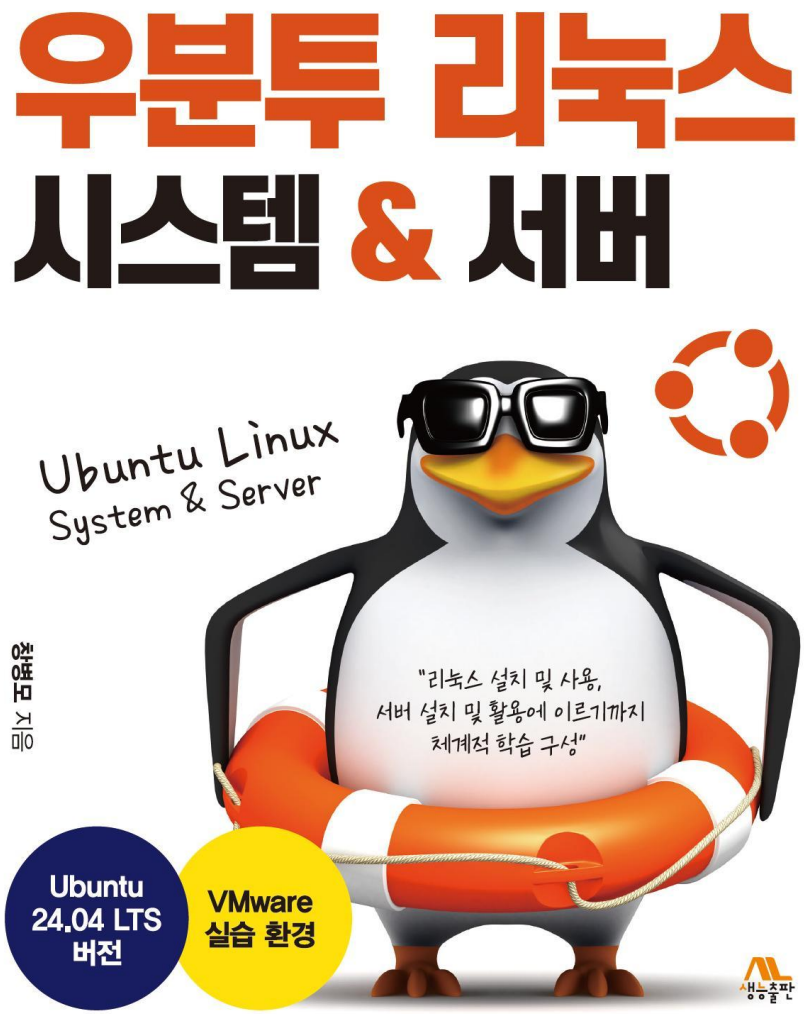
LINUX

Autumn 2025

Daejeon Univ.
Seongbok Baik
sbbaik@dju.ac.kr

Text Book

교재명	우분투 리눅스 시스템 & 서버
저자	창병모
출판사	생능출판사
발행년	2024.07.12
ISBN	979-11-92932-72-9



14.1 PHP 소개

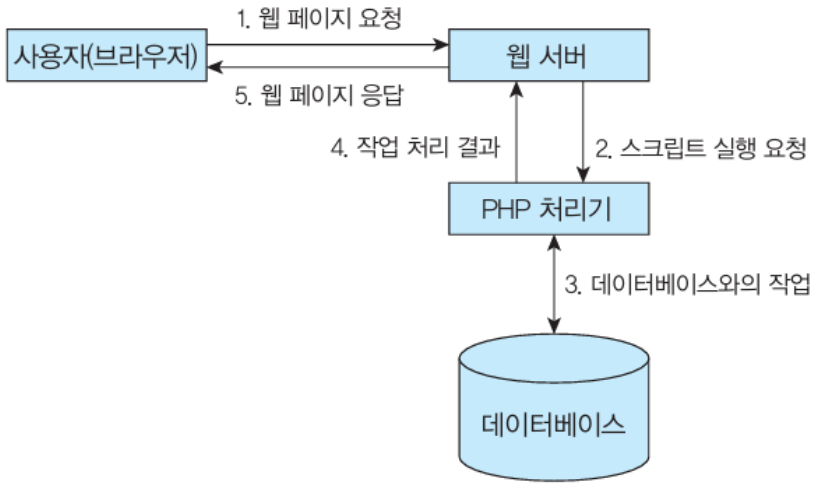
PHP 소개

- 웹 환경에서 동작하는 서버 측 스크립트 언어
  - 동적인 웹 페이지를 생성하는데 사용된다.
  - PHP 코드는 HTML 문서에 포함되며 서버에서 실행되어 클라이언트(브라우저)에는 그 출력 결과가 전달된다.

