

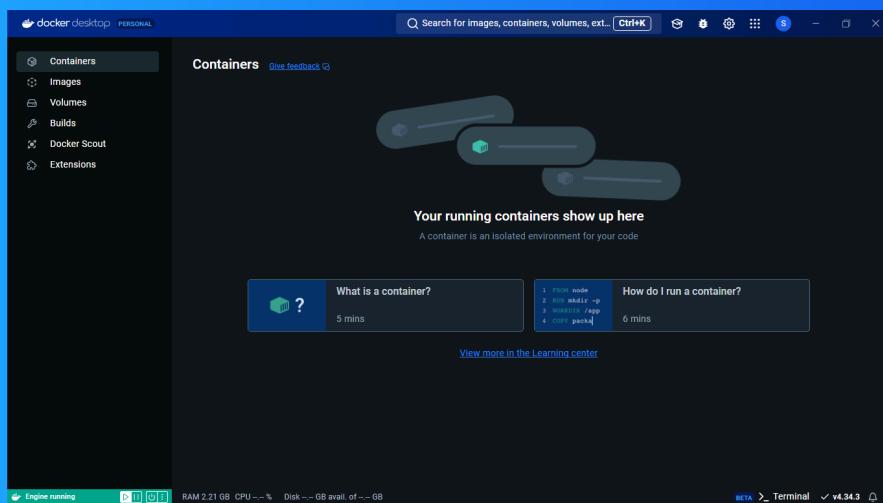


NextWork.org

Containers on Elastic Beanstalk



Sibdou Ibrahim Issifu





Sibdou Ibrahim Issifu

NextWork Student

NextWork.org

Introducing Today's Project!

What is Docker?

Docker is a platform for packaging applications into containers. In today's project, I used Docker with AWS Elastic Beanstalk to deploy and manage our application, simplifying scaling and deployment.

One thing I didn't expect...

The anticipated errors.

This project took me...

About an hour and half.



Sibdou Ibrahim Issifu

NextWork Student

NextWork.org

Understanding Containers and Docker

Containers

A container in Docker is a lightweight, portable, and isolated environment that packages an application and all its dependencies, allowing it to run consistently across different systems.

A container image is a lightweight, standalone, and executable package that includes everything needed to run software: code, runtime, libraries, and system tools. It serves as a blueprint for creating containers.

Docker

Docker is a platform for creating, deploying, and managing containers. Docker Desktop is an application that provides an easy-to-use interface for developing and running containers on a local machine.

The Docker daemon is a background service that manages Docker containers on a host. It listens to API requests, builds, runs, and monitors containers, and communicates with Docker clients to manage container lifecycle and resources efficiently.



Sibdou Ibrahim Issifu
NextWork Student

NextWork.org

Running an Nginx Image

Nginx is an open-source web server and reverse proxy server designed for high performance, scalability, and security. It efficiently handles HTTP traffic, serves static content, balances loads, and can also act as a mail proxy server.

Start your answer with 'The command I ran to start a new container was:
docker run -d -p 80:80 nginx.

The screenshot shows a Windows Command Prompt window and a Microsoft Edge browser window side-by-side. The Command Prompt window displays the output of running Docker commands to pull the Nginx image and start a new container. The browser window shows the 'Welcome to nginx!' landing page, indicating the successful installation and configuration of the Nginx web server.

```
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.

C:\Users\sibdo>docker --version
Docker version 27.2.0, build 3ab4256

C:\Users\sibdo>docker run -d -p 80:80 nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
11dd6fdd0e8a7: Download complete
70850b3ec6b2: Download complete
a480a496ba95: Download complete
40ea07b53d8: Download complete
f3ace1b8ce45: Download complete
f1091daefdf5c: Download complete
6d76794e50f4: Download complete
Digest: sha256:28402db69fec7c17e179ea87882667f1e054391138f77ffaf0c3eb388
efc3Ffb
Status: Downloaded newer image for nginx:latest
65530d8289bcb558fdedd59fb217f82e59e46656e0d3b11d9c4e369eb37f556

C:\Users\sibdo>
```

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

Thank you for using nginx.



Creating a Custom Image

The Dockerfile is a text file containing a set of instructions used to build a Docker image. It defines the base image, application code, dependencies, configurations, and commands to create the final image.

My Dockerfile tells Docker three things: start with the latest Nginx image ('FROM'), copy my 'index.html' file to the default Nginx directory ('COPY'), and open port 80 for web traffic ('EXPOSE').

