**Jobsheet 11 – Docker 2**

**CLOUD COMPUTING**

**JURUSAN TEKNOLOGI INFORMASI**

A picture containing logo

Description automatically generated

**POLITEKNIK NEGERI MALANG**

**OKTOBER 2021**

# BAB 11 : Docker 2

Docker Network Bridge

* Merupakan jenis network bawaan dari Docker
* Ketika Docker service daemon berjalan, akan melakukan konfigurasi sebuah virtual bridge disebut dengan docker0.
* Jika kita tidak menyebutkan spesifik network yang digunakan, docker run –net=<Network> maka Docker daemon akan terhubung ke container menggunakan bridge network.
* Masing-masing container yang dibuat, akan dialokasikan sebuah perangkat virtual Ethernet dan dipetakkan sebagai eth0 pada container.

Graphical user interface, diagram, application

Description automatically generated

* Dalam container interface eth0 akan diberi sebuah IP address dari rentang alamat bridge.
* Docker akan mencari sebuah IP address yang tersedia pada bridge dan akan mengkonfigurasi interface eth0 container dengan IP address.
* Ketika container ingin terhubung ke internet akan menggunakan bridge. Bridge akan secara automatis meneruskan paket antara interface network yang lain asalkan terhubung dan juga diizinkan container berkomunikasi dengan mesin host.

Docker Network Host

* Docker network jenis ini yaitu meletakkan container di dalam stack hostnya network.
* Semua antarmuka jaringan yang telah didefinisikan dapat diakses melalui container.
* Ketika kita menjalankan container menggunakan parameter –net=host, sehingga container akan menggunakan network host.
* Ketika menggunakan host network akan secepat jaringan normal karena tidak ada menggunakan bridge ataupun translation

Graphical user interface, application

Description automatically generated

* ketika ingin membutuhkan mendapatkan performa network terbaik menggunakan host.
* Container yang berjalan di dalam stack hostnya network akan mendapatkan jauh lebih cepat dibandingkan container yang berjalan menggunakan network bridge, tidak membutuhkan traverse pada docker0 bridge dan mappings port iptables.
* Container membagi networknya dari host yang langsung ter-expose ke dunia luar.
* Ketika menggunakan perintah –net=host, container kita akan dapat diakses melalui hostnya IP address.
* Kita butuh berhati-hati ketik menggunkaan network host karena berbahaya.

Docker Network None

* Network None bertujuan untuk tidak mengkonfigurasi networking.
* Tidak ada driver yang digunakan pada Network jenis ini.
* Ketika kita tidak menginginkan container untuk dapat diakses. Perintah yang digunakan dengan menambahkan parameter –net=none

Perintah Networking Docker

* Perintah yang dasar digunakan adalah *docker network* kemudian ditambahkan beberapa parameter setelah perintah dasar.
* Docker network ls; untuk mengetahui network yang tersedia pada host docker
* Docker network create; untuk membuat sebuah network pada host docker.
* Docker network rm; untuk menghapus network yang terdapat pada host docker.
* Docker network connect; untuk menghubungkan container pada spesifik network
* Docker network disconnect; memutus network pada container
* Docker network inspect; untuk mengetahui secara terperinci dari sebuah network, misalkan ip address dan informasi lain

Expose dan mapping port

* Skenario yang umum digunakan ketika kita menginginkan container menerima koneksi yang datang, salah satunya dari container yang lain atau dari luar docker.
* Aplikasi server yang listening pada port 80 atau sebuah database yang menerima request yang datang.
* Sebuah image dapat juga melakukan expose port.
* Expose port adalah container aplikasi kita akan listen pada port yang ter-expose. Sebagai contoh, aplikasi server Tomcat akan listen pada port default yaitu 8080.
* Semua container yang berjalan pada host yang sama dan network yang sama dapat berkomunikasi dengan Tomcat pada port 8080.
* Expose port dapat dilakukan menggunakan 2 cara yaitu salah satunya adalah menggunakan Dockerfile dengan perintah EXPOSE dan menggunakan perintah docker run –expose.
* Ketika container berbeda host atau network agar tetap bisa berkomunikasi menggunakan maping port.
* Mapping port dilakuan dengan menambahkan parameter –p ketika menjalankan container.

