**Name : - Om Butala**

**Roll no :- 323071**

**PRN no :- 22120149**

**Batch :- C3**

**Assignment No 7**

**Title: Deploy a static website using Docker.**

**Theory:**

1. What is Docker ?

* Docker is a software platform that allows developers to easily create, deploy, and run applications in isolated containers. Containers are lightweight, standalone executable packages that contain everything needed to run an application, including code, libraries, and dependencies. Docker provides a consistent and reproducible environment for development, testing, and deployment, regardless of the underlying infrastructure. Docker is widely used in the DevOps community and has become a key tool for building and deploying cloud-native applications.

1. Difference between Docker and Virtual machine

* The main difference between Docker and virtual machines (VMs) is the way they utilize system resources and isolation.
* Virtual machines use a hypervisor to create a virtualized hardware environment, allowing multiple operating systems to run on the same physical machine. Each VM requires its own copy of the operating system, along with all the necessary system libraries and dependencies, resulting in a larger footprint and slower startup times.
* Docker, on the other hand, uses containerization to create isolated environments that share the underlying host operating system kernel. Each container contains only the application and its dependencies, resulting in a smaller footprint and faster startup times. Docker allows multiple containers to run on the same host, sharing the same kernel and resources, but with separate file systems and network interfaces.
* Overall, Docker is more lightweight, efficient, and faster than VMs, but VMs offer stronger isolation and security for running multiple operating systems on the same hardware.

1. Docker Commands

* Here are some commonly used Docker commands:
* docker run: Creates and starts a container.
* docker build: Builds an image from a Dockerfile.
* docker images: Lists all available Docker images.
* docker ps: Lists all running containers.
* docker stop: Stops a running container.
* docker rm: Removes a stopped container.
* docker rmi: Removes an image.
* docker exec: Runs a command in a running container.
* docker-compose: A tool for defining and running multi-container Docker applications.
* docker logs: Displays the logs for a container.

1. Dockerfile

* A Dockerfile is a text file that contains instructions for building a Docker image. It's used to automate the process of building a Docker image, making it easier to create and maintain reproducible, self-contained environments.
* Here are some common instructions used in a Dockerfile:
* FROM: Specifies the base image to build upon.
* RUN: Runs a command in the container.
* COPY: Copies files from the host to the container.
* ADD: Copies files from the host to the container, with support for URLs and archives.
* WORKDIR: Sets the working directory for subsequent instructions.
* EXPOSE: Specifies the ports to be exposed by the container.
* CMD: Specifies the default command to be run when the container starts.

1. Docker-Compose and Docker-swarm

* Docker Compose and Docker Swarm are both tools for orchestrating and managing Docker containers, but they have different use cases.

Docker Compose is a tool for defining and running multi-container Docker applications. It allows you to define your application's services, networks, and volumes in a single YAML file and start your entire application with a single command. Docker Compose is ideal for development and testing environments where you need to spin up multiple containers to support your application.

Docker Swarm, on the other hand, is a native clustering and orchestration tool for Docker. It allows you to deploy and manage a swarm of Docker nodes, creating a highly available and scalable Docker environment. Docker Swarm is ideal for production environments where you need to manage a large number of containers across multiple hosts.

Here are some key differences between Docker Compose and Docker Swarm:

Scalability: Docker Swarm can scale across multiple hosts, while Docker Compose is designed to run on a single host.

High availability: Docker Swarm provides built-in support for high availability, while Docker Compose does not.

Orchestration: Docker Swarm provides advanced orchestration features, such as rolling updates and service discovery, while Docker Compose focuses on defining and running multi-container applications.

In summary, Docker Compose is ideal for development and testing environments where you need to manage multiple containers on a single host, while Docker Swarm is ideal for production environments where you need to manage a large number of containers across multiple hosts with advanced orchestration features.