- HTML에 삽입
  - PHP 코드는 HTML 문서 내에 삽입될 수 있다.
  - <?php와 ?> 태그로 PHP 코드를 구분한다.
- 다양한 용도
  - PHP는 웹 페이지 생성과
  - 데이터베이스 처리, 파일 처리, 세션 관리 등 다양한 작업에 활용된다.
- 데이터베이스 연동
  - PHP는 주로 MySQL과 함께 사용되며,
  - 데이터베이스와의 상호작용을 통해 데이터를 처리할 수 있다.
- 오픈 소스
  - 오픈 소스 무료 소프트웨어
  - 많은 개발자들이 활발한 커뮤니티에 기여하고 있다.

PHP 동작 원리

1. 클라이언트가 웹 브라우저를 통해 웹 서버에 원하는 웹 페이지를 요청한다.
2. 웹 서버는 클라이언트가 요청한 웹 페이지의 로직 및 데이터베이스와의 연동을 위해 PHP 처리기(processor)에 이에 대한 처리를 요청한다.
3. PHP 처리기는 데이터베이스와의 연동이 필요하면 데이터베이스와 연동하여 데이터 처리를 수행한다.
4. PHP 처리기는 웹 페이지의 로직 및 데이터베이스와의 작업 처리 결과를 웹 서버로 전달한다.
5. 웹 서버는 전달받은 데이터로 웹 페이지를 완성하여 웹 브라우저로 응답을 전송한다.



PHP 설치

1. PHP 언어와 mysqli 모듈을 설치한다.

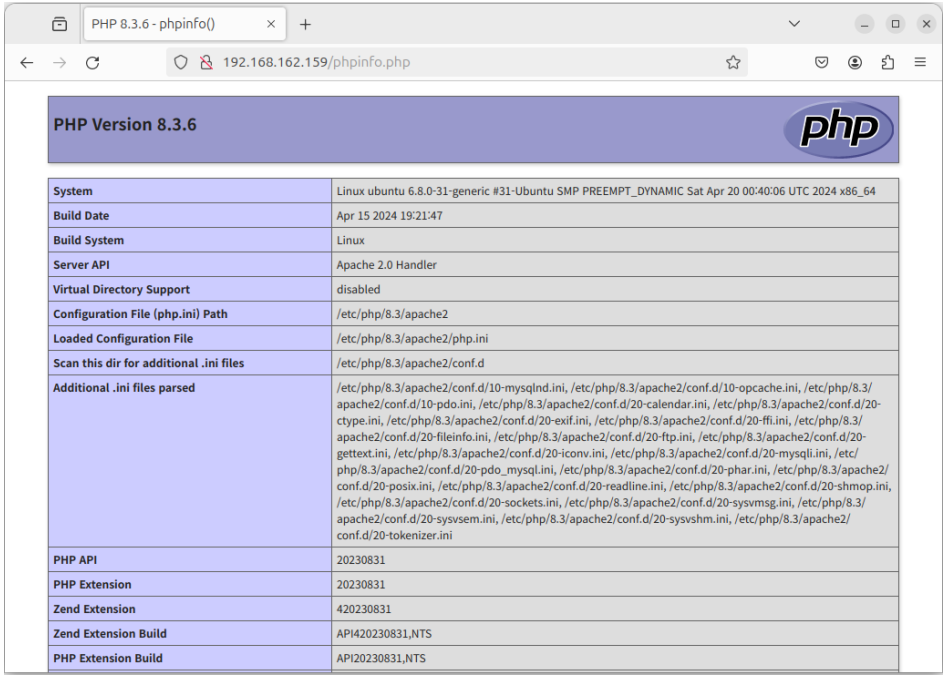
   ```
   # apt install php
   # apt install php8.3-mysqli
   ```

