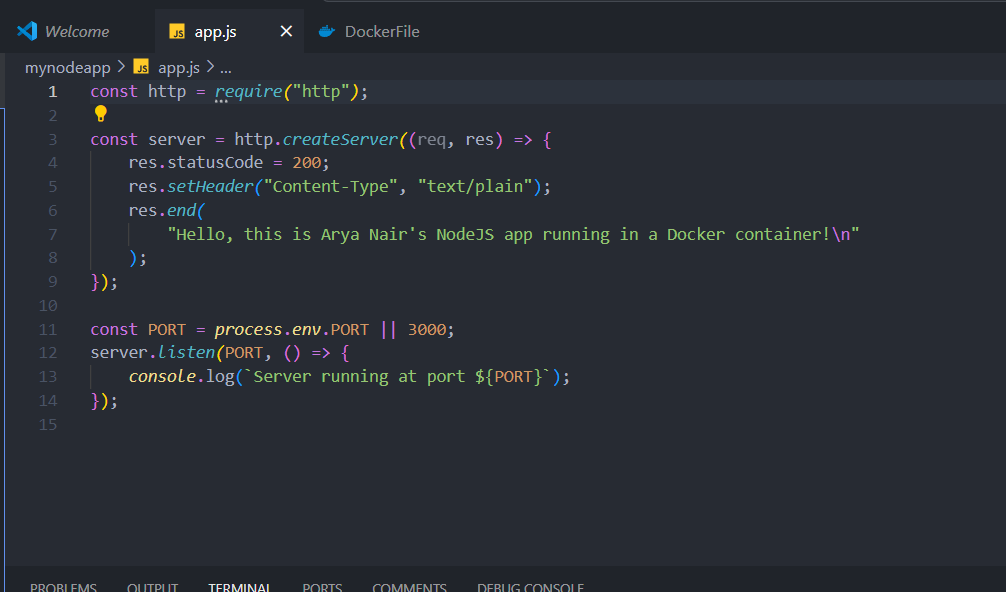
c. Run the created image.

d. Push the image on Docker Hub.