**Implementation:**

**Step 1: Install nginx on windows follow the link:**

[**http://nginx.org/en/docs/windows.html**](http://nginx.org/en/docs/windows.html)

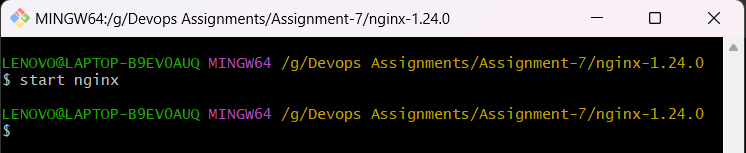
cd c:\

unzip nginx-1.23.4.zip

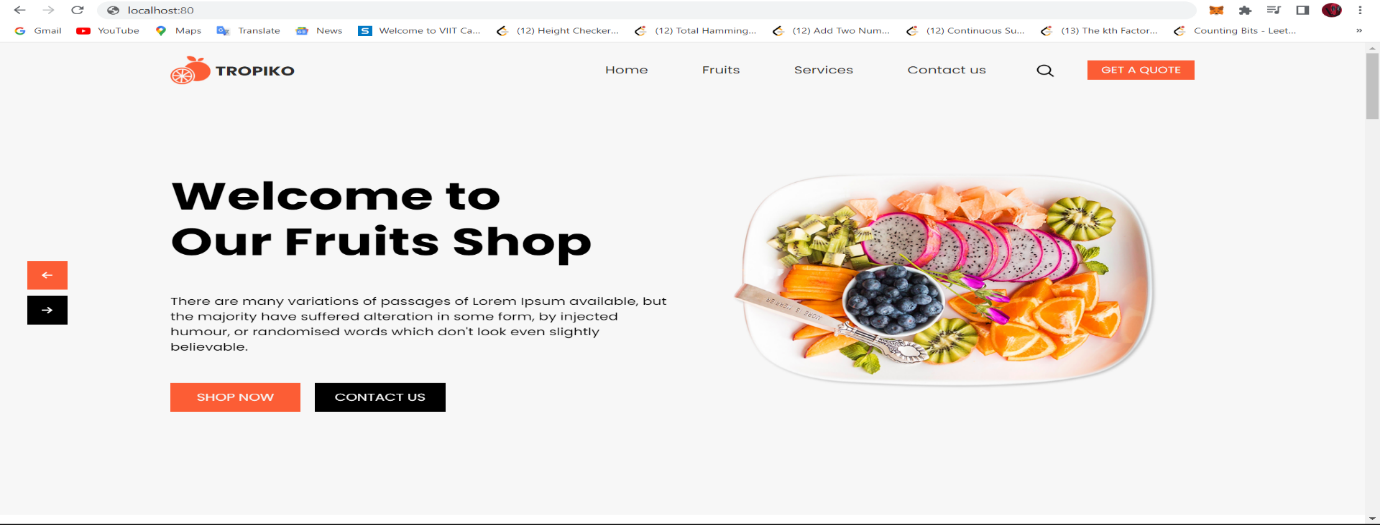
cd nginx-1.23.4

start nginx

**Step 2: Copy the sample-website in “C:\nginx\html\” folder and start nginx**

****

**Step 3: open browser and run “localhost:80”**

****

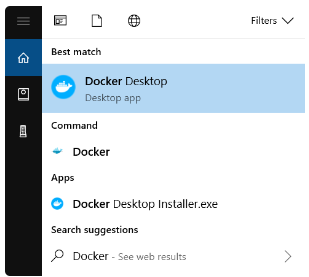
**Step 4: Download Docker for windows, follow the link**

[**https://docs.docker.com/desktop/install/windows-install/**](https://docs.docker.com/desktop/install/windows-install/)

## Step 5: Start Docker Desktop

Docker Desktop does not start automatically after installation. To start Docker Desktop:

