

**C.I.T.L. EXPERIMENT 9**

**Topic : Inventory Management System**

**Submitted By:**

| Akash Panicker | 2021300089 |
| --- | --- |
| Mahesh Patil | 2021300095 |
| Rohit Phalke | 2021300100 |
| Adwait Purao | 2021300101 |

**Submitted To:**

Prof. Sunil Ghane

**Aim:**

Create and use a Docker container interactively.Create a Dockerfile, which allows you to declaratively define your containers. Run detached containers and understand port forwarding

**Problem Statement:**

Develop an inventory management system for a retail store that efficiently tracks and manages the inventory of products. The system should provide real-time updates on stock levels, generate alerts for low stock items, enable easy addition and removal of products, and offer insights into sales trends to optimize restocking decisions.

**Theory:**

* **Containerization**

Containerization is a technology that allows you to package software into standardized units that can be easily deployed and managed. Containers are similar to virtual machines, but they are more lightweight and efficient. This makes them ideal for deploying applications in a cloud environment.

* **Docker**

Docker is a leading platform for containerization. It provides a suite of tools that make it easy to build, run, and manage containers. Docker is a popular choice for deploying applications on a variety of platforms, including Linux, Windows, and macOS.

* **Benefits of Containerization**

There are many benefits to using containerization, including:

* + Portability: Containers can be easily ported from one environment to another.
  + Isolation: Containers are isolated from each other, which means that they cannot interfere with each other.
  + Resource efficiency: Containers are more resource efficient than virtual machines.
  + Scalability: Containers can be easily scaled up or down to meet demand.
* **Creating and Using a Docker Container Interactively**

Docker containers provide a lightweight and portable way to package and run applications. They are self-contained environments that include everything an application needs to run, including its code, libraries, and dependencies. This makes them ideal for deploying applications in a consistent and reliable way across different environments.

* **Creating a Dockerfile**

A Dockerfile is a text file that contains instructions for building a Docker image. The instructions are written in a simple, declarative format that makes it easy to understand and maintain.

* **Building an Image from a Dockerfile**

To build an image from a Dockerfile, you can use the following command:

*docker build -t my-image .*

This command will run a container from the my-image image in interactive mode. This means that you will be able to type commands into the container and see the results.

* **Detached Containers**

A detached container is a container that is not running in interactive mode. This means that you will not be able to type commands into the container after it is started. To run a container in detached mode, you can use the -d flag:

*docker run -d my-image*

* **Port Forwarding**

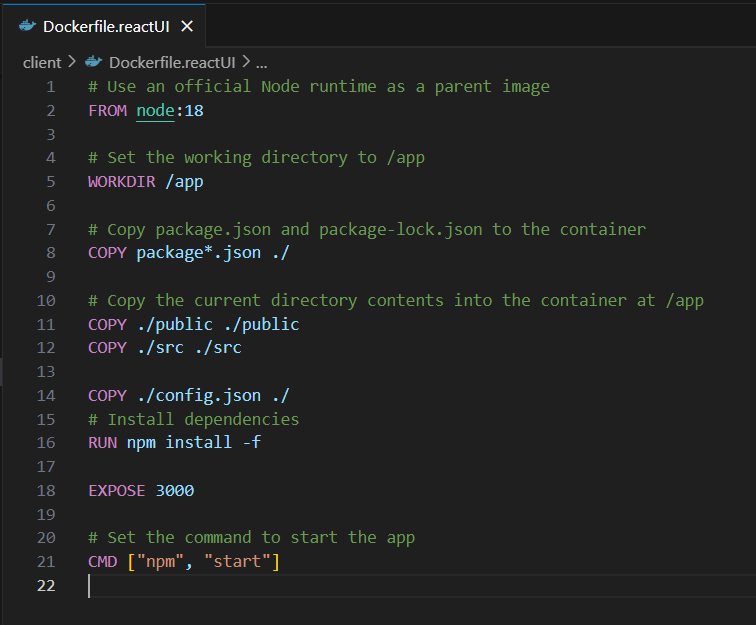
Port forwarding is a way to map a port on the host machine to a port on a container. This allows you to access the container's application from the host machine. To forward a port, you can use the -p flag:

