

# **Tugas MK. ITI**

***“Instalasi nginx, php-fpm, mariadb pada docker dan membuat aplikasi CRUD”***

Dosen Pengampu : R. Dimas Adityo



Disusun Oleh :

Fernando Yunianda Dwi Firmansyah (2314311007)

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNIK  
UNIVERSITAS BHAYANGKARA SURABAYA**

**2024**

# KATA PENGANTAR

Puji syukur saya ucapkan kehadirat Allah SWT yang telah memberikan rahmat serta hidayah kepada kita semua, sehingga berkat Karunia-Nya penulis dapat menyelesaikan makalah ini guna memenuhi Tugas Akhir UAS untuk mata kuliah Infrastruktur Teknologi Informasi, dengan judul *“Instalasi nginx, php-fpm, mariadb pada docker dan membuat aplikasi CRUD”*.

Saya menyadari bahwa dalam penulisan makalah ini tidak terlepas dari bantuan banyak pihak yang dengan tulus memberikan doa, saran dan kritik sehingga makalah ini dapat terselesaikan.

Saya menyadari sepenuhnya bahwa makalah ini masih jauh dari sempurna dikarenakan terbatasnya pengetahuan yang saya miliki. Oleh karena itu, kami mengharapkan segala bentuk saran serta masukan bahkan kritik yang membangun dari berbagai pihak. Dalam penyusunan makalah ini penulis berharap semoga makalah ini dapat bermanfaat bagi penulis sendiri maupun kepada pembaca umumnya.

Penyusun

# Daftar Isi

KATA PENGANTAR.....	2
BAB I .....	4
PENDAHULUAN.....	4
1. 1 Latar Belakang.....	4
1.2 Rumusan Masalah .....	4
1.3 Tujuan Masalah.....	4
BAB II .....	5
PEMBAHASAN .....	5
2.1. Instalasi Nginx meggunakan docker-compose.....	5
2.2 membuat kontainer php dan mengkonfigurasinya.....	6
2.3 membuat kontainer Mariadb dan mengkonfigurasinya .....	9
BAB III.....	11
PENUTUP .....	11
3.1. Kesimpulan.....	11
3.2. Saran.....	11

# BAB I

## PENDAHULUAN

### ***1.1 Latar Belakang***

Aplikasi berbasis web merupakan aplikasi yang mempermudah seseorang untuk mencari informasi, bertransaksi, atau melakukan hal lainnya melalui layanan internet. Aplikasi berbasis web sangat mudah untuk digunakan karena dapat diakses di berbagai platform computer hanya dengan menjalankan web browser.

Dengan kemajuan pengetahuan dalam ilmu komputer, berbagai macam bahasa pemrograman pun telah banyak dibuat untuk menjalankan sebuah aplikasi berbasis web. Bahasa pemrograman yang digunakan antara lain yaitu *php*.

Dengan perkembangan teknologi yang begitu pesat ini, pembuatan aplikasi berbasis web pun ikut mengalami perkembangan dalam hal arsitektur. Arsitektur yang digunakan dalam pembuatan web bernama *Microservice*. *Microservice* adalah sebuah arsitektur dimana aplikasi dipecah-pecah menjadi service-service kecil dengan menggunakan bahasa pemrograman yang sama atau berbeda. Tujuan dari *Microservice* adalah untuk membuat sebuah aplikasi yang mampu berjalan secara independent.

*Docker* sangat mendukung untuk pembuatan *Microservice*, dengan menggunakan *docker* pembuatan aplikasi dan memasukkannya menjadi sebuah image yang nanti dijalankan menjadi sebuah *container*. Adanya *Microservice* membuat aplikasi *website* lebih mudah karena dapat membagi service menjadi kecil-kecil. Akan tetapi dengan adanya hal itu akan mempengaruhi *server* yang bekerja sebagai penyedia layanan dari aplikasi web tersebut serta bahasa pemrograman yang digunakan dalam pembuatan *website* akan mempengaruhi juga.

Pada tugas ini akan dilakukan terhadap bahasa pemrograman *php* dengan menggunakan *docker container*.

### ***1.2 Rumusan Masalah***

1. Bagaimana membuat image dengan menggunakan *dockerfile*?
2. Bagaimana cara menjalankan *Container* di *Docker* menggunakan *docker compose* ?
3. Bagaimana cara cloning code dari github?

### ***1.3 Tujuan Masalah***

1. Untuk mengetahui cara membuat image dengan menggunakan *dockerfile*
2. Untuk mengetahui bagaimana menjalankan *Container* di *Docker* menggunakan *docker compose*.
3. Untuk cloning code dari github.

