1. PHP:Webserver dan Database di Docker
   1. Kita membuat file bernama docker.compose.yml dan diisi dengan code lalu kita compose up (kita namakan container name : redlock-web-2.0 dan port 7077:80)

version: '3.8'

services:

    php-apache-environment:

        container\_name: redlock-web-2.0

        build:

context: ./php

            dockerfile: Dockerfile

        depends\_on:

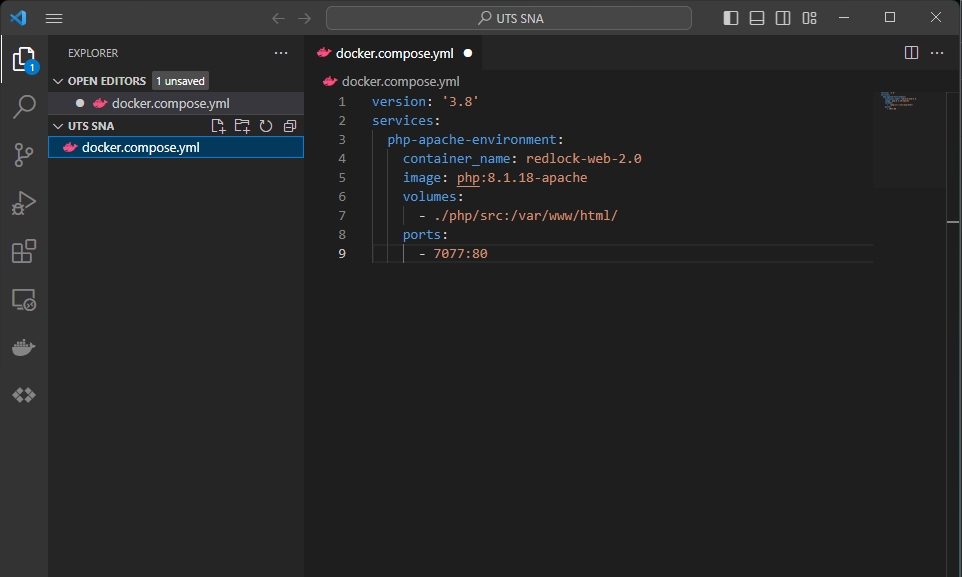
            - db

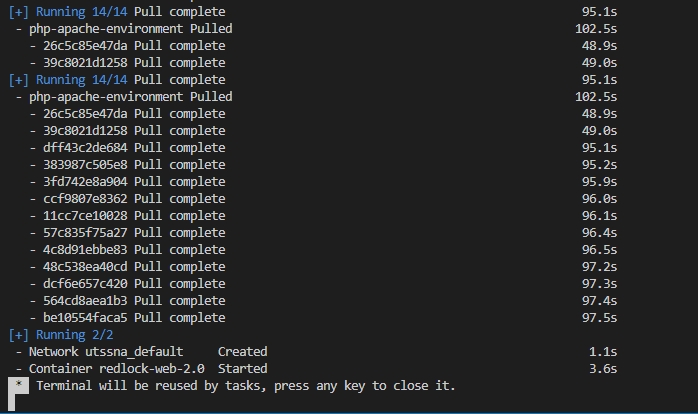
        volumes:

            - ./php/src:/var/www/html/

        ports:

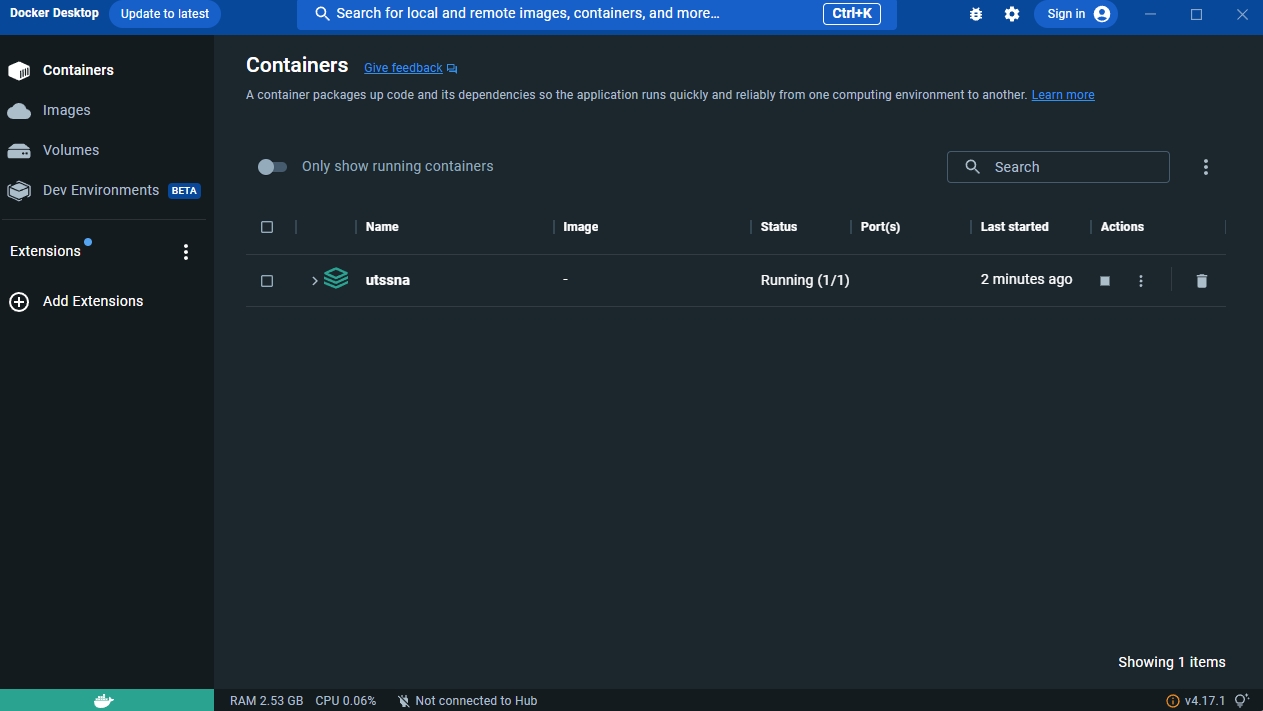
            - 7077:80

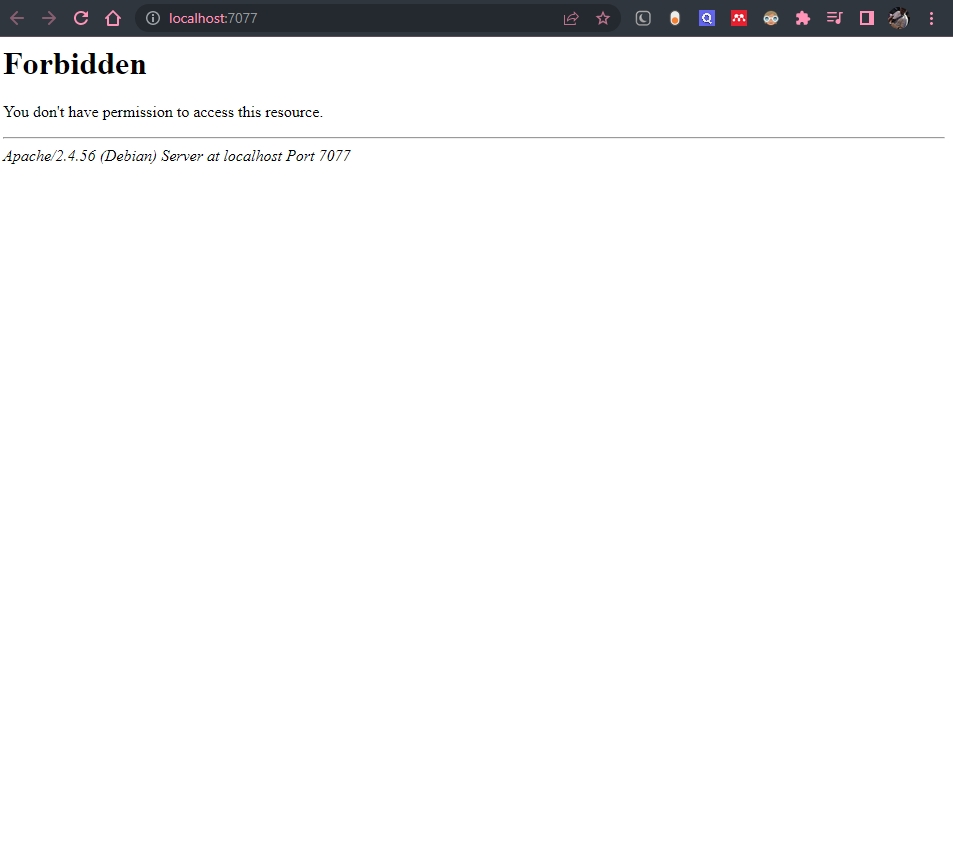




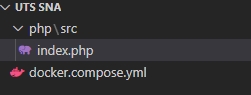
* 1. Setelah sudah di compose up, langsung membuka docker dan masuk kedalam port

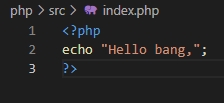






* 1. Selesai memastikan port dapat dibuka, kita membuat index.php dalam folder src untuk memastikan isi web





* 1. Setelah selesai membuat index.php, kita membuat **Dockerfile untuk** untuk php configurationdengan code :

FROM php:8.1.18-apache

RUN docker-php-ext-install mysqli && docker-php-ext-enable mysqli

RUN apt-get update && apt-get upgrade -y

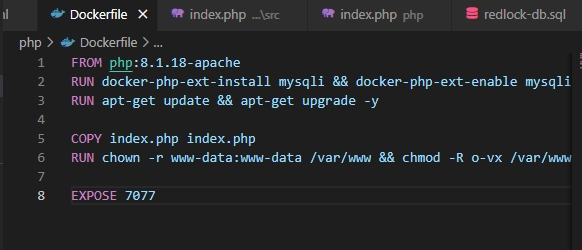
COPY index.php index.php

RUN chown -r www-data:www-data /var/www && chmod -R o-vx /var/www

EXPOSE 7077

Penjelasan:

- Version harus ditulis 8.1.18 sesuai dengan docker  
- “RUN docker-php-ext-install mysqli && docker-php-ext-enable mysqli dan RUN apt-get update && apt-get upgrade -y” digunakan untuk configure mySQL   
