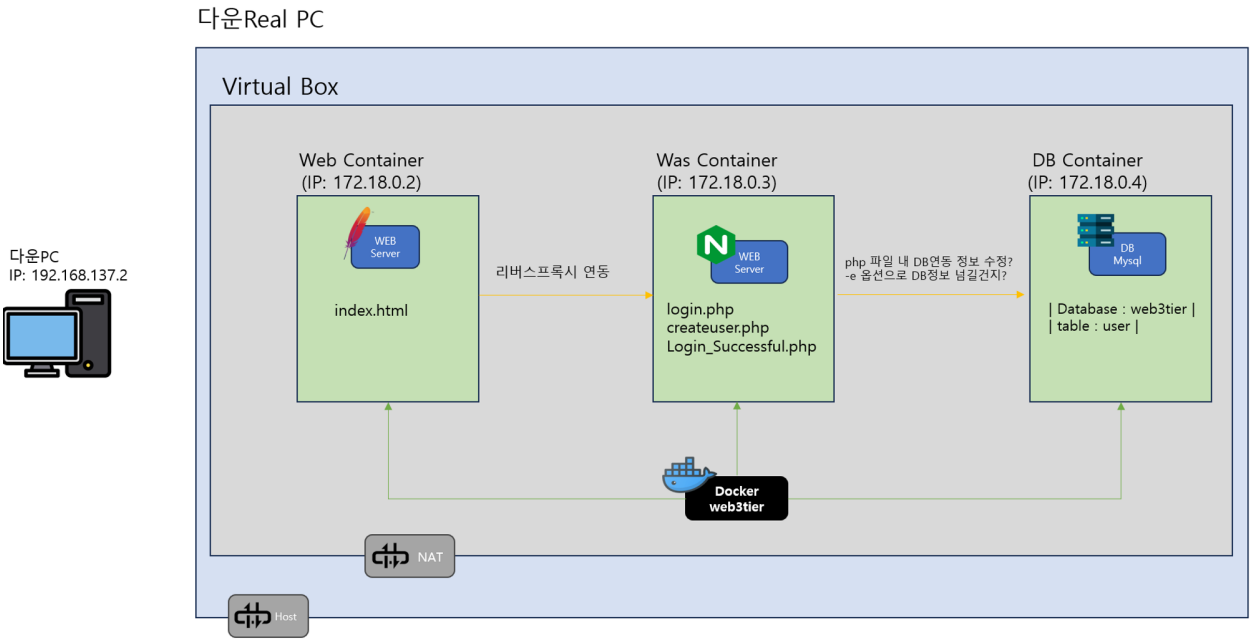
Docker_3Tier

최초 수정일: 24/02/23

최종 제출일: 24/02/26

김민경



1. bridge 네트워크 생성하기

docker network create net3tier

2. web 컨테이너 구성하기

2-1. 컨테이너 생성하기

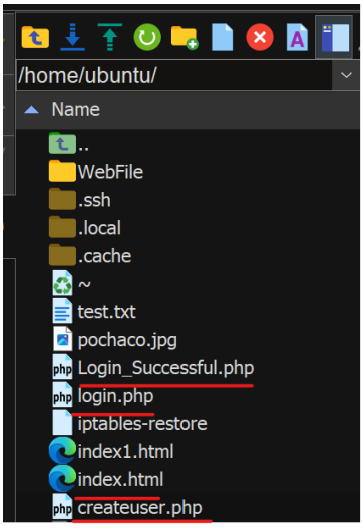
docker run -it -d --name web5 --network net3tier -p 85:80 --hostname web5 httpd

> 💡 **httpd vs ubuntu 이미지**
>
> - httpd: 아파치 컨테이너
> - ubuntu: 우분투컨테이너를 깔고 거기 안에 apache2를 까는 것임

2-2. index.html 환경설정 하기

- U-S에 강사님이 주신 파일들 옮기기
  - index.html 파일
  - Login_Successful.php
  - login.php
  - createuser.php

