

LAPORAN DOCKER



Oleh:

Nama : L HAFIDL ALKHAIR
NIM : 2023903430060
Kelas : TRKJ 1.C
Jurusan : Teknologi Informasi dan Komputer
Program Studi : Teknologi Rekayasa Komputer Jaringan
Dosen Pembimbing : Indrawati, SST. MT



JURUSAN TEKNOLOGI, KOMPUTER, DAN INFORMASI
PRODI TEKNOLOGI REKAYASA KOMPUTER DAN JARINGAN
POLITEKNIK NEGERI LHOKSEUMAWE
TAHUN AJARAN 2023/2024

LEMBAR PENGESAHAN

No. Praktikum : 01 /TIK/TRKJ-1C/ Cloud Computing
Judul : Laporan Cloud Computing
Nama : L HAFIDL ALKHAIR
NIM : 2023903430060
Kelas : TRKJ-1C
Jurusan : Teknologi Informasi Dan Komputer
Prodi : Teknologi Rekayasa Komputer Jaringan
Tanggal Praktikum : 07 Maret 2025
Tanggal Penyerahan : 14 Maret 2025

Buketrata, 14 Maret 2025

Dosen Pembimbing,

Indrawati, SST.MT

Nip. 19740815 200112 2 001

DAFTAR ISI

LEMBAR PENGESAHAN	i
DAFTAR ISI	ii
A. Tujuan	1
B. Dasar Teori.....	1
C. Alat dan bahan	2
D. Lab 1: Docker	3

1.	Lab 1.1 : Docker Image “hello-world.....	3
2.	Lab 1.2: Shell.....	5
3.	Lab 1.3:Membuat akun didocker hub.....	8
4.	Lab 1.4:Membuat images baru dan menyimpan didocker Hub.....	9
E.	Lab 2: Docker	12
1.	Lab 2.1: Docker Image ”node_project”	12
F.	Lab 3 Docker.....	15

A. Tujuan

Praktikum ini bertujuan untuk memberikan pemahaman dasar tentang Docker, termasuk cara:

- Menjalankan container menggunakan image yang tersedia di **Docker Hub**.
- Menggunakan **Shell** dalam container untuk berinteraksi dengan sistem operasi.
- Membuat akun di **Docker Hub** dan menyimpan image yang dibuat.
- Membuat image baru dan mengunggahnya ke **Docker Hub**.
- Menggunakan **Docker Volume** untuk penyimpanan data yang persisten.
- Menjalankan **multi-container** menggunakan **Docker Compose**.

B. Dasar Teori

Docker adalah sebuah platform yang digunakan untuk membuat, menjalankan, dan mengelola aplikasi dalam lingkungan yang terisolasi yang disebut **container**. Container lebih ringan dibandingkan virtual machine karena berbagi kernel dengan sistem operasi host, sehingga lebih cepat dan efisien.

Docker bekerja dengan menggunakan **image**, yaitu file template yang berisi sistem operasi dan aplikasi. Dari image ini, Docker dapat menjalankan **container**, yang merupakan lingkungan aplikasi yang berjalan secara independen. Image bisa diambil dari **Docker Hub**, sebuah repositori online yang menyediakan berbagai image siap pakai.

Dalam penggunaannya, Docker memiliki berbagai perintah dasar seperti `docker pull` untuk mengunduh image, `docker run` untuk menjalankan container, dan `docker ps` untuk melihat container yang sedang berjalan. Untuk menyimpan data agar tidak hilang saat container dihapus, digunakan **Docker Volume**.

Selain itu, Docker juga memiliki **Docker Compose**, yang memungkinkan pengguna menjalankan beberapa container secara bersamaan dengan satu perintah menggunakan file konfigurasi `docker-compose.yml`. Hal ini mempermudah pengelolaan aplikasi yang terdiri dari beberapa layanan, seperti web server dan database.

C. Alat dan bahan

1. Alat

- Komputer dengan sistem operasi yang mendukung Docker (Linux, Windows, macOS).
- Docker Engine yang telah terinstal.
- Docker Compose (untuk pengelolaan multi-container).
- Internet untuk mengakses Docker Hub.

Terminal atau Command Prompt untuk menjalankan perintah Docker.

