

## Continuation on my progress on docker:

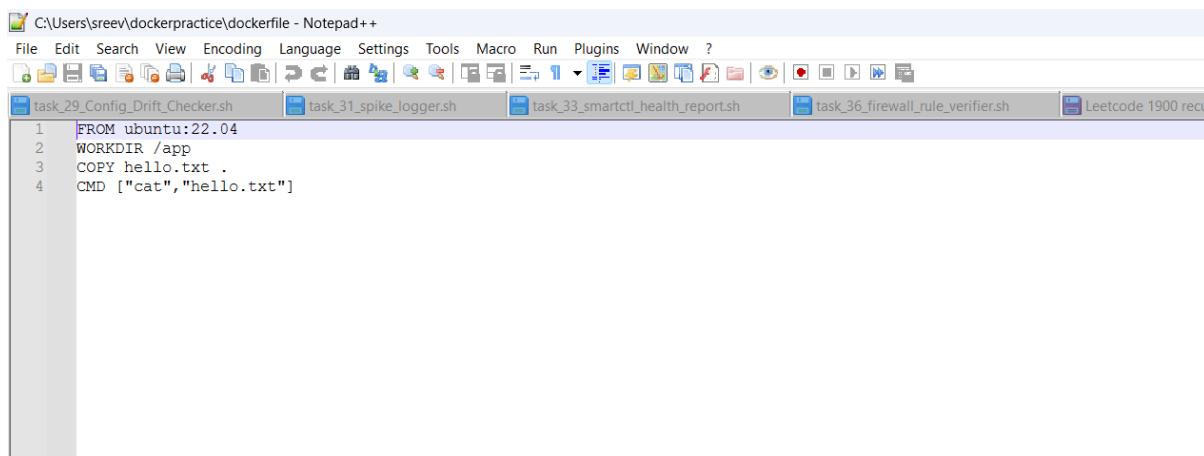
### Dockerfile practicals:

```
PS C:\Users\sreev> mkdir dockerpractice

Directory: C:\Users\sreev

Mode                LastWriteTime     Length Name
----                <-----          ----- 
d-----        12-02-2026      15:22    dockerpractice
```

RAM 1.31 GB CPU 3.26% Disk: 5.06 GB used (limit 1006.85 GB)



```
PS C:\Users\sreev\dockerpractice> docker build -t myhello:1.0 .
[+] Building 1.3s (8/8) FINISHED
=> [internal] load build definition from dockerfile
=> => transferring dockerfile: 111B
=> [internal] load metadata for docker.io/library/ubuntu:22.04
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/3] FROM docker.io/library/ubuntu:22.04@sha256:c7eb020043d8fc2ae0793fb35a37bff1cf33f156d4d4b12ccc7f3ef8706c38b1
=> => resolve docker.io/library/ubuntu:22.04@sha256:c7eb020043d8fc2ae0793fb35a37bff1cf33f156d4d4b12ccc7f3ef8706c38b1
=> [internal] load build context
=> => transferring context: 76B
```

RAM 0.84 GB CPU 0.17% Disk: 5.07 GB used (limit 1006.85 GB)

```
--> CACHED [2/3] WORKDIR /app
--> [3/3] COPY hello.txt .
--> exporting to image
--> exporting layers
--> exporting manifest sha256:d111e8e2edd4e2d2cb230aa9aa55532f909125ea866c63dba1533991ff1fb3e5
--> --> exporting config sha256:c5e21f2704d7d744c5e69fd9de72f067563fcf7979d330c96d402fa164437d
--> --> exporting attestation manifest sha256:241acc42f4cc6aea15d4e2c580de63dec7ba929ec19cb5795cd7563be325dd6
--> --> exporting manifest list sha256:bd7218e035c6a8349b1fb190cf346ac65845475f49486a6a028bb329d70afeaa
--> --> naming to docker.io/library/myhello:1.0
--> --> unpacking to docker.io/library/myhello:1.0
```

