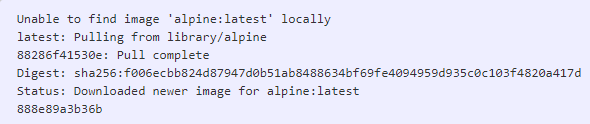
Nama :Temixon Mauboi

Nim :175610065

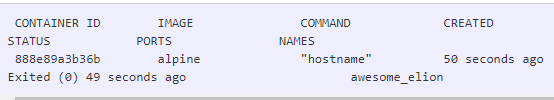
Uas : Teknologi cloud computing





Continer





Run a Docker container and access its shell.



Run the following commands in the container.

ls /

ps aux

cat /etc/issue

exit

Jalankan wadah MySQL baru dengan perintah berikut.

docker container run \

--detach \

--name mydb \

-e MYSQL\_ROOT\_PASSWORD=my-secret-pw \

mysql:latest

--detach akan menjalankan kontainer di latar belakang.

--name akan menamainya mydb.

-e akan menggunakan variabel lingkungan untuk menentukan kata sandi root (CATATAN: Ini seharusnya tidak pernah dilakukan dalam produksi).

Karena gambar MySQL tidak tersedia secara lokal, Docker secara otomatis menariknya dari Docker Hub.

**Daftar wadah yang sedang berjalan.**

docker container ls

**Perhatikan wadah Anda sedang berjalan.**

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

3f4e8da0caf7 mysql:latest "docker-entrypoint..." 52 seconds ago Up 51 seconds 3306/tcp mydb

**Ini menunjukkan log dari wadah MySQL Docker.**

<output truncated>

2017-09-29T16:02:58.605004Z 0 [Note] Executing 'SELECT \* FROM INFORMATION\_SCHEMA.TABLES;' to get a list of tables using the deprecated partition engine. You may use the startup option '--disable-partition-engine-check' to skip this check.

2017-09-29T16:02:58.605026Z 0 [Note] Beginning of list of non-natively partitioned tables

2017-09-29T16:02:58.616575Z 0 [Note] End of list of non-natively partitioned tables

**Mari kita lihat proses yang berjalan di dalam wadah.**

docker container top mydb

**Anda harus melihat daemon MySQL (mysqld) berjalan di wadah.**

PID USER TIME COMMAND

2876 999 0:00 mysqld

**Bangun gambar situs web sederhana**

**Mari kita lihat Dockerfile yang akan kita gunakan, yang membangun situs web sederhana yang memungkinkan Anda mengirim tweet.**

**1.Pastikan Anda berada di direktori linux\_tweet\_app.**

cd ~/linux\_tweet\_app

**2.** **Tampilkan konten Dockerfile**

cat Dockerfile

FROM nginx:latest

COPY index.html /usr/share/nginx/html

COPY linux.png /usr/share/nginx/html

EXPOSE 80 443

CMD ["nginx", "-g", "daemon off;"]

**Mari kita lihat apa yang dilakukan masing-masing baris ini di Dockerfile.**

**FROM menentukan gambar dasar yang akan digunakan sebagai titik awal untuk gambar baru yang Anda buat ini. Untuk contoh ini, kami mulai dari nginx: terbaru.**

**SALIN menyalin file dari host Docker ke dalam gambar, di lokasi yang diketahui. Dalam contoh ini, COPY digunakan untuk menyalin dua file ke dalam gambar: index.html. dan grafik yang akan digunakan di halaman web kami.**

**TAMPILKAN dokumen yang port aplikasi gunakan.**

**CMD menentukan perintah apa yang harus dijalankan ketika sebuah wadah dimulai dari gambar. Perhatikan bahwa kita dapat menentukan perintah, serta argumen run-time.**

**3.Untuk membuat perintah berikut ini lebih ramah salin / tempel, ekspor variabel lingkungan yang berisi DockerID kita (jika kita tidak memiliki DockerID, kita bisa mendapatkannya secara gratis melalui Docker Hub).**

**Kita harus mengetikkan perintah ini secara manual karena membutuhkan DockerID unik kita.**

**export DOCKERID = <id buruh pelabuhan kita>**

**4.Gema nilai variabel kembali ke terminal untuk memastikan itu disimpan dengan benar.**

echo $DOCKERID

**docker image build**

Sending build context to Docker daemon 32.77kB

Step 1/5 : FROM nginx:latest

latest: Pulling from library/nginx

afeb2bfd31c0: Pull complete

7ff5d10493db: Pull complete

d2562f1ae1d0: Pull complete

Digest: sha256:af32e714a9cc3157157374e68c818b05ebe9e0737aac06b55a09da374209a8f9

Status: Downloaded newer image for nginx:latest

---> da5939581ac8

Step 2/5 : COPY index.html /usr/share/nginx/html

---> eba2eec2bea9

Step 3/5 : COPY linux.png /usr/share/nginx/html

---> 4d080f499b53

Step 4/5 : EXPOSE 80 443

---> Running in 47232cb5699f

---> 74c968a9165f

Removing intermediate container 47232cb5699f

Step 5/5 : CMD nginx -g daemon off;

---> Running in 4623761274ac

---> 12045a0df899

Removing intermediate container 4623761274ac

Successfully built 12045a0df899

Successfully tagged <your docker ID>/linux\_tweet\_app:latest

**Perintah jaringan buruh pelabuhan adalah perintah utama untuk mengonfigurasi dan mengelola jaringan kontainer. Jalankan perintah jaringan buruh pelabuhan dari terminal pertama.**

docker network

Usage: docker network COMMAND

Manage networks

Options:

--help Print usage

Commands:

connect Connect a container to a network

create Create a network

disconnect Disconnect a container from a network

inspect Display detailed information on one or more networks

ls List networks

prune Remove all unused networks

rm Remove one or more networks

Run 'docker network COMMAND --help' for more information on a command.

