Name: Ruthik Jadhav
Roll No.: 321087
Gr. No.: 22120243
Division and Batch : A3

Assignment No. 7

Aim: Deploy a Node.js/Python Flask or any web application using Docker.

Theory:

What Is Docker?

Docker is an open-source containerization platform used for developing, deploying, and managing applications in lightweight virtualized environments called containers.

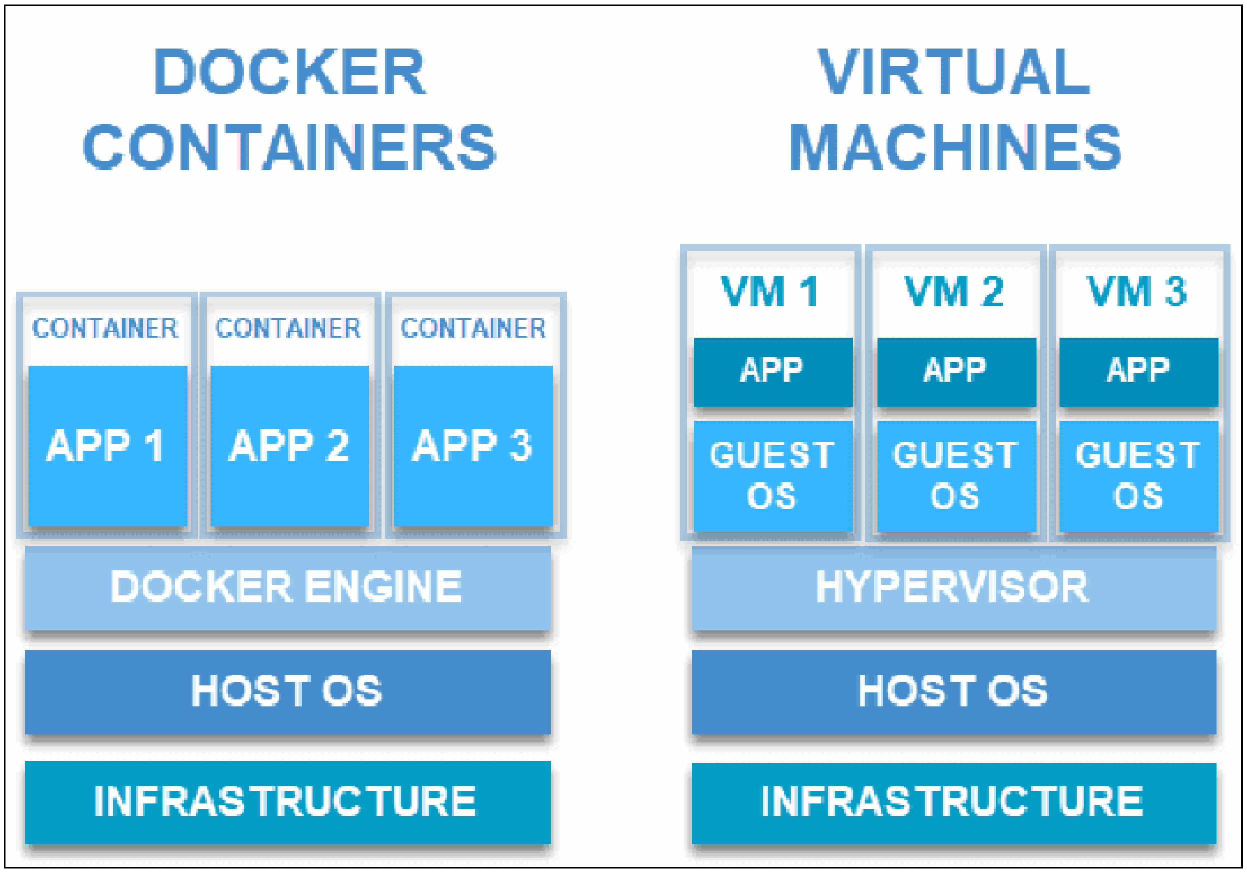
It is mainly used as a software development platform for developing distributed applications that work efficiently in different environments. By making the software system agnostic, developers don't have to worry about compatibility issues. Packaging apps into isolated environments (containers) also makes it easier to develop, deploy, maintain, and use applications.