*docker run -d -p 8000:8000 my-image*

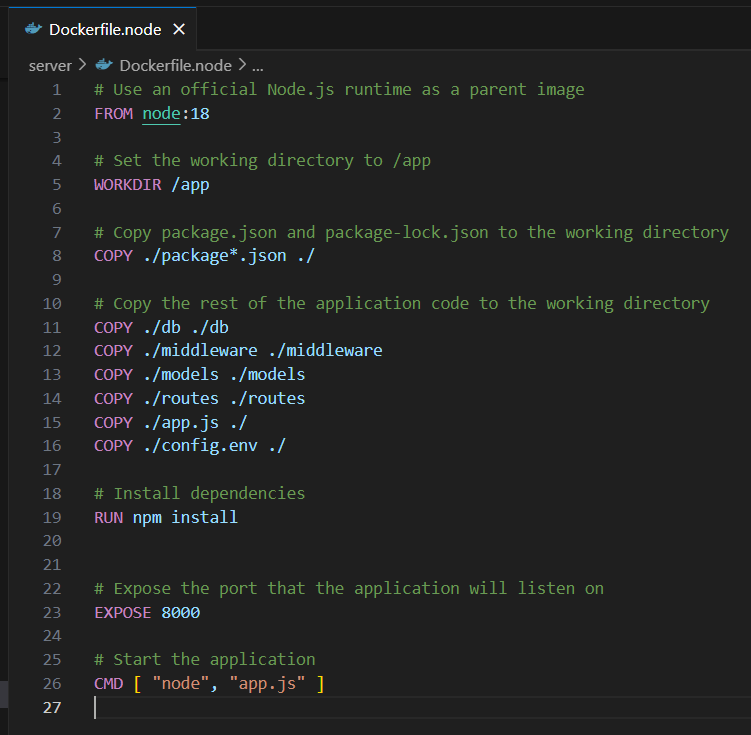
This command will run a container from the my-image image in detached mode and map port 8000 on the host machine to port 8000 on the container. This means that you can access the container's application at http://localhost:8000.