**Langkah 2: Daftar jaringan**

docker network ls

NETWORK ID NAME DRIVER SCOPE

3430ad6f20bf bridge bridge local

a7449465c379 host host local

06c349b9cc77 none null local

**Langkah.3**

docker network inspect bridge

[

{

"Name": "bridge",

"Id": "3430ad6f20bf1486df2e5f64ddc93cc4ff95d81f59b6baea8a510ad500df2e57",

"Created": "2017-04-03T16:49:58.6536278Z",

"Scope": "local",

"Driver": "bridge",

"EnableIPv6": false,

"IPAM": {

"Driver": "default",

"Options": null,

"Config": [

{

"Subnet": "172.17.0.0/16",

"Gateway": "172.17.0.1"

}

]

},

"Internal": false,

"Attachable": false,

"Containers": {},

"Options": {

"com.docker.network.bridge.default\_bridge": "true",

"com.docker.network.bridge.enable\_icc": "true",

"com.docker.network.bridge.enable\_ip\_masquerade": "true",

"com.docker.network.bridge.host\_binding\_ipv4": "0.0.0.0",

"com.docker.network.bridge.name": "docker0",

"com.docker.network.driver.mtu": "1500"

},

"Labels": {}

}

]

**Langkah.4**

docker info

Containers: 0

Running: 0

Paused: 0

Stopped: 0

Images: 0

Server Version: 17.03.1-ee-3

Storage Driver: aufs

<Snip>

Plugins:

Volume: local

Network: bridge host macvlan null overlay

Swarm: inactive

Runtimes: runc

<Snip>

**Bagian # 2 - Jembatan Jaringan**

docker network ls

NETWORK ID NAME DRIVER SCOPE

3430ad6f20bf bridge bridge local

a7449465c379 host host local

06c349b9cc77 none null local

apk update

apk add bridge

brctl show

bridge name bridge id STP enabled interfaces

docker0 8000.024252ed52f7 no

ip a

<Snip>

3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default

link/ether 02:42:52:ed:52:f7 brd ff:ff:ff:ff:ff:ff

inet 172.17.0.1/16 scope global docker0

valid\_lft forever preferred\_lft forever

**Hubungkan sebuah wadah**

docker run -dt ubuntu sleep infinity

Unable to find image 'ubuntu:latest' locally

latest: Pulling from library/ubuntu

d54efb8db41d: Pull complete

f8b845f45a87: Pull complete

e8db7bf7c39f: Pull complete

9654c40e9079: Pull complete

6d9ef359eaaa: Pull complete

Digest: sha256:dd7808d8792c9841d0b460122f1acf0a2dd1f56404f8d1e56298048885e45535

Status: Downloaded newer image for ubuntu:latest

846af8479944d406843c90a39cba68373c619d1feaa932719260a5f5afddbf71

docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

846af8479944 ubuntu "sleep infinity" 55 seconds ago Up 54 seconds heuristic\_boy

brctl show

bridge name bridge id STP enabled interfaces

docker0 8000.024252ed52f7 no vethd630437

docker network inspect bridge

<Snip>

"Containers": {

"846af8479944d406843c90a39cba68373c619d1feaa932719260a5f5afddbf71": {

"Name": "heuristic\_boyd",

"EndpointID": "1265c418f0b812004d80336bafdc4437eda976f166c11dbcc97d365b2bfa91e5",

"MacAddress": "02:42:ac:11:00:02",

"IPv4Address": "172.17.0.2/16",

"IPv6Address": ""

}

},

<Snip>

.

ping -c5 172.17.0.2

PING 172.17.0.2 (172.17.0.2) 56(84) bytes of data.

64 bytes from 172.17.0.2: icmp\_seq=1 ttl=64 time=0.055 ms

64 bytes from 172.17.0.2: icmp\_seq=2 ttl=64 time=0.031 ms

64 bytes from 172.17.0.2: icmp\_seq=3 ttl=64 time=0.034 ms

64 bytes from 172.17.0.2: icmp\_seq=4 ttl=64 time=0.041 ms

64 bytes from 172.17.0.2: icmp\_seq=5 ttl=64 time=0.047 ms

--- 172.17.0.2 ping statistics ---

5 packets transmitted, 5 received, 0% packet loss, time 4075ms

rtt min/avg/max/mdev = 0.031/0.041/0.055/0.011 ms

3.[Application Containerization and Microservice Orchestration](https://training.play-with-docker.com/microservice-orchestration/). Bagian ini menjelaskan tentang Docker Compose untuk App Containerization and Orchestration.

git clone https://github.com/ibnesayeed/linkextractor.git

cd linkextractor

git checkout demo

git checkout step0

tree

.

├── README.md

└── linkextractor.py

0 directories, 2 files

cat linkextractor.py

*#!/usr/bin/env python*

import sys

import requests

from bs4 import BeautifulSoup

res **=** requests.get(sys.argv[**-**1])

soup **=** BeautifulSoup(res.text, "html.parser")

**for** link **in** soup.find\_all("a"):

**print**(link.get("href"))

./linkextractor.py http://example.com/

bash: ./linkextractor.py: Permission denied

ls -l linkextractor.py

-rw-r--r-- 1 root root 220 Sep 23 16:26 linkextractor.py

python linkextractor.py

Traceback (most recent call last):

File "linkextractor.py", line 5, **in** **<**module**>**

from bs4 import BeautifulSoup

ImportError: No module named bs4