Since Docker utilizes virtualization to create containers for storing apps, the concept may seem similar to virtual machines. Although both represent isolated virtual environments used for software development, there are important differences between containers and VMs. The most crucial distinction is that Docker containers are lighter, faster, and more resource efficient than virtual machines.

What Are Containers?

Docker containers are lightweight virtualized runtime environments for running applications. Each container represents a package of software that contains code, system tools, runtime, libraries, dependencies, and configuration files required for running a specific application. They are independent and isolated from the host and other instances running on the host. Containers are based on Docker images. You build a container by running an image on the Docker Engine. As these are the most common Docker terms, make sure you understand the difference between Docker images and Docker containers.

The same hardware can host multiple containers. Unlike virtual machines, containers virtualize at the application level. Therefore, they share the OS kernel with the host and virtualize an operating system on top of it. This means you use less resources and maintain lightweight virtual environments that are quick and easy to configure.



What Is Docker Used For?

Docker is used for:

• Running multiple workloads on fewer resources.

• Isolating and segregating applications.

• Standardizing environments to ensure consistency across development and release cycles.

• Streamlining the development lifecycle and supporting CI/CD workflows.

• Developing highly portable workloads that can run on multi-cloud platforms.

Additionally, it is used as:

• A cost-effective alternative to virtual machines.

• A version control system for an application.

Docker Core Components

The tool consists of multiple components, each playing an important role in the platform.

- Docker Engine

The Docker Engine (DE) is installed on the host machine and represents the core of the Docker system. It is a lightweight runtime system and the underlying client-server technology that creates and manages containers.

Docker Engine consists of three components:

• Server - the Docker daemon (dockerd), which is responsible for creating and managing containers.

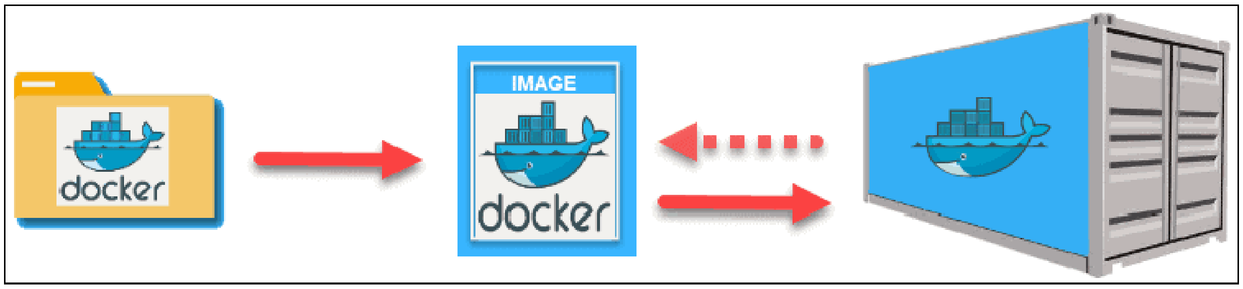
• Rest API - establishes communication between programs and Docker and instructs dockerd what to do.

• Command Line Interface (CLI) - used for running Docker commands.

- Docker Images

Docker images are templates used for building containers. Like snapshots for virtual machines, Docker images are immutable, read-only files that consist of the source code, libraries, dependencies, tools, and any other files necessary for running an application. Each image is created from a Dockerfile, which contains specific instructions for building a particular Docker image.

Once you master creating Docker images from Dockerfiles, you can build images and custom containers are simpler and faster.