2. Bahan

- Docker Image yang digunakan dalam praktikum (misalnya hello-world, alpine, node, go, dll.).
- Source Code untuk membuat container (misalnya salam.go, index.js, dan Dockerfile).
- Akun Docker Hub untuk menyimpan dan mengambil image.

D. Lab 1: Docker

1. Lab 1.1 : Docker Image “hello-world”
Pertama untuk periksa dulu versi docker

\$docker version

```
fid@fid:~$ sudo su
[sudo] password for fid:
root@fid:/home/fid# docker version
Client:
 Version:           26.1.3
 API version:       1.45
 Go version:        go1.22.2
 Git commit:        26.1.3-0ubuntu1~24.04.1
 Built:             Mon Oct 14 14:29:26 2024
 OS/Arch:           linux/amd64
 Context:           default

Server:
 Engine:
  Version:          26.1.3
  API version:      1.45 (minimum version 1.24)
  Go version:       go1.22.2
  Git commit:       26.1.3-0ubuntu1~24.04.1
  Built:            Mon Oct 14 14:29:26 2024
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.7.24
  GitCommit:
 runc:
  Version:          1.1.12-0ubuntu3.1
  GitCommit:
 docker-init:
  Version:          0.19.0
  GitCommit:
```

sudo su disini bertujuan untuk mengasess root pada linux. Dan bisa dilihat pada gambar diatas versi docker yang digunakan.

```

root@fid:/home/fid# docker images
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
fidk/node_project    1.0             72445a68c47d   13 hours ago   1.13GB
fidk/awal            latest          a20b7fc676a4   31 hours ago   888MB
hafidl/awal          latest          5b91262f0b2f   31 hours ago   888MB
golang               latest          29616a01ff27   4 days ago     853MB
node                 latest          e4f23baa3e59   10 days ago    1.12GB
alpine               latest          aded1e1a5b37   3 weeks ago    7.83MB
hello-world          latest          74cc54e27dc4   6 weeks ago    10.1kB

```

\$docker images

Disini ada beberapa images yang sudah terdaftar maka hapus terlebih dahulu dengan perintah berikut ini :

//hentikan proses docker yang masih aktif

\$ docker stop \$(docker ps -a -q)

```

root@fid:/home/fid# docker ps -a -q
0ff21718f530
d3da7c40dcf0
2a3aabb0e4be
bfe778e4cac8
5beebf6e9e0f
23410b56c726
0a6752410174
3f2d28276c49

```

// hapus semua images

\$ docker rmi \$(docker images -a -q)

```

root@fid:/home/fid# docker images -a -q
9c2660d763d7
1c6f831518e7
72445a68c47d
525b2bf054c7
1cee10a3c4ec
3bc3af9a6004
a20b7fc676a4
d0d22d61d7ae
9d3bfc8f5217
c195f3fcd49b
5b91262f0b2f
2ec2a0b16050
9ac5dccfcaa
29616a01ff27
e4f23baa3e59
aded1e1a5b37
74cc54e27dc4

```

// atau ha root@fid:/home/fid# XXXXXXXXXX op

```
root@fid:/home/fid# docker system prune -a
WARNING! This will remove:
- all stopped containers
- all networks not used by at least one container
- all images without at least one container associated to them
- all build cache

Are you sure you want to continue? [y/N]
```

\$ docker system prune -a

Selanjutnya *\$docker run hello-world*

```
root@fid:/home/fid# docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

Docker berhasil dijalankan.

2. Lab 1.2: Shell

Shell disini berfungsi untuk menjalankan system operasi yang ada

\$docker search alpine

Untuk mencari alpine yang terdaftar pada linux

