

## Dockerizing a simple HTML page using Nginx as the web server.

GitHub repository link: [https://github.com/ankuronlyme/Docker\\_html](https://github.com/ankuronlyme/Docker_html)

- Create a plain HTML page named index.html with some content (e.g., "Hello, Docker!").
- Nginx Configuration (nginx.conf):
- Create an Nginx configuration file named nginx.conf that serves the index.html page.
- Configure Nginx to listen on port 80.
- Create a Dockerfile to define the Docker image.
- Use an official Nginx base image.
- Copy the index.html and nginx.conf files into the appropriate location in the container.
- Ensure that the Nginx server is started when the container is run.
- Build the Docker image using the Dockerfile run command:

**docker build -t <your image name> .**

-t is denotes for giving a tag.

- . is denotes for giving the path

- After run this command build is successfully created:

```
PS C:\Users\pc\ Docker_html> docker build -t ankur_docker .
[+] Building 2.7s (9/9) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 171B
=> [internal] load metadata for docker.io/library/nginx:alpine
=> [auth] library/nginx:pull token for registry-1.docker.io
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/3] FROM docker.io/library/nginx:alpine@sha256:6a2f8b28e45c4adea04ec207a251fd4a2df03ddc930f782af51e315ebc76e9a9
=> [internal] load build context
=> => transferring context: 352B
=> CACHED [2/3] COPY index.html /usr/share/nginx/html/index.html
=> CACHED [3/3] COPY nginx.conf /etc/nginx/site-available/nginx.conf
=> exporting to image
=> => exporting layers
=> => writing image sha256:f7500e7a608179cee0d702b071ef7c435d3452e1fcce543ef53db892251e724c
=> => naming to docker.io/library/ankur_docker
```

View build details: [docker-desktop://dashboard/build/default/default/lf11ge9t9cr6sei8m2b577vvy](https://docker-desktop://dashboard/build/default/default/lf11ge9t9cr6sei8m2b577vvy)

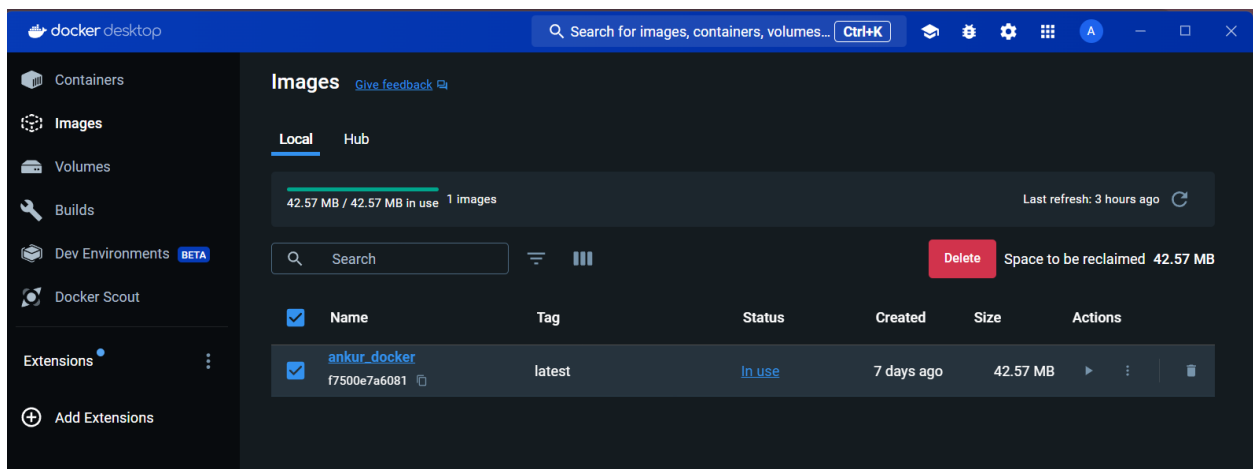
### What's Next?