RAM 0.84 GB CPU 0.08% Disk: 5.07 GB used (limit 1006.85 GB)

```

=> CACHED [2/3] WORKDIR /app
=> [3/3] COPY hello.txt .
=> exporting to image
=> exporting layers
=> >> exporting manifest sha256:d111e8e2dd4e2d2cb230aa9aa55532f909125ea866c63db1533991ff1fb3e5
=> >> exporting config sha256:c5e21f2704d7d744c5e69fd9d72f067563fcfcf7979d6330c96d402fa164437d
=> >> exporting attestation manifest sha256:241ace42f4ce6aea15d4e2c580de63dae/ba929eec19cb5795cd7563be255dd6
=> >> exporting manifest list sha256:bd7218e035c6a8349bf190cf346acc65845475f494866a028b6329d70afeaa
=> >> naming to docker.io/library/myhello:1.0
=> >> unpacking to docker.io/library/myhello:1.0
PS C:\Users\sreev\dockerpractice>

```

RAM 0.84 GB CPU 0.08% Disk: 5.07 GB used (limit 1006.85 GB)

> Terminal ⓘ Update available

```

svish07/myhttpd:2.0    150f18275a55      282MB      77MB
svish07/myubuntu:1.0    c7eb020043d8      119MB      31.5MB U
svish07/myubuntu:2.0    f3bbeebbe790      239MB      78.8MB
ubuntu:22.04             c7eb020043d8      119MB      31.5MB U
ubuntu:latest            cd1dbaa651b30     119MB      31.7MB U
PS C:\Users\sreev\dockerpractice> docker run -it myhello:1.0 bash
root@a9510faa1410:/app# ls
hello.txt
root@a9510faa1410:/app# cat hello.txt
++Hello from docker
root@a9510faa1410:/app#

```

RAM 0.82 GB CPU 0.00% Disk: 5.07 GB used (limit 1006.85 GB)

> Terminal ⓘ Update available

```

svish07/myubuntu:2.0    f3bbeebbe790      239MB      78.8MB
ubuntu:22.04             c7eb020043d8      119MB      31.5MB U
ubuntu:latest            cd1dbaa651b30     119MB      31.7MB U
PS C:\Users\sreev\dockerpractice> docker run -it myhello:1.0 bash
root@a9510faa1410:/app# ls
hello.txt
root@a9510faa1410:/app# cat hello.txt
++Hello from docker
root@a9510faa1410:/app# pwd
/app
root@a9510faa1410:/app#

```

RAM 0.90 GB CPU 0.17% Disk: 5.07 GB used (limit 1006.85 GB)

> Terminal ⓘ Update available

## What is a Dockerfile?

A Dockerfile is a blueprint that defines how to build a Docker image by specifying:

- The base image (FROM)
- The working directory (WORKDIR)
- Files to copy (COPY)
- Commands to run
- The default startup command (CMD)

Docker reads it step-by-step and creates layered images.

In this practical, a Dockerfile was created to define how a custom image should be built using Ubuntu as the base image, setting /app as the working directory, copying a file from the build context into the image, and defining a default startup command. Using docker build -t myhello:1.0 ., Docker read the Dockerfile, processed each instruction as a layer, and created a tagged image. Running the image with docker run created a container that executed the defined CMD and displayed the output.