```
root@fid:/home/fid# docker search alpine
NAME                DESCRIPTION                STARS     OFFICIAL
alpine              A minimal Docker image based on Alpine Linux... 11213     [OK]
alpine/git          A simple git container running in alpine li... 240
alpine/socat        Run socat command in alpine container         105
alpine/helm         Auto-trigger docker build for kubernetes hel... 68
alpine/k8s          Kubernetes toolbox for EKS (kubectl, helm, i... 58
alpine/curl         curl                        8
alpine/terraform    Auto-trigger docker build for terraform whe... 18
alpine/httpie       Auto-trigger docker build for 'httpie' when ... 21
alpine/bombardier   Auto-trigger docker build for bombardier whe... 26
alpine/openssl      openssl                    4
alpine/flake8       Auto-trigger docker build for flake8 via ci c... 2
alpine/ansible      run ansible and ansible-playbook in docker     24
alpine/semver       Docker tool for semantic versioning             5
alpine/jmeter       run jmeter in Docker                        8
alpine/gcloud       Auto-trigger docker build for gcloud (google... 0
alpine/bundle       This repository has been archived by the own... 1
alpine/xml          several xml tools to work on xml file as jq ... 1
alpine/mysql        mysql client                      3
alpine/cfn-nag      Auto-trigger docker build for cfn-nag when n... 0
alpine/psql         psql - The PostgreSQL Command-Line Client      1
alpine/mongosh      mongosh - MongoDB Command Line Database Tools  2
alpine/dfimage      reverse Docker images into Dockerfiles        70
alpine/java         Repo containing the build scripts to produce... 2
alpine/cfn-lint     CloudFormation linting tool                0
alpine/doctl        DigitalOcean command line with kubernetes an... 0
root@fid:/home/fid#
```

\$docker pull alpine

```
root@fid:/home/fid# docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
Digest: sha256:a8560b36e8b8210634f77d9f7f9efd7ffa463e380b75e2e74aff4511df3ef88c
Status: Image is up to date for alpine:latest
docker.io/library/alpine:latest
```

\$docker images

alpine	latest	aded1e1a5b37	3 weeks ago	7.83MB
ubuntu	latest	a04dc4851cbc	5 weeks ago	78.1MB
hello-world	latest	74cc54e27dc4	6 weeks ago	10.1kB

\$docker pull alpine:latest

```
root@fid:/home/fid# docker pull alpine:latest
latest: Pulling from library/alpine
Digest: sha256:a8560b36e8b8210634f77d9f7f9efd7ffa463e380b75e2e74aff4511df3ef88c
Status: Image is up to date for alpine:latest
docker.io/library/alpine:latest
root@fid:/home/fid#
```

\$docker run -it alpine