2. PHP의 동작을 확인하기 위해서 phpinfo.php 파일을 만든다.

   ```
   <?php phpinfo(); ?>
   ```

3. 다음과 같이 서버 주소로 웹 서버에 접속하여 서비스를 확인할 수 있다

   http://서버주소/phpinfo.php



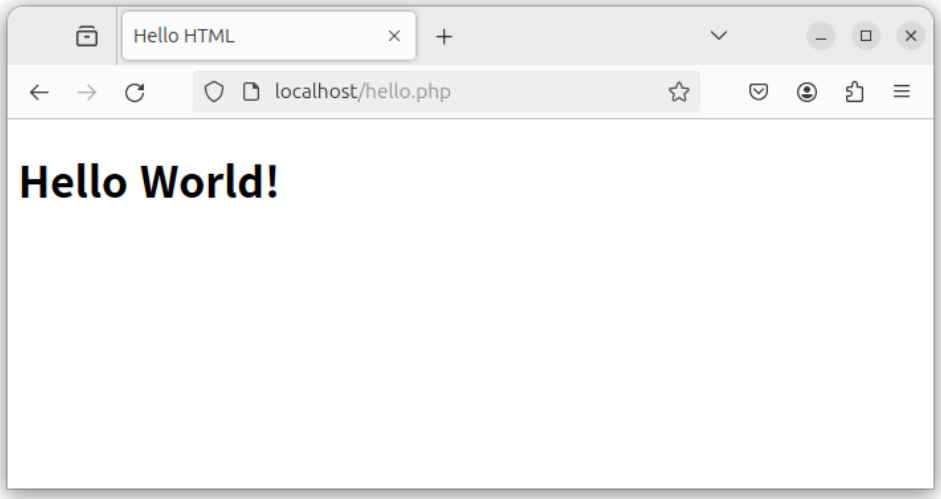
System	Linux ubuntu 6.8.0-31-generic #31-Ubuntu SMP PREEMPT_DYNAMIC Sat Apr 20 00:40:06 UTC 2024 x86_64
Build Date	Apr 15 2024 19:21:47
Build System	Linux
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.3/apache2
Loaded Configuration File	/etc/php/8.3/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/8.3/apache2/conf.d
Additional .ini files parsed	/etc/php/8.3/apache2/conf.d/10-mysqlnd.ini, /etc/php/8.3/apache2/conf.d/10-opcache.ini, /etc/php/8.3/apache2/conf.d/10-pdo.ini, /etc/php/8.3/apache2/conf.d/20-calendar.ini, /etc/php/8.3/apache2/conf.d/20-ctype.ini, /etc/php/8.3/apache2/conf.d/20-exif.ini, /etc/php/8.3/apache2/conf.d/20-ffi.ini, /etc/php/8.3/apache2/conf.d/20-fileinfo.ini, /etc/php/8.3/apache2/conf.d/20-ftp.ini, /etc/php/8.3/apache2/conf.d/20-gettext.ini, /etc/php/8.3/apache2/conf.d/20-iconv.ini, /etc/php/8.3/apache2/conf.d/20-mysqli.ini, /etc/php/8.3/apache2/conf.d/20-pdo_mysql.ini, /etc/php/8.3/apache2/conf.d/20-phar.ini, /etc/php/8.3/apache2/conf.d/20-posix.ini, /etc/php/8.3/apache2/conf.d/20-readline.ini, /etc/php/8.3/apache2/conf.d/20-shmop.ini, /etc/php/8.3/apache2/conf.d/20-sockets.ini, /etc/php/8.3/apache2/conf.d/20-sysvmsg.ini, /etc/php/8.3/apache2/conf.d/20-sysvsem.ini, /etc/php/8.3/apache2/conf.d/20-sysvshm.ini, /etc/php/8.3/apache2/conf.d/20-tokenizer.ini
PHP API	20230831
PHP Extension	20230831
Zend Extension	420230831
Zend Extension Build	API420230831,NTS
PHP Extension Build	API20230831,NTS

14.1 PHP 언어

예제 14.1 hello.php

- PHP 코드는 <?php 와 ?> 사이에 작성한다.

```
<!DOCTYPE html>
<html>
<head>
<title>Hello HTML</title>
</head>
<body>
<h1>
<?php
  echo "Hello World!";
?>
</h1>
</body>
</html>
```



예제 14.2 date.php

```
<!DOCTYPE html>
<html>
<body>
<h1>오늘의 날짜와 현재 시간은
<?php
    echo date("Y-m-d H-m-s");
?>
입니다.
</h1>
</body>
</html>
```

출력 결과

오늘의 날짜와 현재 시간은 2023-12-07 15-12-44입니다.

