Modul Praktikum 5

Containerization dengan Docker

1. Maksud

Praktikum ini bertujuan untuk memberikan pemahaman kepada peserta tentang konsep containerization menggunakan Docker. Peserta akan mempelajari cara membuat, menjalankan, dan mengelola container Docker pada sistem operasi Windows 11, dengan fokus pada aplikasi berbasis PHP.

---

2. Tujuan

Setelah menyelesaikan praktikum ini, peserta diharapkan mampu:

a) Memahami konsep dasar containerization dan peran Docker dalam pengembangan aplikasi.

b) Menginstal dan mengonfigurasi Docker Desktop pada Windows 11.

c) Membuat container Docker untuk menjalankan aplikasi berbasis PHP.

d) Mengelola container Docker (start, stop, delete, dan inspect).

e) Menerapkan containerization dalam studi kasus sederhana.

---

3. Dasar Teori

Apa itu Containerization?

Containerization adalah teknologi yang memungkinkan pengemasan aplikasi dan dependensinya ke dalam unit yang ringan dan portabel, sehingga dapat dijalankan secara konsisten di berbagai lingkungan.

Apa itu Docker?

Docker adalah platform open-source yang menyediakan alat untuk membuat, mengelola, dan menjalankan container. Docker memungkinkan pengembang untuk mengemas aplikasi mereka beserta semua dependensi ke dalam container yang dapat dijalankan di mana saja tanpa masalah kompatibilitas.

Komponen Utama Docker:

a) Docker Engine : Inti dari Docker yang bertanggung jawab untuk menjalankan container.

b) Docker Image : Template read-only yang digunakan untuk membuat container.

c) Docker Container : Instance runtime dari sebuah image.

d) Dockerfile : File teks yang berisi instruksi untuk membangun image Docker.

Keuntungan Docker:

a) Portabilitas: Aplikasi dapat dijalankan di berbagai lingkungan tanpa perubahan konfigurasi.

b) Isolasi: Setiap container berjalan secara independen, menghindari konflik antar aplikasi.

c) Efisiensi: Lebih ringan dibandingkan virtual machine karena tidak memerlukan sistem operasi tambahan.

---

4. Prosedur Praktik

Langkah 1: Instalasi Docker Desktop

a) Unduh Docker Desktop dari situs resmi: *https://www.docker.com/products/docker-desktop* .

b) Jalankan installer dan ikuti petunjuk instalasi.

c) Pastikan WSL 2 (Windows Subsystem for Linux) sudah diaktifkan di Windows 11. Jika belum, aktifkan melalui PowerShell:

wsl --install

d) Setelah instalasi selesai, buka Docker Desktop dan pastikan statusnya "Running".

Langkah 2: Membuat Container PHP

a) Buka terminal atau PowerShell.

b) Jalankan perintah berikut untuk menarik image PHP dari Docker Hub:

docker pull php:8.2-cli

c) Buat direktori kerja dan file PHP sederhana:

mkdir php-app
cd php-app
echo "<?php echo 'Hello, Docker!'; ?>" > index.php

d) Jalankan container PHP dengan perintah berikut:

docker run -d -p 8080:80 --name php-container -v ${PWD}:/var/www/html php:8.2-cli php -S 0.0.0.0:80

Keterangan:

1) **-d**: Menjalankan container dalam mode detached.

2) **-p 8080:80**: Memetakan port 8080 host ke port 80 container.

3) **--name php-container**: Memberi nama container.

4) **-v ${PWD}:/var/www/html**: Memetakan direktori lokal ke direktori container.

e) Buka browser dan akses **http://localhost:8080/index.php**. Anda akan melihat output "Hello, Docker!".