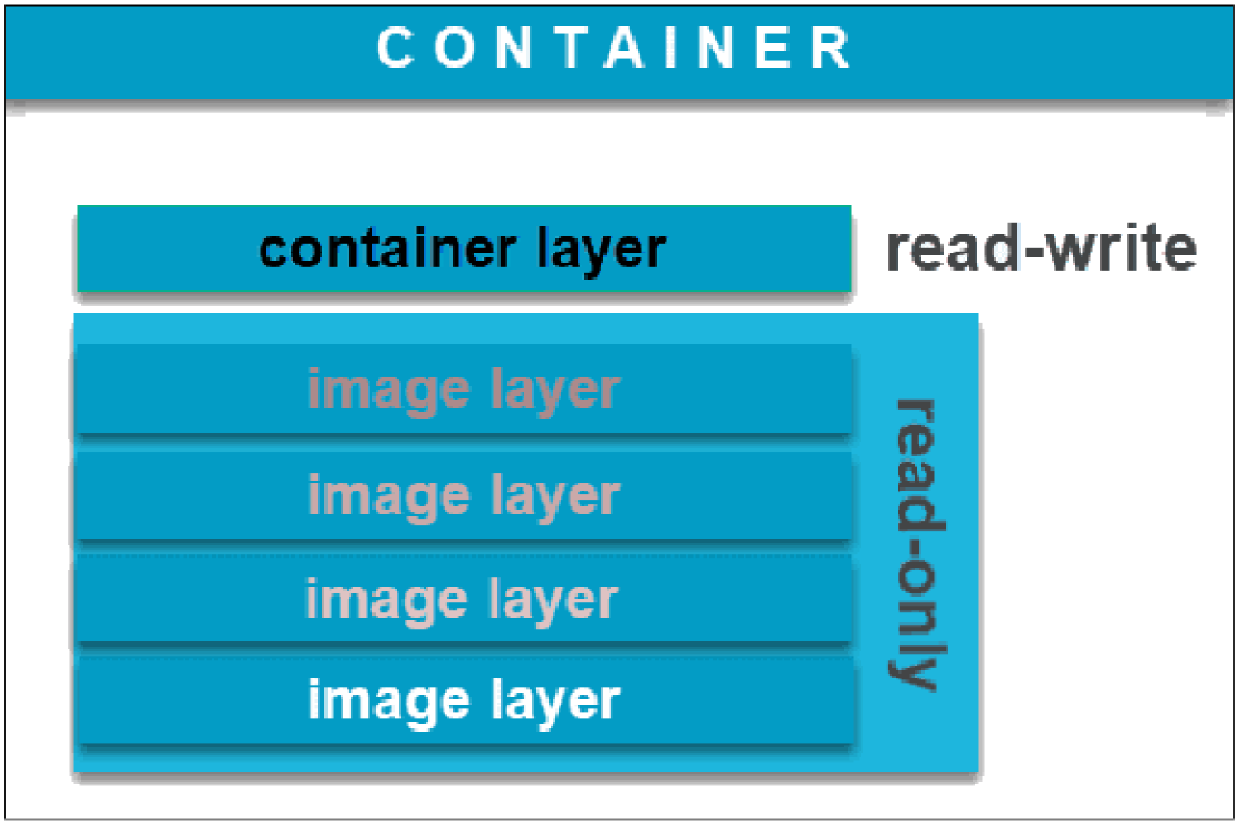


Apart from speeding up Docker builds, images are also useful for increasing reusability and essentially decreasing disk use. Since you want to keep your containers lightweight and fast, it is vital to maintain small images. Utilizing a lighter image base, avoiding unnecessary layers, and using the .dockerignore file are just a few ways of keeping your Docker images small.

Dockerfile

A Dockerfile is a script that consists of a set of instructions on how to build a Docker image. These instructions include specifying the operating system, languages, Docker environment variables, file locations, network ports, and other components needed to run the image. All the commands in the file are grouped and executed automatically.

An image has multiple layers. Once you run a Docker image to create a container, a new read-write layer is added. This is sometimes referred to as the container layer. The additional layer allows you to make changes to the base image, which you can commit to create a new Docker image for future use.



Docker Hub

Docker Hub is the largest cloud-based repository of container images provided by Docker. It supplies over 100,000 images available for use created by open-source projects, software vendors, and the Docker community.

