

TRƯỜNG ĐẠI HỌC NGÂN HÀNG TP. HCM



MÔN HỌC: LẬP TRÌNH WEB

CHƯƠNG 5 LẬP TRÌNH WEB VỚI PHP VÀ MYSQL

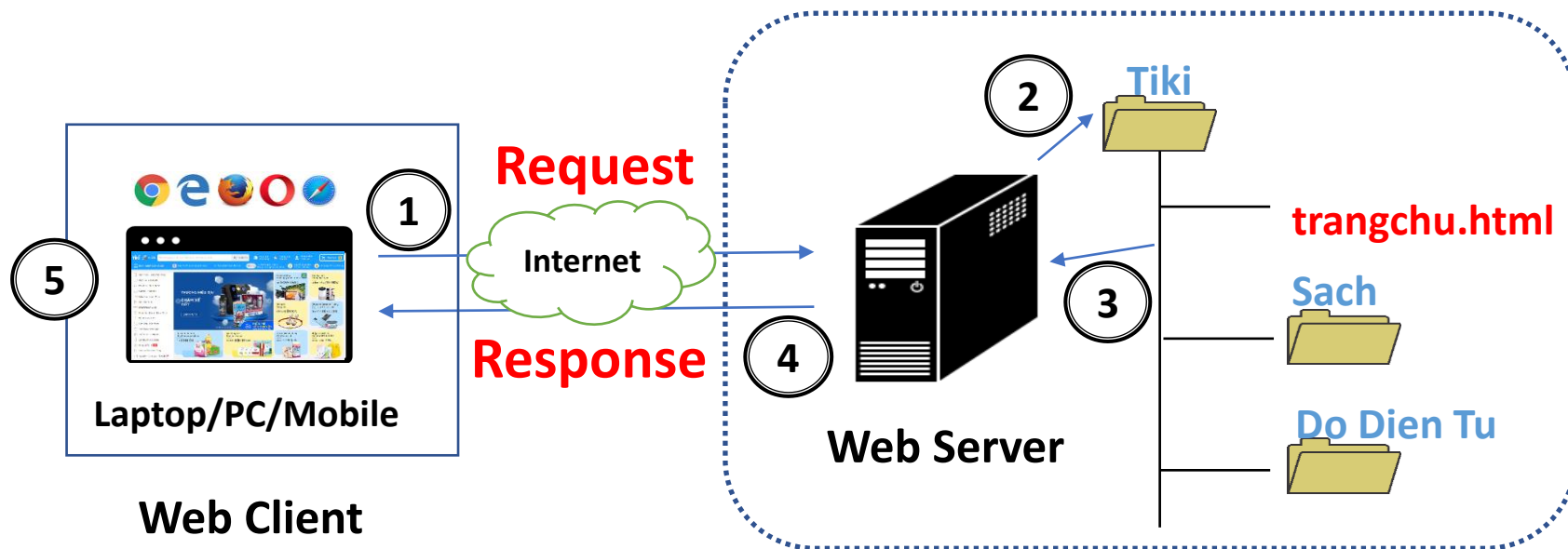


1. Mô hình hoạt động của website
2. Web Form
3. Kết nối PHP và MySQL
4. Sessions, Cookies và ứng dụng

- Website hoạt động theo mô hình khách - chủ (client – server)
- Trình khách (web client) gồm các ứng dụng (trình duyệt, . . .) được sử dụng để truy cập và gửi các yêu cầu đến trình chủ.
- Trình chủ là một máy chủ web (web server) nhận các yêu cầu từ trình khách và trả về kết quả dưới dạng HTML.
- Website tĩnh và website động có một số điểm khác biệt trong quá trình xử lý.