```
root@fid:/home/fid# docker run -it alpine
/# id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel),11(floppy),20(dialout),26(tape),27(video)
/# hostname
994481bf075f
/# netstat -r
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
default 172.17.0.1 0.0.0.0 UG 0 0 0 eth0
172.17.0.0 * 255.255.0.0 U 0 0 0 eth0
/# exit
root@fid:/home/fid#
```

\$docker ps

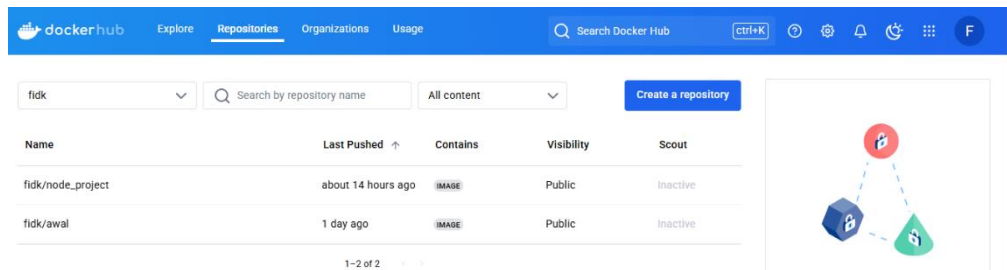
```
root@fid:/home/fid# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
root@fid:/home/fid#
```

\$docker ps -a

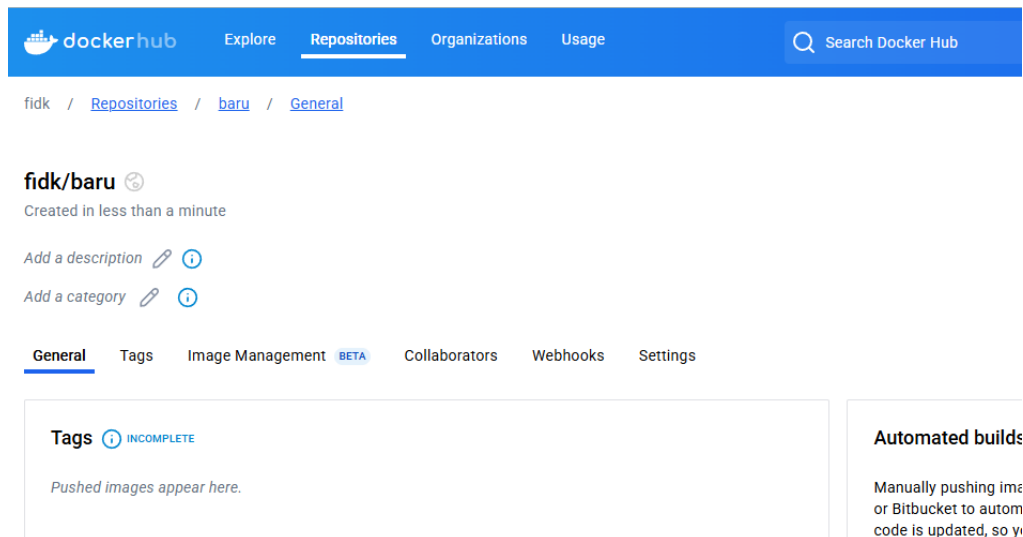
CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS	PORTS
994481bf075f	alpine	angry_wilson	"/bin/sh"	About a minute ago	Exited (0)	About a minute ago
c172f268dfa6	hello-world	gifted_villani	"/hello"	15 minutes ago	Exited (0)	15 minutes ago
7a896bb9b634	ubuntu	youthful_cohen	"bash"	16 minutes ago	Exited (0)	15 minutes ago
2bd2c0cdc8e1	hello-world		"/hello"	16 minutes ago	Exited (0)	16 minutes ago

3. Lab 1.3: Membuat akun didocker hub

Login menggunakan github atau google disini saya menggunakan github yang sudah ada



Buat repository baru seperti ini



4. Lab 1.4: Membuat images baru dan menyimpan didocker Hub
Buat folder baru untuk menyimpan images baru

`$mkdir baru`

Untuk membuat direktori baru yang Bernama “baru”

Kemudian untuk masuk ke dalam folder baru tersebut gunakan

`$cd baru/`

`$nano baru.go`

nano merupakan teks editor yang ada pada linux

```
GNU nano 7.2
package main
import "fmt"
func main () {
fmt . Println (" Salam dari L Hafidl Alkhair Kelompok 1")
}
```

`$nano Dockerfile`

membuat file Dockerfile

```
GNU nano 7.2
FROM golang : latest
# fidk - hafidalkhair27@gmail.com
WORKDIR / app
COPY . .
RUN go build baru .go
CMD ["/ app / baru "]
```

\$docker build -t fidk/baru

```
Sending build context to Docker daemon 3.072kB
Step 1/5 : FROM golang:latest
----> 29616a01ff27
Step 2/5 : WORKDIR /app
----> Using cache
----> 2ec2a0b16050
Step 3/5 : COPY . .
----> a22dc8837a31
Step 4/5 : RUN go build baru.go
----> Running in 6e6426c671fa
----> Removed intermediate container 6e6426c671fa
----> aabb66bf9a86
Step 5/5 : CMD ["/app/baru "]
----> Running in e58c5228b9de
----> Removed intermediate container e58c5228b9de
----> ffbc0d896c41
Successfully built ffbc0d896c41
Successfully tagged fidk/baru:latest
root@fid:/home/fid/baru#
```

\$docker images

```
root@fid:/home/fid/baru# docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
fidk/baru           latest      ffbc0d896c41 About a minute ago 888MB
```

hasil yang telah di buat. Kemudian simpan ke docker hub.

\$docker login -u fidk

Harus disesuaikan dengan username yang ada pada docker yang telah dibuat tadi

```
root@fid:/home/fid/baru# docker login -u fidk
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
Login Succeeded
```

Jika seperti ini maka login berhasil.

\$docker push fidk/baru

```
root@fid:/home/fid/baru# docker push fidk/baru
Using default tag: latest
The push refers to repository [docker.io/fidk/baru]
94562367213f: Pushed
d143e4d2ff7b: Pushed
```

\$docker pull fidk/baru