The platform allows you to ship your applications anywhere quickly, collaborate with teammates, and automate builds for faster integration to a development pipeline.

Like GitHub, developers push and pull container images from Docker Hub and decide whether to keep them public or private

Docker Volumes

Instead of adding new layers to an image, a better solution to preserve data produced by a running container is using Docker volumes. This helpful tool allows users to save data, share it between containers, and mount it to new ones. Docker volumes are independent of the container life cycle as they are get stored on the host.

There are different ways to create and mount a Docker volume while launching a container. Learn more in Docker Volumes: How to Create & Get Started.

Docker Compose

When running and managing multiple containers simultaneously, Docker Compose is a useful tool designed to simplify the process. It strings multiple containers needed to work together and controls them through a single coordinated command.

Docker Compose is used to launch, execute, communicate, and close containers with a command. This is done using a YAML file which configures the application's services.

5

Docker Desktop

Docker Desktop, formerly known as Docker for Windows and Docker for Mac, is an application that allows you to start creating and running containers on Windows and Mac within minutes. It is a simple way of installing and setting up the entire Docker development environment. It includes Docker Engine, Docker Compose, Docker CLI client, Docker Content

Trust, Kubernetes, and Credential Helper.

The tool is used for building and sharing containerized applications and microservices in multiple languages and frameworks, on any cloud platform.

Docker Advantages

• Consistency. Docker ensures reliability that your app runs the same across multiple environments. Developers working on different machines and operating systems can work together on the same application without environment issues.

• Automation. The platform allows you to automate tedious, repetitive tasks and schedule jobs without manual intervention.

• Faster deployments. Since containers virtualize the OS, there is no boot time when starting up containers instances. Therefore, you can do deployments in a matter of seconds. Additionally, you can share existing containers to create new applications.

• Support of CI/CD. Docker works well with CI/CD practices as it speeds up deployments, simplifies updates, and allows teammates to work efficiently together.

• Rollbacks and image version control. A container is based on a Docker image which can have multiple layers, each representing changes and updates on the base. Not only does this feature speed up the build process, but it also provides version control over the container. This allows developers to roll back to a previous version if the need arises.

• Modularity. Containers are independent and isolated virtual environments. In a multi-container application, each container has a specific function. By segregating the app, developers can easily work on a particular part without taking down the entire app.

• Resource and cost-efficiency. As containers do not include guest operating systems, They are much lighter and smaller than VMs. They take up less memory and reuse components thanks to data volumes and images. Also, containers don't require large physical servers as they can run entirely on the cloud.

Docker Disadvantages

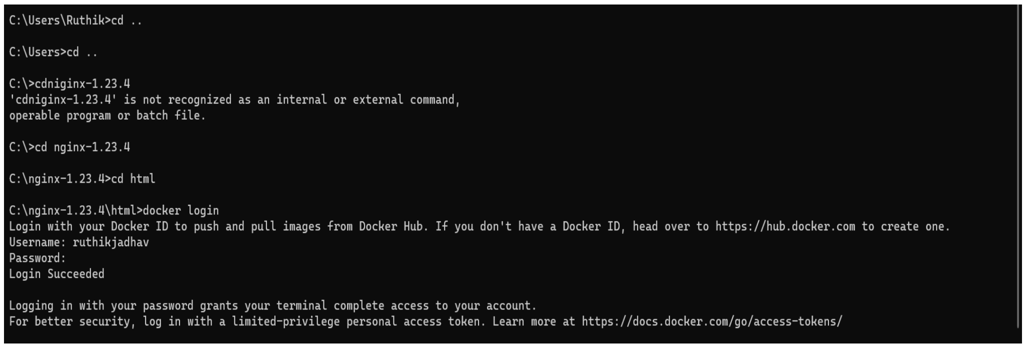
• No graphical interface. Docker is not the best choice if you want to run apps that require a graphical interface. It is mainly for hosting applications that run on the command line.
• Security issues. Although Docker provides security by isolating contains from the host and each other, there are certain Docker-specific security risks. Many potential security issues may arise while working with containers, so make sure to adopt best Docker security practices that can help you prevent attacks and privilege breaches.
• Learning curve. Even developers experienced with the VM infrastructure need some time to get used to Docker concepts and how they work. If switching to Docker, make sure to take into account the necessary learning curve.