- “COPY index.php index.php” ditulis untuk content dari web karena harus dicopy menggunakan perintah dari dalam dockerfile  
- “RUN chown -r www-data:www-data /var/www” digunakan untuk kepemilikikan docker  
- “chmod -R o-vx /var/www” digunakan untuk menghilang control list ‘w’ dan ‘x’  
- “EXPOSE 7077” digunakan untuk port 7077



* 1. Setelah itu, kita edit docker.compose.yml dalam index untuk memasukan image database mysql dalam container dan compose up.

version: '3.8'

services:

    php-apache-environment:

        container\_name: redlock-web-2.0

        build:

            context: ./php

            dockerfile: Dockerfile

        depends\_on:

            - db

        volumes:

            - ./php/src:/var/www/html/

        ports:

            - 7077:80

    db:

        container\_name: db

        image: mysql

        restart: always

        environment:

            MYSQL\_ROOT\_PASSWORD: MYSQL\_ROOT\_PASSWORD

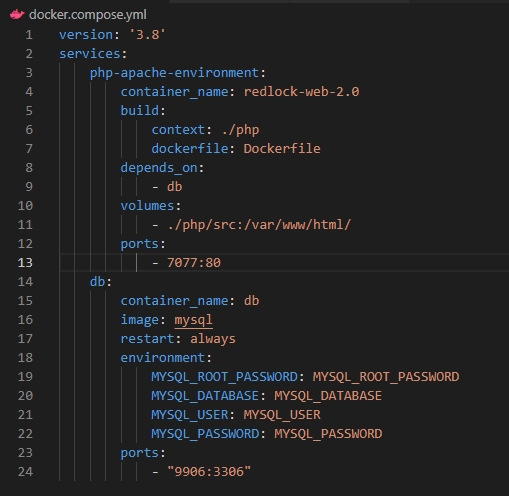
            MYSQL\_DATABASE: MYSQL\_DATABASE