PHP 기본 문법

- 변수는 $ 기호로 시작한다.
  $name, $age 등
- 조건문
  if, else, elseif 등
- 반복문
  for, while, foreach
- 함수
  function 키워드로 함수를 정의한다.
- 웹 개발에 유용한 다양한 내장 함수와 라이브러리를 제공한다.

변수

- 변수를 선언할 때에는 $ 기호로 시작하며,
- echo 함수를 사용하여 그 값을 출력할 수 있다.

- 예제 14.3 intro.php

```
<?php
    $name = "홍길동";
    $age = 23;
    echo "안녕하세요, 저는 " . $name . "이고, " . $age . "살입니다.";
?>
```

- 출력 결과

안녕하세요, 저는 홍길동이고, 23살입니다.

조건문

- 예제 14.4 grade.php

```
<?php
$grade = 85;
if ($grade >= 90) {
   echo "A 학점";
} elseif ($grade >= 80) {
   echo "B 학점";
} else {
   echo "C 학점";
}
?>
```

- 출력 결과

B 학점

반복문

- for 문

```
for (초기화; 조건; 증감) {
// 반복해서 실행할 코드
}
```

- 예제 14.5 for.php

```
<?php
for ($i = 1; $i <= 5; $i++) {
    echo $i . "<br>";
}
?>
```

출력 결과

1
2
3
4
5

배열과 반복문

- 예제 14.6 cities.php

```
<?php
$cities = array("서울", "대전", "부산");
foreach ($cities as $city) {
  echo $city . "<br>";
}
?>
```

- 출력 결과

서울
대전
부산

연관 배열(associative array)

- 연관 배열
  - 키-값 쌍(key-value pair)으로 데이터를 저장하는 데이터 구조

- ```
  <?php
  $person = array("name" => "홍길순", "age" => 23, "city" => "서울");
  echo $person["name"]; // "홍길순" 출력
  ?>
  ```
  ```

함수

- 예제 14.7 sale.php

```
<?php
function saleprice($price, $sale) {
   return $price * (1 - $sale / 100);
}
echo saleprice(500, 20);
?>
```

- 출력 결과

```
400
```

사용자 입력과 폼 처리

- HTML 폼(form)
  - 웹 페이지에서 사용자로부터 입력을 받아 이들을 서버 프로그램에 전달
  - method는 폼 데이터를 서버에 전송하는 방법(GET 또는 POST)

```
<form action="서버 프로그램" method="전송방법">
<input type="입력타입" name="이름">
...
</form>
```

- GET 방식
  - GET 방식은 입력 정보를 서버로 전달하는 방식
  - 입력된 폼 데이터를 URL의 쿼리 문자열로 첨부하여 서버에 전송
  - 이 방식은 URL에 데이터가 노출되기 때문에 보안에 취약하다.
- POST 방식
  - POST 방식은 입력된 폼 데이터를 HTTP 요청의 본문(body)에 담아 서버에 전송한다.
  - 보안적으로 더 안전하며, 데이터의 양이나 타입에 제한이 없다.

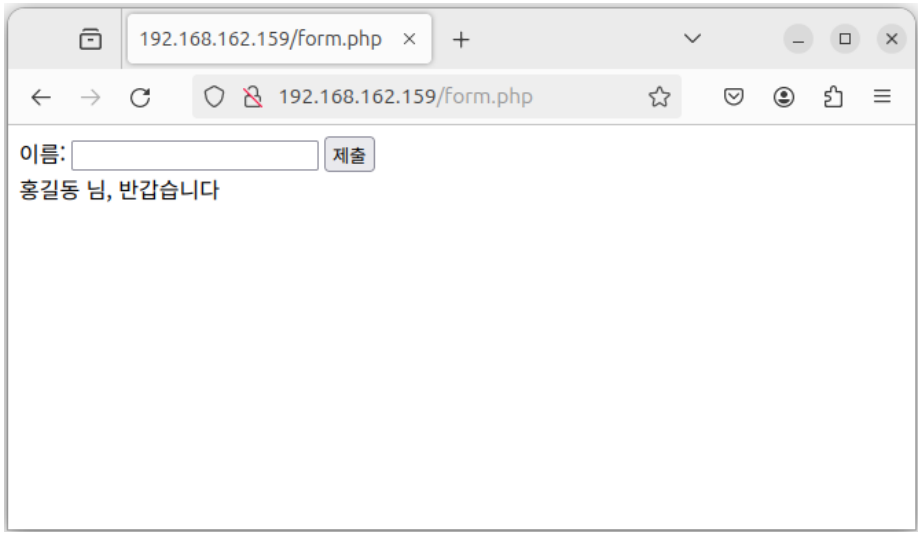
입력 타입

- <input type="입력타입" name="이름">

입력 타입	의미
button	클릭할 수 있는 버튼
text	텍스트를 입력할 수 있는 텍스트 입력 필드
password	비밀번호를 안전하게 입력할 수 있는 필드
radio	여러 옵션 중에서 하나를 선택할 수 있는 라디오 버튼
number	숫자를 입력할 수 있는 입력 필드
checkbox	여러 옵션 중에서 여러 개를 선택할 수 있는 체크박스
email	이메일 주소를 입력할 수 있는 입력 필드
submit	폼에 입력한 정보를 제출하는 버튼

사용자 입력 예제: form.php