1. Search for Docker, and select **Docker Desktop** in the search results.

****

## Step 6: Open Powershell and check Docker installation using commands:

## docker –version

## 

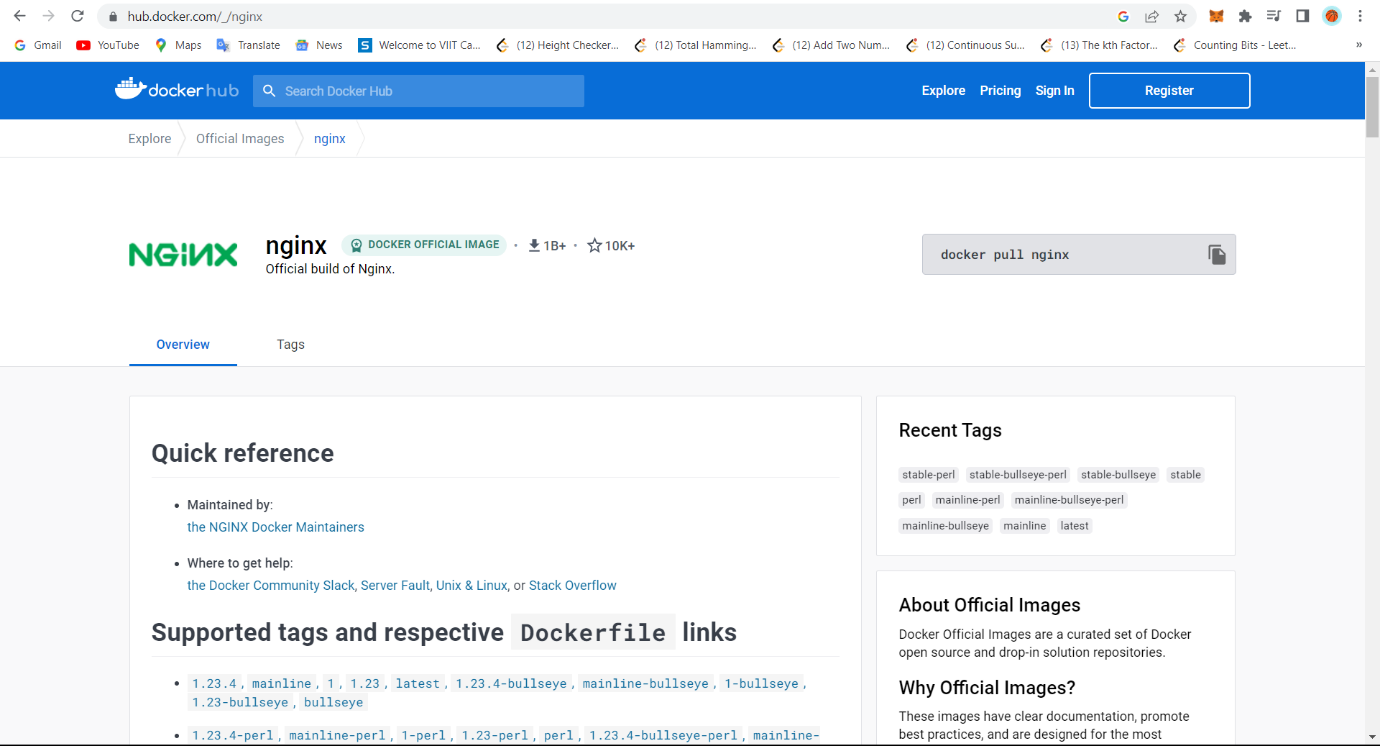
## docker info

## docker version --format '{{json .}}'

* **Steps to run the “Sample website” in Docker container**

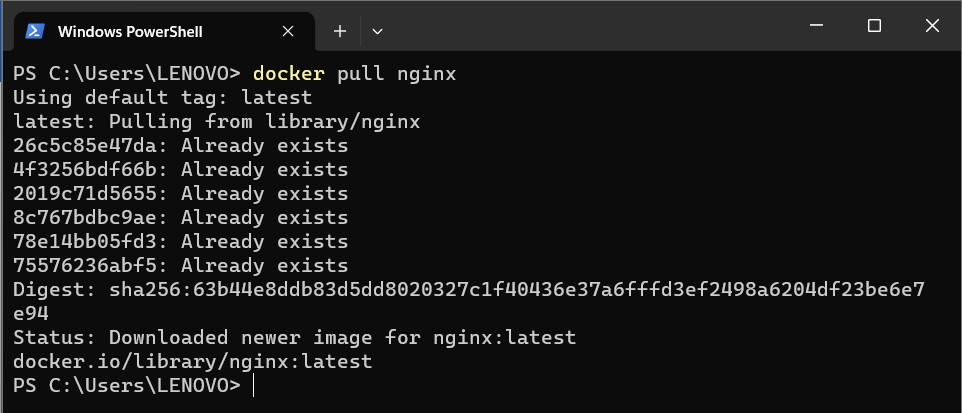