IMPLEMENTATION

Step 1 - Create a Directory for the Website

Make sure that you have your HTML files already in the current directory.

Step 2 – Install Docker and nginx and check if it is installed properly or not.

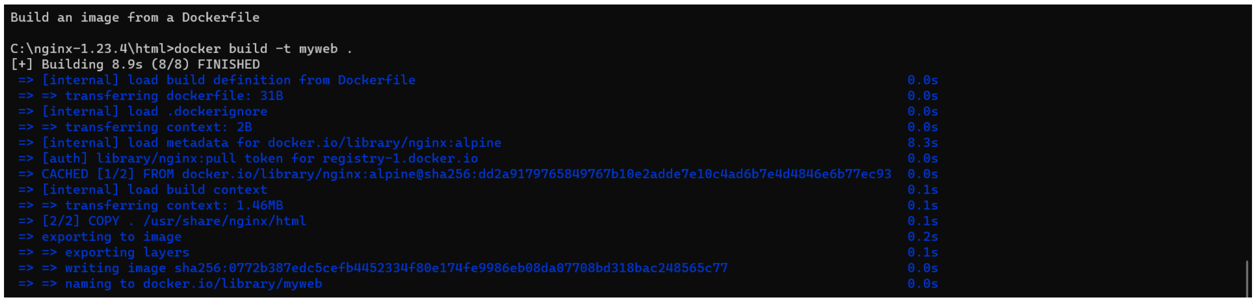


```
C:\Users\Ruthik>cd ..

C:\Users>cd ..

C:\>cdniginx-1.23.4
'cdniginx-1.23.4' is not recognized as an internal or external command,
operable program or batch file.

C:\>cd nginx-1.23.4

C:\nginx-1.23.4>cd html

C:\nginx-1.23.4\html>docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: ruthikjadhav
Password:
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/
```

Step 3 - Build the Docker Image for the HTML Server

Run the following command:

docker build -t myweb .



```
Build an image from a Dockerfile

C:\nginx-1.23.4\html>docker build -t myweb .
[+] Building 8.9s (8/8) FINISHED
 => [internal] load build definition from Dockerfile                                                      0.0s
 => => transferring dockerfile: 31B                                                                       0.0s
 => [internal] load .dockerignore                                                                         0.0s
 => => transferring context: 2B                                                                           0.0s
 => [internal] load metadata for docker.io/library/nginx:alpine                                           8.3s
 => [auth] library/nginx:pull token for registry-1.docker.io                                              0.0s
 => CACHED [1/2] FROM docker.io/library/nginx:alpine@sha256:dd2a9179765849767b10e2adde7e10c4ad6b7e4d4846e6b77ec93  0.0s
 => [internal] load build context                                                                         0.1s
 => => transferring context: 1.46MB                                                                       0.1s
 => [2/2] COPY . /usr/share/nginx/html                                                                    0.1s
 => exporting to image                                                                                    0.2s
 => => exporting layers                                                                                   0.1s
 => => writing image sha256:0772b387edc5cefb4452334f80e174fe9986eb08da07708bd318bac248565c77              0.0s
 => => naming to docker.io/library/myweb                                                                  0.0s
```

Step 4 -

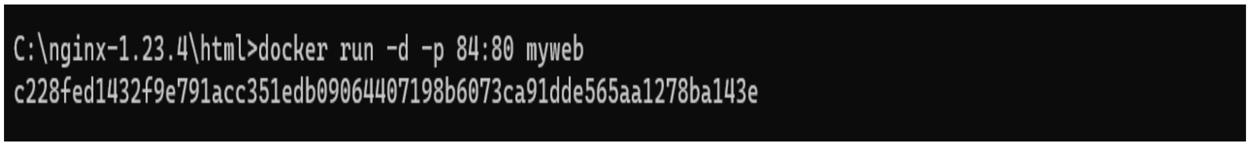
Run the Docker Container

Run the following command to run the HTML container server:

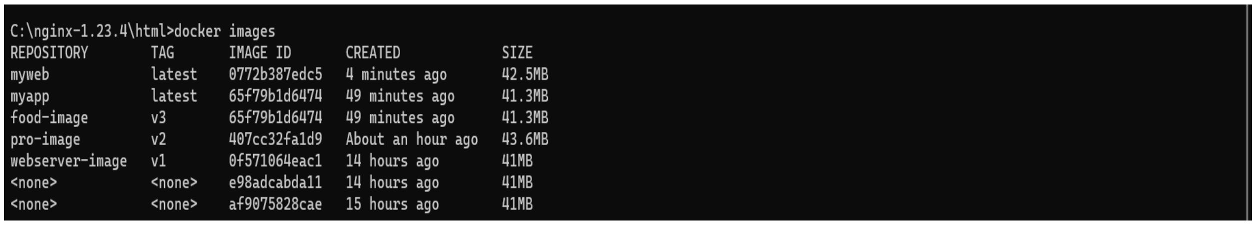
docker run -d -p 84:80 myweb

You can confirm that this has worked by running the command:

docker images

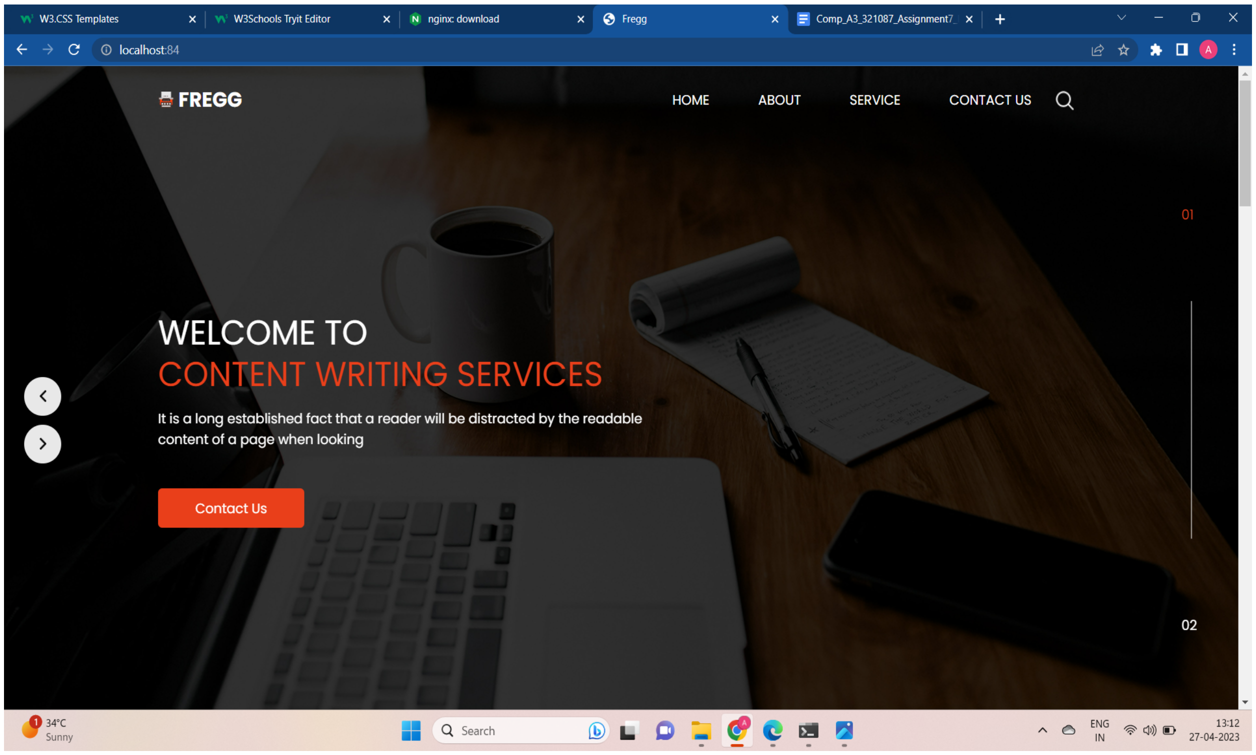


```
C:\nginx-1.23.4\html>docker run -d -p 84:80 myweb
c228fed1432f9e791acc351edb09064407198b6073ca91dde565aa1278ba143e
```