```
<!DOCTYPE html>
<html>
<body>
<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
    이름: <input type="text" name="name">
    <input type="submit" value="제출">
</form>
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = $_POST['name' ];
    echo $name . " 님, " . "반갑습니다";
}
?>
</body>
</html>
```



PHP 내장 변수

- $_POST 변수
  - POST 방식으로 전달된 데이터를 가지고 있는 연관 배열
  - 폼을 통해 입력된 데이터를 이 변수를 통해 접근할 수 있다.
  - $_POST['name']은 입력된 'name'의 값을 나타낸다.

- $_SERVER 변수
  - 서버와 관련된 정보를 담고 있는 변수로, 전송 방식, 서버의 헤더, 파일 위치, 실행 스크립트 등의 정보를 포함한다.
  - $_SERVER[“REQUEST_METHOD”]는 입력 정보 전송 방식

- $_SERVER[“PHP_SELF”]
  - PHP에서 현재 실행 중인 스크립트 자체의 이름을 나타내며
  - 이 변수는 현재 PHP 스크립트의 파일 경로명을 포함한다.

내장 변수

PHP 내장 변수	설명
$_GET	URL의 쿼리 문자열에서 GET 방식으로 전달된 데이터를 가지고 있는 연관배열이다. 예를 들어, $_GET['id']는 URL에서 ?id=123과 같이 전달된 값을 가져온다.
$_POST	POST 방식으로 전송된 데이터를 가지고 있는 연관 배열로 폼(form)을 통해 전송된 데이터를 이 변수를 통해 접근할 수 있다.
$_SERVER	서버와 관련된 정보를 담고 있는 변수로, 서버의 헤더, 파일 위치, 실행 스크립트 등의 정보를 포함한다. 예를 들어, $_SERVER["REQUEST_METHOD"]는 입력 정보 전송 방식을 제공한다.
$_SESSION	세션 변수를 저장하는 데 사용되며, 여러 페이지나 요청 사이에서 사용자 정보를 유지하는 데 유용하다.
$_COOKIE	쿠키(cookie) 값을 가지고 있는 변수로 사용자의 브라우저에 저장된 쿠키 정보에 접근할 때 사용된다.
$_FILES	파일 업로드에 관련된 정보를 가지고 있는 변수로, 업로드된 파일의 정보를 포함한다.
$_ENV	환경 변수를 담고 있는 변수로, 서버 환경 설정과 관련된 정보를 포함한다.
$GLOBALS	현재 스크립트에서 사용 가능한 모든 변수의 전역화된 배열로 스크립트 내에서 정의된 모든 변수들을 포함한다.

14.3 MariaDB 데이터베이스

MariaDB 소개

- 오픈 소스
  - MariaDB는 GPL(General Public License) 라이선스 무료
- MySQL 호환성
  - MySQL과의 호환성을 유지하면서도,
  - MariaDB는 MySQL보다 추가 기능을 포함
- 성능 향상
  - MariaDB는 더 나은 성능
- 보안 강화
  - 데이터 암호화, SSL 지원, 접근 제어 및 보안 기능을 강화
- 활발한 커뮤니티
  - MariaDB는 활발한 오픈 소스 커뮤니티와 개발자

MariaDB 서버 설치

1. MariaDB 서버를 설치한다.