**Step 1) visit to Docker hub web site:** [**https://hub.docker.com/**](https://hub.docker.com/)

Step 2) search for “nginx” image on site

****

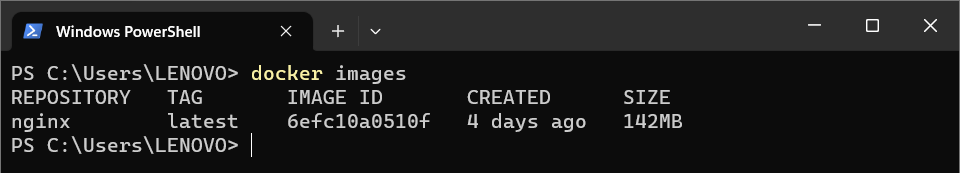
**Step 3) pull the latest image of nginx using command**

**“docker pull nginx”**

****

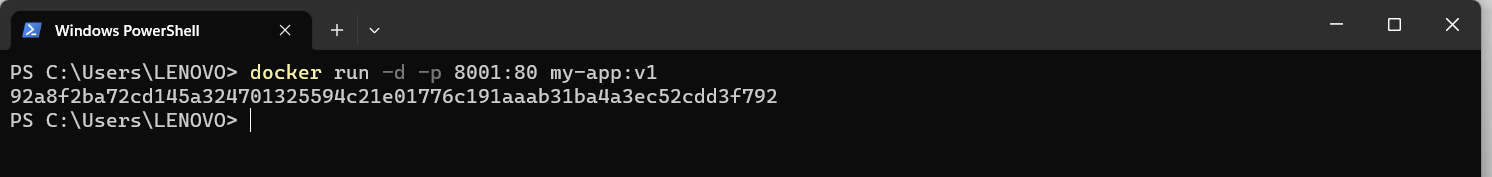
**Step 4) check the docker images on your desktop by using command:**

