# Placement Empowerment Program

***Cloud Computing and DevOps Centre***

Build and Run a Custom Docker Image: Create a Dockerfile to package your static website into a Docker container and run it locally.

Name: Yasvanth Rajan E Department:AML



# Introduction

With the increasing adoption of containerization in modern software development, **Docker** has become a key technology for packaging applications in a consistent and portable way. In this Proof of Concept (POC), we explore how to **containerize a static website** using **Docker and Nginx**.

By creating a **Docker image** for our website, we ensure that it can run consistently across different environments without worrying about dependencies, configurations, or setup issues. This POC is especially useful for developers and DevOps engineers who want to deploy static sites in a **lightweight and efficient manner**.

# Overview

This POC demonstrates how to:

1. Create a **Dockerfile** to define a containerized static website.
2. Use **Nginx** as a web server to serve the website inside a container.
3. Build a **Docker image** for the static site.
4. Run a **Docker container** to host and test the website locally.

By the end of this POC, we will have a working **Dockerized static website** that can be easily deployed and shared.

The key goals of this POC are:

* 1. **Understand the basics of Docker and Dockerfiles.**
  2. **Learn how to use Nginx to serve static files inside a container.**
  3. **Practice building and running Docker containers for web applications.**
  4. **Ensure the website runs consistently across different systems.**
  5. **Prepare for real-world deployment scenarios using containerized environments.**

# Importance

1. **Portability:** The website runs the same way on any system with Docker installed.
2. **Consistency:** No dependency issues since everything is inside the container.
3. **Fast Deployment:** Running the website takes just a few commands.
4. **DevOps Skill Development:** Provides hands-on experience with Docker, an essential tool in DevOps.
5. **Scalability:** Can be extended for cloud deployments using AWS, Azure, or Kubernetes.

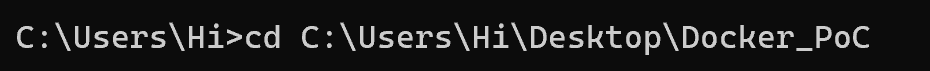
## Step 1:

Create a folder (Docker\_PoC)

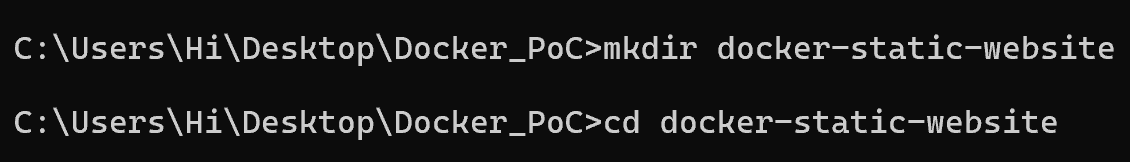


## Step 2:

Open Command Prompt and navigate to the folder which is created.



## Step 3:

Create a new Directory **mkdir docker-static-website cd docker-static-website**

Create a Folder for Your Static Website

**mkdir html**



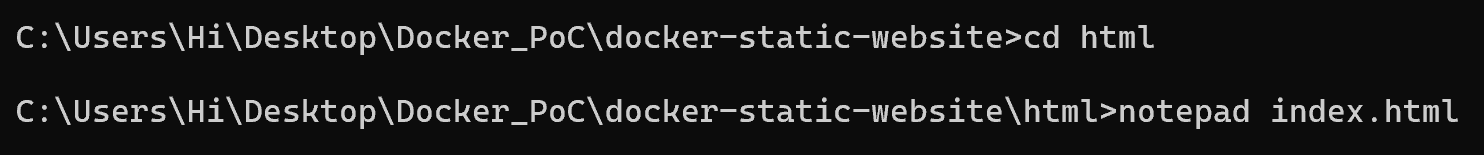
## Step 5:

Create a Simple index.html File

Inside html, create a new file named index.html:

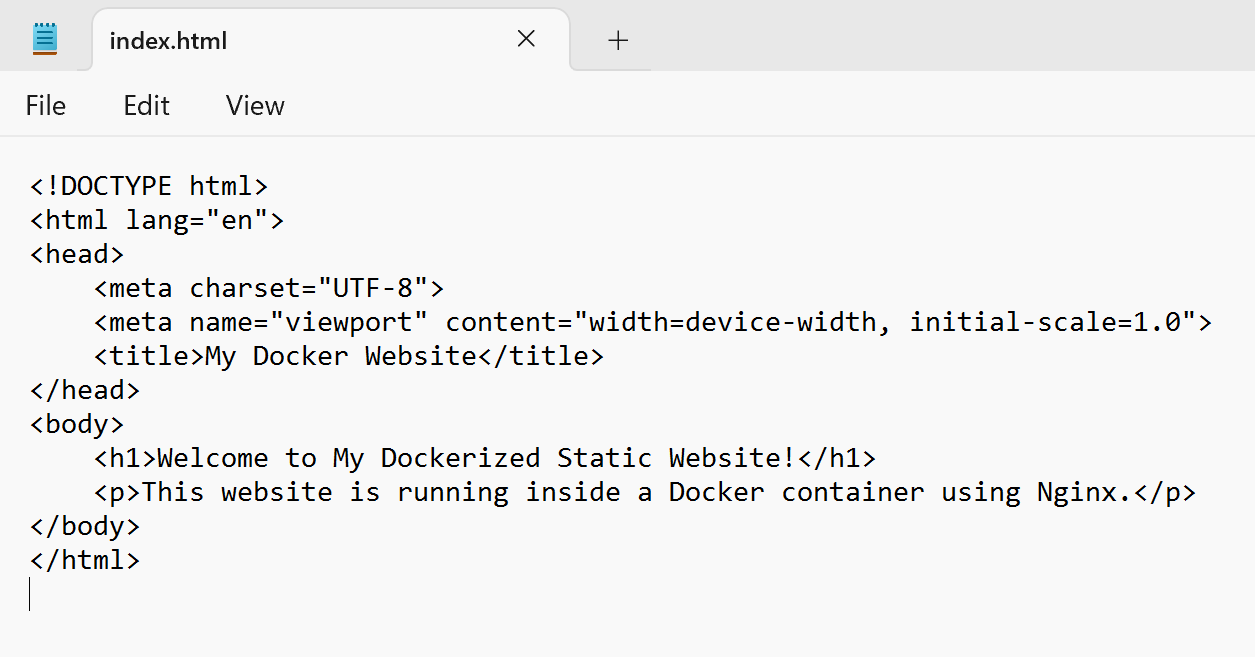
**cd html**

**notepad index.html**



## Step 6:

Add the following simple HTML code:

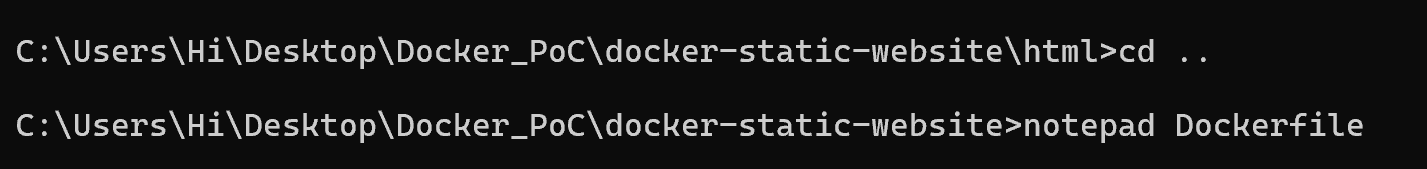


Go Back to the Main Project Folder

**cd ..**

Create a New File Named Dockerfile

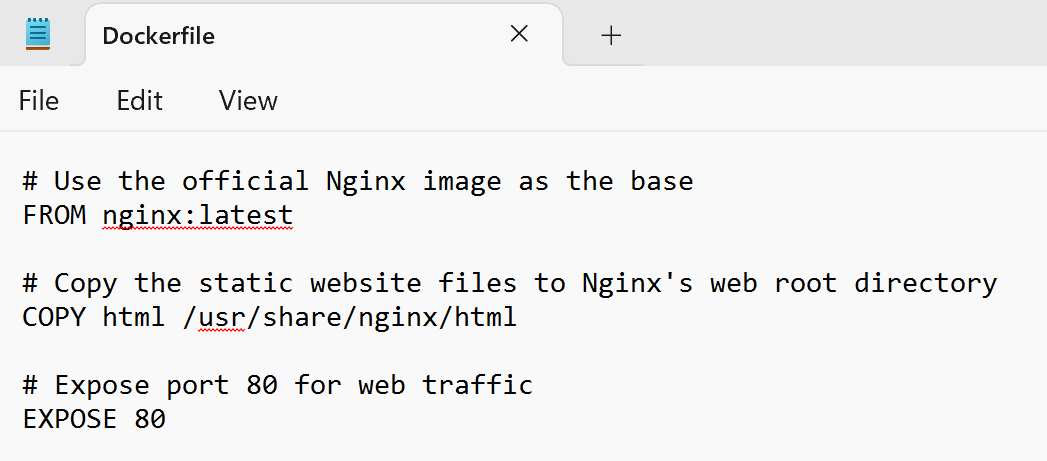
**notepad Dockerfile**



## Step 8:

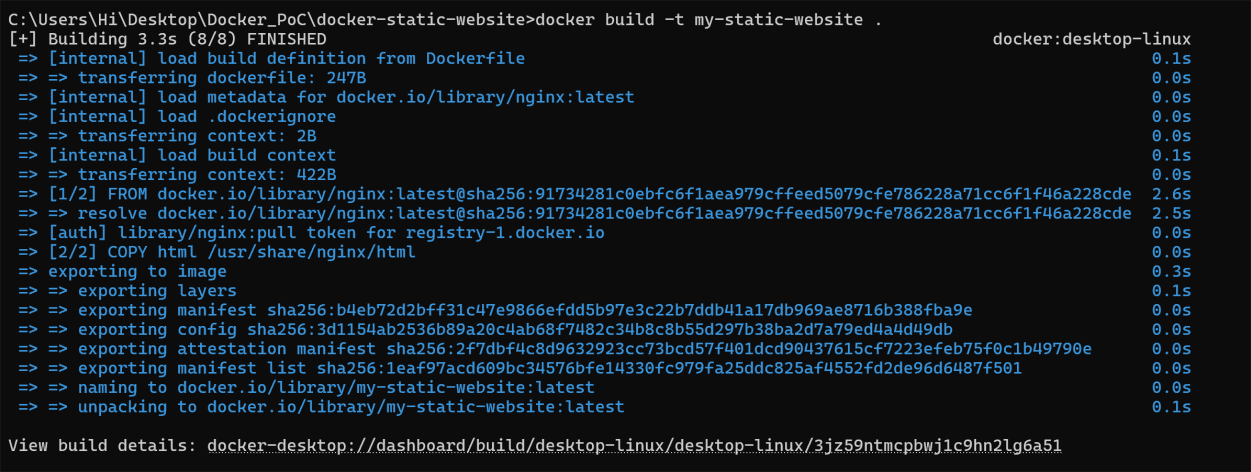
Add the Following Content to the Dockerfile Click **File → Save**

Close Notepad



Build the Docker Image

**docker build -t my-static-website .**

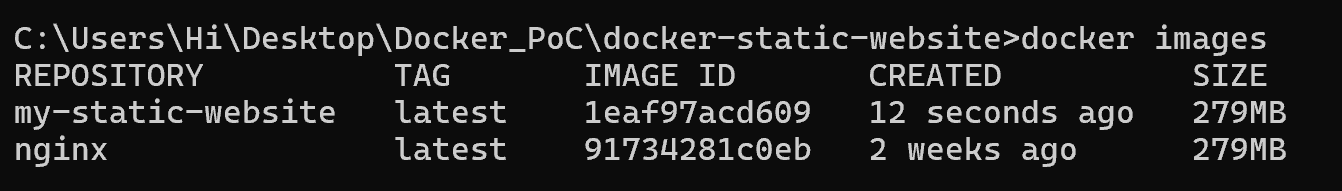


## Step 10:

Once the build is complete, check if the image was created successfully:

**docker images**

You should see a list of Docker images, including **my-static-website**.