- 위 파일 4개에 777 권한 부여하기




```
root@ubuntu:/home/ubuntu# ll
total 164
drwxr-x--- 6 ubuntu ubuntu  4096 Feb 23 08:45  ./
drwxr-xr-x 5 root   root    4096 Jan 23 07:48  ../
-rw-r--r-- 1 root   root    3240 Jan 24 10:26 '~'
-rw------- 1 ubuntu ubuntu 18841 Feb 23 02:33  .bash_history
-rw-r--r-- 1 ubuntu ubuntu   220 Jan  6  2022  .bash_logout
-rw-r--r-- 1 ubuntu ubuntu  3771 Jan  6  2022  .bashrc
drwx------ 2 ubuntu ubuntu  4096 Dec 22 07:57  .cache/
-rwxrwxrwx 1 ubuntu ubuntu  1354 Feb 23 08:45  createuser.php*
-rw-rw-r-- 1 ubuntu ubuntu 10671 Jan  4 08:44  index1.html
-rwxrwxrwx 1 ubuntu ubuntu   329 Feb 23 08:31  index.html*
-rw-r--r-- 1 ubuntu ubuntu 16384 Feb 19 07:01  .index.html.swp
-rw-rw-r-- 1 ubuntu ubuntu  5586 Feb  1 16:31  iptables-restore
-rw------- 1 ubuntu ubuntu    20 Jan 23 07:42  .lesshst
drwxrwxr-x 3 ubuntu ubuntu  4096 Jan 23 06:37  .local/
-rwxrwxrwx 1 ubuntu ubuntu   809 Feb 23 08:45  login.php*
-rwxrwxrwx 1 ubuntu ubuntu   950 Feb 23 08:45  Login_Successful.php*
-rw------- 1 ubuntu ubuntu    13 Jan 23 07:41  .mysql_history
```

- U-S → web5 컨테이너로 index.html 옮기기
  - `docker cp {U-S에서 index.html 경로} {컨테이너 이름:컨테이너 경로}`

    docker cp /home/ubuntu/index.html web5:/usr/local/apache2/htdocs/index.html

  - 잘 옮겨졌는지 확인하기

    cat index.html

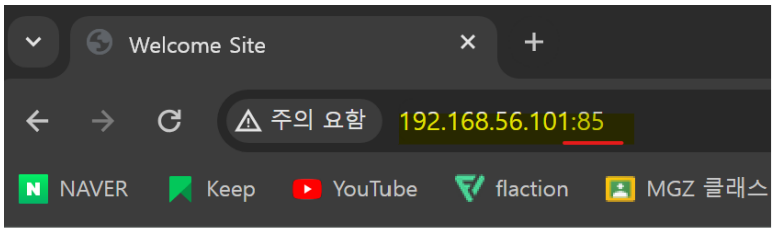
```
root@ubuntu:/home/ubuntu# cat index.html

<html>
   <head>
      <title> Welcome Site </title>
   </head>

   <h1> [ BABO Site ] </h1>

   <form action="login.php">
      <input type="submit" value="Login">
   </form>

   <form action="createuser.php">
      <input type="submit" value="CREATE NEW USER">
   </form>
</html>
root@ubuntu:/home/ubuntu#
```

2-3. RealPC에서 WEB(index.html) 테스트

- WEB 컨테이너 생성 시 포트포워딩을 85:80으로 설정했음

```
PORTS                                                  NAMES
0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp   mysql
0.0.0.0:86->80/tcp, :::86->80/tcp                      was
0.0.0.0:85->80/tcp, :::85->80/tcp                      web5
```

⇒ RealPC에서 접근 시 192.168.56.101:**85**로 접근하기



[BABO Site]

Login

CREATE NEW USER

2-4. 리버스 프록시 설정하기

▼ 프록시 개념

프록시

1) 개념

- '대리'라는 뜻
- 서버와 서버 사이의 중개자 역할(클리이언트 - 웹 서버 간)
    - 클라이언트로부터 요청을 대신 받아 웹 서버에 전달하고, 웹 서버의 응답을 클라이언트에게 전달하는 역할

2) 사용 이유

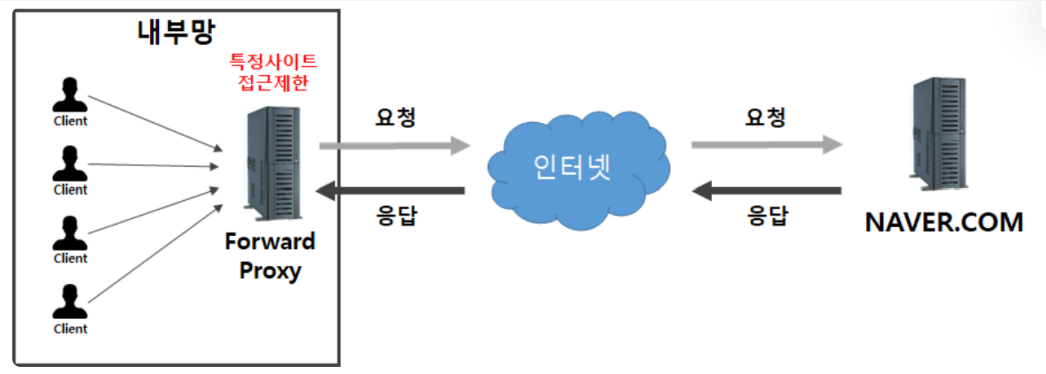
- 보안상 이유로 직접 통신할 수 없는 2 지점 사이에서 대리로 통신을 수행