**“docker images”**

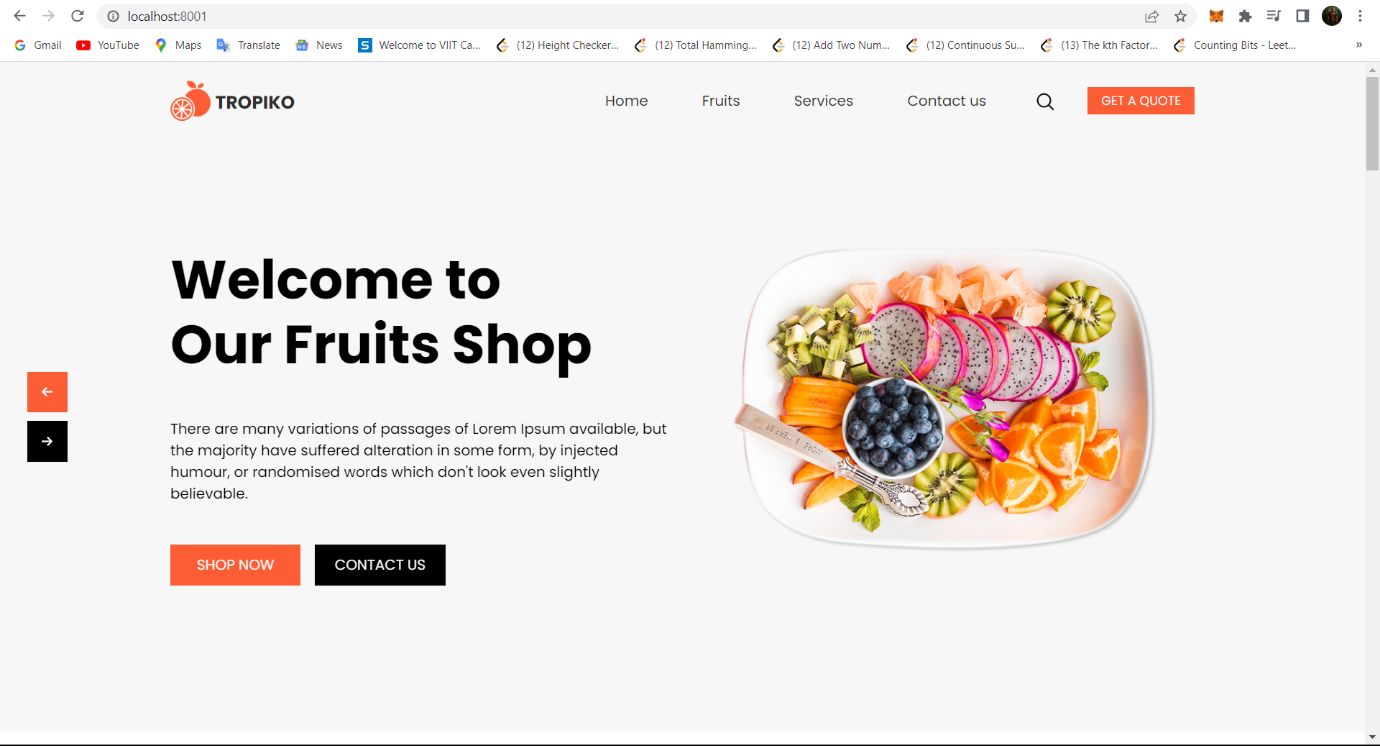
****

**Step 5) go in the “SampleWebsite” folder and then Create a container using the docker command and sync the “SampleWebsite” folder with folder inside the container folder. (This is called Mount Bind”)**