e. Deployment code for Kubernetes

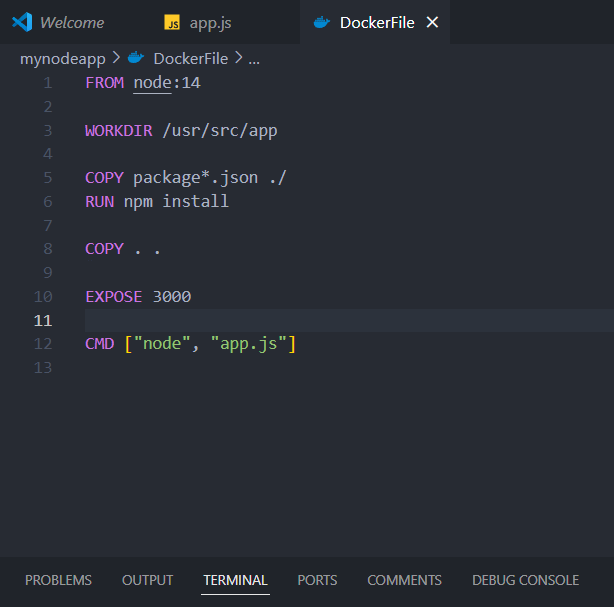
f. Running application

4. Push Updated code and see the changes in running application

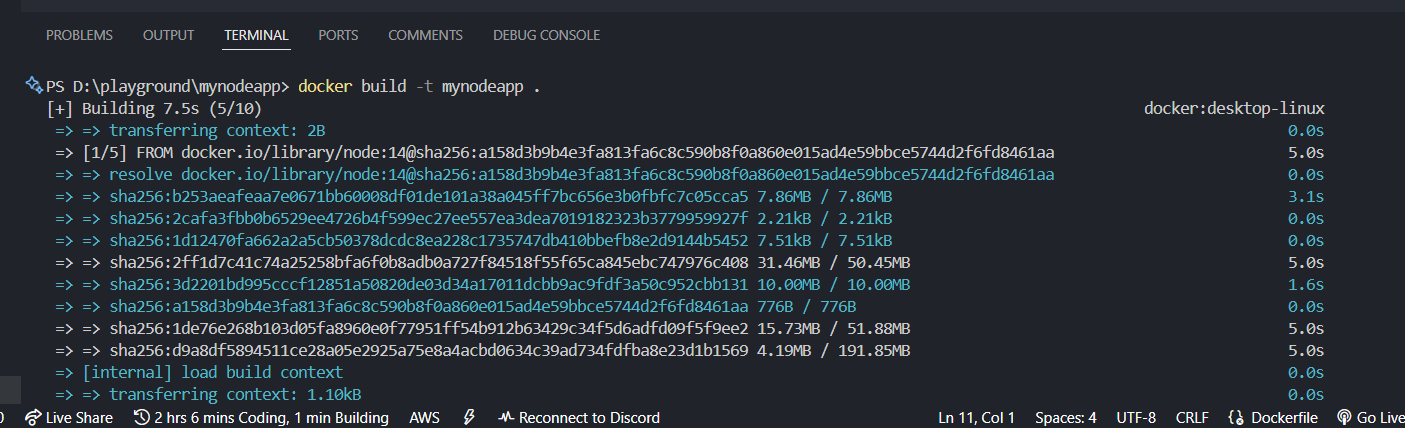


My node application



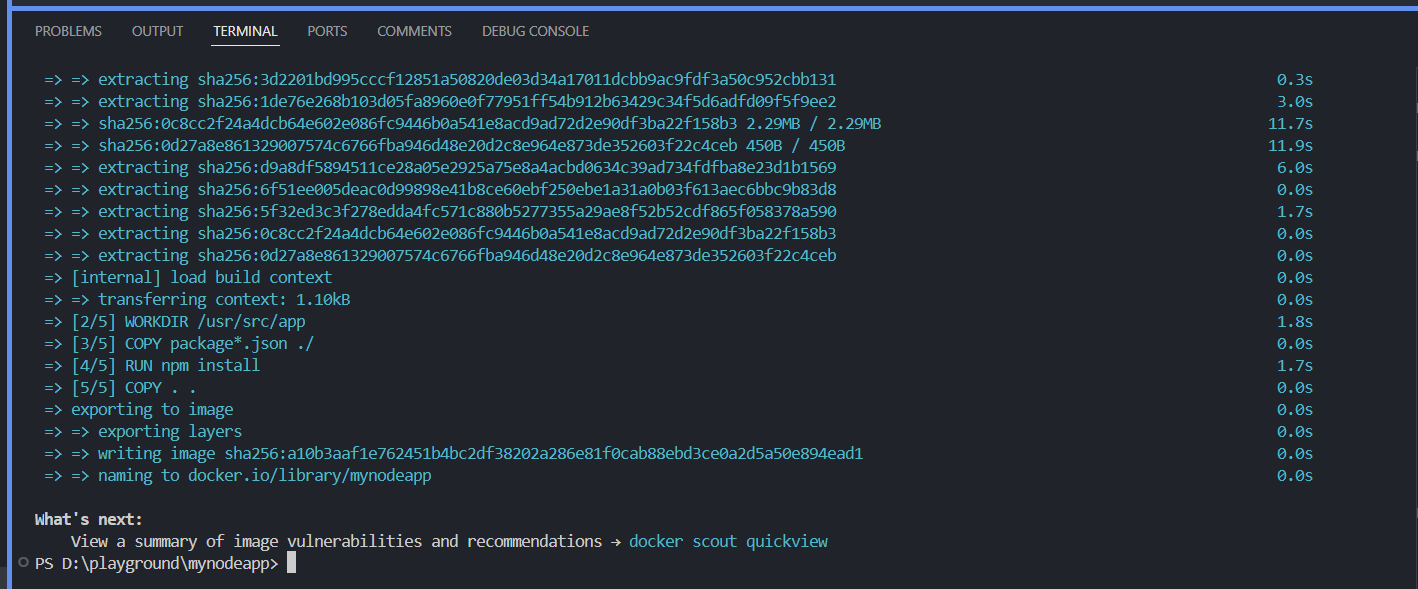