```
# apt install mariadb-server
```

2. 먼저 mariadb를 시작하고(start), 서비스를 활성화한다(enable). 실행 상태(status)를 확인

```
# systemctl start mariadb
# systemctl enable mariadb
# systemctl status mariadb
● mariadb.service - MariaDB 10.6.12 database server
   Loaded: loaded(/lib/systemd/system/mariadb.service;enabled; vendor preset: enabled)
   Active: active (running) since Wed 2023-12-06 13:48:05 KST; 3min 38s ago
   Docs: man:mariadbd(8)
...
```

3. MariaDB 연결

터미널에서 mariadb 명령을 실행하여 MariaDB에 연결하고 SQL 명령어를 사용할 수 있다.

```
# mariadb
Welcome to the MariaDB monitor. Commands end with ; or ₩g.
Your MariaDB connection id is 56
Server version: 10.6.12-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '₩h' for help. Type '₩c' to clear the current input statement.
MariaDB [(none)]>
```

14.4 MariaDB 데이터베이스 SQL 명령어

데이터베이스 관련 SQL

- 데이터베이스 생성 CREATE DATABASE
  - CREATE DATABASE 명령을 사용하여 데이터베이스를 생성할 수 있다.

```
> CREATE DATABASE my_database;
Query OK, 1 row affected (0.000 sec)
```

- 데이터베이스 선택 USE
  - 예를 들어, 다음과 같이 USE 문을 사용하여 데이터베이스를 선택할 수 있다.

```
> USE my_database;
```

- 테이블 생성 CREATE TABLE
  - CREATE TABLE 명령을 사용하여 테이블을 생성할 수 있다.

```
> CREATE TABLE users (
  username VARCHAR(50),
  email VARCHAR(50),
  password VARCHAR(200)
);
```

데이터베이스 관련 SQL

- 테이블 구조 확인 EXPLAIN

```
> EXPLAIN users
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| username | varchar(50)  | YES  |     | NULL    |       |
| email    | varchar(50)  | YES  |     | NULL    |       |
| password | varchar(200) | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
```

- 데이터베이스 및 테이블 삭제 DROP
  - DROP 문을 사용하여 테이블 또는 데이터베이스를 삭제할 수 있다.

```
> DROP TABLE users;
> DROP DATABASE my_database;
```

레코드 관련 SQL

- 테이블에 데이터 삽입 INSERT
  - INSERT 문을 사용하여 테이블에 데이터를 추가할 수 있다.

```
> INSERT INTO users (username, email, password) VALUES
   ('hong', 'gildong@gmail.com', ' 1234' );
```

- 테이블에서 데이터 조회 SELECT
  - SELECT 문을 사용하여 테이블의 모든 데이터를 조회할 수 있다.

```
> SELECT * FROM users;
+----------+-------------------+----------------------+
| username | email             | password             |
+----------+-------------------+----------------------+
| hong     | gildong@gmail.com | $2y$10$Bg ...        |
+----------+-------------------+----------------------+
1 rows in set (0.000 sec)
```

```
> SELECT email FROM users;
+---------------------+
||email                |
+---------------------+
| gildong@gmail.com |
+---------------------+
1 rows in set (0.000 sec)
```

- 테이블에서 데이터 삭제 DELETE
  - DELETE 문을 사용하여 테이블에서 특정 데이터를 삭제할 수 있다.