## Advanced practical on Dockerfile:

```
Terminal
PS C:\Users\sreev\dockerpractice> mkdir dockerfileadv

Directory: C:\Users\sreev\dockerpractice

Mode                LastWriteTime     Length Name
----                <-----          ----- 
d----       12-02-2026      17:15          dockerfileadv

RAM 0.91 GB CPU 0.75% Disk: 5.23 GB used (limit 1006.85 GB) >_ Terminal ⚡ Update available
```

```
Terminal
PS C:\Users\sreev\dockerpractice\dockerfileadv> notepad dockerfileadv
PS C:\Users\sreev\dockerpractice\dockerfileadv> ls

Directory: C:\Users\sreev\dockerpractice\dockerfileadv

Mode                LastWriteTime     Length Name
----                <-----          ----- 
-a---       12-02-2026      17:20        135 dockerfile.txt
-a---       12-02-2026      17:16          0 dockerfileadv.txt

RAM 0.92 GB CPU 3.50% Disk: 5.23 GB used (limit 1006.85 GB) >_ Terminal ⚡ Update available
```

```
Terminal
PS C:\Users\sreev\dockerpractice\dockerfileadv> ren dockerfile.txt dockerfile
PS C:\Users\sreev\dockerpractice\dockerfileadv> ls

Directory: C:\Users\sreev\dockerpractice\dockerfileadv

Mode                LastWriteTime     Length Name
----                <-----          ----- 
-a---       12-02-2026      17:20        135 dockerfile

RAM 0.92 GB CPU 0.17% Disk: 5.23 GB used (limit 1006.85 GB) >_ Terminal ⚡ Update available
```

```
Terminal
PS C:\Users\sreev\dockerpractice\dockerfileadv> docker build -t advpractice .
[+] Building 44.4s (7/7) FINISHED
   => [internal] load build definition from dockerfile
   => => transferring dockerfile: 174B
   => [internal] load metadata for docker.io/library/ubuntu:22.04
   => [internal] load .dockerrcignore
   => => transferring context: 2B
   => [1/3] FROM docker.io/library/ubuntu:22.04@sha256:c7eb020043d8fc2ae0793fb35a37bfff1cf33f156d44d4b12ccc7f3ef8706c38b1
   => => resolve docker.io/library/ubuntu:22.04@sha256:c7eb020043d8fc2ae0793fb35a37bfff1cf33f156d44d4b12ccc7f3ef8706c38b1
   => CACHED [2/3] WORKDIR /app
   => [3/3] RUN apt update && apt install -y vim curl && apt clean && rm -rf /var/lib/apt/lists/*
docker:desktop-linux
  0.1s
  0.0s
  1.2s
  0.1s
  0.0s
  0.1s
  0.1s
  0.0s
  36.8s

RAM 0.92 GB CPU 0.08% Disk: 5.23 GB used (limit 1006.85 GB) >_ Terminal ⚡ Update available
```

```
Terminal
PS C:\Users\sreev\dockerpractice\dockerfileadv> docker run -it --name advprac advpractice
root@d419682d131a:/app# curl -v
curl: (7) Failed to open URL: No URL specified!
```

```
root@d419682d131a:/app# curl --version
curl 7.81.0 (x86_64-pc-linux-gnu) libcurl/7.81.0 OpenSSL/3.0.2 zlib/1.2.11 brotli/1.0.9 zstd/1.4.8 libidn2/2.3.2 libpsl/0.21.0 (+libidn2/2.3.2)
libssh/0.9.6/openssl/zlib nghttp2/1.43.0 librtmp/2.3 OpenLDAP/2.5.20
Release-Date: 2022-01-05
Protocols: dict file ftp ftps gopher http https imap imaps ldap ldaps mqtt pop3 pop3s rtmp rtsp scp sftp smb smbs smtp smtps telnet tftp

Features: alt-svc AsynchDNS brotli GSS-API HSTS HTTP2 HTTPS-proxy IDN IPv6 Kerberos Largefile libz NTLM NTLM_WB PSL SPNEGO SSL TLS-SRP UnixSocke
ts zstd
root@d419682d131a:/app# vim --version
VIM - Vi IMproved 8.2 (2019 Dec 12, compiled Apr 02 2025 12:39:01)
Included patches: 1-16, 647, 17-579, 1969, 580-647, 678, 648-1848, 4975, 5016, 5023, 5072, 2068, 1849-1854, 1857, 1855-1857, 1331, 1858, 1858-18
```

```

Terminal

59, 1873, 1860-1969, 1992, 1970-1992, 2010, 1993-2068, 2106, 2069-2106, 2108, 2107-2109, 2109-3995, 4563, 4646, 4774, 4895,
3, 1840, 1846-1847, 2110-2112, 2121
Modified by team+vim@tracker.debian.org
Compiled by team+vim@tracker.debian.org
Huge version without GUI. Features included (+) or not (-):
+acl          +file_in_path    +mouse_urxvt   -tag_any_white
+arabic       +find_in_path   +mouse_xterm    -tcl
+autocmd      +float         +multi_byte    +termguiccolors
+autochdir    +folding        +multi_lang     +terminal
-autoservername -footer       -mzscheme      +terminfo
-balloon_eval  +fork()        +netbeans_intg +termresponse

RAM 0.92 GB CPU 4.02% Disk: 5.23 GB used (limit 1006.85 GB)