docker file to containerise the node app

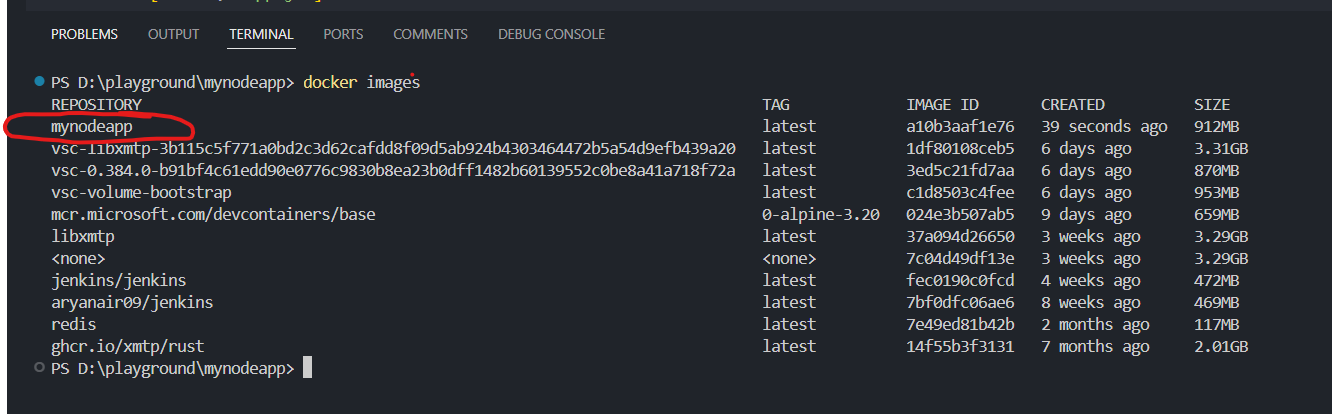


build docker image

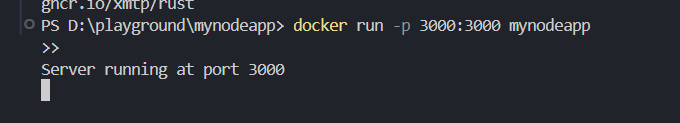


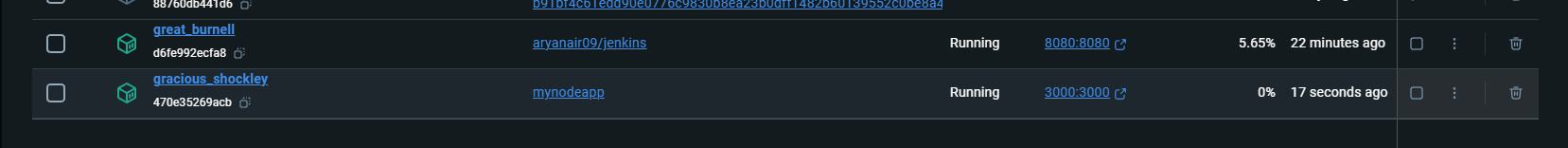


see docker imags

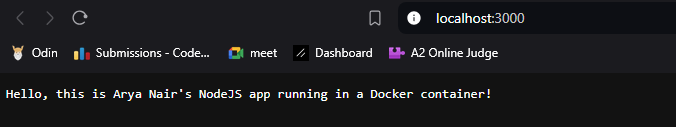


run docker container



