Placement Empowerment Program

Cloud Computing and DevOps Centre

BUILD AND RUN A CUSTOM DOCKER IMAGE

Name: Shalini D Department: IT

Introduction

Containerization has revolutionized the way applications are developed, deployed, and managed. Docker provides a lightweight and efficient way to package applications with all dependencies, ensuring consistency across different environments. In this POC, we will build and run a custom Docker image that serves a simple HTML web page using NGINX.

Overview :

This POC demonstrates how to: 1. Create a Dockerized static website using NGINX. 2. Build a custom Docker image with an HTML file. 3. Run a Docker container to serve the HTML content. 4. Access the webpage via http://localhost:8080.

Step-by-Step Overview

Step 1:

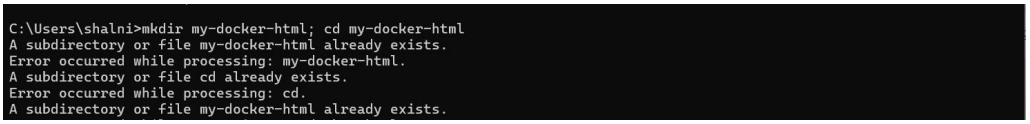
Install Docker (If Not Installed)

1. Open Command Prompt (cmd) and check if Docker is installed docker - -version

Step 2

Create a Project Directory Open Command Prompt and run: mkdir my-

docker-html && cd mydocker-html mkdir my-docker-html → Creates a new directory. cd my-docker-html → Moves inside the directory.



Step 3

Create an HTML File (index.html) Run this command to create an empty index.html file: type nul > index.html Open the file in Notepad: notepad index.html Add the following content and save the file

Step 4

Create a Dockerfile Run this command to create an empty Dockerfile: type nul > Dockerfile Open the file in Notepad: notepad Dockerfile Add the following content and save the file:

```
C:\Users\shalni>type nul>Dockerfile

C:\Users\shalni>notepad Dockerfile
```

```
FROM nginx:latest

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

CMD ["nginx", "-g", "daemon off;"]
```

Step 5:

Build the Docker Image Run the following command inside the mydocker-html directory: docker build -t my-html-image. -t my-html-image → Names the image myhtml-image. → Uses the current directory (where the Dockerfile is located).

```
Usage:  docker buildx build [OPTIONS] PATH | URL | -

Start a build

C:\Users\shalni>docker build -t my-html-image .
[+] Building 5.1s (8/8) FINISHED                                  docker:desktop-linux
 => [internal] load build definition from Dockerfile                              0.3s
 => => transferring dockerfile: 144B                                              0.1s
 => [internal] load metadata for docker.io/library/nginx:latest                   0.1s
 => [internal] load .dockerignore                                                 0.1s
 => => transferring context: 2B                                                   0.0s
 => [internal] load build context                                                 0.1s
 => => transferring context: 357B                                                 0.0s
 => [1/2] FROM docker.io/library/nginx:latest@sha256:9d6b58feebd2dbd3c56ab5853333d6  3.2s
 => => resolve docker.io/library/nginx:latest@sha256:9d6b58feebd2dbd3c56ab5853333d6  2.9s
 => [auth] library/nginx:pull token for registry-1.docker.io                      0.0s
 => [2/2] COPY index.html /usr/share/nginx/html/index.html                        0.1s
 => exporting to image                                                            0.8s
 => => exporting layers                                                           0.3s
 => => exporting manifest sha256:1120b1a6855cb0fc535ada95ebbcecbd0ceab7890b744a1c58  0.0s
 => => exporting config sha256:c8f9d7359208f05789d59b808fb0dc2f8c0d3d361925d4159855  0.0s
 => => exporting attestation manifest sha256:0728f7fb2ae9bc96f811005fe067fd0f0247ac  0.1s
 => => exporting manifest list sha256:a6bb96cdaf92f2414d5b5bb82b9866e5cd1b9cf13b2af  0.0s
 => => naming to docker.io/library/my-html-image:latest                           0.0s
 => => unpacking to docker.io/library/my-html-image:latest                        0.2s

C:\Users\shalni>
```

Step 6:

Run the Docker Container Run this command to start a container from the image and expose it on port 8080: docker run -d -p 8080:80 my-html-image -d → Runs the container in detached mode (background). -p 8080:80 → Maps port 8080 on your computer to port 80 inside the container.

```
C:\Users\shalni>docker run -d -p 8080:80 my-html-image
23573e3c39ce4cd94f0bd1f8d2378ca2a6dac17e626e232c1ac192180a9bd0b8

C:\Users\shalni>
```

Step 7:

View the Web Page in a Browser Open a browser and go to: http://localhost:8080 You should see the "Hello from Docker!" message.

Outcome:

Successfully built and ran a Docker container hosting an HTML webpage. Accessed the webpage using http://localhost:8080. Understood how Docker images and containers work for web applications. Gained practical experience in containerization using Docker.