```

```

Terminal

system vimrc file: "$VIM/vimrc"
 user vimrc file: "$HOME/.vimrc"
 2nd user vimrc file: "~/.vim/vimrc"
 user exrc file: "$HOME/.exrc"
 defaults file: "$VIMRUNTIME/defaults.vim"
 fall-back for $VIM: "/usr/share/vim"

Compilation: gcc -c -I. -Iproto -DHAVE_CONFIG_H -Wdate-time -g -O2 -ffile-prefix-map=/build/vim-300zQI/vim-8.2.3995=. -fno-strict-aliasing -fno-fat-lto-objects -fstack-protector-strong -Wformat -Werror=format-security -D REENTRANT -U FORTIFY_SOURCE -D FORTIFY_SOURCE=1
Linking: gcc -Wl,Bsymbolic-functions -fno-fat-lto-objects -fno-strict-aliasing -Wl,-z,relro -Wl,-z,now -Wl,-as-needed -o vim -lm -ltinfo -lselinux -lsodium -lssl -latkr -lgpm -L/usr/lib/python3.10/config-3.10-x86_64-linux-gnu -lpython3.10 -lcrypt -ldl -lm -lncurses
root@d419682d131a:/app#

```

## Summary:

In this practical, I went to create a custom Docker image using Ubuntu as the base, defined a working directory with WORKDIR /app, and installed system tools (curl and vim) using a carefully structured RUN instruction that chained apt update, installation, and cleanup commands with && to ensure controlled execution and reduced image size. By combining installation and cache removal (apt clean and rm -rf /var/lib/apt/lists/\*) in a single layer, we followed an optimized and elegant image-building approach. The image was built using docker build -t mytools:1.0 ., tagged meaningfully, and then run interactively to verify that the installed tools were correctly available inside the container. This practical demonstrated layered image construction, efficient package management, and runtime validation in a clean, reproducible Docker workflow.

## Docker Image Build Pipeline and Runtime Override Practical:

```

Terminal

PS C:\Users\sreev\dockerpractice\dockerfileadv> mkdir dockerfull

Directory: C:\Users\sreev\dockerpractice\dockerfileadv

Mode                LastWriteTime         Length Name
----                -              ----- 
d----- 12-02-2026 17:49                 dockerfull

RAM 0.87 GB CPU 0.08% Disk: 5.23 GB used (limit 1006.85 GB)

```

```

PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> rm .dockerignore.txt
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> ren dockerfile.txt dockerfile
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> ls

Directory: C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull

Mode                LastWriteTime         Length Name
----                <-----           <-----  --

```

Terminal

```

PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> type dockerfile
FROM ubuntu:22.04
ENV APP_ENV=production
WORKDIR /app
RUN apt update && apt install -y curl && apt clean && rm -rf /var/lib/apt/lists/*
COPY index.html .
EXPOSE 8080
VOLUME ["/app/data"]
ENTRYPOINT ["echo"]
CMD ["Container started in production mode"]

RAM 0.86 GB CPU 0.17% Disk: 5.24 GB used (limit 1006.85 GB)

```

Terminal

```

PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker build -t fullapp:1.0 .
[+] Building 43.8s (9/9) FINISHED
=> [internal] load build definition from dockerfile
=> => transferring dockerfile: 300B
=> [internal] load metadata for docker.io/library/ubuntu:22.04
=> [internal] load .dockerignore
=> => transferring context: 69B
=> [1/4] FROM docker.io/library/ubuntu:22.04@sha256:c7eb020043d8fc2ae0793fb35a37bfff1cf33f156d4d4b12ccc7f3ef8706c38b1
=> => resolve docker.io/library/ubuntu:22.04@sha256:c7eb020043d8fc2ae0793fb35a37bfff1cf33f156d4d4b12ccc7f3ef8706c38b1
=> [internal] load build context
=> => transferring context: 101B

RAM 0.86 GB CPU 0.08% Disk: 5.24 GB used (limit 1006.85 GB)