## BAB II

### PEMBAHASAN

#### 2.1. Instalasi kontainer nginx

Untuk menginstall nginx di docker dengan menggunakan docker compose meliputi:

##### 1. Buat file dengan nama docker-project

Ketik code berikut di cmd :

```
mkdir ~/docker-project  
cd ~/docker-project
```

##### 2. Buat file *docker-compose.yml*

Ketika sudah membuat file docker-compose.yml di folder docker-project buka docker-compose.yml dengan text editor kalian disini kami menggunakan text editor VS Code ketik perintah `code .` untuk membuka vs code

##### 3. Tambahkan konfigurasi berikut ke *docker-compose.yml* file:

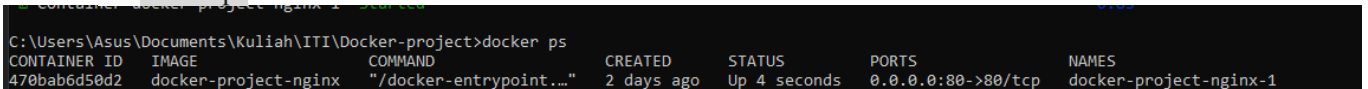
```
version: "3.9"  
services:  
  nginx:  
    image: nginx:latest  
    container_name: nginx-container  
    ports:  
      - 80:80
```

##### 4. Jalankan perintah

```
Docker-compose up -d
```

##### 5. Cek apakah container berjalan dengan perintah

```
docker ps
```



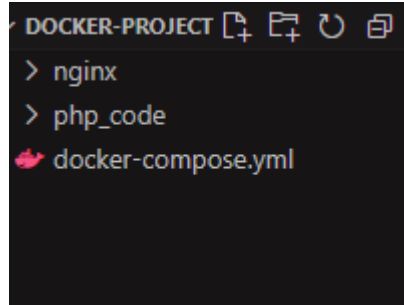
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
470bab6d50d2	docker-project-nginx	"/docker-entrypoint...."	2 days ago	Up 4 seconds	0.0.0.0:80->80/tcp	docker-project-nginx-1

## 2.2 Membuat kontainer php

### 1. Buat direktori *php\_code* di dalam folder *docker-project*

```
mkdir ~/docker-project/php_code
```

susunan file yang ada di dalam *docker-project* seperti :



### 2. Kloning code dari github untuk mendapatkan source code php crud dengan menggunakan git bash

```
Git clone https://github.com/chapagain/crud-php-simple.git
```

### 3. Buat *dockerfile* untuk container *php\_code*

```
mkdir ~/docker-project/php_code/Dockerfile
```

### 4. Tambahkan kode berikut di dalam *dockerfile*

```
FROM php:7.0-fpm
RUN docker-php-ext-install mysqli pdo pdo_mysql
RUN docker-php-ext-enable mysqli
```

### 5. Buat direktori *nginx* di dalam *docker-project* dan buat file *default.conf* didalam file *nginx*

```
mkdir ~/docker-project/nginx
mkdir ~/docker-project/nginx/default.conf
```

### 6. Masukkan kode berikut kedalam *default.conf*

```
server {

    listen 80 default_server;
    root /var/www/html;
    index index.html index.php;

    charset utf-8;

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

    location = /favicon.ico { access_log off; log_not_found off; }
    location = /robots.txt { access_log off; log_not_found off; }

    access_log off;
    error_log /var/log/nginx/error.log error;
```

```

sendfile off;

client_max_body_size 100m;

location ~ .php$ {
    fastcgi_split_path_info ^(.+\.php)(/.+)$;
    fastcgi_pass php:9000;
    fastcgi_index index.php;
    include fastcgi_params;
    fastcgi_read_timeout 300;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    fastcgi_intercept_errors off;
    fastcgi_buffer_size 16k;
    fastcgi_buffers 4 16k;
}

location ~ /\.ht {
    deny all;
}

```

**7. Buat dockerfile di dalam folder nginx dan masukan kode berikut**

```
mkdir ~/docker-project/nginx/Dockerfile
```

```
FROM nginx
COPY ./default.conf /etc/nginx/conf.d/default.conf
```

**8. Ubah konfigurasi di dalam docker-compose.yml menjadi seperti ini:**

```

version: "3.9"
services:
  nginx:
    build: ./nginx/
    ports:
      - 80:80

  volumes:
    - ./php_code/:/var/www/html/

  php:
    build: ./php_code/
    expose:
      - 9000
    volumes:
      - ./php_code/:/var/www/html/

```

**9. Lalu jalankan kontainer dengan perintah**

```
Docker-compose up -d
```

**10. Lalu lihat apakah kontainer berjalan dengan perintah**

```
docker ps
```

```
C:\Users\Asus\Documents\Kuliah\ITI\docker-project>docker ps
CONTAINER ID   IMAGE                COMMAND                  CREATED        STATUS        PORTS                    NAMES
470bab6d50d2   docker-project-nginx "/docker-entrypoint..." 2 days ago    Up 4 seconds   0.0.0.0:80->80/tcp      docker-project-nginx-1
5f40fe66d792   docker-project-php  "/docker-php-entrypoi..." 2 days ago    Up 4 seconds   9000/tcp                docker-project-php-1
3b59fa1ded60   mariadb             "/docker-entrypoint.s..." 2 days ago    Up 4 seconds   3306/tcp                docker-project-db-1
```