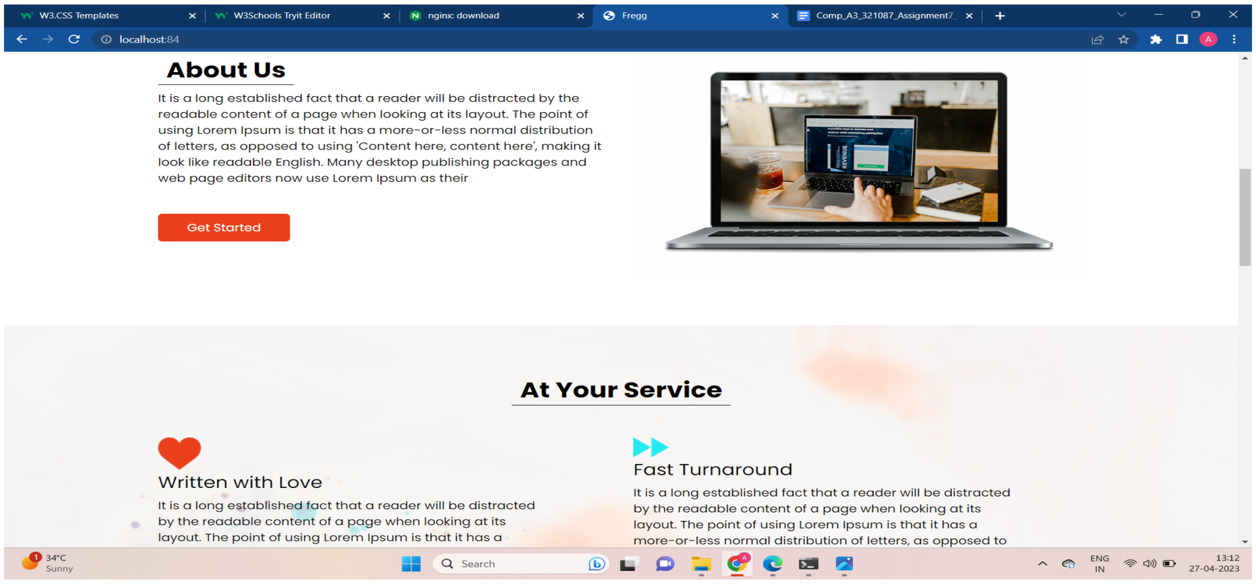
```
C:\nginx-1.23.4\html>docker images
REPOSITORY         TAG       IMAGE ID       CREATED            SIZE
myweb              latest    0772b387edc5   4 minutes ago      42.5MB
myapp              latest    65f79b1d6474   49 minutes ago     41.3MB
food-image         v3        65f79b1d6474   49 minutes ago     41.3MB
pro-image          v2        407cc32fa1d9   About an hour ago  43.6MB
webserver-image    v1        0f571064eac1   14 hours ago       41MB
<none>             <none>    e98adcabda11   14 hours ago       41MB
<none>             <none>    af9075828cae   15 hours ago       41MB
```

Step 5 – Check if the website is running or not on the specified host.



WELCOME TO
CONTENT WRITING SERVICES

It is a long established fact that a reader will be distracted by the readable content of a page when looking



About Us

It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their

At Your Service

Written with Love

It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a

Fast Turnaround

It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to

Conclusion:

Thus, I’ve successfully used docker and deployed a web app using docker.