```

Terminal

```

=> [4/4] COPY index.html .
=> exporting to image
=> => exporting layers
=> => exporting manifest sha256:2f8aa854c07c59c335e1dd307b70abc5057f8f1fe7ddda255da49b2578f6767d
=> => exporting config sha256:8bb0c2d2ddceef72af979dd1d4e7fb62e9802d1704b8d96df1dbb657d50c7f642
=> => exporting attestation manifest sha256:c1b1fa4cab1accbe2d4f9d040ff464a062315e6a83bb43830d7abafb698bf4cc
=> => exporting manifest list sha256:c5d981fffebbsbadibiffia7170659af2e1a6e1d1dc2652d9d86e3a79ec4b207
=> => naming to docker.io/library/fullapp:1.0
=> => unpacking to docker.io/library/fullapp:1.0
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker run -it --name myfullapp fullapp:1.0
Container started in production mode

RAM 0.86 GB CPU 0.00% Disk: 5.24 GB used (limit 1006.85 GB)

```

```

PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker run -it --name myfullapp fullapp:1.0
Container started in production mode
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker run -it --name myfullapp fullapp:1.0 bash
docker: Error response from daemon: Conflict. The container name "/myfullapp" is already in use by container "ecf260eaeb850f6708fb871e68295af2b60a9d5c5313f660f27fd81e0621bd". You have to remove (or rename) that container to be able to reuse that name.

Run 'docker run --help' for more information
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull>

RAM 0.86 GB CPU 0.42% Disk: 5.24 GB used (limit 1006.85 GB)

```

```

Run 'docker run --help' for more information
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker exec -it myfullapp bash
Error response from daemon: container ecf260eaeb850f6708fb871e68295af2b60a9d5c5313f660f27fd81e0621bd is not running
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker start myfullapp
myfullapp
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker exec -it myfullapp bash
Error response from daemon: container ecf260eaeb850f6708fb871e68295af2b60a9d5c5313f660f27fd81e0621bd is not running
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull>

RAM 0.86 GB CPU 0.00% Disk: 5.24 GB used (limit 1006.85 GB)

```

```

Terminal
Container started in production mode
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker run -it --name myfullapp fullapp:1.0 bash
docker: Error response from daemon: Conflict. The container name "/myfullapp" is already in use by container "ecf260eaeb850f6708fb871e68295af2b60fa9d5c5313f660f27fd81e0621bd". You have to remove (or rename) that container to be able to reuse that name.

Run 'docker run --help' for more information
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker exec -it myfullapp bash
Error response from daemon: container ecf260eaeb850f6708fb871e68295af2b60fa9d5c5313f660f27fd81e0621bd is not running
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker start myfullapp
myfullapp
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker exec -it myfullapp bash

RAM 0.88 GB CPU 0.08% Disk: 5.24 GB used (limit 1006.85 GB)

```

```

Terminal
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker exec -it myfullapp bash
Error response from daemon: container ecf260eaeb850f6708fb871e68295af2b60fa9d5c5313f660f27fd81e0621bd is not running
PS C:\Users\sreev\dockerpractice\dockerfileadv\dockerfull> docker run -it --entrypoint bash fullapp:1.0
root@ad28cbf7c48f:/app# curl --version
curl/7.81.0 (x86_64-pc-linux-gnu) libcurl/7.81.0 OpenSSL/3.0.2 zlib/1.2.11 brotli/1.0.9 zstd/1.4.8 libidn2/2.3.2 libpsl/0.21.0 (+libidn2/2.3.2)
libssh/0.9.6/openssl/1.1.1g zlib nghttp2/1.43.0 librtmp/2.3 OpenLDAP/2.5.20
Release-Date: 2022-01-05
Protocols: dict file ftp ftps gopher gophers http https imap imaps ldaps mqtt pop3 pop3s rtmp rtsp scp sftp smb smbs smtp smtps telnet tftp
Features: alt-svc AsyncDNS brotli QSS-API HSTS HTTP2 HTTPS-proxy IDN IPv6 Kerberos Largefile libz NTLM NTLM_WB PSL SPNEGO SSL TLS-SRP UnixSocke
ts zstd

RAM 0.88 GB CPU 0.00% Disk: 5.24 GB used (limit 1006.85 GB)

```

In this practical, I moved beyond simply building an image and instead explored the full choreography of Docker's build-time and run-time mechanics: starting with a controlled build context using `.dockerignore` to consciously shape what the daemon receives, then designing a layered Dockerfile that defined a base system (`FROM`), structured execution space (`WORKDIR`), controlled installation and cleanup through chained `RUN` commands, and intentional metadata through `ENV`, `EXPOSE`, and `VOLUME`. We examined how `ENTRYPOINT` establishes a container's core executable while `CMD` supplies its default arguments, discovering through experimentation that a container lives only as long as its primary process runs. When the container exited instantly, we learned that behaviour is not failure but design and that runtime overrides like `--entrypoint` and `docker exec` allow inspection and flexibility without rebuilding images. Altogether, the exercise revealed Docker as a disciplined separation of immutable image architecture and mutable runtime behaviour, where context, layering, execution flow, and lifecycle awareness converge into elegant container engineering.