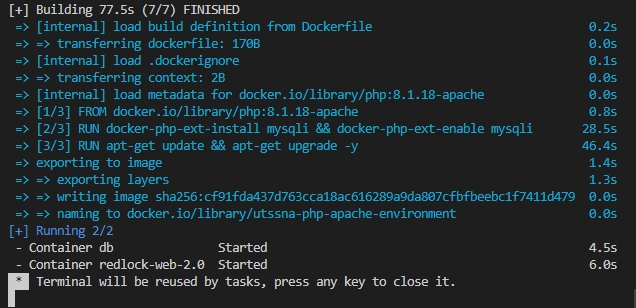
            MYSQL\_USER: MYSQL\_USER

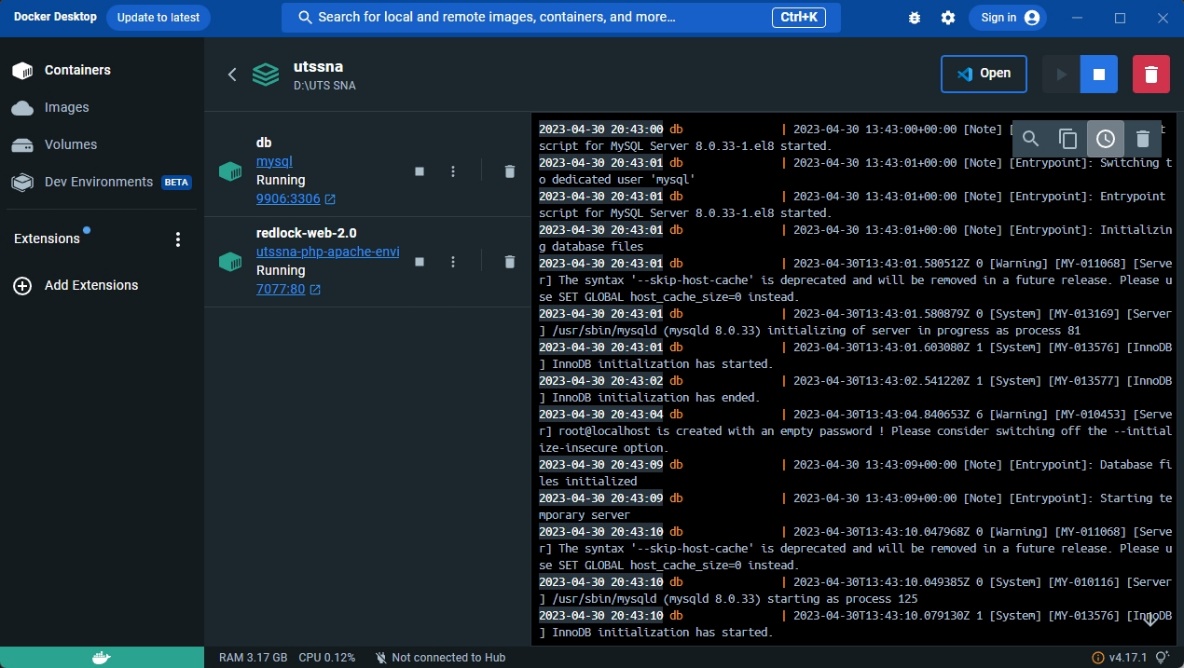
            MYSQL\_PASSWORD: MYSQL\_PASSWORD

        ports:

            - "9906:3306"







* 1. Setelah kita compose up, kita edit Kembali index.php untuk running SQL query dengan PHP scripts

<?php

$host = 'db';

$user = 'MYSQL\_USER';

$pass = 'MYSQL\_PASSWORD';

$conn = new mysqli($host, $user, $pass);