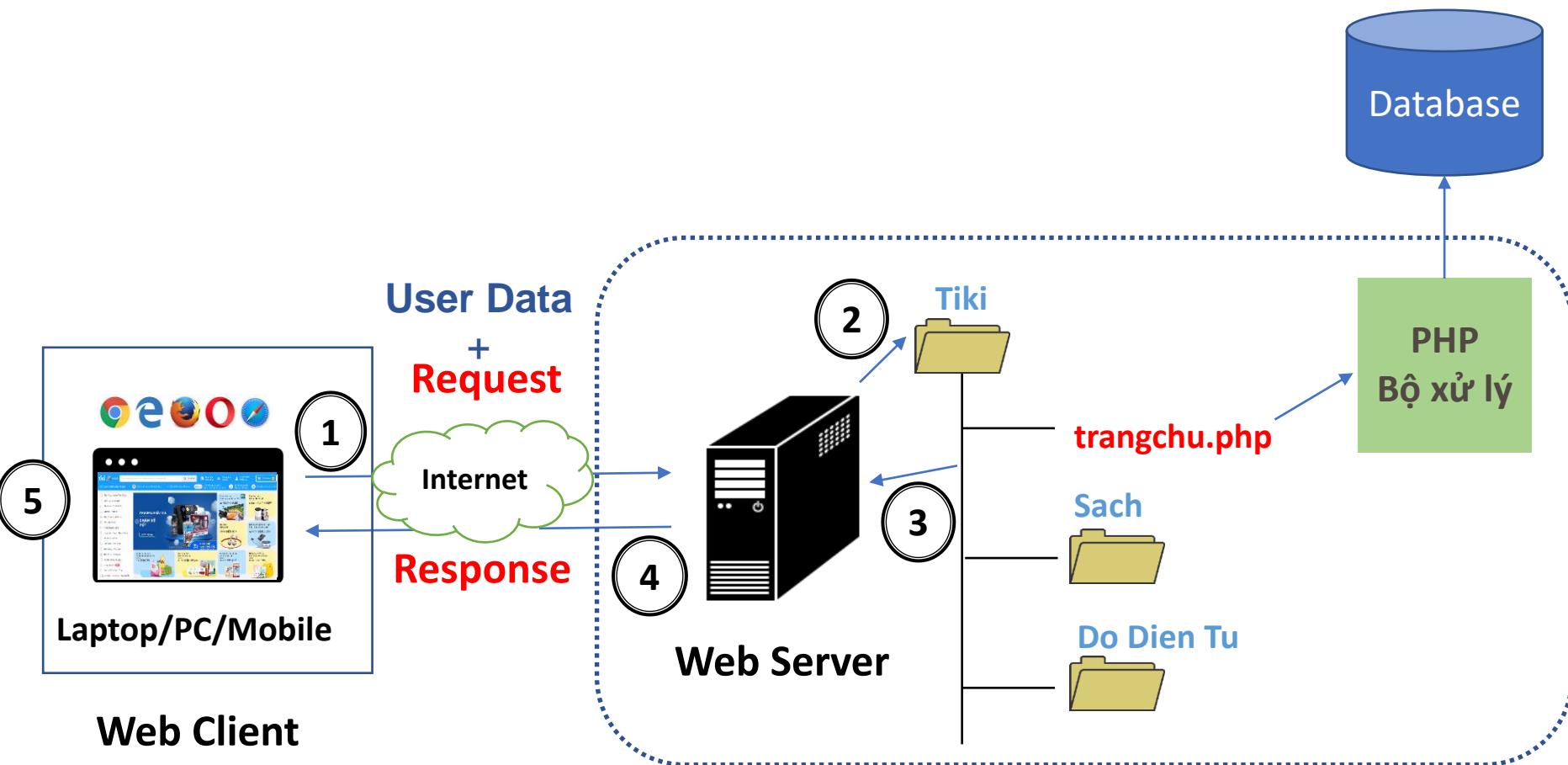
QUÁ TRÌNH XỬ LÝ CỦA WEBSITE TĨNH

- Website tĩnh có phần mở rộng là .html
- Khi nhận yêu cầu truy cập một file .html, máy chủ web tìm kiếm file và trả về nội dung (không xử lý) của file cho trình duyệt