## Advanced networking simulation practicals:

```

Terminal
PS C:\Users\sreev> docker run -dit --name server --network mybridge nginx
6c65bd039301b7d6670c8b994f0c0004e89069b8745b47a8d03869e36be54
PS C:\Users\sreev> docker run -dit --name client --network mybridge busybox sh
Unable to find image 'busybox:latest' locally
latest: Pulling from library/busybox
61dfb50712f5: Pull complete
96cfb76e59bd: Download complete
Digest: sha256:b3255c7dfbcd10cb367af0d409747d511aeb66dfac98cf30e97e87e4207dd76f
Status: Downloaded newer image for busybox:latest
d0fffb0e07ba420ab28147f684f304551dc5c123285eb52af26077e3d61724a84
PS C:\Users\sreev> docker exec -it client sh

RAM 1.01 GB CPU 0.08% Disk: 5.25 GB used (limit 1006.85 GB)

```

```
Terminal
PS C:\Users\sreev> docker exec -it client sh
/ # ping server
PING server (172.18.0.2): 56 data bytes
64 bytes from 172.18.0.2: seq=0 ttl=64 time=0.647 ms
64 bytes from 172.18.0.2: seq=1 ttl=64 time=0.104 ms
64 bytes from 172.18.0.2: seq=2 ttl=64 time=0.147 ms
64 bytes from 172.18.0.2: seq=3 ttl=64 time=0.142 ms
64 bytes from 172.18.0.2: seq=4 ttl=64 time=0.108 ms
64 bytes from 172.18.0.2: seq=5 ttl=64 time=0.078 ms
64 bytes from 172.18.0.2: seq=6 ttl=64 time=0.196 ms
64 bytes from 172.18.0.2: seq=7 ttl=64 time=0.433 ms
RAM 1.00 GB CPU 0.00% Disk: 5.25 GB used (limit 1006.85 GB)
>_ Terminal ⚡ Update available
```

```
Terminal
64 bytes from 172.18.0.2: seq=24 ttl=64 time=0.168 ms
64 bytes from 172.18.0.2: seq=25 ttl=64 time=0.135 ms
64 bytes from 172.18.0.2: seq=26 ttl=64 time=0.207 ms
64 bytes from 172.18.0.2: seq=27 ttl=64 time=0.144 ms
^C
--- server ping statistics ---
28 packets transmitted, 28 packets received, 0% packet loss
round-trip min/avg/max = 0.078/0.212/0.975 ms
/ # wget -qO- http://server
<!DOCTYPE html>
<html>
RAM 1.00 GB CPU 0.00% Disk: 5.25 GB used (limit 1006.85 GB)
>_ Terminal ⚡ Update available
```

```
Terminal
/ # wget -qO- http://server
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
RAM 1.01 GB CPU 0.00% Disk: 5.25 GB used (limit 1006.85 GB)
>_ Terminal ⚡ Update available
```

```
Terminal
working. Further configuration is required.</p>
<p>For online documentation and support please refer to
<a href="http://nginx.org/">http://nginx.org/</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">http://nginx.com/</a>.</p>
<p><em>Thank you for using nginx.</em></p>
</body>
</html>
/ #
RAM 1.00 GB CPU 0.08% Disk: 5.25 GB used (limit 1006.85 GB)
>_ Terminal ⚡ Update available
```