**“docker run -d -p 8001:80 -v ${PWD}:/usr/share/nginx/html --name web-site nginx”**



**Step 6)verify the website open browser and chec “localhost:8001”. Now this website is running inside your container.**



DockerFile

**Step 1) Create a Directory structure like**

App

tropiko

Dockerfile

**Step 2) Write a following script into “Dockerfile”**

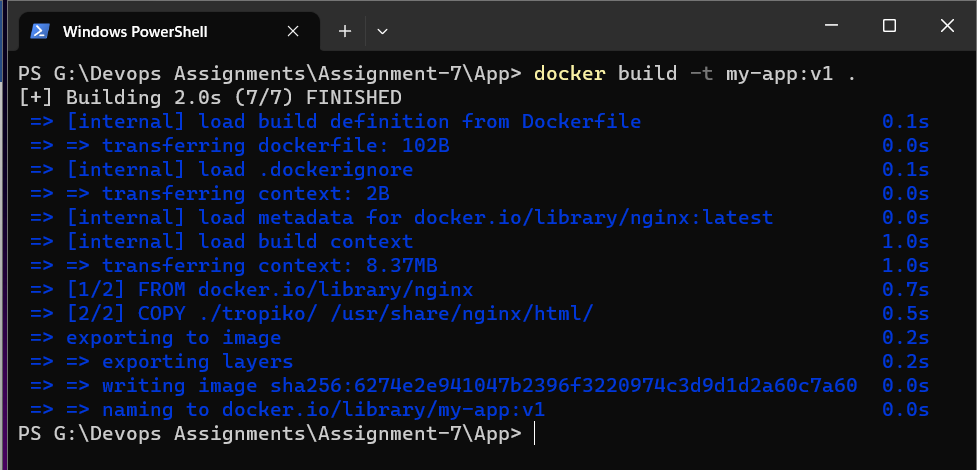
FROM nginx:latest

COPY ./tropiko/ /usr/share/nginx/html/

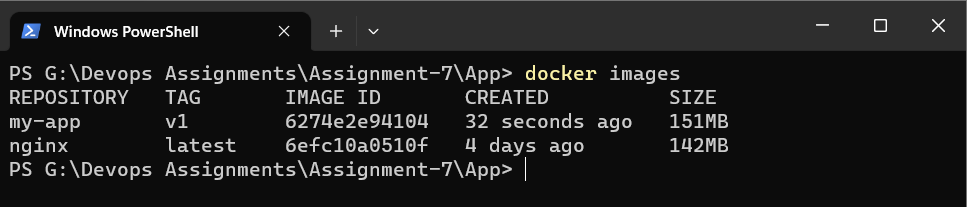
EXPOSE 80

**Step 3) build image from docker file using command**

**“docker build -t my-app:v1 . ”**

****

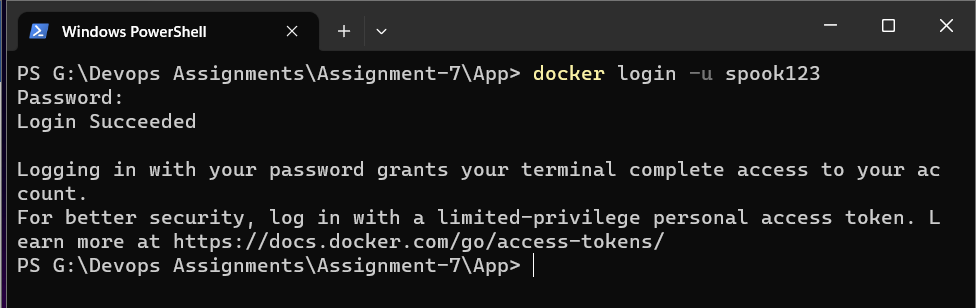
**Step 4) check images using command: docker images**

****

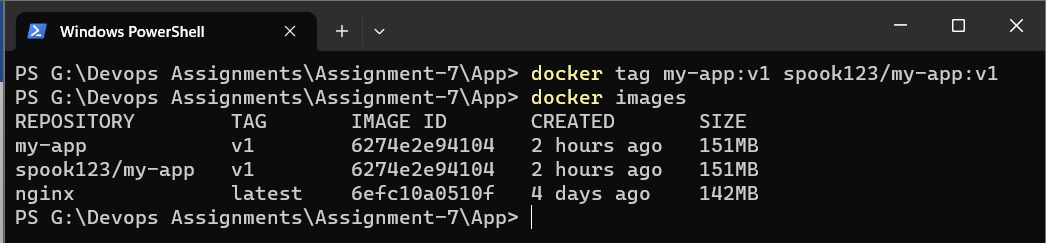
PUSH Image to “DockerHub”

