# 

# PROJECT REPORT

**Deploying A Cloud Service With Docker And Docker Swarm**

**Student:** Ngô Lê Hạnh Dung

**ID:** 20520961 **Class:** KHBC2020

**Course:** Cloud Computing - CSBU015.M11.KHBC

**Lecturer:** Bùi Thanh Bình

# 

**Contents**

[**PROJECT REPORT**](#_dlp5gkpmz0i) **0**

[**Introduction**](#_mlqws2y8zosz) **1**

[**Deploying a Cloud Service with Docker Swarm**](#_iq5oqkh1j6br) **1**

[Install Docker in all nodes](#_rc4fhxfmio5t) 2

[Build a static HTML site image based on Nginx image](#_3lihedvavq44) 3

[Initialize a Swarm](#_ayysou45y9o6) 6

[Create a Registry Service for the Web Application](#_m6suiysy8g09) 7

[Create a Replicated Docker Service to the Swarm](#_9fohxtwqioww) 8

[Create a Global Docker Service to the Swarm](#_jwpcs090sefg) 11

[**Conclusion**](#_8b93p13hxm2p) **13**

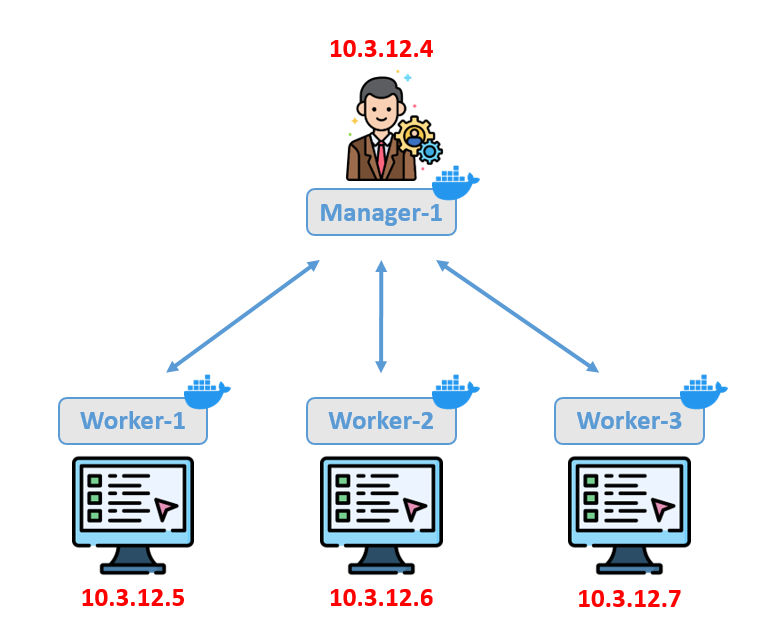
# 

# Introduction

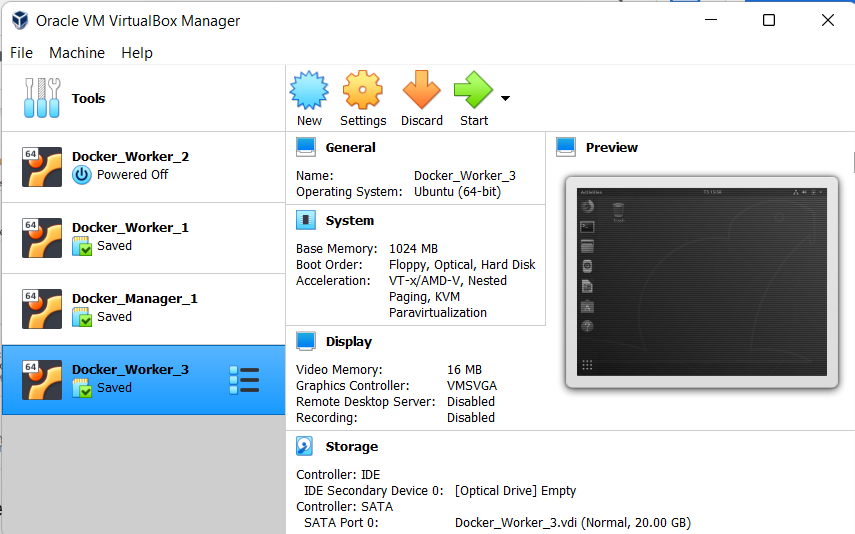
Cloud computing is the most popular Internet technology nowadays and one of the most important components of a cloud computing system is virtualization technology. In computing, virtualization refers to the process of generating a virtual (rather than physical) version of something, such as a virtual machine, an operating system (OS), a storage device, or computer network resources. Docker is a new type of virtualization technology. In this project, we will deploy a multi-node cloud service with Docker and Docker Swarm.