**Screenshots:**

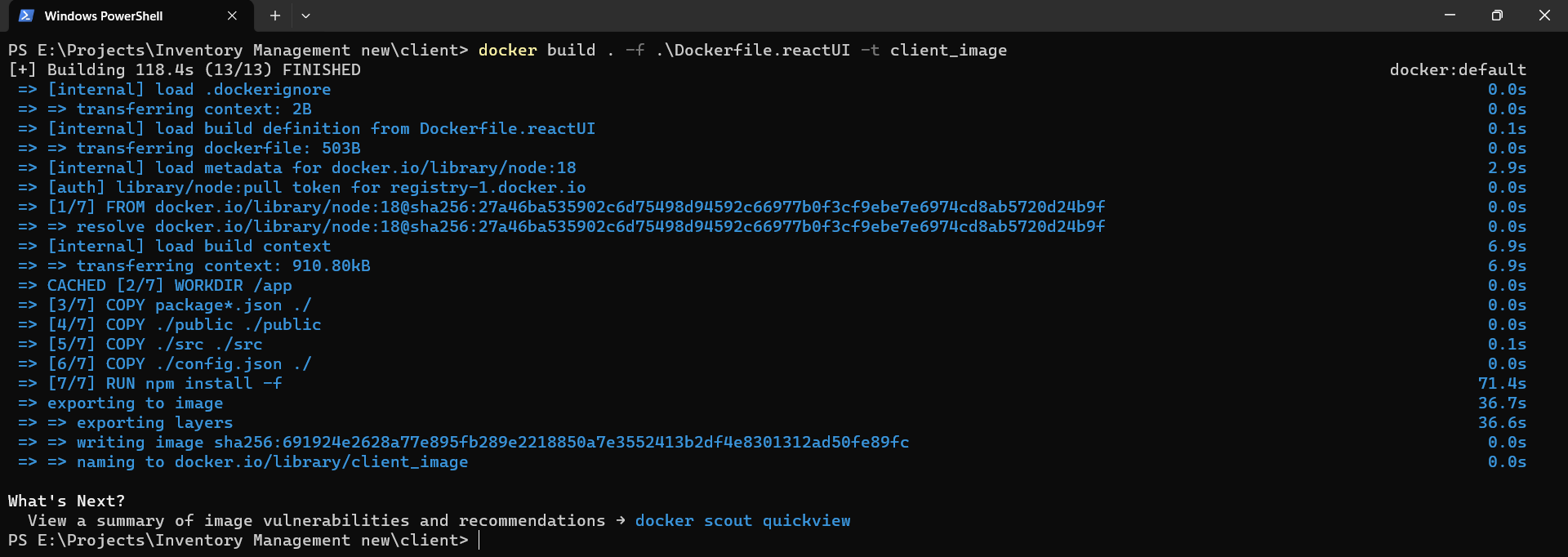
**Docker file for ReactJS frontend**

****

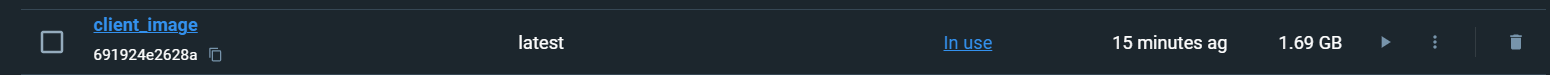
**Docker file for NodeJS backend**

****

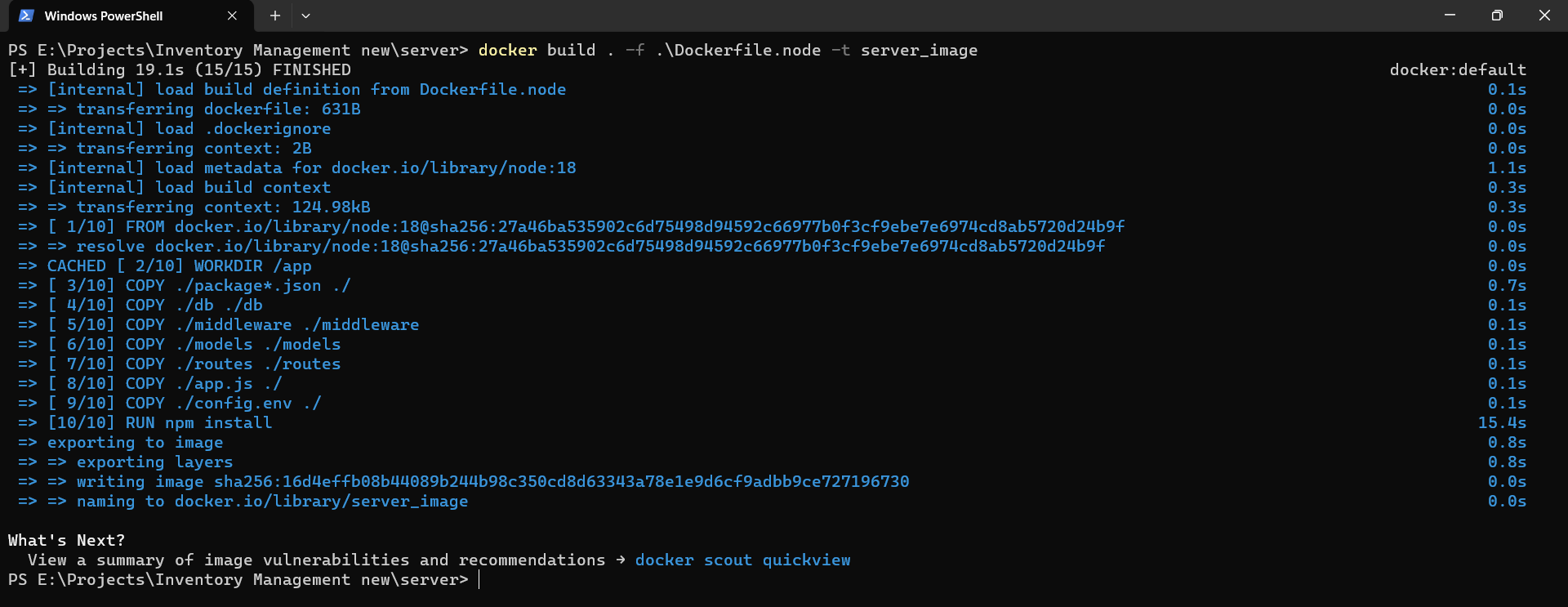
**Building image of frontend**

****

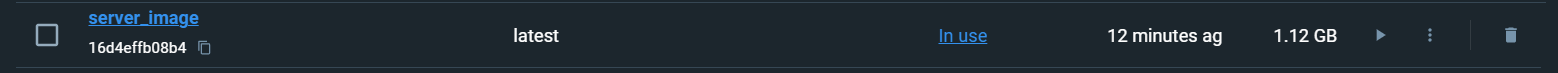
**Frontend image created**

****

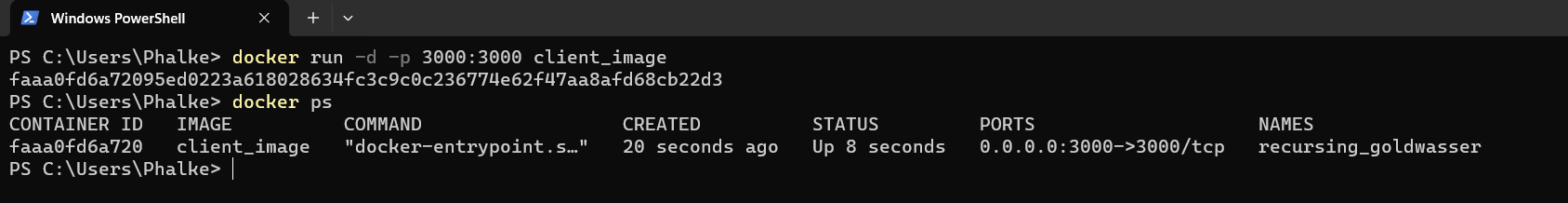
**Building image of backend**

****

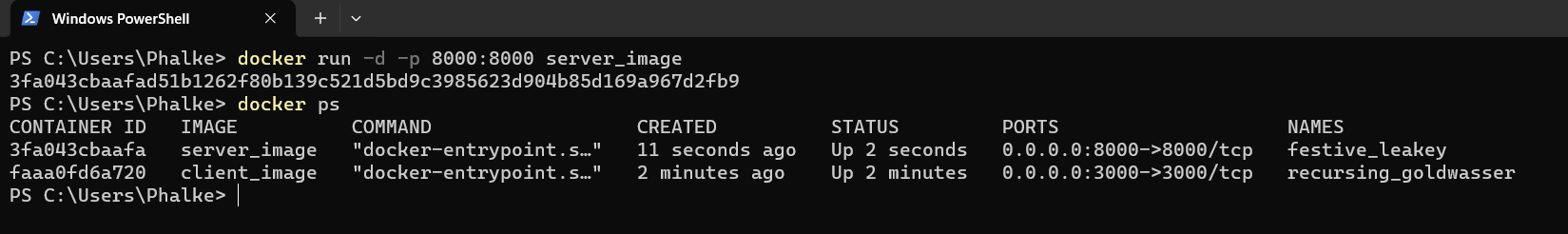
**Backend image created**

****

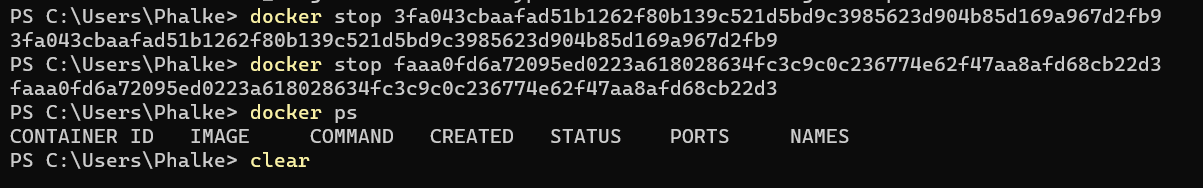
**Creating container and Running frontend image on port 3000 in detached mode**

****

**Creating container and Running backend image on port 8000 in detached mode**

****

**Images stopped**

****

**Conclusion:**

Docker containers have become a popular choice for developing, deploying, and managing applications due to their lightweight, portable, and efficient nature. They provide a self-contained environment that encapsulates an application and its dependencies, ensuring consistency and predictability across different deployment environments. Docker's portability and isolation capabilities make it well-suited for cloud environments, enabling rapid application delivery and flexible resource management. In summary, Docker containers have transformed the way we build and run applications, offering a powerful and versatile solution for modern application development and deployment.