## Practical: Customizing and Publishing an HTTPD Docker Image

```
Terminal
PS C:\Users\sreev> docker pull httpd:latest
latest: Pulling from library/httpd
Digest: sha256:b89c19e390514d6767e8c62f29375d0577190be448f63b24f5f11d6b03f7bf18
Status: Image is up to date for httpd:latest
docker.io/library/httpd:latest
PS C:\Users\sreev> docker images
IMAGE          ID          DISK USAGE   CONTENT SIZE   EXTRA
adpractice:latest  1c74231c7281    214MB      52.3MB      U
busybox:latest    b3255e7dfbcd    6.77MB     2.22MB      U
RAM 1.77 GB CPU 0.00% Disk: 5.42 GB used (limit 1006.85 GB)
>_ Terminal ⚡ Update available
```

```
Terminal
PS C:\Users\sreev> docker tag httpd:latest svish07/myhttpd:practice
PS C:\Users\sreev> docker login
Authenticating with existing credentials... [Username: svish07]
Info To login with a different account, run 'docker logout' followed by 'docker login'

Login Succeeded
PS C:\Users\sreev> docker push svish07/myhttpd:practice
The push refers to repository [docker.io/svish07/myhttpd]
996228ca33ab: Layer already exists

RAM 1.78 GB CPU 0.33% Disk: 5.42 GB used (limit 1006.85 GB)
>_ Terminal ⚡ Update available
```

```
Terminal
ncae846799
PS C:\Users\sreev> docker run -it --name practice1 httpd:latest bash
root@315547f154db:/usr/local/apache2# apt update && apt install -y vim curl && apt clean
Hit:1 http://deb.debian.org/debian trixie InRelease [47.3 kB]
Get:2 http://deb.debian.org/debian-security trixie-security InRelease [43.4 kB]
Get:3 http://deb.debian.org/debian trixie/main amd64 Packages [9670 kB]
Get:4 http://deb.debian.org/debian trixie/main amd64 Packages [5412 B]
Get:5 http://deb.debian.org/debian-security trixie-security/main amd64 Packages [108 kB]
Get:6 http://deb.debian.org/debian-security trixie-security/main amd64 Packages [108 kB]
Fetched 9874 kB in 3s (3069 kB/s)
All packages are up to date.

RAM 1.78 GB CPU 0.00% Disk: 5.42 GB used (limit 1006.85 GB)
>_ Terminal ⚡ Update available
```

```
Terminal
UnixSockets zstd
root@315547f154db:/usr/local/apache2# vim --version
VIM - Vi IMproved 9.1 (2024 Jan 02, compiled May 23 2025 00:48:59)
Included patches: 1-948, 950-1230, 1242, 1244
Modified by team+vim@tracker.debian.org
Compiled by team+vim@tracker.debian.org
Huge version without GUI. Features included (+) or not (-):
+acl          +file_in_path    +mouse_urxvt   -tag_any_white
+arabic       +find_in_path    +mouse_xterm    -tcl
+autocmd      +float          +multi_byte    +termguicolors
+autochdir    +folding         +multi_lang    +terminal
RAM 1.78 GB CPU 0.00% Disk: 5.42 GB used (limit 1006.85 GB)
>_ Terminal ⚡ Update available
```

```
Terminal
PS C:\Users\sreev> docker commit practice1 svish07/myhttpd:done
sha256:b6ba4773f31742a94dd9621e295fac03ad2eeef630544001ebf99657c810cf5be
PS C:\Users\sreev> docker push svish07/myhttpd:done
The push refers to repository [docker.io/svish07/myhttpd]
fe1dbdd446f: Layer already exists
30973b009bab: Layer already exists
d077bbcef6ed0: Pushed
4f4fb700ef54: Layer already exists
85b0d820aa56: Layer already exists
996228ca33ab: Layer already exists
0c8d554a5c0d: Layer already exists

RAM 1.78 GB CPU 0.08% Disk: 5.42 GB used (limit 1006.85 GB)
>_ Terminal ⚡ Update available
```

In this practical, we transitioned from merely consuming an official httpd image to owning and evolving it, pulling the base image, re-tagging it under a personal namespace, authenticating with Docker Hub, and publishing it as versioned artifacts. By overriding the default runtime to access the container shell, we introduced additional tooling (vim, curl), then captured those changes through docker commit, effectively converting a transient container state into a new immutable image layer. Finally, by pushing the updated image to Docker Hub and exposing it via port mapping, we demonstrated the full lifecycle: base image acquisition, customization, versioning, distribution, and runtime validation, showcasing Docker not just as a container runtime, but as a disciplined system for image ownership, reproducibility, and controlled evolution.