  → 안성, 성능, 안정성을 향상
- 중복 요청에 대해 빨리 응답 가능함
    - 클라이언트에겐 빠른 속도의 서비스를, 서버에게는 불필요한 부하를 줄이는 효과를 낼 수 있음

3) 종류

- 리버스 프록시
- 포워드 프록시

포워드 프록시



1) 개념

- 클라이언트 바루 뒤에 있음

- 같은 내부망에 존재하는 클라이언트의 요청을 받아 인터넷을 통해 외부 서버에서 데이터를 가져와 클라이언트에게 응답해줌
    - 즉, 클라이언트가 서버에 접근하고자 할때, 클라이언트는 타겟 서버의 주소를 포워드 프록시에 전달하여, 포워드 프록시가 인터넷으로 요청된 내용을 가져오는 방식

2) 필요성

① 보안 강화

- 보통 정부, 학교, 기업 등과 같은 기관이 해당 기관에 속한 사람들의 제한적인 인터넷 사용을 위해 방화벽을 사용
    - 포워드 프록시 서버에 룰을 추가해서 특정 사이트에 접속하는 것을 막을 수 있음

② 캐싱

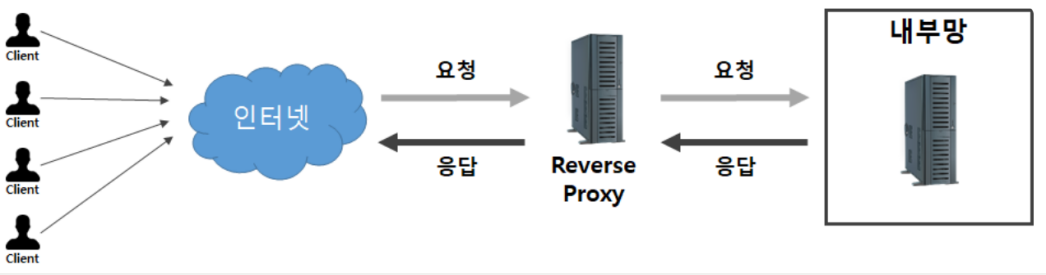
- 자주 사용되는 정적 파일(이미지, CSS, JavaScript)을 캐시에 저장함

  ⇒ 서버 부하 줄이고, 응답 시간 단축시켜 웹 서비스의 성능 향상 시킴

③ 암호화

- 클라이언트 IP 감추어줌
    - 클라이언트 요청은 프워드 프록시 서버를 통과할 때 암호화됨
    - 본 서버에서 IP 주소 역추적해도 프록시 서버를 사용하면 정체를 파악하기 어려움 (프록시 서버IP 만 보이기 때문)

리버스 프록시



1) 개념

- 웹서버/WAS 앞에 있음
- 보통 기업의 네트워크환경에서는DMZ라고 부르는 내부네트워크/외부네트워크 사이에 위치하는 구간이 존재함 (내부네트워크/외부네트워크에 둘다 접근할 수 있는 공간)
    - 이 구간에는 보통 메일 서버, 웹 서버, FTP 서버 등 외부 서비스를 제공하는 서버가 위치함
    - WAS를 DMZ에 놓고 서비스해도 되지만 보안상문제가 있음
        - WAS는 DB서버와 연결되어 있으므로, WAS가 해킹당할 경우 DB서버까지 해킹당할 수 있는 문제가 발생할 수 있음
        - so, 리버스 프록시 서버를 DMZ에 두고 실제 서비스 서버는 내부망에 위치시킨 후 서비스 하는 것이 일반적임

2) 필요성

① **서버 부하 분산 (Load Balancing)**

- 웹 서비스에 동시에 多 사용자가 접속할 경우, 서버에 부하가 집중되어 성능 저하 및 서비스 중단 현상이 발생할 수 있음

  ⇒ 리버스 프록시가 들어오는 요청을 여러 대의 서버로 분산 시켜 각 서버의 분산 줄이고, 서버의 가용성 높여 안정적인 서비스 제공이 가능하도록 함