```
> DELETE FROM users WHERE username = 'hong';
```

데이터베이스 사용자 생성 및 권한 부여

- 사용자 생성 CREATE USER
  - username이라는 데이터베이스 사용자를 localhost에서 생성하고, 비밀번호 설정
  - CREATE USER 'username'@'localhost' IDENTIFIED BY 'password' ;

  > CREATE USER 'admin'@'localhost' IDENTIFIED BY '1234' ;

- 권한 부여 GRANT ALL PRIVILEGES
  - database_name 데이터베이스에 대해 username 사용자에게 모든 권한을 부여
  - GRANT ALL PRIVILEGES ON database_name.* TO 'username'@'localhost' ;

  > GRANT ALL PRIVILEGES ON my_database.* TO 'admin'@'localhost' ;

- 변경사항 적용 FLUSH PRIVILEGES
  - 권한 변경사항을 즉시 적용하기 위해 FLUSH PRIVILEGES 명령을 실행한다.

  > FLUSH PRIVILEGES;

14.5 웹 애플리케이션 프로그램

회원 가입

- 이 예제에서 사용할 데이터베이스 my_database와 회원 정보 테이블 users를 생성

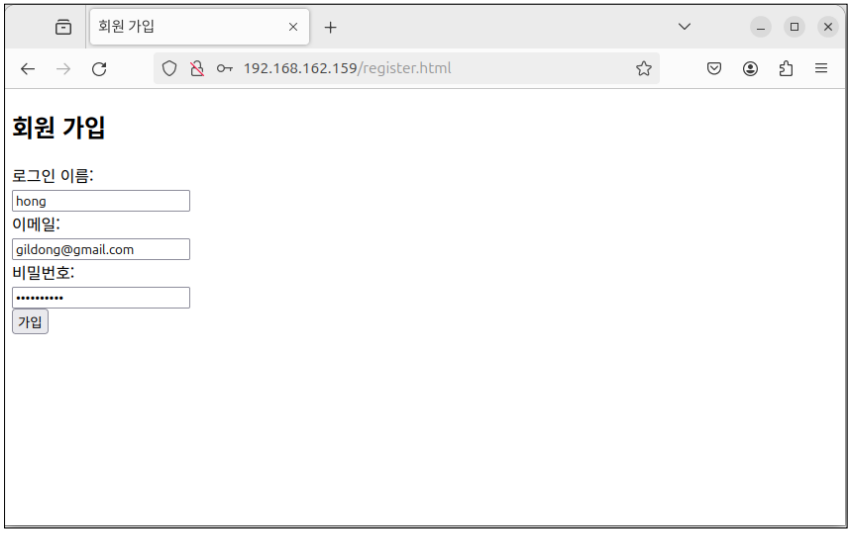
```
> CREATE DATABASE my_database;
> USE my_database;
> CREATE TABLE users (
    username VARCHAR(50),
    email VARCHAR(50),
    password VARCHAR(200)
);
```

- 데이터베이스 사용자 admin를 생성하고 my_database에 대한 모든 권한을 부여

```
> CREATE USER 'admin'@'localhost' IDENTIFIED BY '1234';
> GRANT ALL PRIVILEGES ON my_database.* TO 'admin'@'localhost';
> FLUSH PRIVILEGES;
```

회원 가입 폼 (register.html)

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>회원 가입</title>
</head>
<body>
  <h2>회원 가입</h2>
  <form action="register.php" method="post">
    <label for="username">로그인 이름:</label><br>
    <input type="text" name="username"><br>
    <label for="email">이메일:</label><br>
    <input type="text" name="email"><br>
    <label for="password">비밀번호:</label><br>
    <input type="password" name="password"><br>
    <input type="submit" value="가입">
  </form>
</body>
</html>
```



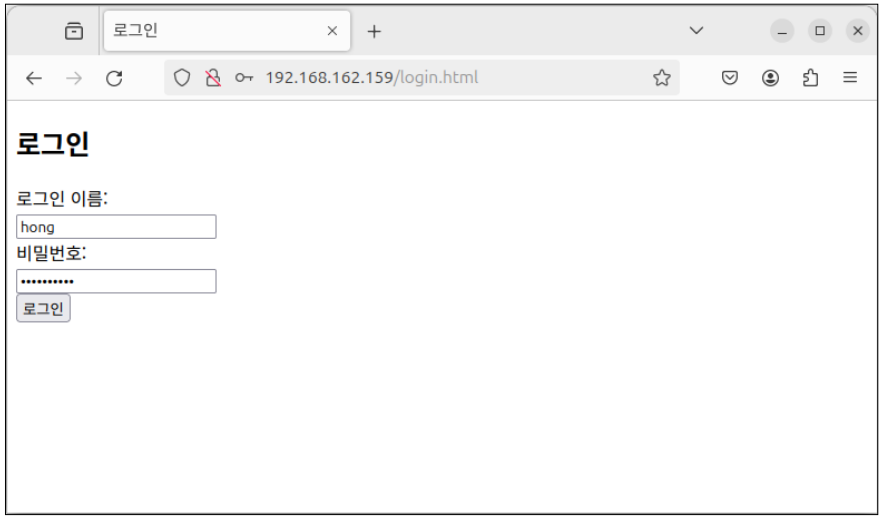
회원 가입 프로그램 (register.php)

```php
<?php
// 데이터베이스 연결 설정
$servername = "localhost";
$username = "admin";
$password = "1234";
$dbname = "my_database";
// 데이터베이스에 연결
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) { // 연결 확인
   die("Connection failed: " . $conn->connect_error);
}
// POST로 전송된 회원가입 정보 받기
if ($_SERVER["REQUEST_METHOD"] == "POST") {
   $username = $_POST['username' ];
   $email = $_POST['email' ];
   $password = $_POST['password'];
```