**Step 1) login to docker hub using command**

1. **docker login -u spook123**

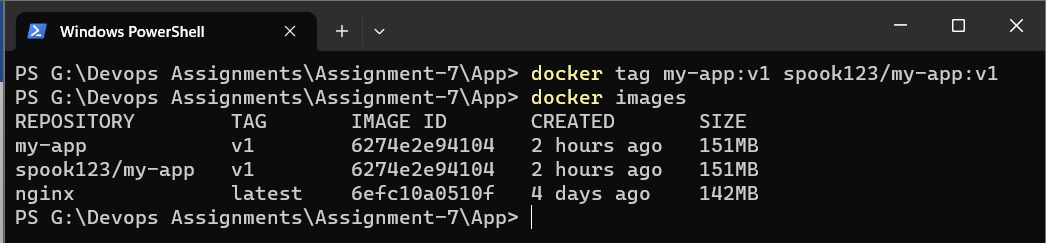
****

**2) docker images**

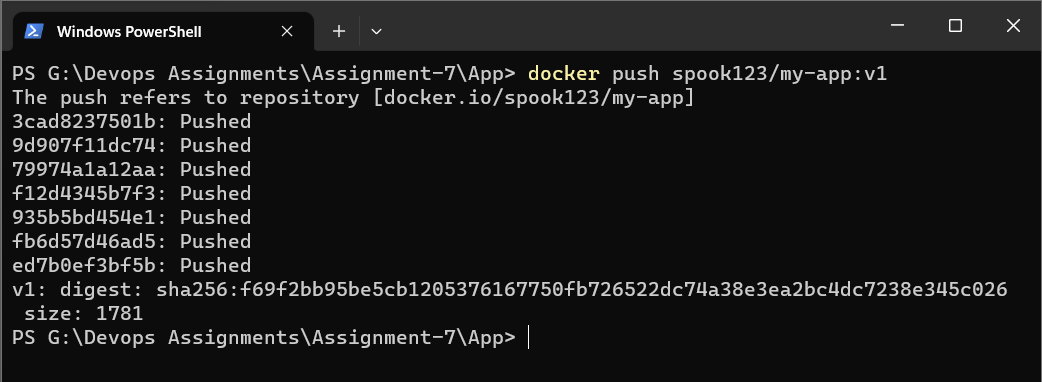
****

**3) docker tag (old image name) spook123/newname**

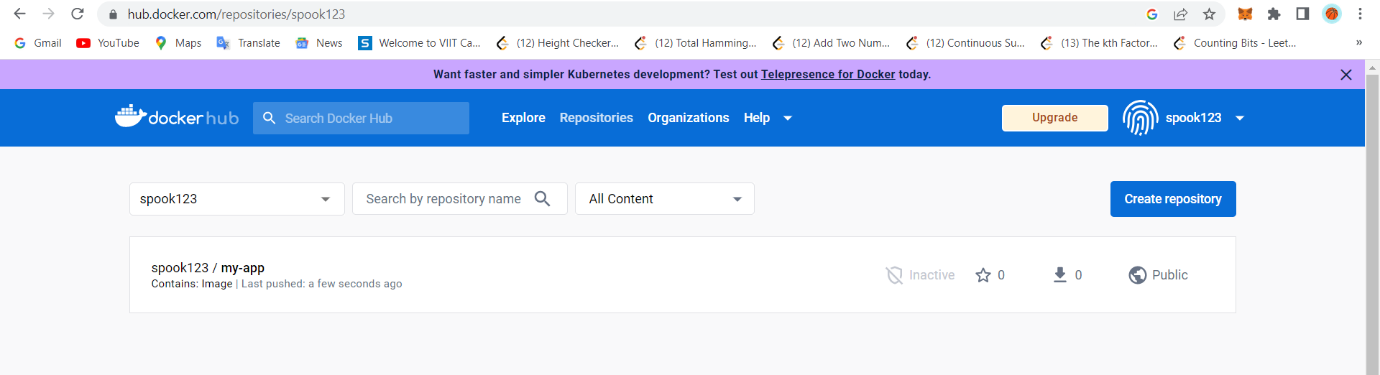
**docker tag my-web:v1 spook123/newapp**

****

**4) docker push spook123/newapp**

****

**Step 5) Login to Docker Hub and check the repository**

****

Now you can share this image with anyone with running nginx and your web application.