# Deploying a Cloud Service with Docker Swarm

This is a schematic diagram of our system:



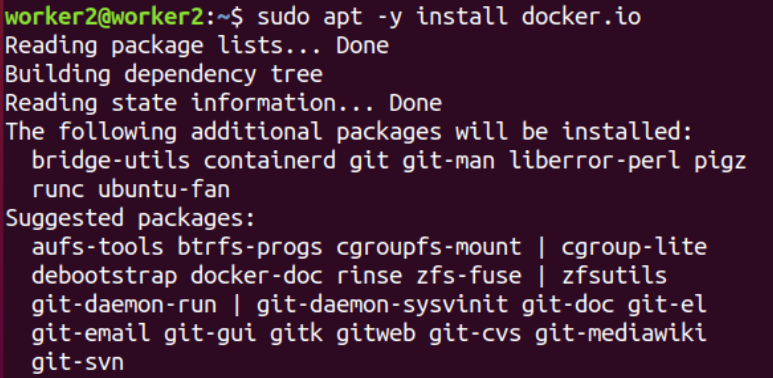
Our cloud service has 4 nodes: 1 manager and 3 workers using *ubuntu18.04* virtual machines. We deployed the *Manager-1*, *Worker-1*, *Worker-2*, *Worker-3* nodes in Virtual Box as shown below.



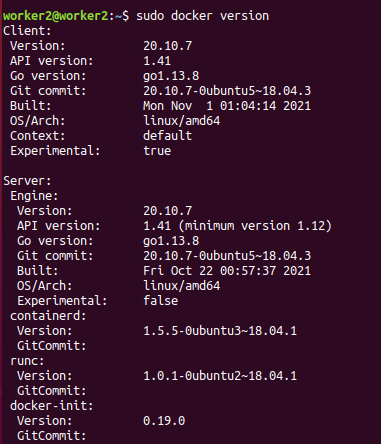
In order to test the connection between nodes and through the Internet, we build a static HTML site based on the Nginx image. The process with the following steps will be presented as below.

## Install Docker in all nodes

First, we will install Docker for all 4 nodes with the following command:

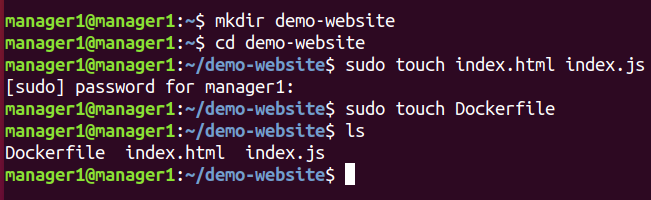


After installation, we can check the version of Docker as below:

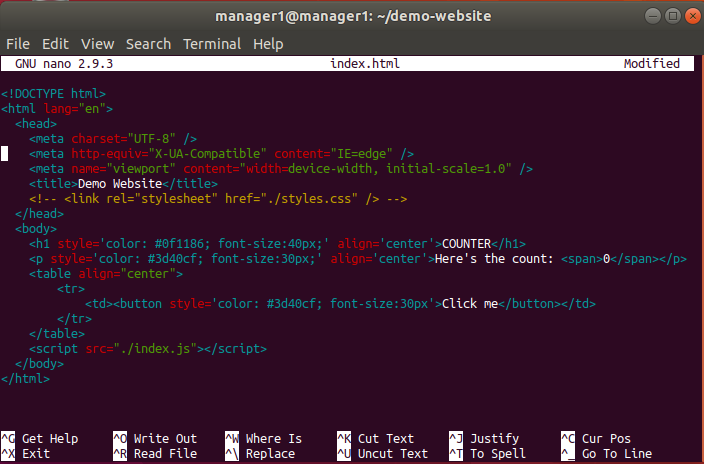


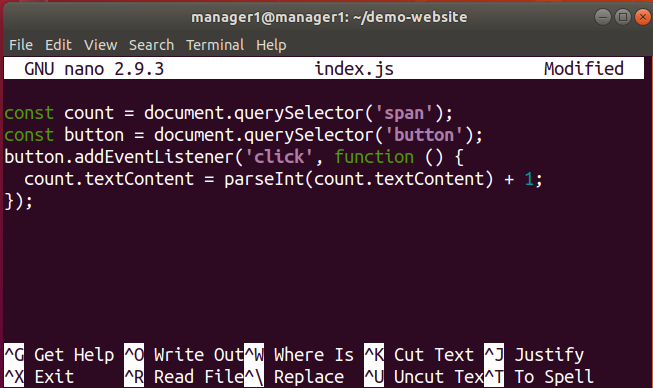
## Build a static HTML site image based on Nginx image

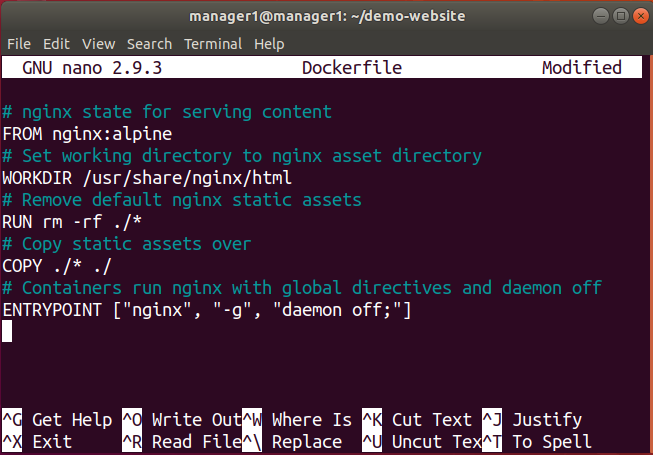
In this step, we build an HTML site image based on the Nginx image in the *Manager-1* node. First, we create a simple static site with some assets. We can make a directory called demo-website and, within that directory, let’s make an index.html file, an index.js file and a Dockerfile for building image.



Next up, put the following content in each file. This ends up being a pretty bare-bones site with a dynamic counter.



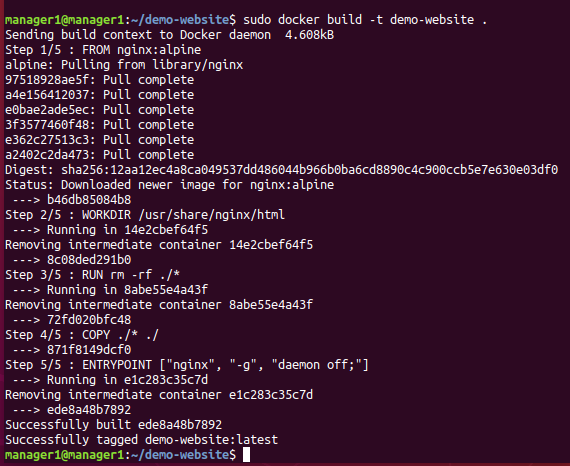


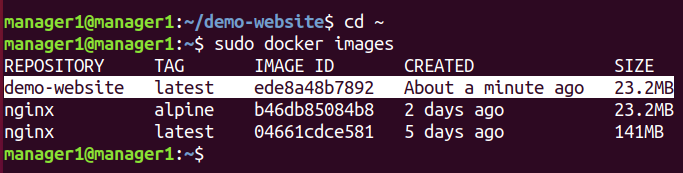


We run the application without Docker first to ensure that it works well:

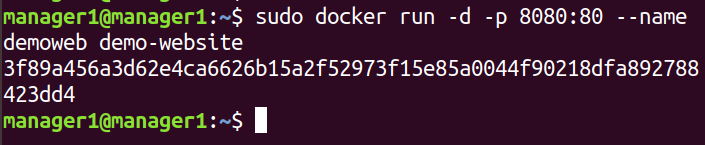


Now let’s build an image called *demo-website*:



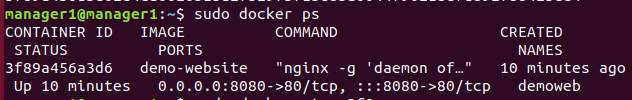


Our image is built, we can start a container with the following command, which will serve our app on port 8080.