View a summary of image vulnerabilities and recommendations → [docker scout quickview](#)

- Also check is image is created or not in desktop Docker run command : **docker images**

```
PS C:\Users\pc\ Docker_html> docker images
REPOSITORY      TAG         IMAGE ID      CREATED        SIZE
ankur_docker    latest     f7500e7a6081  7 days ago    42.6MB
```

Images is created in the desktop Docker



- To run image into the container execute command:

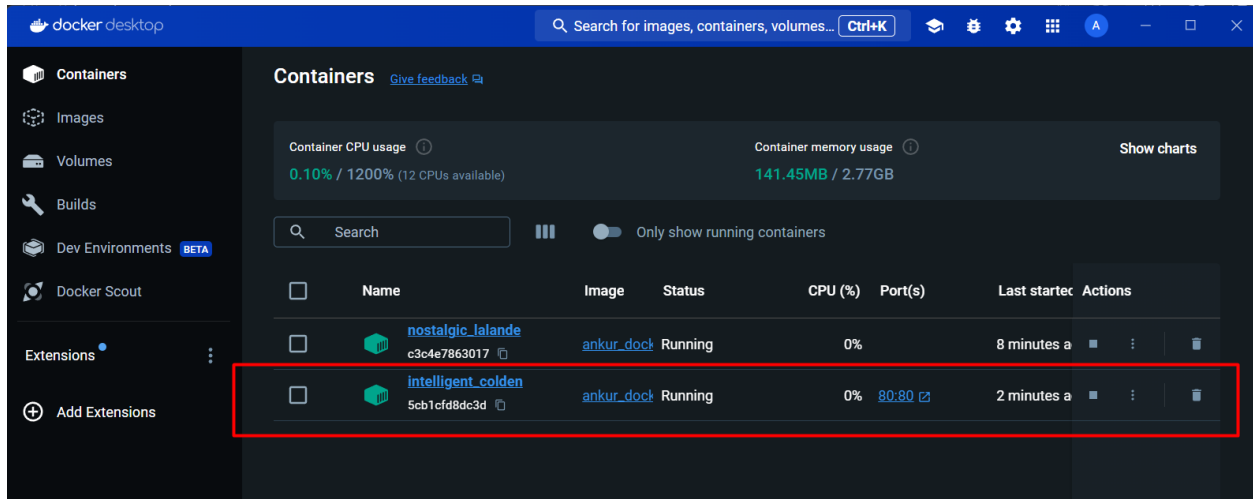
**docker run -it -p 80:80 -d <your – image- name>**

-it denotes for to run in interactive mode TTY

-p denotes for ports

-d denotes for detached mode (The Docker container running in the background).

```
PS C:\Users\pc\Docker_html> docker run -it -p 80:80 -d ankur_docker  
5cb1cf8dc3d1975c5ae29ff8f8854880fb8787a7b82b9a8dedff8d7783f5e51
```



Our webpage is also running on localhost:



**Hello, Docker!**

- **Now we have to push our image into ECR (Elastic Container Registry)**

Step: 1: Go to your AWS account.

Step: 2: Go to the services and search with ECR or Elastic Container Registry.

Step: 3: Create a public repository in ECR.

- After create a repository now we have to install AWS CLI in our machine. For download the AWS CLI prefer official site: <https://awscli.amazonaws.com/AWSCLIV2.msi>
- After setup the AWS CLI into the machine, now have to configure AWS CLI with our AWS account we need "I AM" or "Secret keys" and "Access Keys"
- Now go to your command prompt and run command: `aws configure`; after run the command just copy paste your access key and secret key, region name.

```
Command Prompt
Microsoft Windows [Version 10.0.22631.3296]
(c) Microsoft Corporation. All rights reserved.

C:\Users\pc>aws configure
AWS Access Key ID [*****PVUT]:
AWS Secret Access Key [*****Iow9]:
Default region name [eu-west-2]:
Default output format [json]:
```

- Now AWS configure successfully.
- Next step is we have to push or image into ECR (Elastic Container Repository). Below are the steps for that:

Step: 1: Go to your ECR Repository there is a button called “View Push command” click on it, then you will get the commands:

**Push commands for ankur\_docker**

Make sure that you have the latest version of the AWS CLI and Docker installed. For more information, see [Getting Started with Amazon ECR](#).