Persistent Storage (Volume)

* Persistent storage pada docker bertujuan agar data pada container tetap terjaga walaupun container telah dihapus
* Persitence storage juga dapat berbagi data antara container yang lain ataupun dengan host docker.
* Sebagai contoh ketika kita membuat sebuah container yang bertujuan untuk menyimpan data transaksi pada sebuah database transactional, ketika menggunakan volume data yang tersimpan pada database transactional akan tetap persistent walaupun kita sudah menghapus container.

Perintah Docker Volume

* Docker volume create; untuk membuat sebuah volume
* Docker volume inspect; menampilkan detail informasi pada satu atau lebih dari satu volume.
* Docker volume ls; menampilkan daftar volume.
* Docker volume rm; menghapus satu atau lebih volume yang terdapat pada docker host.
* Docker volume prune; menghapus semua volume yang tidak digunakan, semua volume yang sudah tidak digunakan oleh container akan dihapus.

Docker Volume Driver

Sama halnya dengan network yang memiliki driver, volume juga memiliki jenis-jenis volume agar mampu terintegrasi dengan jenis penyimpanan yang lain.

* Docker volume driver for Azure file storage; sebuah driver docker volume yang digunakan Azure file storage untuk mount file yang dishare pada cloud ke Docker container sebagai volume.
* IPFS(InterPlanetary File System); sebuah plugin volume open source yang mengizinkan menggunakan sebuah IPFS filesystem sebagai sebuah volume. IPFS adalah sebuah sistem storage yang sangat menarik dan menjanjikan, memungkinkan dalam mendistribusikan high volume data dengan efisiensi yang tinggi.
* Keywhiz; kita dapat menggunakan drive ini untuk membuat container kita berkomunikasi secara remote pada server Keywhiz. Keywhiz adalah sebuah sistem untuk mengelola dan mendistribusikan data yang bersifat rahasia, seperti TLS certificate/keys, GPG keys, API tokens, dan database credential. Selain meletakkan data di dalam file-file config atau meng-copy file-file tersebut, keywhiz membuat pengelolaaan akan lebih mudah dan lebih aman.

Docker Compose

* Docker compose adalah sebuah tool untuk mendefiniskan, launching, dan mengelola service, dimana sebuah service adalah didefinisikan sebagai satu atau lebih dari sebuah docker container.
* Service dan sistem service didefinisikan di dalam file YAML dan mengelolanya menggunakan perintah command-line docker-compose.
* Beberapa hal yang bisa dilakukan menggunakan docker compose adalah build docker image, menjalankan aplikasi container sebagai service, menjalankan full system dari service, mengelola secara individu service pada sebuah sistem, melakukan scaling up atau down, dan melihat log dari kumpulan container yang berjalan sebagai service.

Perintah Dasar Docker Compose

* Docker-compose up; membuat dan menjalankan container, ketika ditambakan parameter –d maka container akan berjalan sebagai daemon.
* Docker-compose down; menghentikan dan menghapus semua resource
* Docker-compose logs; melihat logging pada container
* Docker-compose ps; melihat container yang sedang berjalan

## Praktikum

1. **Docker Network**

Pada praktikum yang pertama akan melakukan pembuatan network pada docker agar container yang dibuat bisa terhubung ke server lain atau berkomunikasi dengan dunia luar, dan juga container yang lain.

* 1. **Membuat dan Memeriksa Docker Network**

Beberapa perintah yang dapat digunakan adalah sebagai berikut

* Pertama ketik perintah di bawah ini untuk mengetahui network yang tersedia pada docker host kita

A screenshot of a computer

Description automatically generated with medium confidence

docker network ls

* Selanjutnya untuk mengetahui perintah-perintah dasar yang bisa digunakan untuk docker network

docker network help

Text

Description automatically generated

* Untuk membuat network bisa menggunakan perintah create seperti pada gambar di bawah ini

docker network create myNetwork

Text

Description automatically generated

* Selanjutnya cek network yang sudah dibuat menggunkan perintah ls, seharusnya akan menambahkan network. Hasilnya adalah sebagai berikut

docker network ls

A screenshot of a computer

Description automatically generated with medium confidence

* Terakhir, ketika ingin mengetahui informasi detail dari network yang telah kita buat bisa menggunakan perintah inspect

docker network inspect myNetwork

Text

Description automatically generated

Dari praktikum pertama kita telah membuat network yaitu myNetwork, network tersebut selanjutnya bisa digunakan untuk menghubungkan container satu dengan yang lain.