After stopping and removing this container, we will go to the next step to create a swarm.

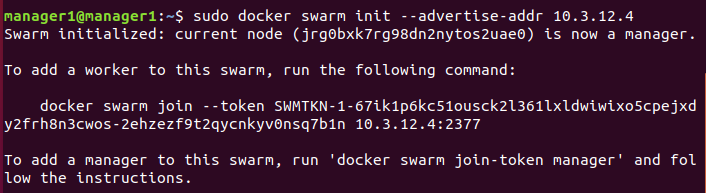




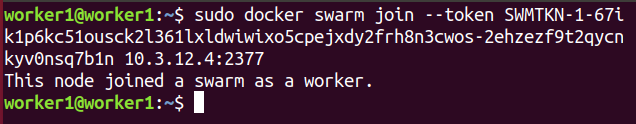


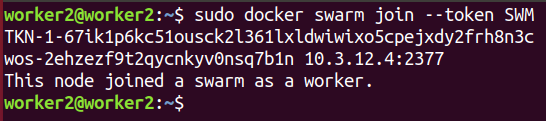
## Initialize a Swarm

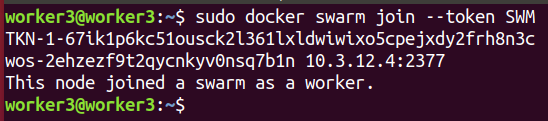
We will initialize a swarm at the *Manager-1* node so that it becomes the primary manager.



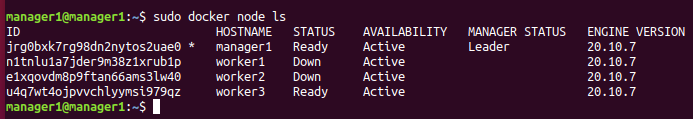
Then, we add the other nodes one by one to the swarm as below:





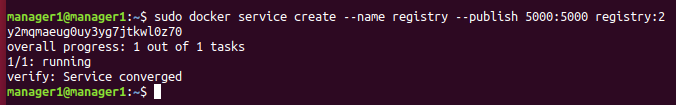


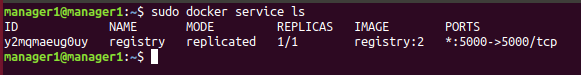
After adding 3 nodes, we come back to the Manager-1 node to see all nodes in the swarm:



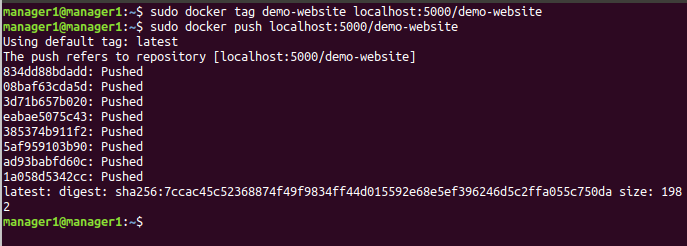
## Create a Registry Service for the Web Application

For our application can be hosted in all nodes within the swarm, we need to create a “*registry*” service for storing the application and publish its port to make sure it can be reached from every node.

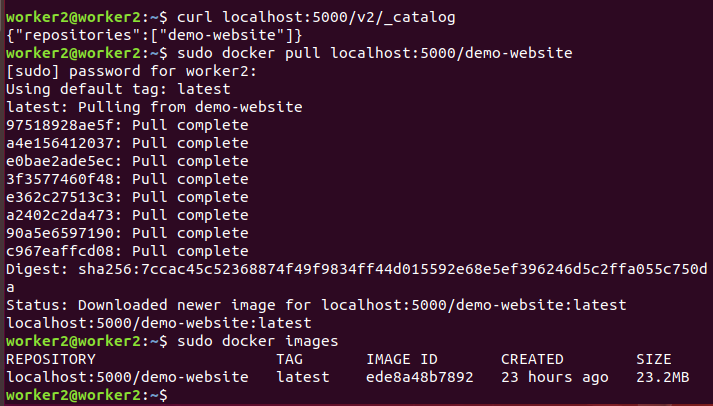




After creating the registry service, we will tag and push the application image into it and reach via *‘localhost:5000’*.