Use the following steps to authenticate and push an image to your repository. For additional registry authentication methods, including the Amazon ECR credential helper, see [Registry Authentication](#).

1. Retrieve an authentication token and authenticate your Docker client to your registry.  
Use the AWS CLI:  

```
aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password-stdin public.ecr.aws/s7f2n3x3
```

Note: If you receive an error using the AWS CLI, make sure that you have the latest version of the AWS CLI and Docker installed.
2. Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions [here](#). You can skip this step if your image is already built:  

```
docker build -t ankur_docker .
```
3. After the build completes, tag your image so you can push the image to this repository:  

```
docker tag ankur_docker:latest public.ecr.aws/s7f2n3x3/ankur_docker:latest
```
4. Run the following command to push this image to your newly created AWS repository:  

```
docker push public.ecr.aws/s7f2n3x3/ankur_docker:latest
```

Close

Step: 2: Copy first command and execute into your VS code terminal.

## Note\*:-

Now you may facing an error in execution of the command so for resolving this issue we have to follow few steps:

Step: 1: Goto your .docker folder it is present in the c drive for my system the path is "C:\Users\pc\.docker" .

Step: 2: There is file with name "Config.json", make a duplicate file of itv and name it "Configbackup.json".

Step : 3: We have to make some correction in the main file "Config.json", there is a line "credsStore": "desktop", remove this and save.

Step: 4 : Now go to your docker folder C:\Program Files\Docker\Docker\resources\bin, upto bin folder and rename a file "docker-credential-wincred" to "docker-credential-wincred.backup". Just add.backup in the end and save it.

- Now again execute the first command again, and its running fine:

```
PS C:\Users\pc\Docker_html> aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password-stdin public.ecr.aws/s7f2n3x3
WARNING! Your password will be stored unencrypted in C:\Users\pc\.docker\config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

Step 3: Run second command to build:

Step 4: Run third command:

```
View a summary of image vulnerabilities and recommendations -> docker scout quickview
PS C:\Users\pc\Docker_html> docker tag ankur_docker:latest public.ecr.aws/s7f2n3x3/ankur_docker:V1.0
PS C:\Users\pc\Docker_html>
```

Make changes in the, if you want you can execute version.

Step: 5: Run fourth command:

If have already pushed this image with the same name so its shows layer already exist: but in your case it will show the image pushing:

```
PS C:\Users\pc\Docker_html> docker push public.ecr.aws/s7f2n3x3/ankur_docker:V1.0
The push refers to repository [public.ecr.aws/s7f2n3x3/ankur_docker]
ed1bf9db37df: Layer already exists
badbe6e20a95: Layer already exists
667a247707f0: Layer already exists
d8527026595f: Layer already exists
2593b08e5428: Layer already exists
9909978d630d: Layer already exists
c5140fc719dd: Layer already exists
3137f8f0c641: Layer already exists
718db50a47c0: Layer already exists
aedc3bda2944: Layer already exists
V1.0: digest: sha256:fec769b58462deb89699016a0332e2b772bc5580992b1aa65940c880aa41deab size: 2403
```

- After pushing the image it will reflect in the ECR: [public.ecr.aws/s7f2n3x3/ankur\\_docker:V1.0](https://public.ecr.aws/s7f2n3x3/ankur_docker:V1.0)

Amazon ECR > Public Registry > Repositories > ankur\_docker

## ankur\_docker

[View public listing](#) [View push commands](#) [Edit](#)

Images (1)

[Refresh](#) [Delete](#) [Details](#)

< 1 > ⚙

<input type="checkbox"/>	Image tag	Artifact type	Pushed at	Size (M)		Digest
<input type="checkbox"/>	V1.0	Image	March 14, 2024, 13:50:11 (UTC+05.5)	17.97	<div>✔ Image URI copied ✕ Copy URI</div>	sha256:fec769b58462de...

\*\*\*\*\*