if ($conn->connect\_error) {

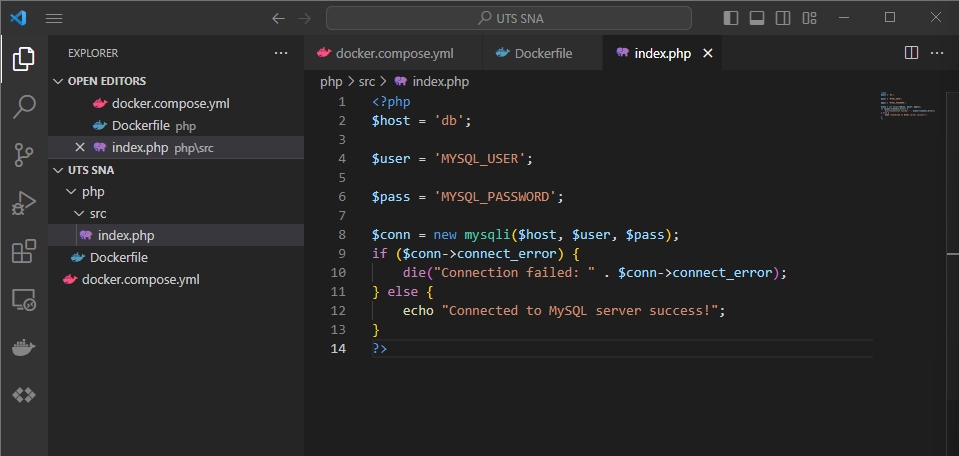
die("Connection failed: " . $conn->connect\_error);

} else {

echo "Connected to MySQL server successfully!";

}

?>





* 1. Setelah itu, kita edit docker.compose.yml untuk memasukan image phpMyAdmin dan compose

version: '3.8'

services:

    php-apache-environment:

        container\_name: redlock-web-2.0

        build:

            context: ./php

            dockerfile: Dockerfile

        depends\_on:

            - db

        volumes:

            - ./php/src:/var/www/html/

        ports:

            - 7077:80

    db:

        container\_name: db

        image: mysql

        restart: always

        environment:

            MYSQL\_ROOT\_PASSWORD: MYSQL\_ROOT\_PASSWORD

            MYSQL\_DATABASE: MYSQL\_DATABASE

            MYSQL\_USER: MYSQL\_USER

            MYSQL\_PASSWORD: MYSQL\_PASSWORD

        ports:

            - "9906:3306"

    phpmyadmin:

        image: phpmyadmin/phpmyadmin

        ports:

            - '8080:80'

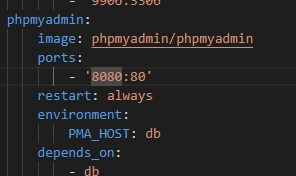
        restart: always

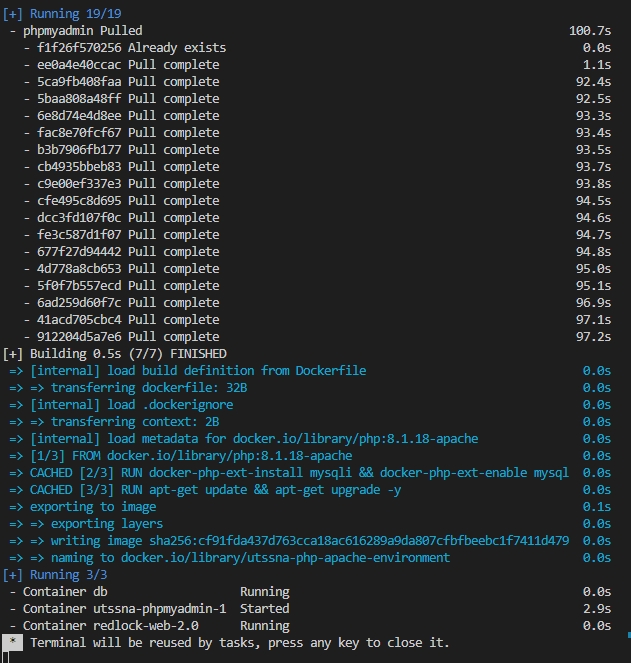
        environment:

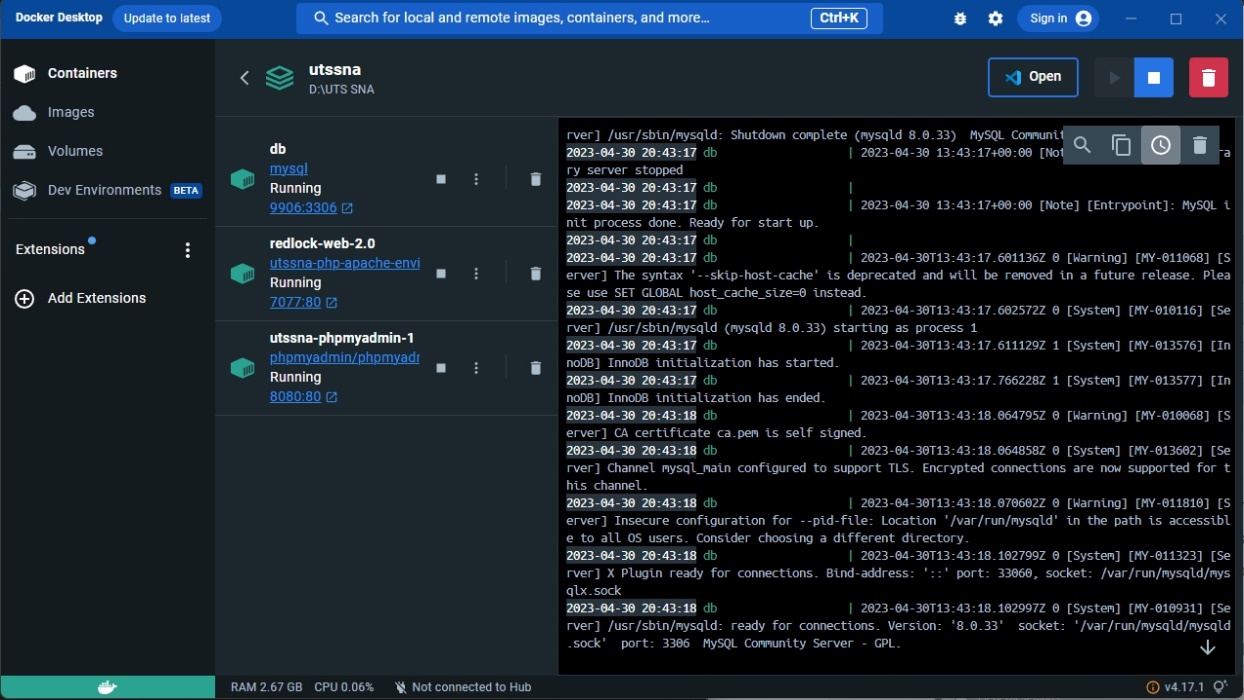
            PMA\_HOST: db

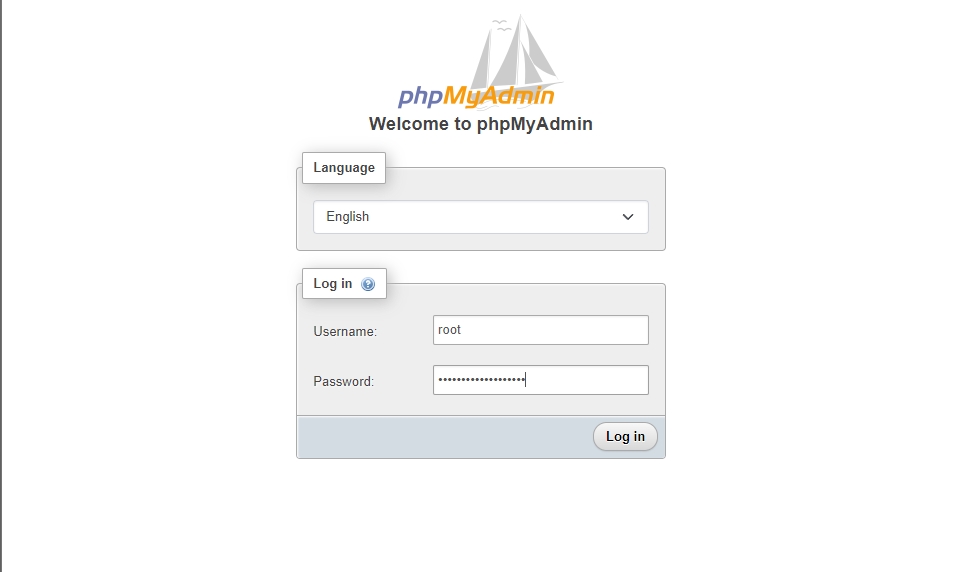
        depends\_on:

            - db





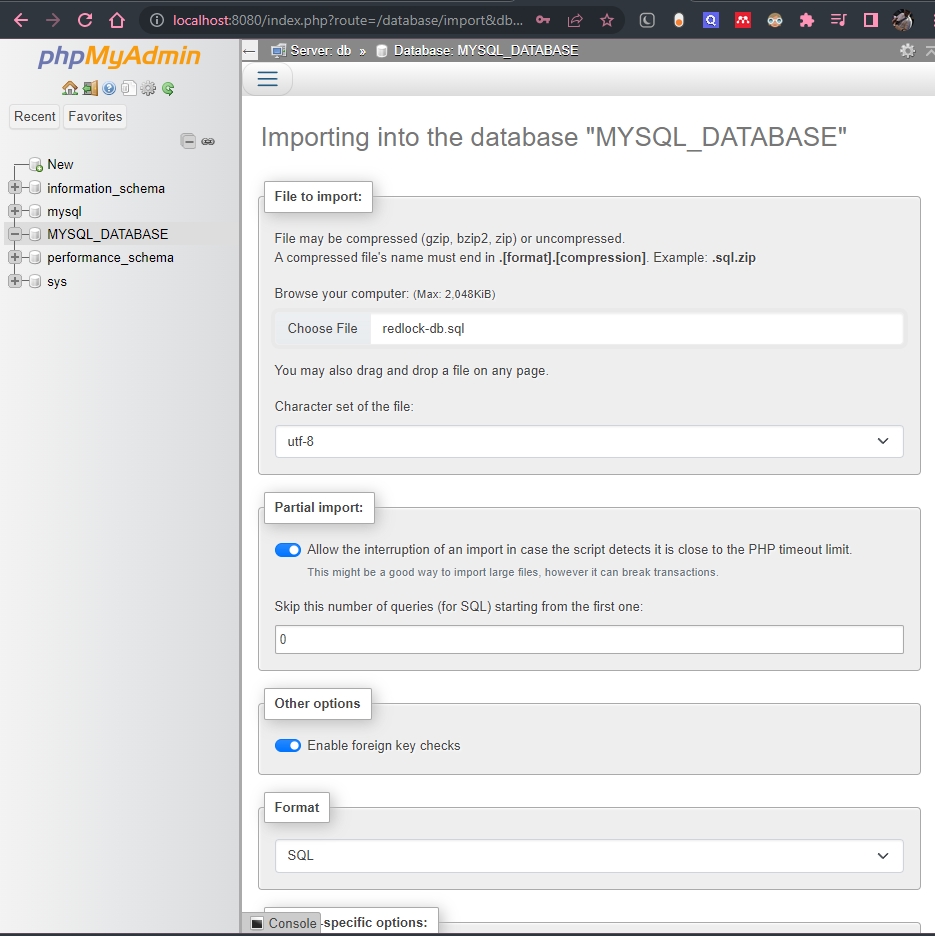


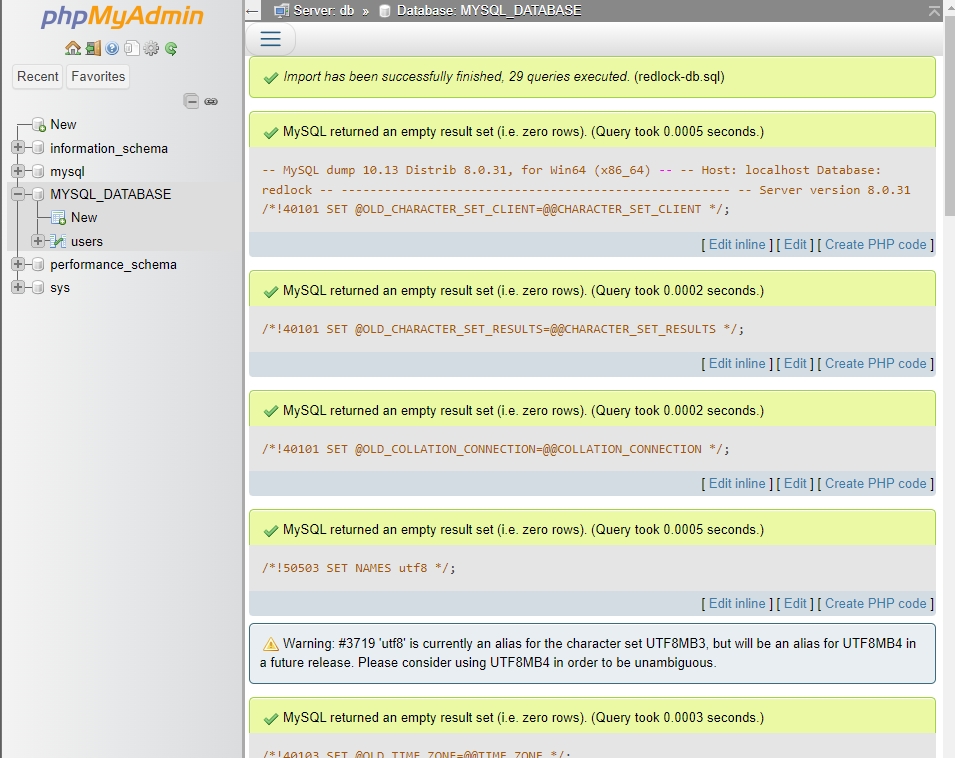


Username : root  
Password : MYSQL\_ROOT\_PASSWORD

* 1. Setelah itu, kita membuat database dari SQL workbench yang dissave dalam folder yang sama dengan dockerfile dan import kedalam PHPMYadmin







* 1. Setelah kita import, kita edit index.php untuk men setup port sesuai dengan format mySQL yang sudah kita edit, lalu kita save dan refresh website pada port 7077,maka seharusnya sudah terubah

<?php

$host = 'db';

$user = 'MYSQL\_USER';

$pass = 'MYSQL\_PASSWORD';

$mydatabase = 'MYSQL\_DATABASE';

$conn = new mysqli($host, $user, $pass, $mydatabase);

$sql = 'SELECT \* FROM users';

if ($result = $conn->query($sql)) {

    while ($data = $result->fetch\_object()) {

        $users[] = $data;

    }

}

foreach ($users as $user) {

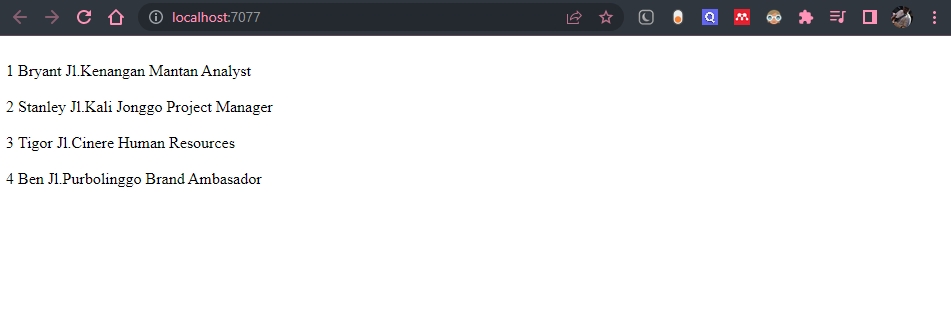
    echo "<br>";

    echo $user->ID . " " . $user->Nama . " " . $user->Alamat . " " . $user->Jabatan;

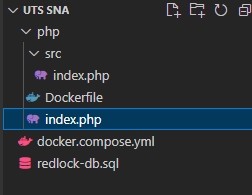
    echo "<br>";

}

?>



* 1. Untuk menggunakan perintah dari dalam dockerfile, kita buat file index.php dengan code yang sama namun file berada diluar src



1. Git & Jenkins
2. Sebutkan minimal 3 (tiga) contoh keadaan / situasi dimana kita harus memakai Ansible / CICD Tools
   1. Ansible dapat kita gunakan Ketika kita harus mengkonfigurasi server. Dengan ansible, kita dapat menkonfigurasi menjadi otomatis dan memastikan bahwa semua server dalam infrastruktur dikonfigurasi dengan cara yang sama sehingga konsisten
   2. CICD tools dapat kita gunakan dalam deployment. Dengan CICD, kita dapat melakukan deplotment secara otomasi setiap terjadi perubahan dalam code yang terdapat pada repository.
   3. Ansible dapat kita gunakan Ketika kita sedang maintaining infrastruktur berskala besar. Dengan ansible, kita dapat melakukan instaslasi package baru dan update kedalam seluruh server dengan efisien