회원 가입 프로그램 (register.php)

```
    // 비밀번호 해싱
    $hashed_password = password_hash($password, PASSWORD_DEFAULT);
    // 사용자 정보 삽입 쿼리 실행
    $sql = "INSERT INTO users (username, email, password) VALUES
            ('$username', '$email', '$hashed_password')";
    if ($conn->query($sql) === TRUE) {
        echo "회원가입이 완료되었습니다!";
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }
    $conn->close();
}
?>
```

로그인 폼 (login.html)




```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>로그인</title>
</head>
<body>
  <h2>로그인</h2>
  <form action="login.php" method="post">
    <label for="username">로그인 이름:</label><br>
    <input type="text" id="username" name="username"><br>
    <label for="password">비밀번호:</label><br>
    <input type="password" id="password" name="password"><br>
    <input type="submit" value="로그인">
  </form>
</body>
</html>
```

로그인 처리 (login.php)

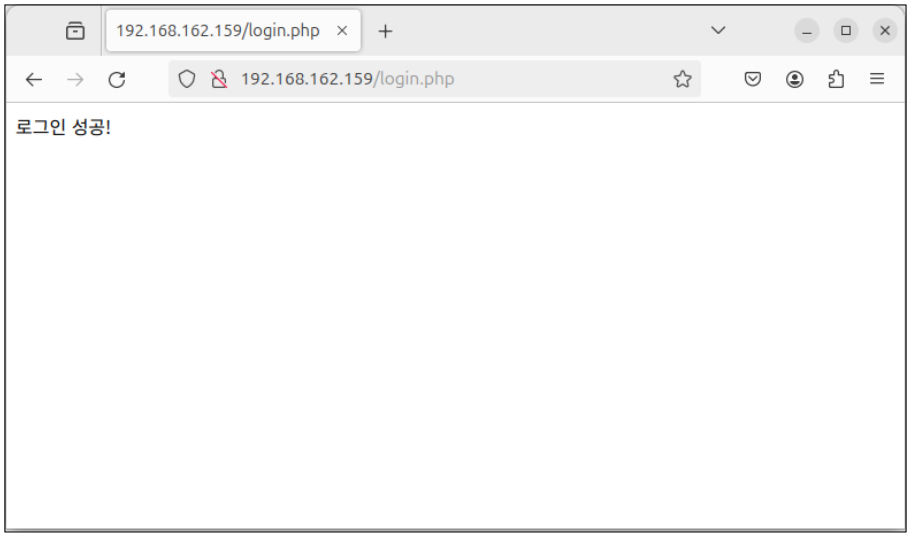
```php
<?php
// 데이터베이스 연결 설정
$servername = "localhost";
$username = "admin";
$password = "1234";
$dbname = "my_database";

// 데이터베이스에 연결
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) { // 연결 확인
  die("Connection failed: " . $conn->connect_error);
}

// POST로 전송된 로그인 정보 받기
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  $username = $_POST['username' ];
  $password = $_POST['password' ];
```

로그인 처리 (login.php)

```
    // 사용자 정보 조회 쿼리 실행
    $sql = "SELECT * FROM users WHERE username='$username'";
    $result = $conn->query($sql);
    if ($result->num_rows == 1) {          // 사용자가 존재하는 경우
        $row = $result->fetch_assoc();
        if (password_verify($password, $row['password'])) {// 비밀번호확인
            echo "로그인 성공!";
            // 여기에서 세션을 시작하는 등의 작업을 수행한다.
        } else { // 비밀번호 불일치
            echo "비밀번호가 일치하지 않습니다.";
        }
    } else {          // 사용자가 존재하지 않는 경우
        echo "사용자가 존재하지 않습니다.";
    }
    $conn->close();
}
?>
```



핵심 개념

- PHP는 웹 환경에서 동작하는 서버 측 스크립트 언어로 주로 동적인 웹 페이지를 생성하는 데 사용된다.
- PHP 코드는 <?php 와 ?> 사이에 작성한다.
- HTML 폼(form)을 사용하면 웹 페이지에서 사용자로부터 다양한 종류의 데이터를 입력받아 서버에 제출할 수 있다.