* 1. **Menghubungkan Container ke Network**

Silakan menjalankan perintah-perintah di bawah ini untuk memanfaatkan network yang sebelumnya telah kita buat

* Buat terlebih dahulu contoh container, misalkan kita akan membuat container linux yaitu busybox.
* d

Text

Description automatically generated

* Buat container yang kedua menggunakan image linux alphine, buka windows yang baru agar container1 tetap berjalan
* docker run -it --name container2 --net=myNetwork --rm alpine:latest

Graphical user interface

Description automatically generated with medium confidence

Jangan lupa mengubah nama container dari yang sebelumnya.

* Kemudian cek masing-masing container tersebut menggunakan perintah ps untuk mengetahui status container yang sedang jalan. Buka windows baru, jalankan perintah berikut
* docker ps

Graphical user interface, text

Description automatically generated

* Kemudian jalankan inspect untuk mengetahui informasi detail dari masing-masing container.
* docker inspect container1 | grep IPAddress

Text

Description automatically generated

* Untuk membuktikan silakan ping dari container1 ke container2, seharusnya akan reply.

Text

Description automatically generated

Setelah melakukan konfigurasi network selanjutnya bisa dimanfaatkan untuk diimplementasikan pada image aplikasi yang kita buat sendiri.

* 1. **Membuka Port Container**

Scenario yang umum sering kita lakukan agar aplikasi dapat menerima request adalah dengan mengizinkan akses pada port tertentu, sama halnya dengan container. Harus dilakukan expose agar dapat menerima request dari lingkungan luar, beberapa hal yang dapat dilakukan adalah sebagai berkut

* Misalkan kita ingin membuat image dari apache, layanan httpd. Bisa dilakukan dengan perintah di bawah ini

Text

Description automatically generated

* Silakan cek menggunakan telnet terhadap port yang di-expose, seharusnya akan menghasilkan output seperti berikut

Text

Description automatically generated

* Atau Anda bisa ngetik pada browser seperti ditunjukkan di bawah ini

Graphical user interface, text, application, chat or text message

Description automatically generated

* Kemudian untuk mengetahui port yang dilakukan expose pada container, bisa menggunakan perintah di bawah ini

Graphical user interface, text, application, chat or text message

Description automatically generated

Setelah mengetahui cara untuk melakukan expose dan mapping port, dapat diterapkan ketika Anda membuat container dan container tersebut bisa menerima request dari luar.

1. **Docker Volume**

Ketika kita membuat container dan dalam keadaan tertentu data-data yang terdapat container tersebut harus tetap dipertahankan walaupun container sudah dihapus, hal ini tetap bisa dilakukan dengan konsep persistent storage.

* 1. **Membuat Volume**

Ada beberapa scenario yang bisa dilakukan, yaitu membuat volume di host container untuk di-share ke semua container dan membuat volume di docker untuk di-share ke semua container.

* Buat terlebih dahulu folder di host(laptop/computer/vps), misalkan saya buat folder docker-shared.

Graphical user interface, text, application, website

Description automatically generated

* Selanjutkan jalankan perintah dengan menambahkan parameter -v seperti berikut untuk mapping ke container yang akan kita buat, jangan lupa untuk masuk direktori volume, dan membuat file log.txt

A screenshot of a computer

Description automatically generated with medium confidence

* Cek di folder local Anda, bisa menggunakan perintah ls atau dir. Dapat juga dilihat menggunakan windows explorer.