② 보안 강화

- 외부에서 직접 서버에 접근하지 x하도록 함

③ 캐싱 및 가속화

- 자주 사용되는 정적 파일(이미지, CSS, JavaScript)을 캐시에 저장함

  ⇒ 서버 부하 줄이고, 응답 시간 단축시켜 웹 서비스의 성능 향상시킴

포워드 프록시 vs 리버스 프록시 차이점

1) 프록시 서버 위치

- 포워드 프록시 서버 ⇒ 클리이언트 앞
- 리버스 프록시 서버 ⇒ 웹서버/was 앞

2) 프록시 서버 통신 대상

- 포워드 프록시 서버 ⇒ 내부망에서 **클라이언트**와 **Proxy 서버**가 통신하여 인터넷을 통해 외부에서 데이터를 가져옴
- 리버스 프록시 서버 ⇒ 내부망에서 **Proxy 서버**와 **내부망 서버**가 통신하여 인터넷을 통해 요청이 들어오면 Proxy 서버가 받아 응답함

3) 감춰지는 대상

- 포워드 프록시 서버 ⇒ **서버에게 클라이언트가 누구인지 감춰줌**
  - 내부망에서 인터넷 상에 있는 서버에 요청할때 먼저 포워드 프록시 서버를 호출하고 프록시가 서버에게 요청을 보내게 되는데, 이로써 서버에게클라이언트가 누구인지 감출수 있음
- 리버스 포륵시 서버 ⇒ **본 서버의 ip 정보**를 감춰줌
  - Forward Proxy는 직접 서버 url로 요청을 보내지만, Reverse Proxy는 프록시 서버 url로만 접근이 가능하기 때문

https://inpa.tistory.com/entry/NETWORK-📡-Reverse-Proxy-Forward-Proxy-정의-차이-정리

https://aday7.tistory.com/entry/리버스-프록시Reverse-Proxy-쉽게-이해하기-개념부터-필요성-오픈-소스-솔루션까지

docker exec -it web5 /bin/bash //web서버에 접속하기

apt-get update

apt install vim

dpkg -l | grep vim //vim 설치 확인하기

cd /usr/local/apache2

cd ./conf/

ls

vim httpd.conf

- 아래와 같이 수정하기

```
ServerAdmin webmaster@localhost

<VirtualHost *:80>
    ServerName localhost

    ProxyPass "/login.php" "http://was/login.php"
    ProxyPass "/createuser.php" "http://was/createuser.php"
    ProxyPass "/Login_Successful.php" "http://was/Login_Successful.php"
</VirtualHost>


#
# ServerName gives the name and port that the server uses to identify itself.
# This can often be determined automatically, but we recommend you specify
# it explicitly to prevent problems during startup.
#
# If your host doesn't have a registered DNS name, enter its IP address here.
#
#ServerName www.example.com:80
```

```
LoadModule proxy_module modules/mod_proxy.so
#LoadModule proxy_connect_module modules/mod_proxy_connect.so
#LoadModule proxy_ftp_module modules/mod_proxy_ftp.so
LoadModule proxy_http_module modules/mod_proxy_http.so
```

- **apachectl configtest** // 아파치가 현재의 설정 파일을 로드할 수 있는지 확인하고 구성에 오류가 있는지 확인하기

  ⇒ 하지만 **오류** 뜸

```
root@web5:/usr/local/apache2/conf# apachectl configtest
AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using
 172.20.0.2. Set the 'ServerName' directive globally to suppress this message
Syntax OK
```

  - 이 오류는 아파치 서버가 시작될 때 발생하는 것으로, 아파치 구성 파일에 전역적으로 `ServerName` 지시문이 설정되어 있지 않아 발생함
  - 아파치의 설정 파일에 `ServerName`을 전역적으로 설정해야함

```
ServerAdmin webmaster@localhost

<VirtualHost *:80>
    ServerName localhost

    ProxyPass "/login.php" "http://was/login.php"
    ProxyPass "/createuser.php" "http://was/createuser.php"
    ProxyPass "/Login_Successful.php" "http://was/Login_Successful.php"
</VirtualHost>


#
# ServerName gives the name and port that the server uses to identify itself.
# This can often be determined automatically, but we recommend you specify
# it explicitly to prevent problems during startup.
#
# If your host doesn't have a registered DNS name, enter its IP address here.
#
ServerName webmaster@localhost
```