## 2.3 Membuat kontainer mariadb

### 1. Ubah kode didalam docker-compose.yml dengan code

```
version: "3.9"
services:
  nginx:
    build: ./nginx/
    ports:
      - 80:80

    volumes:
      - ./php_code:/var/www/html/

  php:
    build: ./php_code/
    expose:
      - 9000
    volumes:
      - ./php_code:/var/www/html/

  db:
    image: mariadb
    volumes:
      - mysql-data:/var/lib/mysql
    environment:
      MYSQL_ROOT_PASSWORD: nando123
      MYSQL_DATABASE: dbcrud

volumes:
  mysql-data
```

### 2. Jalankan perintah

```
docker-compose up -d
```

### 3. Masuk ke cli mariadb dengan perintah

```
docker exec -it 6b59fa1ded60 /bin/sh
```

### 4. Akses MariaDB sebagai pengguna root:

```
mariadb -u root -pnando123
```

### 5. Buat database dengan nama dbcrud dan buat tabel didalamnya dengan code seperti berikut:

```
Create database dbcrud;

CREATE TABLE `mahasiswa` (
  `id` int(11) NOT NULL,
  `nim` varchar(25) NOT NULL,
  `nama` varchar(25) NOT NULL,
  `alamat` varchar(25) NOT NULL,
```

```

`fakultas` varchar(25) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

ALTER TABLE `mahasiswa`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `nim` (`nim`);

ALTER TABLE `mahasiswa`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=8;
COMMIT;

```

## 6. Konfigurasi index.php

Pastikan konfigurasi untuk koneksi ke database benar berikut konfigurasi koneksi kami:

```

<?php
$host      = "db";
$user      = "nando";
$pass      = "nando123";
$db        = "dbcrud";

```

## 7. Buka web browser untuk mengecek aplikasi apakah berjalan

<http://localhost>

## 8. Satukan code docker-compose.yml menjadi 1 dengan code:

```

version: "3.9"
services:
  nginx:
    build: ./nginx/
    ports:
      - 80:80

  volumes:
    - ./php_code:/var/www/html/

  php:
    build: ./php_code/
    expose:
      - 9000
    volumes:
      - ./php_code:/var/www/html/

  db:
    image: mariadb
    volumes:
      - mysql-data:/var/lib/mysql

```

environment:  
MYSQL\_ROOT\_PASSWORD: nando123  
MYSQL\_DATABASE: dbnando  
volumes:  
mysql-data:

## 9.membuat web crud cloning dari github:

1.git clone dari github letakkan di folder php\_code

```
Git clone https://github.com/chapagain/crud-php-simple.git
```

2.lalu buat databae di dalam mysql dengan code:

```
Create database dbcrud;
```

```
CREATE TABLE `mahasiswa` (  
  `id` int(11) NOT NULL,  
  `nim` varchar(25) NOT NULL,  
  `nama` varchar(25) NOT NULL,  
  `alamat` varchar(25) NOT NULL,  
  `fakultas` varchar(25) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

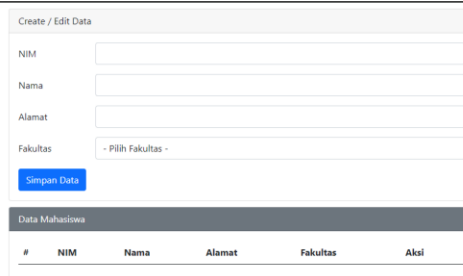
```
ALTER TABLE `mahasiswa`  
  ADD PRIMARY KEY (`id`),  
  ADD UNIQUE KEY `nim` (`nim`);
```

```
ALTER TABLE `mahasiswa`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=8;  
COMMIT;
```

3.ubah setingan di index.php sebagai berikut

```
<?php  
$host = "db";  
$user = "root";  
$pass = "nando123";  
$db = "dbnando";
```

4.lalu buka google dan buka localhost maka akan tampil seperti ini



The screenshot shows a web interface for managing student data. At the top, there is a 'Create / Edit Data' form with the following fields: NIM (text input), Nama (text input), Alamat (text input), and Fakultas (dropdown menu with the option '- Pilih Fakultas -'). Below the form is a blue button labeled 'Simpan Data'. Underneath the form is a table titled 'Data Mahasiswa' with the following columns: #, NIM, Nama, Alamat, Fakultas, and Aksi. The table is currently empty.

## **BAB III**

### **PENUTUP**

#### ***3.1. Kesimpulan***

Setelah kita membuat webserver dengan menggunakan docker kami mengetahui sangat mudah untuk membuat webserver dengan menggunakan docker compose

#### ***3.2. Saran***

Bagi pembaca makalah ini, semoga makalah ini dapat menjadi bermanfaat bagi semua. Kepada generasi muda untuk lebih bijak dalam penggunaan perkembangan teknologi saat ini.