```
root@fid:/home/fid/baru# docker pull fidk/baru
Using default tag: latest
latest: Pulling from fidk/baru
Digest: sha256:04c78c16e7a8736cd9e9c6aebd22172c6311a404948cba5301a92d32f70a2169
Status: Image is up to date for fidk/baru:latest
docker.io/fidk/baru:latest
root@fid:/home/fid/baru#
```

\$docker run fidk/baru

```
root@fid:/home/fid/baru# docker run fidk/baru
Salam dari L Hafidl Alkhair Kelompok 1
root@fid:/home/fid/baru#
```

Jika tampil seperti gamabr maka ini sudah berhasil.

E. Lab 2: Docker

1. Lab 2.1: Docker Image "node_project"

\$mkdir node_project

\$cd node_project

Setelah membuat direktori baru jangan lupa install

\$sudo apt install npm

\$npm -v

Untuk melihat versi dari npm yang ada pada linux

```
root@fid:/home/fid/node_project# npm -v
9.2.0
root@fid:/home/fid/node_project#
```

\$npm init

```
Press ^C at any time to quit.
package name: (node_project)
version: (1.0.0)
description: Cloud Computing
entry point: (index.js)
test command:
git repository:
keywords:
author: L Hafidl Alkhair
license: (ISC)
About to write to /home/fid/node_project/package.json:
{
  "name": "node_project",
  "version": "1.0.0",
  "description": "Cloud Computing",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "L Hafidl Alkhair",
  "license": "ISC"
}
```

\$npm install express

```
found 0 vulnerabilities
root@fid:/home/fid/node_project# cat package.json
{
  "name": "node_project",
  "version": "1.0.0",
  "description": "Cloud Computing",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "L Hafidl Alkhair",
  "license": "ISC",
  "dependencies": {
    "express": "^4.21.2"
  }
}
```

Buat file dengan index.js seperti berikut:

```
root@fid:/home/fid/node_project# cat index.js
const express = require("express")
const app = express()
const port = 7000

app.get("/", (req, resp) => {
  resp.send("Nama saya Hafidl dari kelompok 1");
})
app.listen(port, ()=> console.log(`listening on port ${port}`))
```

Kemudian jalan kan port dengan perintah

\$node index.js

```
root@fid:/home/fid/node_project# node index.js
listening on port 7000
```



Nama saya Hafidl dari kelompok 1


```

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

Build an image from a Dockerfile
root@fid:/home/fid/node_project# docker build -t hafidl/node_project:1.0 .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/

Sending build context to Docker daemon 2.85MB
Step 1/7 : FROM node:latest
--> e4f23baa3e59
Step 2/7 : WORKDIR /app
--> Using cache
--> 84f3a335599e
Step 3/7 : COPY package.json .
--> Using cache
--> 26a8b3a8740f
Step 4/7 : RUN npm install
--> Using cache
--> cd40a5c7ff88
Step 5/7 : COPY . .
--> Using cache
--> cacc2b658d5f
Step 6/7 : EXPOSE 7000
--> Using cache
--> 622b23e8e38e
Step 7/7 : CMD ["node", "index.js"]
--> Using cache
--> adda581a6d6a
Successfully built adda581a6d6a
Successfully tagged hafidl/node_project:1.0
root@fid:/home/fid/node_project#

```

Selanjutnya buat file dengan nama Dockerfile

```

FROM node : latest
WORKDIR / app
COPY package . json .
RUN npm install
COPY . .
EXPOSE 7000
CMD [" node ", " index .js"]

```

\$docker build -t hafidl/node_project :1.0 .

\$docker ps untuk melihat status images

```

root@fid:/home/fid/node_project# docker ps

```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
b37171d6e475	hafidl/node_project:1.0	"docker-entrypoint.s..."	8 minutes ago	Up 8 minutes	7000/tcp
58f9339f6fc6	hafidl/node_project:1.0	"docker-entrypoint.s..."	12 minutes ago	Up 12 minutes	0.0.0.0:80->7000/tcp, :::80->7000/tcp