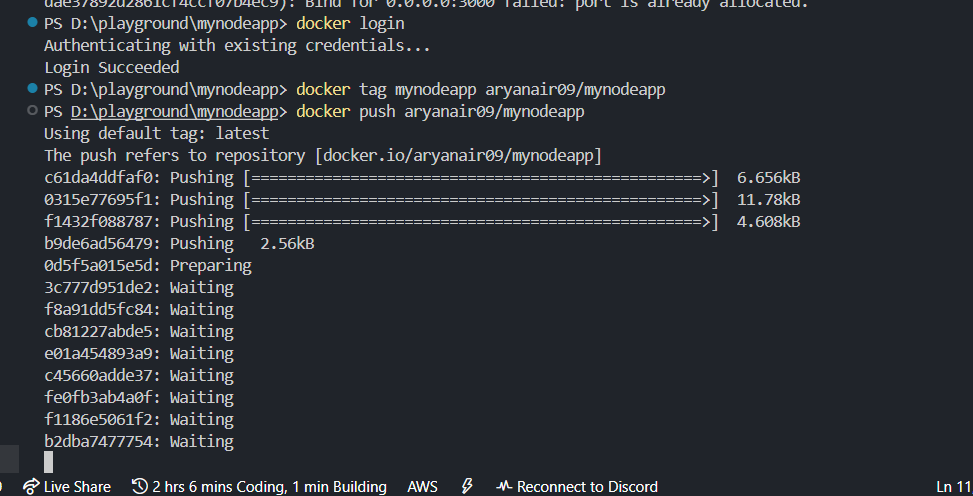


host it locally to check

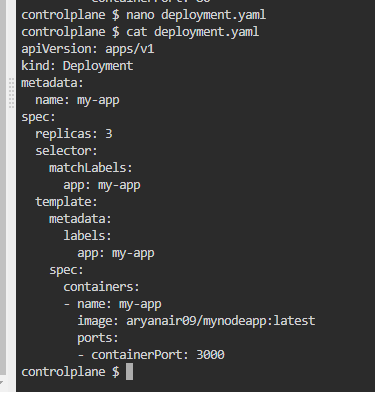


push docker image on docker hub

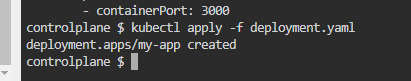




use kllercoda and create deployment.yaml

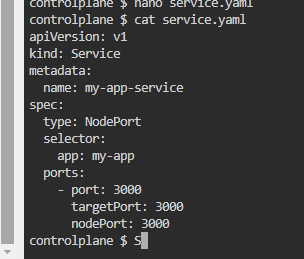