Langkah 3: Membuat dan Menjalankan Dockerfile

Dockerfile adalah file teks yang berisi instruksi untuk membangun image Docker. Berikut adalah langkah-langkah membuat dan menjalankan Dockerfile untuk aplikasi PHP:

a. Persiapan Direktori Kerja

1. Buat direktori kerja baru untuk proyek Anda:

   mkdir php-docker-app
   cd php-docker-app
2. Buat file PHP sederhana bernama **index.php**:

   <?php
   echo "Hello from Dockerized PHP App!";
   ?>

b. Membuat Dockerfile

1. Di dalam direktori **php-docker-app**, buat file bernama **Dockerfile** (tanpa ekstensi):

   touch Dockerfile

2. Buka file **Dockerfile** menggunakan editor teks (misalnya Notepad, VS Code, atau Nano), lalu tambahkan konten berikut:

   # Menggunakan base image PHP
   FROM php:8.2-cli
   # Menyalin file PHP ke dalam container
   COPY . /usr/src/app
   # Mengatur direktori kerja
   WORKDIR /usr/src/app
   # Perintah yang akan dijalankan saat container dimulai
   CMD ["php", "-S", "0.0.0.0:80"]

Keterangan:

1) **FROM php:8.2-cli**: Menggunakan image PHP versi 8.2 CLI sebagai base image.
2) **COPY . /usr/src/app**: Menyalin semua file dari direktori lokal ke **/usr/src/app** di container.
3) **WORKDIR /usr/src/app**: Mengatur direktori kerja di dalam container.
4) **CMD ["php", "-S", "0.0.0.0:80"]**: Menjalankan server PHP bawaan pada port 80.

c. Membangun Image Docker

1. Di terminal, jalankan perintah berikut untuk membangun image Docker berdasarkan Dockerfile:

   docker build -t php-docker-app .

Keterangan:

- **-t php-docker-app**: Memberi nama image dengan tag **php-docker-app**.
- **.**: Menunjukkan bahwa Dockerfile berada di direktori saat ini.

2. Setelah proses build selesai, verifikasi image yang telah dibuat:

   docker images

Anda akan melihat image **php-docker-app** dalam daftar.

d. Menjalankan Container dari Image

1. Jalankan container dari image yang telah dibuat:

   docker run -d -p 8080:80 --name php-docker-container php-docker-app

Keterangan:

- **-d**: Menjalankan container dalam mode detached.
- **-p 8080:80**: Memetakan port 8080 host ke port 80 container.
- **--name php-docker-container**: Memberi nama container.

2. Buka browser dan akses **http://localhost:8080/index.php**. Anda akan melihat output "Hello from Dockerized PHP App!".

Langkah 4: Mengelola Container

a) Melihat daftar container yang sedang berjalan:

   docker ps

b) Menghentikan container:

   docker stop php-container

c) Memulai kembali container:

   docker start php-container

d) Menghapus container:

   docker rm php-container

---

5. Latihan

Studi Kasus:

Anda diminta untuk membuat container Docker untuk menjalankan aplikasi PHP yang menampilkan waktu saat ini.

Penyelesaian:

1. Buat file PHP baru bernama **time.php** di direktori **php-app**:

   ```
   <?php
   echo "Current time: " . date('Y-m-d H:i:s');
   ?>
   ```

2. Jalankan container PHP seperti pada prosedur sebelumnya:

   docker run -d -p 8080:80 --name php-time-container -v ${PWD}:/var/www/html php:8.2-cli php -S 0.0.0.0:80

3. Buka browser dan akses **http://localhost:8080/time.php**. Anda akan melihat waktu saat ini.

---

6. Tugas

Tugas 1: Membuat Container PHP dengan MySQL

Buat container Docker untuk menjalankan aplikasi PHP yang terhubung ke database MySQL. Ikuti langkah-langkah berikut:

1. Buat file PHP bernama **db-test.php** yang menghubungkan ke database MySQL dan menampilkan data dari tabel **users**.
2. Gunakan image **mysql:8.0** untuk menjalankan container MySQL.
3. Hubungkan container PHP dan MySQL menggunakan Docker network.

Tugas 2: Membuat Dockerfile

Buat Dockerfile untuk membangun image PHP kustom yang sudah mencakup ekstensi MySQL. Kemudian gunakan image tersebut untuk menjalankan aplikasi PHP Anda.