```

root@fid:/home/fid/node_project#

```

\$docker run -d --name node_project -p 80:7000 hafidl / node_project :1.0

\$docker push hafidl/node_project:1.0

Perintah ini digunakan untuk mengunggah (push) image ke Docker Hub atau registry Docker lainnya.

\$docker pull hafidl/node_project:1.0

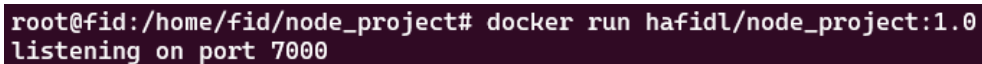
Perintah ini digunakan untuk mengunduh (pull) image dari Docker Hub ke sistem lokal.

\$docker run hafidl/node_project:1.0

Perintah ini digunakan untuk menjalankan container dari image yang sudah ada.

Jika

tampilan



```
root@fid:/home/fid/node_project# docker run hafidl/node_project:1.0
listening on port 7000
```

nya seperti itu maka sudah berhasil.

F. Lab 3 Docker

1. Lab 3.1 : Docker Volume

Docker volume dibutuhkan oleh kontainer karena kontainer tidak dapat menyimpan data secara persisten. Docker volume menawarkan fisik disk yang ada di Host sebagai media penyimpanan untuk kontainer. Dengan demikian jika kontainer selesai bekerja dan hilang dari Host, maka data tersebut tetap ada.

\$docker info

- Menampilkan informasi lengkap tentang konfigurasi Docker, termasuk jumlah container yang berjalan, jumlah image yang tersedia, versi Docker, dan detail lainnya.

\$docker images

- Menampilkan daftar container yang sedang berjalan.
- Gunakan `docker ps -a` untuk menampilkan semua container, termasuk yang telah berhenti.

\$doceker ps

- Menampilkan daftar container yang sedang berjalan.
- Gunakan `docker ps -a` untuk menampilkan semua container, termasuk yang telah berhenti.

\$docker volume ls

- Menampilkan daftar volume yang tersedia di mesin Docker Anda.
- Volume ini digunakan untuk menyimpan data yang persisten, bahkan setelah container dihentikan atau dihapus.

\$docker system prune -a

```
root@Hafid:~# docker system prune -a
WARNING! This will remove:
- all stopped containers
- all networks not used by at least one container
- all images without at least one container associated to them
- all build cache

Are you sure you want to continue? [y/N] y
Total reclaimed space: 0B
```

\$docker run -it -v /home/hafid/localdata:/mydata k1 alpine

```
root@Hafid:/home/hafid# docker run -it -v /home/hafid/localdata:/mydata --name k1 alpine
/ # cd /mydata
/mydata # touch f1 f2 f3
/mydata # ls -al
total 8
drwxr-xr-x  2 root    root      4096 Mar 13 10:25 .
drwxr-xr-x  1 root    root      4096 Mar 13 12:20 ..
-rw-r--r--  1 root    root         0 Mar 13 12:21 f1
-rw-r--r--  1 root    root         0 Mar 13 12:21 f2
-rw-r--r--  1 root    root         0 Mar 13 12:21 f3
/mydata #
root@Hafid:/home/hafid# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
6a3ca160e6ac   alpine    "/bin/sh" 2 hours ago Up 2 hours      k5
a656ae61d2dc   alpine    "/bin/sh" 2 hours ago Up 2 hours      k4
root@Hafid:/home/hafid# ls /home/hafid/localdata
f1 f2 f3
```

Menjalankan container berbasis image k1 dengan sistem operasi alpine.

```
a656ae61d2dc   alpine    "/bin/sh" 2 hours ago
root@Hafid:/home/hafid# ls /home/hafid/localdata
f1 f2 f3
root@Hafid:/home/hafid# docker start -i k1
/ # ls mydata
f1 f2 f3
```

Kemudian buat star kembali kontainer k1 nya seperti diatas

```
root@Hafid:/home/hafid# docker run -it -v /myfolder --name k2 alpine
/ # cd myfolder
/myfolder # touch f4 f5 f6
/myfolder #
```

Setelah itu buka terminal 2 untuk jalan kan perintah

\$docker ps

```
root@Hafid:/home/hafid# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS        NAMES
5b78fd0d82b0   alpine    "/bin/sh"               3 minutes ago Up 3 minutes          k2
```

\$docker volume ls