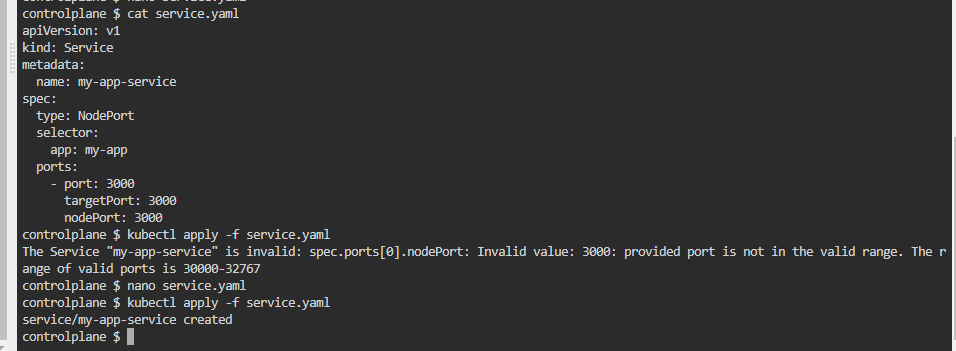
apply the deployment.yaml



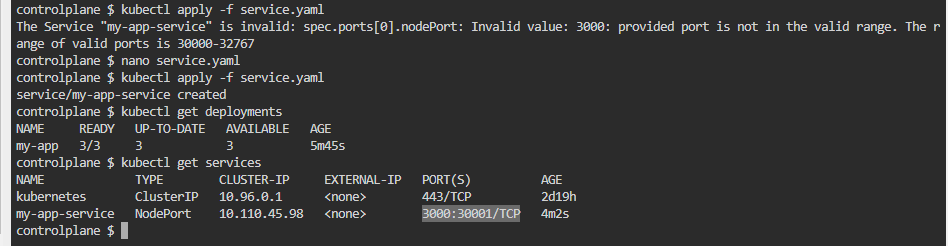
create service.yaml and apply





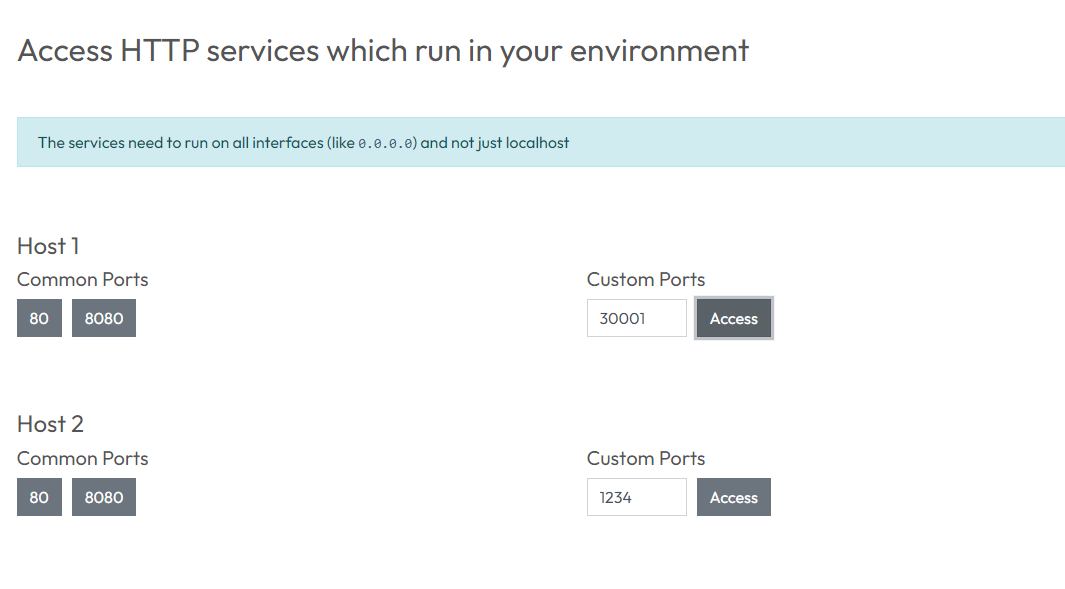


view deployments and services

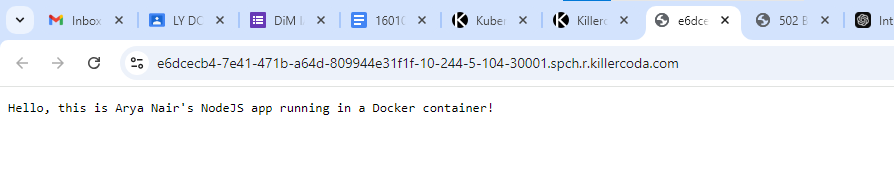


