

**C.I.T.L. EXPERIMENT 10**

**Topic : Inventory Management System**

**Submitted By:**

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

**Submitted To:**

Prof. Sunil Ghane

**Aim:**

Use docker-compose to run a multi-container web application

**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 and Docker**

Containerization is a lightweight virtualization technology that allows developers to package software applications and their dependencies into self-contained units called containers. Containers are isolated from each other, making them highly portable and scalable. Docker is the most popular containerization platform, providing a comprehensive set of tools for building, running, and managing containers.

* **Multi-Container Web Applications**

Web applications often consist of multiple components, such as a web server, a database, and a cache. Traditionally, these components would be deployed on separate servers, which can be complex to manage and scale. Containerization offers a more efficient and flexible solution for deploying multi-container web applications.

* **Docker Compose**

Docker Compose is a tool for defining and running multi-container Docker applications. It simplifies the process of managing multiple containers, allowing developers to start, stop, and update all containers in a stack with a single command. Docker Compose also handles networking and dependency management between containers.

* **Running a Multi-Container Web Application with Docker Compose**

To run a multi-container web application with Docker Compose, you typically follow these steps:

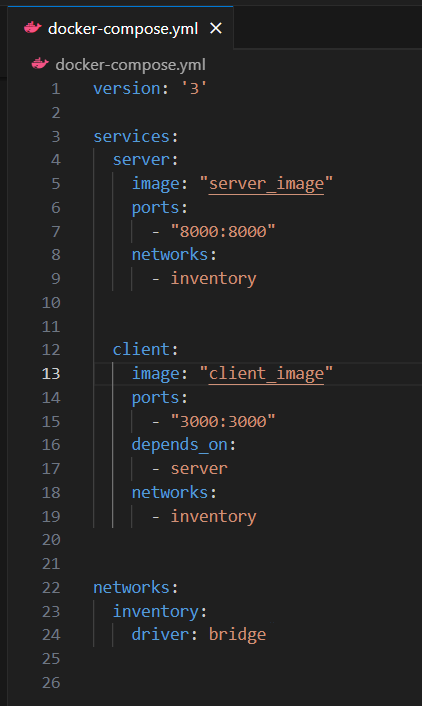
* + **Create a docker-compose.yml file:** This file defines the services that make up your application, specifying their images, ports, volumes, and other configuration options.
  + **Build the application images:** If you have Dockerfiles for your application components, you need to build them before running Docker Compose.
  + **Start the application:** Run the docker-compose up command to start all the containers defined in the docker-compose.yml file.

Docker Compose will handle the communication and dependency management between the containers, allowing your web application to run seamlessly.

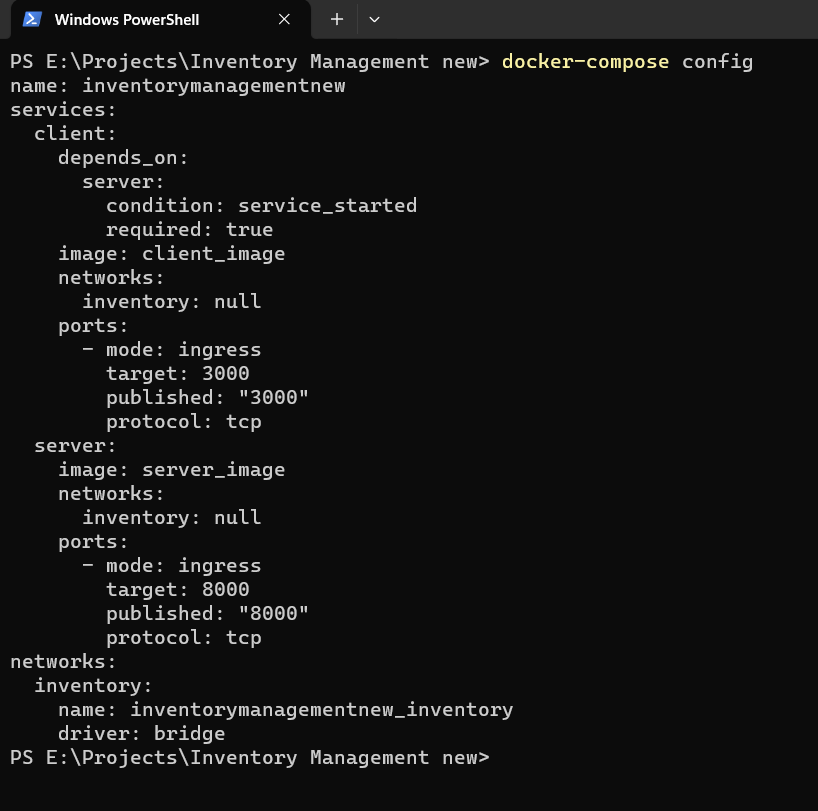
* **Benefits of Using Docker Compose for Multi-Container Web Applications**
  + Docker Compose offers several benefits for running multi-container web applications:
  + Simplified deployment and management: With Docker Compose, you can start, stop, and update all containers in your application with a single command.
  + Portable and scalable: Docker Compose applications can be easily deployed across different environments, from local development machines to production servers.
  + Isolated and secure: Containers are isolated from each other, making them more secure and preventing conflicts between applications.
  + Resource-efficient: Containers share the host machine's kernel, making them more resource-efficient than traditional virtualization technologies.