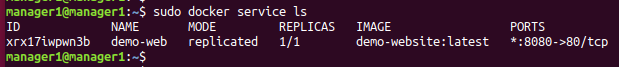
From worker nodes, we can access the registry and pull the image from it:



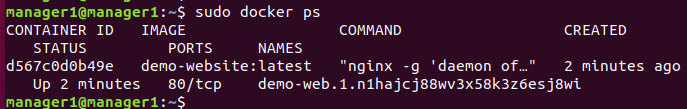
## Create a Replicated Docker Service to the Swarm

We start to create a new service for our HTML web application in the Manager-1 node:

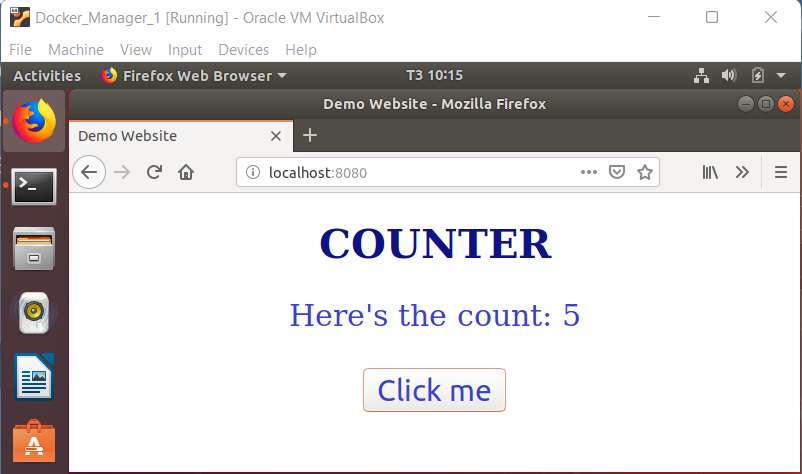
## 



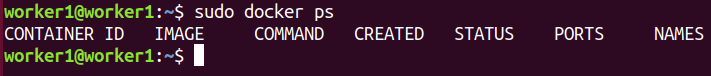
We can see that it also appear in the container list of Manager-1 node and it is running:

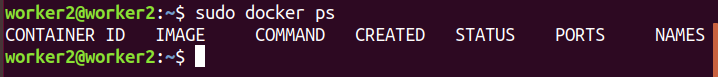


We can check if the web application is running using the browser:

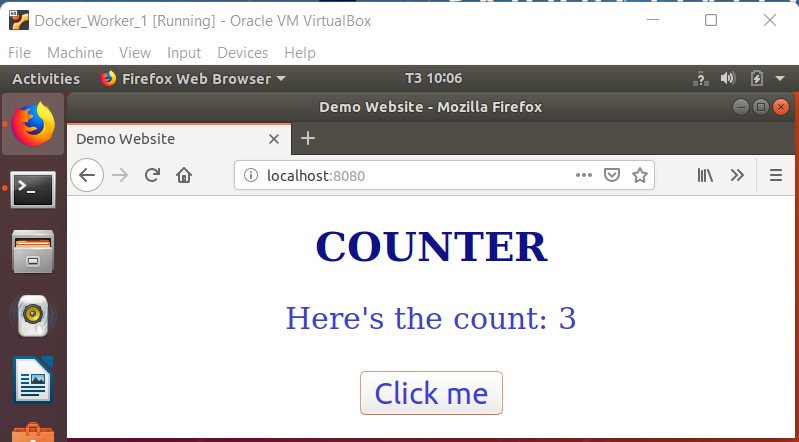


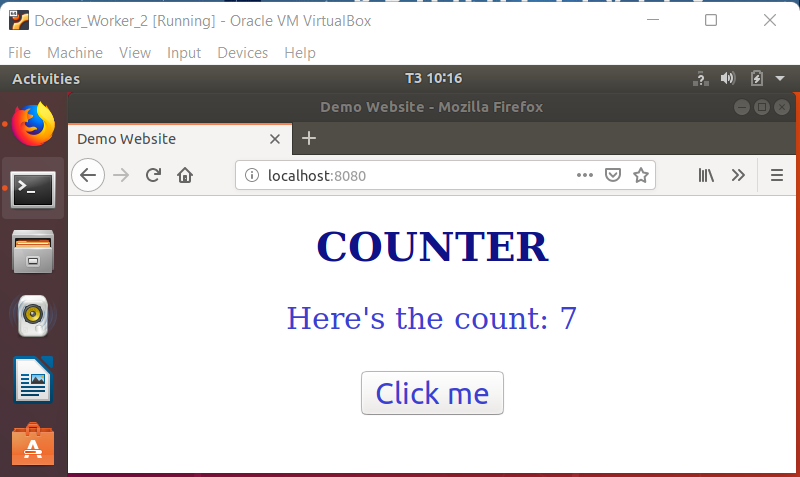
Now, we will come to the Worker-1 and Worker-2 nodes and see that the application is not being hosted in those two:





However, when we browse the application, we can get the same application as shown below. This is because all the nodes in the same cluster can see the application that we’ve started. Since the application is in replicated mode, it’s the only host in thenode that created it (*Manager-1*).

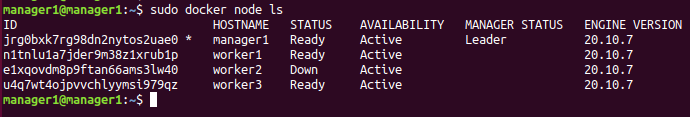




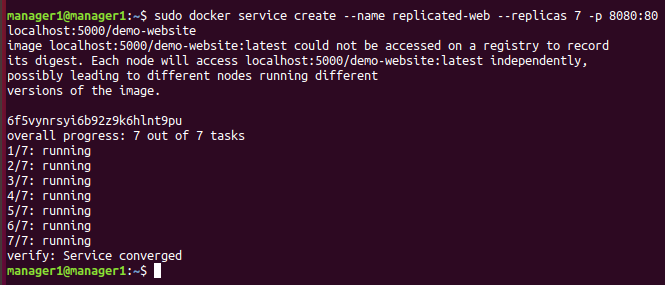
Now we will remove this service and create a new one with a number of tasks and see how Docker Swarm scale them.



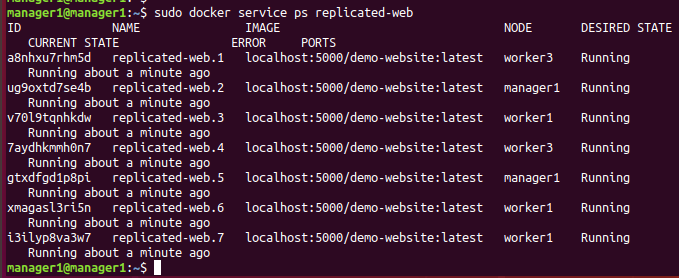
We will use Manager-1, Worker-1, Worker-3 nodes to run this service.



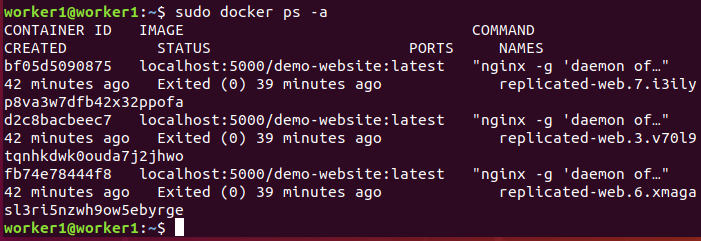
Now, we create the service with 7 tasks (replicas) using the image that we pushed in the registry. This will help other nodes can also host the application.



From the picture below, we see that the swarm has scaled 3 tasks for Worker-1, 2 for Manager-1 and 2 for Worker-2 by looking at the Node column.