- 결과 다시 확인하기

  apachectl configtest

```
root@web5:/usr/local/apache2/conf# apachectl configtest
Syntax OK
```

apachectl restart //변경사항 적용하기

3. was 컨테이너 구성하기

3-1. 컨테이너 생성하기

docker run -it -d --name was --network net3tier -p 86:80 --hostname was ubuntu

3-2. 파일 옮기기 및 수정하기

- U-S → was 컨테이너로 강사님이 주신 파일들 옮기기

  docker cp /home/ubuntu/login.php was:/var/www/html //login.php 파일 옮기기

  ⇒ 하지만 오류 뜸

```
root@ubuntu:~# docker cp /home/ubuntu/login.php was:/var/www/html/login.php
Successfully copied 2.56kB to was:/var/www/html/login.php
Error response from daemon: Could not find the file /var/www/html in container was
```

  - var 밑에 www 디렉토리가 없다는 의미

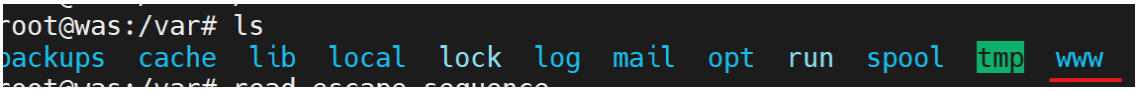
⇒ 이는 apache2 설치를 해줘야 생김

- apache2 설치하기

  apt-get update

  apt install apache2 -y

  - 설치 후 var/www/html이 생김



- php 설치하기

  apt-get install php8.1-mysql php8.1-mysqli php8.1 php8.1-cli

> 💡 **php & php-mysql**
>
> - **php**
>   - PHP 스크립트를 실행하는 데 사용되는 프로그래밍 언어
> - **php-mysql**
>   - PHP에서 MySQL 데이터베이스에 연결하고 상호 작용하기 위한 MySQL 관련 확장 기능을 제공
>   - 일반적으로 PHP를 사용하여 웹 애플리케이션을 개발할 때 MySQL과 같은 데이터베이스와 통합해야 할 때가 많음
>   - 이때 PHP에서 MySQL과의 상호 작용을 지원하는 확장 기능인 `php-mysql`이 필요함
>   - 이 확장 기능은 MySQL 데이터베이스에 연결하고 쿼리를 실행하며 결과를 처리하는 데 사용됨

- U-S → was 컨테이너로 강사님이 주신 파일들 옮기기

docker cp /home/ubuntu/login.php was:/var/www/html //login.php 파일 옮기기

docker cp /home/ubuntu/createuser.php was:/var/www/html
//createuser.php 파일 옮기기

docker cp /home/ubuntu/Login_Successful.php was:/var/www/html
//Login_Successful.php 파일 옮기기

> 💡 **docker cp 명령어**
>
> - 컨테이너가 실행중일 때만 사용 가능

- 파일이 제대로 옮겨졌는지 확인하기

```
root@was:/var/www/html# ls
Login_Successful.php  createuser.php  index.html  login.php
```