access http port from killercoda

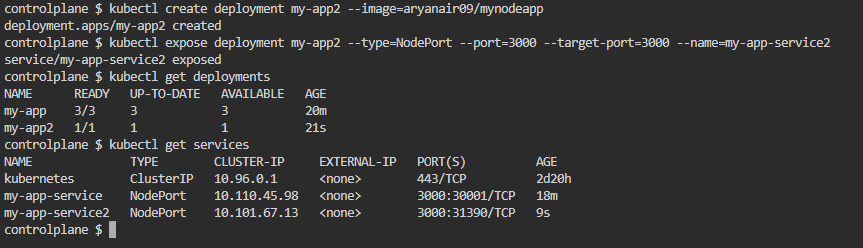




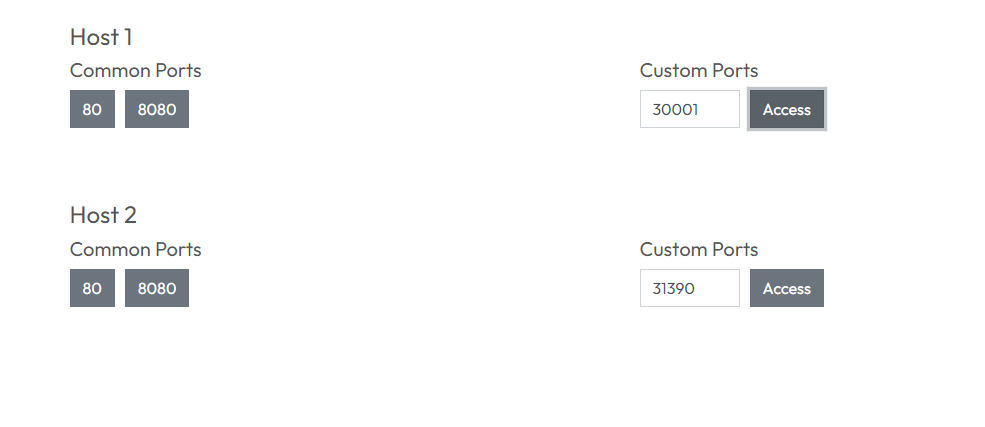
view the mapped port from killercoda



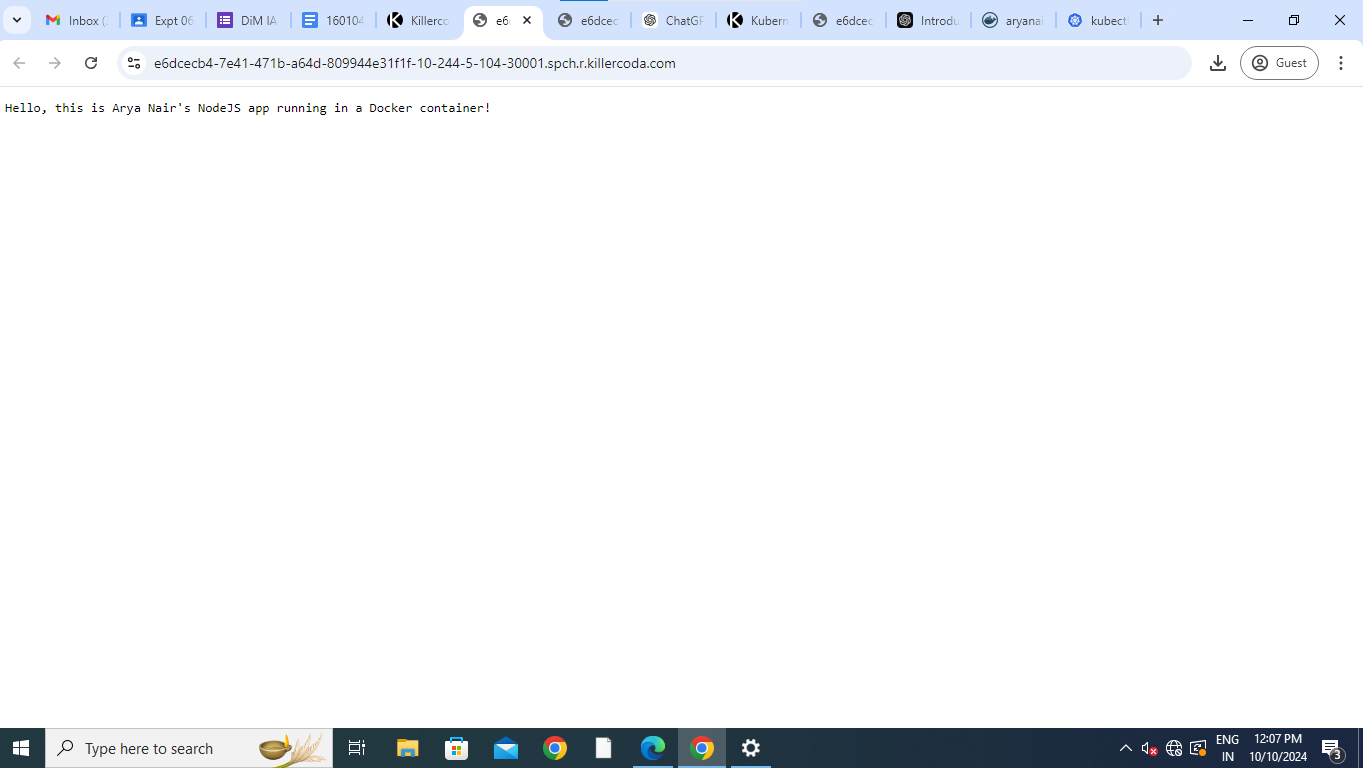
create deployment using cli command, and also expose service