Graphical user interface, application

Description automatically generated with medium confidence

Setelah kita melakukan mapping volume dari container ke local direktori, kemudian kita akan membuat volume pada docker.

* Perintah pertama yang bisa dijalankan adalah sebagai berikut, mengecek volume yang telah ada sebelumnya.

Text

Description automatically generated

* Untuk membuat volume bisa menggunakna perintah di bawah ini

Text

Description automatically generated

* Jalankan container dengan menyertakan volume yang telah kita sebelumnya dan buat file log.txt, jangan lupa masuk ke direktori volume terlebih dahlulu.

Graphical user interface, text, application

Description automatically generated

* Buat container yang lain, dengan mengulangi langkah yang sama sebelumnya. Ketik perintah ls untuk mengetahui isi folder volume

Text

Description automatically generated

* 1. **Menghapus Volume**

Untuk menghapus volume yang telah kita buat sebelumnya dapat dilakukan dengan cara di bawah ini

Graphical user interface, text, application, chat or text message

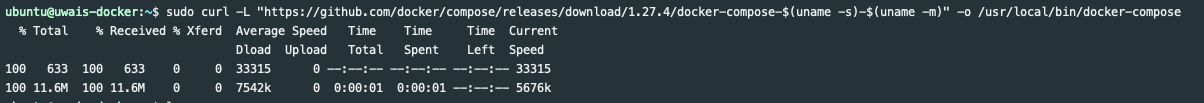
Description automatically generated

Sebelum menghapus volume, pastikan terlebih dahulu volume tersebut tidak digunakan pada container.

1. **Docker Compose**

Dengan menggunakan docker compose Anda bisa menggabungkan beberapa container dengan sekali jalan, biasanya menggunakan file YAML untuk konfigurasinya. Sebelumnya silakan mengikuti Langkah-langkah berikut ini untuk mencoba docker compose.

* Silakan masuk ke oracle cloud dan ke VM Anda, kemudian install docker compose dengan perintah *sudo curl -L "https://github.com/docker/compose/releases/download/1.27.4/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose*



* Ubah docker compose menjadi executable menggunakan command berikut



Cek versi docker compose untuk memastikan proses installasi berhasil dilakukan dengan perintah



* Clone repo yang telah disediakan menggunakan git atau download secara manual.



Masuk ke direktori tersebut dan ganti branch menjadi cloud-docker dengan perintah seperti berikut ini

Text

Description automatically generated

* Build image tersebut menggunakan perintah build pada docker.



* Selanjutnya buat folder data di bawah /opt agar data mysql tidak hilang ketika container dihapus



* Jalankan docker compose menggunakan perintah berikut ini



Jangan lupa untuk open port 8090, karena port tersebut yang di-expose.

* Selanjutnya cek hasilnya pada browser dengan mengakses VM Anda, hasilnya jika tidak terjadi error seperti berikut

Graphical user interface, text, application

Description automatically generated

* Untuk mengetahui container yang sedang berjalan menggunakan docker compose bisa dilakukan menggunakan cara sebagai berikut

Graphical user interface, text

Description automatically generated

## Tugas

1. Buatlah laporan dan dokumentasi dari praktikum yang Anda lakukan.
2. Silakan konfigurasi aplikasi yang terdiri lebih dari satu container menggunakan docker compose kemudian jalankan di Oracle Cloud
3. Kirim laporan dalam bentuk pdf dengan format :

Kelas\_Absen\_Nama\_KomputasiAwan\_jobsheet07.pdf

contoh : TI3A\_25\_Alonso\_KomputasiAwan\_jobsheet10.pdf

# **Referensi**

[1] J. Nickoloff, *Docker in action*. Shelter Island, NY: Manning Publications, Co, 2016.

[2] S. Gallagher, *Mastering Docker: rethink what’s possible with Docker - become an expert in the innovative containerization tool to unlock new opportunities in the way you use and deploy software*. Birmingham Mumbai: Packt Publishing, 2015.

[3] J. Krochmalski, *Docker and Kubernetes for Java developers: scale, deploy, and monitor multi-container applications*, First published. Birmingham Mumbai: Packt Publishing, 2017.