- 파일 수정을 위한 vim 설치하기

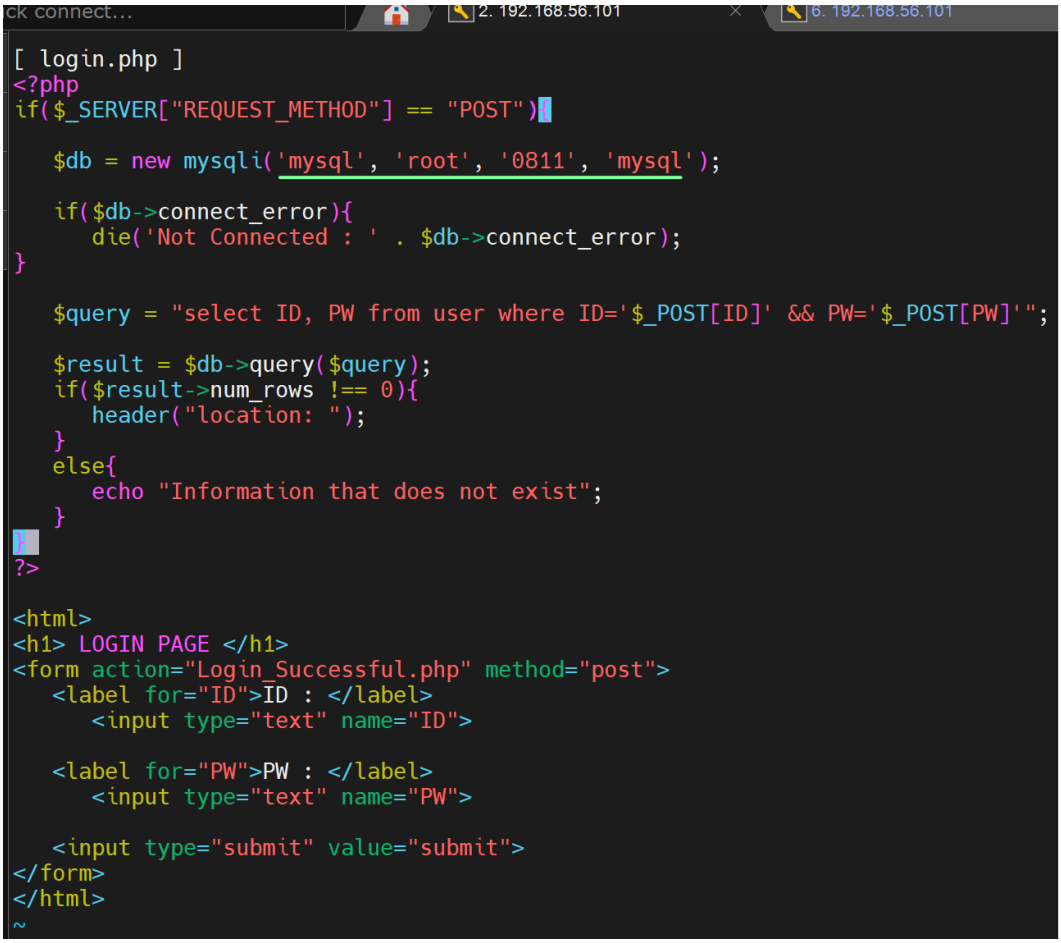
  apt-get update

  apt install vim

  dpkg -l | grep vim //vim 설치 확인하기

- 파일들 수정하기

  vim login.php //login.php 파일 수정하기



💡 **db= new mysqli('mysql', 'root' '0811', 'test')**

- PHP에서 MySQL 데이터베이스에 연결하기 위한 것
- `'db'` : MySQL 데이터베이스 호스트 (Docker 컨테이너의 이름)
  - Docker 컨테이너 내부 네트워크에서 서로 통신할 때 사용하는 호스트 이름
- `'root'` : MySQL 데이터베이스에 연결할 사용자 이름
  - 일반적으로는 관리자인 'root'를 사용함
- `'1234'` : MySQL 데이터베이스에 연결할 사용자의 비밀번호
- `'test'` : 연결할 MySQL 데이터베이스의 이름

`vim createuser.php` //createuser.php 파일 수정

```
[ createuser.php ]
<?php

if ($_SERVER["REQUEST_METHOD"] == "POST"){
   $db = new mysqli('mysql', 'root', '0811', 'mysql');
   if($db->connect_error){
      die('Not Connected : ' . $db->connect_error);
   }


$sql2 = "select ID from user where ID='$_POST[ID]'";
   $result2 = $db->query($sql2);
   if($result2->num_rows !== 0){
      die("User Exists!");
   }
   $sql = "insert into user (ID,PW) values ('$_POST[ID]','$_POST[PW]')";
   $result = $db->query($sql);
   if($result){
      echo '<p>' .  "List of users" . '</p>';
      echo "----------------------------------------------------------";

         $conn=mysqli_connect('mysql', 'root', '0811', 'mysql');
         $sql="select * from user";
         $result=mysqli_query($conn, $sql);

         while( $row = mysqli_fetch_array( $result ) ) {
          echo '<p>' ."ID : ". $row[ 'ID' ] . "  " . "PW : ". $row[ 'PW' ] . '<
';
      }

   }
   else{
      echo "ERROR ";
   }
```

vim Login_Successful.php //Login_Successful.php 파일 수정

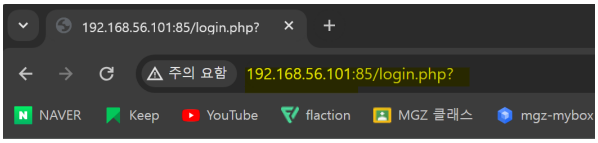
ck connect... 2. 192.168.56.101 6. 192.168.56.101

```
[ Login_Successful.php ]
<html>
<div style="text-align:center"> <b> <font size="40"> Login Successful </font> </b>
</html>

<?php
if($_SERVER["REQUEST_METHOD"] == "POST"){

   $db = new mysqli('mysql', 'root', '0811', 'mysql');

   if($db->connect_error){
      die('Not Connected : ' . $db->connect_error);
}

   $query = "select ID, PW from user where ID='$_POST[ID]' && PW='$_POST[PW]'";

   $result = $db->query($query);
   if($result->num_rows !== 0){
      header("location: Login_Successful.php");
   }
   else{
      echo "Information that does not exist";
   }
}
?>

<html>
<h1> LOGIN PAGE </h1>
<form action="" method="post">
   <label for="ID">ID : </label>
      <input type="text" name="ID">

   <label for="PW">PW : </label>
      <input type="text" name="PW">

   <input type="submit" value="submit">
</form>
</html>
```