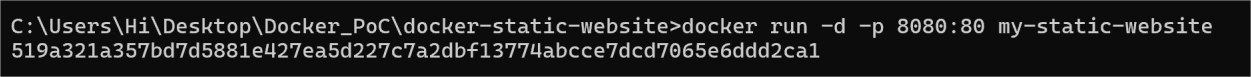


Now, we will create and start a container from the **my-static-website**

image.

Run the Container :

**docker run -d -p 8080:80 my-static-website**



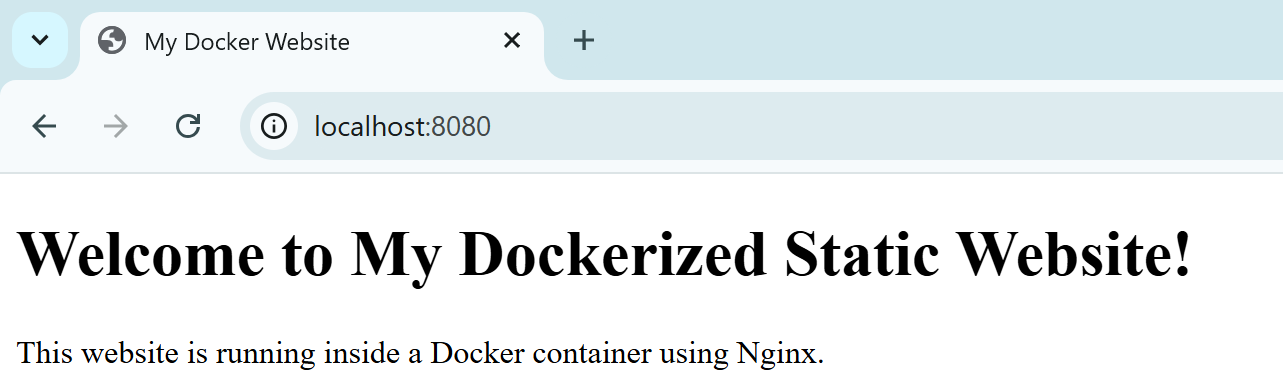
## Step 12:

Test the Website

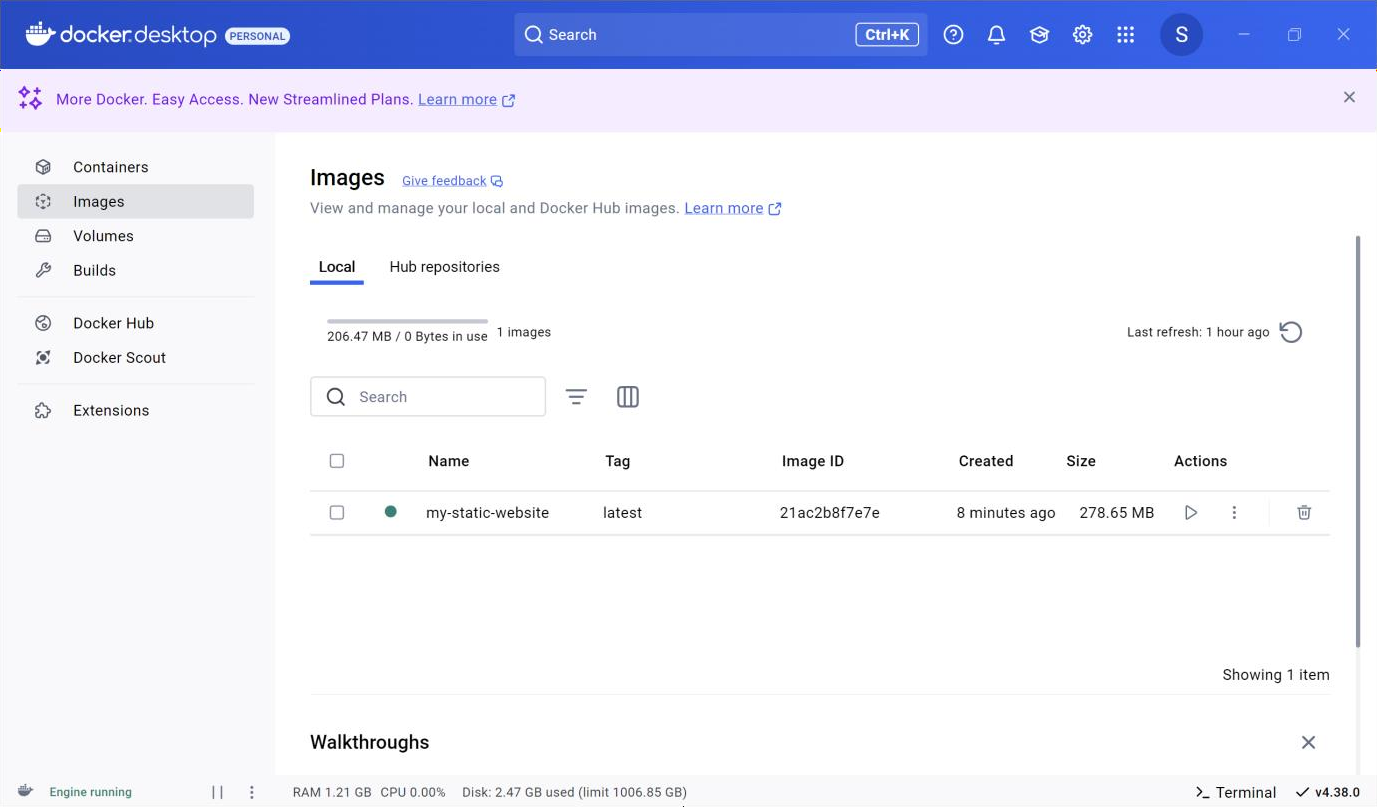
Open your browser and visit:

[**http://localhost:8080**](http://localhost:8080/)

If everything is correct, you should see **your static website running!**



You can also see the Docker Images in Docker Desktop.

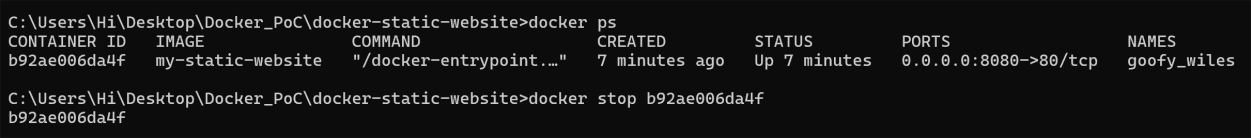


## Step 14:

Stop and Remove the Container (Optional) If you want to stop the running container: **docker ps # Get the container ID docker stop <container\_id>**

To remove the container:

**docker rm <container\_id>**



# Outcomes

By completing this POC, you will:

1. **Create and Configure a Dockerfile** – Learn to define a containerized static website using Dockerfile commands.
2. **Build a Docker Image** – Package the static website into a Docker image using docker build.
3. **Run a Docker Container** – Deploy the website inside a container using Nginx as the web server.
4. **Expose and Access the Website** – Map ports to access the running container via a web browser.
5. **Manage Docker Containers** – Use essential Docker commands to start, stop, and remove containers.
6. **Understand Containerization Benefits** – Explore how Docker simplifies deployment, improves portability, and streamlines DevOps workflows.