```
root@Hafid:/home/hafid# docker volume ls
DRIVER      VOLUME NAME
local       6cccb614d9560f8ef82222baea75dabe62a1974301e5c2d92ec8c70b5aa8d358
local       7b72acbaaf4a9baf9b6b2bf08ed728ca3b6d40f098dd3a4e8bbd1a0e17f388e2
local       7bfc0e4d859cac9371bf30c098e5e30a2e55f33ec669fe1f979e0e593baad44
local       99bf394f63ab76d18957b26861bd39991f094e8349743770eeb050867f9bc5bd
local       ourdata
local       ourdate
```

\$docker inspect k2

```
root@Hafid:/home/hafid# docker inspect k2
[
  {
    "Id": "5b78fd0d82b0907c58a8654be5632f423bf687f4efeb0f9953e53aa4baf44a60",
    "Created": "2025-03-13T12:31:30.471144661Z",
    "Path": "/bin/sh",
    "Args": [],
    "State": {
      "Status": "running",
      "Running": true,
      "Paused": false,
      "Restarting": false,
      "OOMKilled": false,
      "Dead": false,
      "Pid": 50596,
      "ExitCode": 0,
      "Error": "",
      "StartedAt": "2025-03-13T12:31:30.762107861Z",
      "FinishedAt": "0001-01-01T00:00:00Z"
    }
  }
]
```

\$docker volume inspect

Asynchronous volume ini dapat diakses bersama dengan kontainer lain. Buka dari terminal 2, buat kontainer baru dengan nama k3 menggunakan volume dari kontainer k2 tersebut

Buka di terminal 2 seperti gambar dibawah

```
root@Hafid:/home/hafid# docker run -it --volumes-from k2 --name k3 alpine
/# cd myfolder
/myfolder # echo "ini dari k3" > pesan
/myfolder #
```

Gambar di atas Menjalankan container k3 yang dapat mengakses semua data atau volume yang digunakan oleh container k2. Cocok untuk berbagi data antar-container.

```
/myfolder # touch f4 f5 f6
/myfolder # cat pesan
ini dari k3
/myfolder #
```

Ini tampilan pada terminal 1

Ketik di terminal 1 \$echo "balasan dari k2 untuk k3" >> pesan

```
/myfolder # cat pesan
ini dari k3
balasan dari k2 untuk k3
/myfolder #
```

2. Lab 3.2 multi kontainer dengan docker-compose

```
$mkdir -p $HOME/wp-compose/{database,html}
```

```
$cd $HOME/wp-compose
```

```
REPOSITORY TAG IMAGE ID CREATED SIZE
alpine latest aded1e1a5b37 3 weeks ago 7.83MB
root@Hafid:/home/hafid# mkdir -p $HOME/wp-compose/{database,html}
root@Hafid:/home/hafid# cd $HOME/wp-compose/
root@Hafid:~/wp-compose# docker-compose --version
docker-compose version 1.29.2, build unknown
root@Hafid:~/wp-compose# vi docker-compose.yml
root@Hafid:~/wp-compose# docker-compose up -d
```

\$vi docker-compose.yml

```
version: '3.3'

services:
  mariadb:
    image: mariadb
    volumes:
      - ./database:/var/lib/mysql
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: rootpassword
      MYSQL_DATABASE: wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD: wordpress

  wordpress:
    depends_on:
      - mariadb
    image: wordpress:latest
    ports:
      - "8001:80"
    restart: always
    environment:
      WORDPRESS_DB_HOST: mariadb:3306
      WORDPRESS_DB_USER: wordpress
      WORDPRESS_DB_PASSWORD: wordpress
      WORDPRESS_DB_NAME: wordpress
    volumes:
      - ./html:/var/www/html

"docker-compose.yml" 29 lines, 602 bytes
```

Setelah dicopy tekan i untuk masuk mode insert kemudian paste, lalu *ctrl + shift + : qw*

Untuk save file

Kemudian *\$docker - compose up -d* Menjalankan semua layanan yang didefinisikan di docker-compose.yml secara otomatis dan berjalan di latar belakang.

Jalankan di browser <http://localhost:8001> untuk melihat wordpress yang telah dibuat

| localhost:8001/wp-admin/install.php



English (United States)

Afrikaans

አማርኛ

Aragonés

العربية

العربية المغربية

অসমীয়া