- 변경사항 적용하기

  service apache2 reload

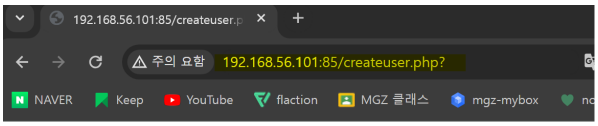
3-3. RealPC에서 WAS(login.php, createuser.php) 테스트



[login.php]

LOGIN PAGE

ID : PW : submit



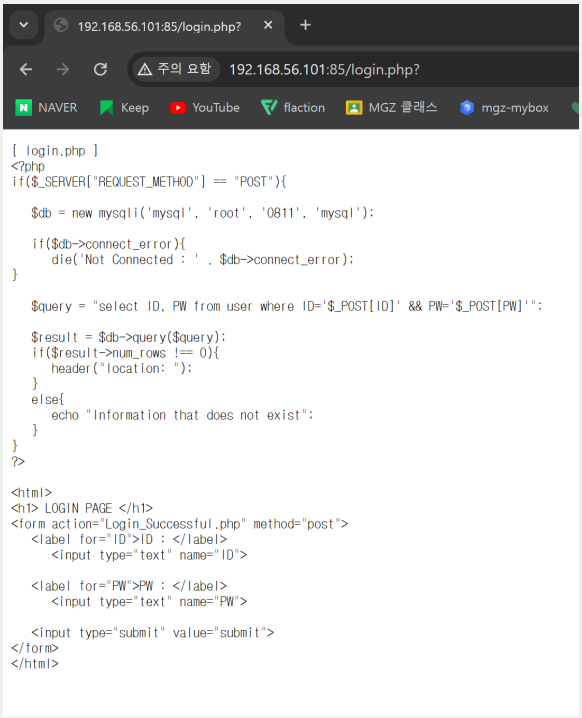
[createuser.php]

CREATE A NEW USER!

NEW ID : NEW PW : submit

💡 **TS**

- 원래는 WAS 컨테이너를 ubuntu가 아닌 httpd로 만들었음
- httpd이기 때문에 apache2를 따로 설치하지는 않았음
- 그런데 [3-3. RealPC에서 WAS(login.php, createuser.php) 테스트] 과정에서 login과 createuser 버튼을 누르면 아래와 같이 텍스트가 불러져 왔었음



- **이는 아마 was 서버에 강사님이 주신 login.php, createuser.php, Login_Successful.php 파일만 옮긴 것이 아니라 index.html까지 옮겨서 저 텍스트 파일이 나온 것 같음**

4. DB 컨테이너 구성하기

4-1. 컨테이너 생성하기

```
docker run -d -p 3306:3306 -v mk-sql-vol2:/var/lib/mysql \
> -e MYSQL_ROOT_PASSWORD=0811 \
```

**> -e MYSQL_DATABASE=mysql **
**> --name mysql **
**> --hostname mysql **
> mysql:5.7

```
root@ubuntu:~# docker run -d -p 3306:3306 -v mk-sql-vol2:/var/lib/mysql \
> -e MYSQL_ROOT_PASSWORD=0811 \
> -e MYSQL_DATABASE=mysql \
> --name mysql \
> --hostname mysql \
> mysql:5.7
8b336e7df70ca639730f79c5acded693c7260ae1e35257c193c51bfa38065483
```

docker network inspect net3tier 으로 확인해보니 mysql2 컨테이너는 연결이 되어있지 x

```
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "45f9878da9488550eab8faa7f840d52209271d39e9fd14723dfb0997ba5e708c": {
                "Name": "web5",
                "EndpointID": "c842d90da6d96736d6c04b1e78494dc9d27c27a7793ac0f8bb725c3e20d0c0f7",
                "MacAddress": "02:42:ac:14:00:02",
                "IPv4Address": "172.20.0.2/16",
                "IPv6Address": ""
            },
            "5e48ad7438f60d1f0f94c27a3e81eb469e79bc7076db15d9162fdba95361c0bd": {
                "Name": "was",
                "EndpointID": "043d0381e28dbf86ddf207dd6aac2f39dc512b5c8dcc7fbd6d57f8c45ac8de0d",
                "MacAddress": "02:42:ac:14:00:03",
                "IPv4Address": "172.20.0.3/16",
                "IPv6Address": ""
```

