**Docker Worksheet – Part 2**

**Docker Images and Dockerfile**

**Objective**

By the end of this session, you will:

* Create a Dockerfile and build a custom image
* Understand tagging, volumes, and port mapping
* Push your image to DockerHub and pull it back

**Part 1: Dockerfile Basics**

**Step 1: Create Project Folder**

mkdir my-nginx

cd my-nginx

**Step 2: Create an HTML file**

Create a file called index.html with this content:

<h1>Hello from my custom Docker image!</h1>

**Step 3: Create a Dockerfile**

In the same folder, create a file named Dockerfile (no extension) with:

FROM nginx:alpine

COPY index.html /usr/share/nginx/html/index.html

This means:

* Start from a lightweight NGINX image
* Copy your custom index.html into the default web server directory

**Part 2: Build and Tag the Custom Image**

**Step 4: Build the Docker Image**

docker build -t my-nginx-image .

**What this does:**

* -t my-nginx-image: Tags the image name
* .: Build context is the current directory

You’ll see a success message like: Successfully built <image\_id>

**Part 3: Run and Test the Custom Image**

**Step 5: Run the Image with Port Mapping**

docker run -d -p 8081:80 my-nginx-image

Now open your browser and visit:

http://localhost:8081

You should see your custom “Hello from my custom Docker image!” page.

**Part 4: Volumes & Live Editing**

**Step 6: Mount Local HTML File**

docker run -d -p 8082:80 -v $(pwd)/index.html:/usr/share/nginx/html/index.html nginx

Now:

* Edit index.html locally.
* Refresh http://localhost:8082 — changes will reflect live.

This demonstrates **volumes** and how containers can use external files.

**Part 5: DockerHub Push & Pull**

**Step 7: Tag the Image for DockerHub**

docker tag my-nginx-image YOUR\_DOCKERHUB\_USERNAME/my-nginx:v1

**Step 8: Log in and Push**

docker login

docker push YOUR\_DOCKERHUB\_USERNAME/my-nginx:v1

Once pushed, your image is public (unless you use a private repo).

**Step 9: Pull and Run on Any Machine**

docker pull YOUR\_DOCKERHUB\_USERNAME/my-nginx:v1

docker run -d -p 8083:80 YOUR\_DOCKERHUB\_USERNAME/my-nginx:v1

You should see the same webpage from the cloud-hosted version of your image.