

DevOps Lab Program 1

Title: Build a Docker Container from a Custom Dockerfile

Objective:

To create a custom Docker image from a user-provided Dockerfile (based on Ubuntu) and run it inside a Docker container, verifying the build and run process.

Software Requirements:

- Docker (installed on host)
- (Optional) VS Code for editing files

Files Used:

1. Dockerfile – Custom Dockerfile based on ubuntu:latest

Screenshot:

The screenshot shows a dark-themed interface of VS Code. On the left, the Explorer sidebar shows a project named 'FIRST' containing a 'Dockerfile' and a file named 'sampl.txt'. The main editor area displays a Dockerfile with the following content:

```
1 #Getting latest ubuntu
2 FROM ubuntu:latest
3
4 #Information about the author
5 LABEL maintainer="wwwshreya19@gmail.com"
6
7 #Label version
8 LABEL version="1.0"
9
10 #Date of creation
11 LABEL date="2024-06-10"
12
13 #Updating the Os
14 RUN apt-get update
15
16 #Installing the curl
17 RUN apt-get install -y curl
18
19 #defining current working directory
20 WORKDIR /app
21
22 #copy all files
23 COPY ./ ./app
24
25 #Setting the environment variable
26 ENV APP_ENV=development
27
```

Below the editor, the Terminal tab is active, showing the command-line output of a Docker build:

```
● IRV24MC100_Shreya@ivobook:~/first$ sudo docker build -t p1 .
[sudo] password for IRV24MC100 Shreya:
[+] Building 8.8s (11/11) FINISHED
=> [internal] Load build definition from Dockerfile
=> => transferring dockerfile: 655B
=> [internal] Load metadata for docker.io/library/ubuntu:latest
=> [internal] Load .dockerrcignore
=> => transferring context: 2B
=> [internal] Load build context
=> => transferring context: 588
=> CACHED [2/6] RUN apt-get update
=> CACHED [3/6] RUN apt-get install -y curl
=> CACHED [4/6] WORKDIR /app
=> CACHED [5/6] COPY ./ ./app
=> CACHED [6/6] RUN echo "Dockerfile execution completed successfully"
=> exporting to image
=> => exporting layers
=> => writing image sha256:8d8e329a5614b203725c455c1655ce1df9c7ac3210031ff814f24974548b545d
=> => naming to docker.io/library/p1
● IRV24MC100_Shreya@ivobook:~/first$ sudo docker run -d -p 8080:8080 p1
8f6c29125ea853793f93196ffb4fd32ad13da461affe81916cd2d4291b0e4ce
```

Procedure:

1. Create project folder and place Dockerfile and any app files inside it.
2. Open a terminal in the project folder.
3. Build the Docker image:
 `sudo docker build -t p1 .`
4. Run the Docker container:
 `sudo docker run -d -p 8080:8080 p1`
5. Verify running containers:
 `sudo docker ps`
6. Access application at:
 `http://localhost:8080`

Dockerfile Summary:

- FROM ubuntu:latest
- LABEL maintainer and version metadata
- RUN apt-get update && apt-get install -y curl
- WORKDIR /app
- COPY . /app
- ENV APP_ENV=development
- RUN echo "Dockerfile execution completed successfully"

Result:

A Docker image named 'p1' was built successfully and a container was started with port mapping 8080:8080.

Conclusion:

This exercise demonstrates how to create a Docker image from a custom Dockerfile and run it as a container.

Document generated on: 2025-11-04 19:21:05