⇒ so mysql2 컨테이너도 연결해주기

docker network connect net3tier mysql

docker network inspect net3tier

```
},
"ConfigOnly": false,
"Containers": {
    "45f9878da9488550eab8faa7f840d52209271d39e9fd14723dfb0997ba5e708c": {
        "Name": "web5",
        "EndpointID": "c842d90da6d96736d6c04b1e78494dc9d27c27a7793ac0f8bb725c3e20d0c0f7",
        "MacAddress": "02:42:ac:14:00:02",
        "IPv4Address": "172.20.0.2/16",
        "IPv6Address": ""
    },
    "5e48ad7438f60d1f0f94c27a3e81eb469e79bc7076db15d9162fdba95361c0bd": {
        "Name": "was",
        "EndpointID": "043d0381e28dbf86ddf207dd6aac2f39dc512b5c8dcc7fbd6d57f8c45ac8de0d",
        "MacAddress": "02:42:ac:14:00:03",
        "IPv4Address": "172.20.0.3/16",
        "IPv6Address": ""
    },
    "ed84c26b746c388c924ec1bbc309d97d1ba10bfb501e226833780109dcfbdc01": {
        "Name": "mysql",
        "EndpointID": "e3db3629a6489db6765091ea26df4cccfce54737c33be5f5129c5420a5ad517a",
        "MacAddress": "02:42:ac:14:00:04",
        "IPv4Address": "172.20.0.4/16",
        "IPv6Address": ""
```


4-2. 테이블 생성하기

docker exec -it mysql mysql -u root -p

//첫번째 mysql : db 컨테이너 이름

//두번째 mysql : mysql db 이름

```
root@ubuntu:~# docker exec -it mysql mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

show databases;

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.03 sec)
```

use mysql;

```
mysql> use mysql;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

CREATE TABLE user (

ID VARCHAR(30) NOT NULL PRIMARY KEY,

PW VARCHAR(30) NULL

);

⇒ 하지만 **오류** 뜸

```
Database changed
mysql> CREATE TABLE user (
    -> ID VARCHAR(30) NOT NULL PRIMARY KEY,
    -> PW VARCHAR(30) NULL
    -> );
ERROR 1050 (42S01): Table 'user' already exists
mysql>
mysql> show tables;
+---------------------------+
| Tables_in_mysql           |
+---------------------------+
| 4team                     |
| columns_priv              |
| db                        |
| engine_cost               |
| event                     |
| func                      |
| general_log               |
| gtid_executed             |
| help_category             |
| help_keyword              |
| help_relation             |
| help_topic                |
| innodb_index_stats        |
| innodb_table_stats        |
| ndb_binlog_index          |
| plugin                    |
| proc                      |
| procs_priv                |
| proxies_priv              |
| server_cost               |
| servers                   |
| slave_master_info         |
| slave_relay_log_info      |
| slave_worker_info         |
| slow_log                  |
| tables_priv               |
| time_zone                 |
| time_zone_leap_second     |
| time_zone_name            |
| time_zone_transition      |
| time_zone_transition_type |
| user                      |
+---------------------------+
```

- 따라서 user table 삭제하고 다시 생성함

  DROP TABLE user;

```
mysql> DROP TABLE user;
Query OK, 0 rows affected (0.00 sec)
```

  ⇒ 하지만 **오류** 뜸

```
mysql> CREATE TABLE user (
    -> ID VARCHAR(30) NOT NULL PRIMARY KEY,
    -> PW VARCHAR(30) NULL
    -> );
ERROR 1030 (HY000): Got error 168 from storage engine
```

  저장공간 지우고 다시 하고 있습니다:) 금방해요:)

show tables;

```
mysql> show tables;
+---------------------------+
| Tables_in_mysql           |
+---------------------------+
| 4team                     |
| columns_priv              |
| db                        |
| engine_cost               |
| event                     |
```

INSERT INTO 4team (id, pw) VALUES ('test1', '1234');

```
mysql> use mysql;INSERT INTO 4team (id, pw) VALUES ('test1', '1234');
Database changed
Query OK, 1 row affected (0.02 sec)
```

SELECT * FROM 4team;

```
mysql> SELECT * FROM 4team;
+-------+------+
| ID    | PW   |
+-------+------+
| test1 | 1234 |
+-------+------+
1 row in set (0.00 sec)
```