The command I used to build a custom image with my Dockerfile was `docker build -t my-web-app .`. The `.` at the end of the command means the build context is the current directory, where Docker will look for the Dockerfile and other resources.

```
FROM nginx:latest
COPY index.html /usr/share/nginx/html/
EXPOSE 80
```



Running My Custom Image

There was an error when I ran my custom image because another container was already using port 80. I resolved this by stopping the containing useing port 80.

In this example, the container image is the blueprint created from the Dockerfile, containing Nginx and the `index.html` file. The container is the running instance of that image, serving the `index.html` content on port 80.

The screenshot shows a Windows desktop environment. On the left, a PowerShell window titled "Windows PowerShell" displays command-line output related to Docker operations. On the right, a web browser window titled "My Web App" shows the text "Hello from Sibdou's custom Docker image!"

```
PS C:\Users\sibdo\Desktop\Compute> docker run -d -p 88:80 my-web-app
e2a6300513a39df4f25ff3966c1ad6418a4830e0c8b1c2eb28ad9728839f79u01
docker: Error response from daemon: driver failed programming external c
onnectivity on endpoint gracius_turing (42c4ice964f81574996a69559984b89
1fd4454c29972ef0f47bb3f150677be5a): Bind for 0.0.0.0:80 failed: port is
already allocated.
PS C:\Users\sibdo\Desktop\Compute> docker ps --filter "publish=80"
CONTAINER ID IMAGE COMMAND CREATED STATUS
S PORTS NAMES
65530d8289bc nginx: "docker-entrypoint..." 27 minutes ago Up 27
minutes 0.0.0.0:80->80/tcp dreamy_brown
PS C:\Users\sibdo\Desktop\Compute>

> docker run -d -p 88:80 my-web-app
cd6d6408fd4c190f30bf60b23ec619af66f60aebl17b4cd7e7feb94871b10906
PS C:\Users\sibdo\Desktop\Compute>
```



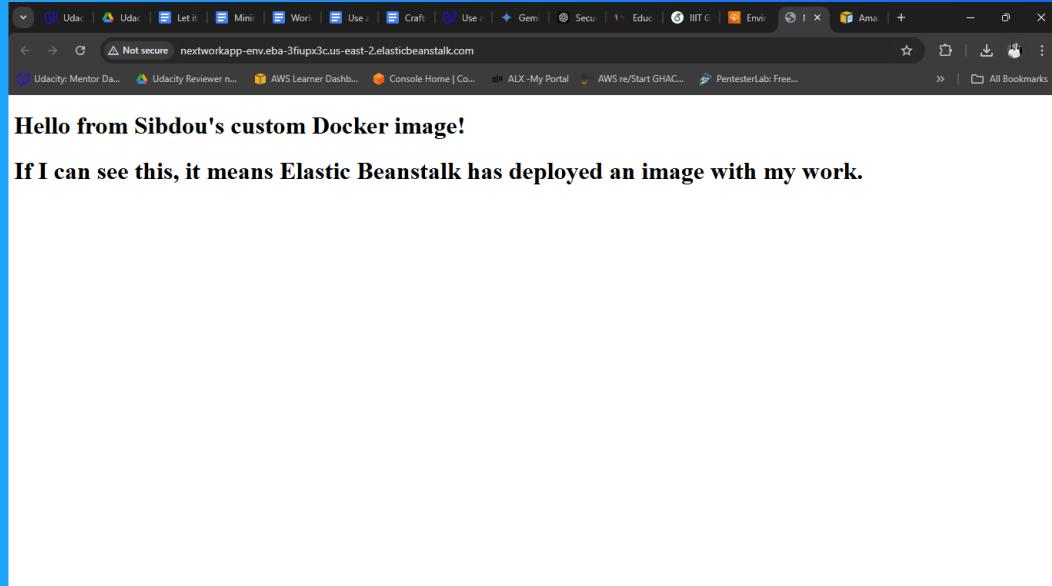
Sibdou Ibrahim Issifu
NextWork Student

NextWork.org

Elastic Beanstalk

Elastic Beanstalk is a fully managed service by AWS that simplifies deploying and scaling web applications and services. It automatically handles infrastructure provisioning, load balancing, and monitoring, allowing developers to focus on coding.

Deploying my custom image with Elastic Beanstalk took me almost an hour and half.





NextWork.org

Everyone should be in a job they love.

Check out nextwork.org for
more projects