**Screenshots:**

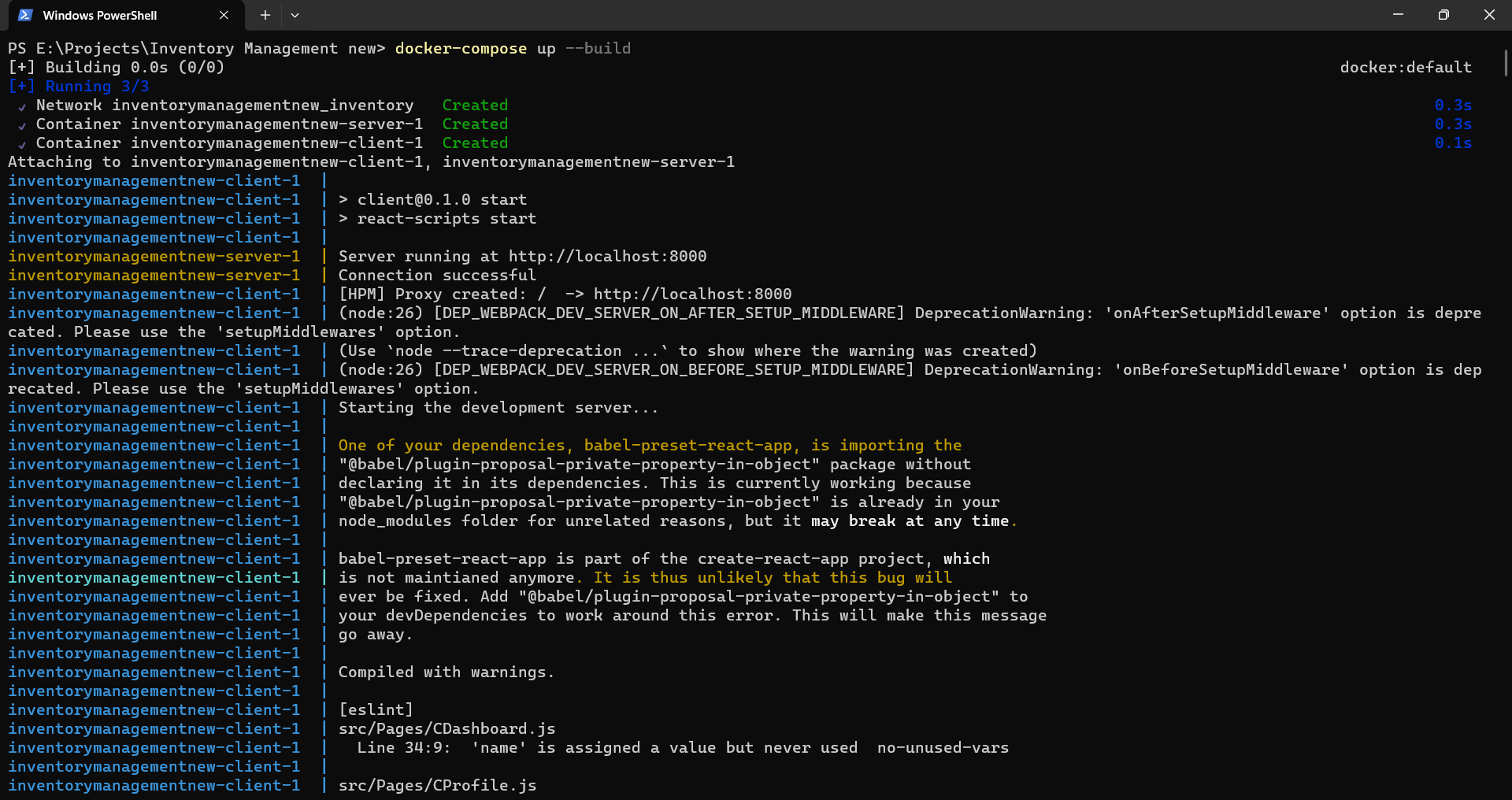
**docker-compose.yml file**

****

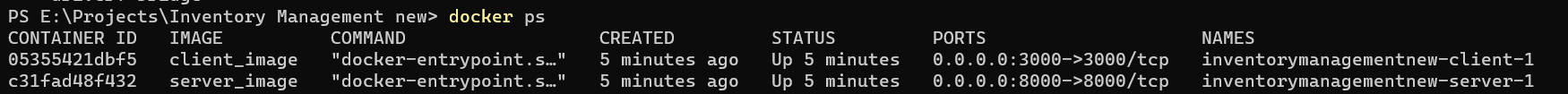
**Docker compose configuration**

****

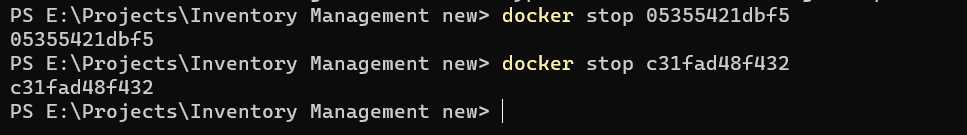
**Running docker-compose.yml file thus running a multiple container application**

****

**Multiple containers running**

****

**Containers stopped**

****

**Conclusion:**

Docker Compose is a valuable tool for simplifying the development and management of multi-container web applications. It provides a straightforward way to define, run, and scale complex application stacks, offering numerous advantages over traditional deployment methods. By leveraging Docker Compose, developers can streamline their workflow, enhance portability, improve resource utilization, and strengthen application security.