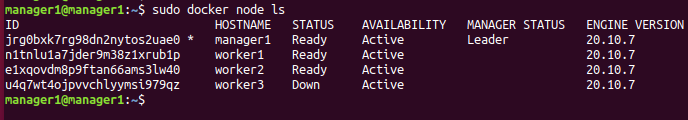


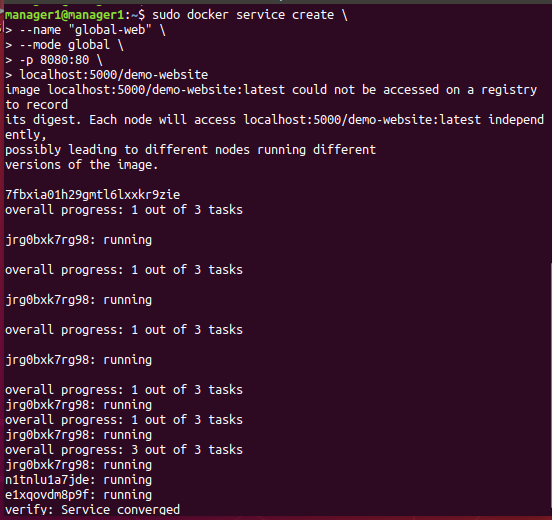
And because we use the image from the registry, the worker nodes also host the application by themselves.



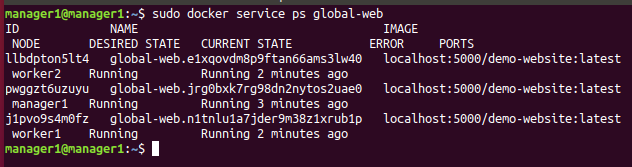
## Create a Global Docker Service to the Swarm

In this step, we create a service for our application with global mode as shown below. We will use *Manager-1*, *Worker-1, Worker-2* nodes to run this service.

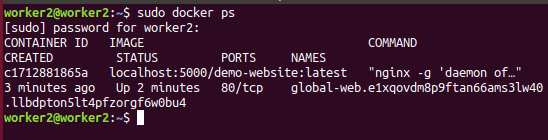




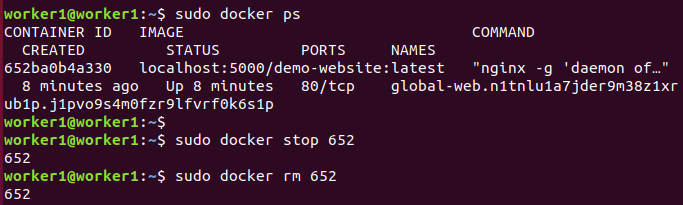
After creating, the service’s tasks are running in all nodes:



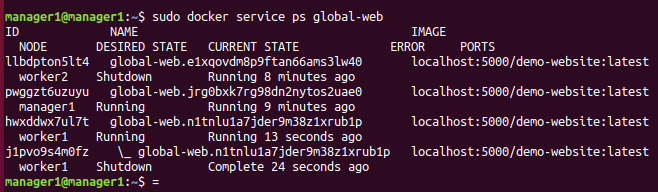
Since we use the image from the registry, the application also appears in the container list of worker nodes.



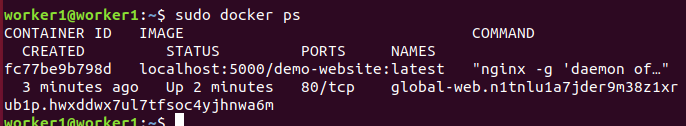
Now we will stop and remove the task in the Worker-1 node and see if it has any changes.



From the manager node, we know that the swarm assume that the removed task is completed and start a new one in Worker-1.



If we come back to the Worker-1 node and check the running container list, we can see that new task.



# **Conclusion**

Cloud computing has been widely used by both individuals and businesses for the past several years. Docker and container technologies were employed, which would be beneficial to everyone in terms of load balancing, security, and so on. We also set up a docker swarm in the form of a cluster and ran the services on it to see how it works. In conclusion, it is a helpful tool that helps us develop, shipping, and run applications more quickly.