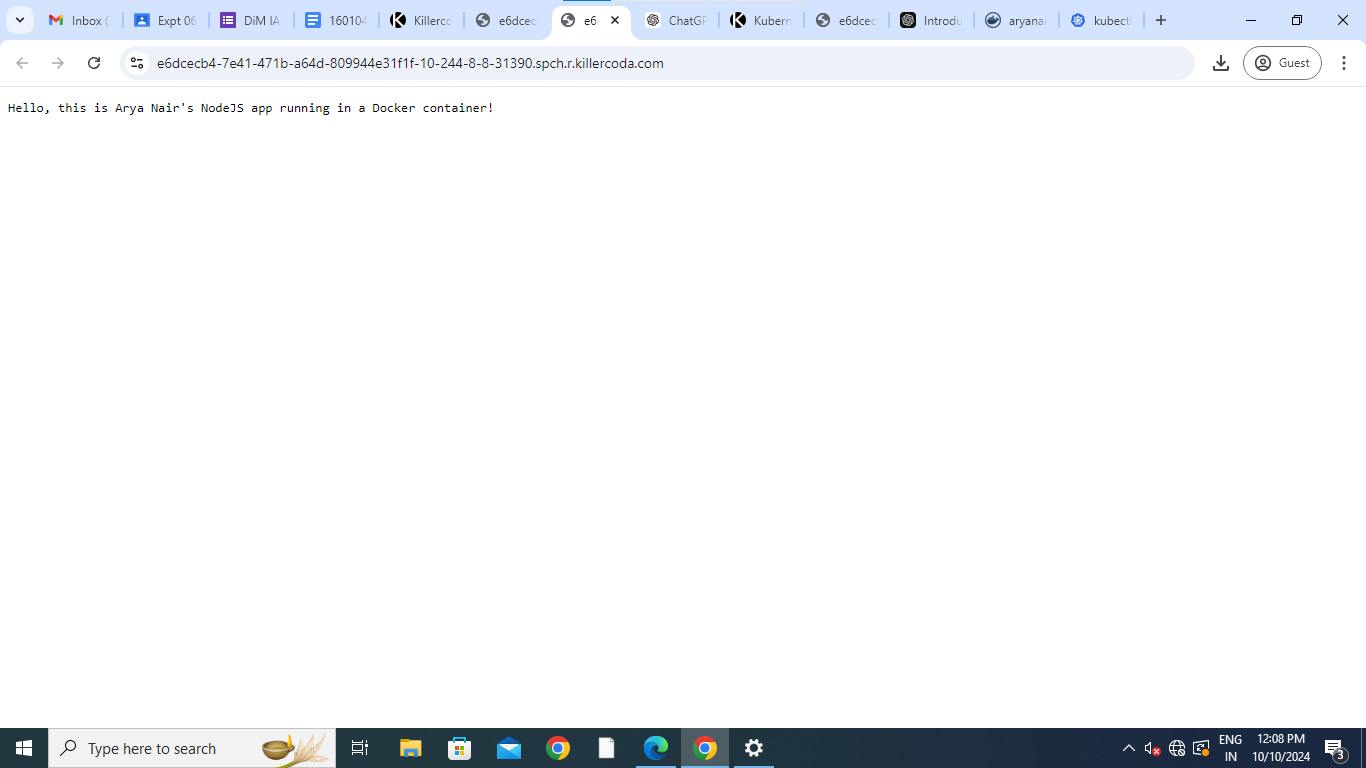




we can see the final result of the same







after every change run docker push and then run kubectl rollout restart **Questions:**

1. What is Docker file and Docker compose?

A Dockerfile is a text file that contains instructions for Docker to build a container image. It defines everything needed to create the environment, such as the base image (e.g., Ubuntu, Node.js), dependencies, and application code. By running the docker build command with a Dockerfile, you can create a custom image for your application.

Docker Compose is a tool that simplifies the management of multi-container Docker applications. Using a docker-compose.yml file, you can define multiple services (containers), networks, and volumes, and then manage them with a single command. It's commonly used for defining environments with interconnected services like web servers, databases, and caching layers.

2. Explain two containers other than Docker.

Aside from Docker, there are other containerization platforms that offer similar functionality for creating, deploying, and managing containers. Two popular alternatives are:   
1. Podman:   
Podman is a container engine that allows users to manage containers and pods without requiring a running daemon (unlike Docker, which relies on dockerd). It offers Docker-compatible commands (podman run, podman build), making it easy for users to transition. One of Podman’s key features is that it allows rootless containers, meaning users can run containers without root privileges, improving security.

2. LXC (Linux Containers):   
LXC is a Linux-based containerization method that provides system-level virtualization. Unlike Docker, which focuses on application-level containers, LXC provides lightweight, virtualized environments similar to virtual machines but without the overhead. It offers more control over the container environment, allowing users to run a full Linux system inside a container, making it ideal for more complex, multi-process systems.





**Outcomes:CO4 Explain code deployment and monitoring systems and their tool support.**





**Conclusion: (Conclusion to be based on the Results and outcomes achieved)** made a docker container and have deployment scripts for kubectl





**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of faculty in-charge with date**



**References:**   
**Books/ Journals/ Websites:**

1. https:/[/www.digit](http://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-)a[locean.com/community/tutorials/how-to-install-and-use-docker-on-](http://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-) ubuntu-18-04   
2. https:/[/www.simplilearn.com/tutorials/docke](http://www.simplilearn.com/tutorials/docker-tutorial/install-docker-on-windows)r[-tutorial/install-docker-on-windows](http://www.simplilearn.com/tutorials/docker-tutorial/install-docker-on-windows) 3. https:/[/www.simplilearn.com/tutorials/docke](http://www.simplilearn.com/tutorials/docker-tutorial/how-to-install-docker-on-ubuntu)r[-tutorial/how-to-install-docker-on-ubuntu](http://www.simplilearn.com/tutorials/docker-tutorial/how-to-install-docker-on-ubuntu) 4. [https://www.whitesourcesoftware.com/free-developer-tools/blog/docker-images-vs-](https://www.whitesourcesoftware.com/free-developer-tools/blog/docker-images-vs-docker-containers/) [docker-containers/](https://www.whitesourcesoftware.com/free-developer-tools/blog/docker-images-vs-docker-containers/)   
5. Docker Commands Tutorial: <https://www.edureka.co/blog/docker-commands/>6. Docker Basic Commands | Docker Commands with Examples | Docker   
 Commands Tutorial | Intellipaat:   
 <https://www.youtube.com/watch?v=nXV6qihj5uw>  
7. <https://www.simplilearn.com/tutorials/docker-tutorial/what-is-dockerfile>  
8. [https://medium.com/codex/dockerfile-explained-dockerize-nodejs-application-](https://medium.com/codex/dockerfile-explained-dockerize-nodejs-application-411dadbc3412) [411dadbc3412](https://medium.com/codex/dockerfile-explained-dockerize-nodejs-application-411dadbc3412)   
9. How To Push a Docker Image To Docker Hub Using Jenkins   
 [https://medium.com/codex/how-to-push-a-docker-image-to-docker-hub-using-jenkins-](https://medium.com/codex/how-to-push-a-docker-image-to-docker-hub-using-jenkins-487fb1fcbe25) [487fb1fcbe25](https://medium.com/codex/how-to-push-a-docker-image-to-docker-hub-using-jenkins-487fb1fcbe25)   
10. Effective DevOps: Building a Culture of Collaboration, Affinity, and Tooling at Scale, Jennifer Davis, Ryn Daniels, O'Reilly Media June 2016.



11. Practical DevOps: Implement DevOps in your organization by effectively building, deploying, testing, and monitoring code,Joakim Verona, Packt Publishing, 2nd Edition,2016