<http://www.tiki.vn/trangchu.html>

QUÁ TRÌNH XỬ LÝ CỦA WEBSITE ĐỘNG



<http://www.tiki.vn/trangchu.php>

QUÁ TRÌNH XỬ LÝ CỦA WEBSITE ĐỘNG

- Website động có phần mở rộng phụ thuộc vào ngôn ngữ lập trình được sử dụng (PHP, Java). Ví dụ: .php, .jsp, . . .
- Có thể gửi kèm dữ liệu của người dùng khi gửi yêu cầu truy cập đến máy chủ web.
- Khi nhận yêu cầu truy cập một file .php, **máy chủ web** tìm kiếm file và gọi bộ xử lý của PHP để xử lý các đoạn mã bên trong file và trả về nội dung cho trình duyệt dưới dạng HTML.

QUÁ TRÌNH XỬ LÝ CỦA WEBSITE ĐỘNG

- Lập trình **front-end**: sử dụng HTML, CSS, Javascript để thiết kế, xây dựng giao diện website để người dùng tương tác (nhập dữ liệu, tìm kiếm thông tin, . . .)
- Lập trình **back-end**: sử dụng các ngôn ngữ lập trình: PHP, Java, . . . để xây dựng các đoạn mã xử lý các dữ liệu được gửi từ giao diện người dùng theo yêu cầu cụ thể và trả về kết quả cho người dùng.

URL (UNIFORM RESOURCE LOCATOR)

- Cú pháp đầy đủ:

scheme://<host>[:port][<path>[?<querystring>]]

- Đối với website:

- scheme: http(s) = HyperText Transfer Protocol (Secure)
(Giao thức truyền tải siêu văn bản)
- host: Tên miền hoặc địa chỉ IP của máy chủ
- port: nếu không có thông tin thì mặc định port là 80
- path: đường dẫn tới trang cần truy cập
- querystring: thông tin dữ liệu được gửi đến máy chủ



MÔ HÌNH HOẠT ĐỘNG CỦA WEBSITE

URL (UNIFORM RESOURCE LOCATOR)

- Cú pháp đầy đủ:

http://laptrinhweb.edu.vn:8080/xuly.php?key=value

↑
scheme

↑
host

↑
port

↑
path

↑
query string

GIỚI THIỆU

- Web Form là một trong những cách để người dùng của website tương tác với PHP và MySQL.
- Web Form hỗ trợ người dùng nhập và gửi các dữ liệu đến máy chủ để xử lý.
- Xử lý Web Form gồm nhiều công đoạn: Tạo form; gửi và nhận dữ liệu; xử lý dữ liệu, . . .

TẠO FORM

- Cặp thẻ HTML `<form>` `</form>` được sử dụng tạo form cho website.
- Thẻ `<form>` có các thuộc tính để xác định phương thức gửi dữ liệu và địa chỉ sẽ nhận và xử lý dữ liệu được gửi từ form.
- Nội dung bên trong cặp thẻ `<form>` `</form>` cần có các thẻ input để nhập dữ liệu.

TẠO FORM

```
<form action="file_name.php" method="post" >  
    Họ tên:    <input type = "text" name = "ho_ten"/><br/>  
    Năm sinh: <input type = "text" name = "nam_sinh"/><br/>  
    <input type="submit" value="Gửi">  
</form>
```

Họ tên:

Năm sinh:

CÁC THẺ NHẬP DỮ LIỆU TRONG FORM

■ Textbox

```
<input type="text" name="name" size="size"
      maxlength="length" value="value">
```

- Ví dụ: `<input type="text" name="ho_ten" >`

■ Checkbox

```
<input type="checkbox" name="name" value="value"
      checked="checked">
```

- Ví dụ: `<input type="checkbox" name="language">English`
`<input type="checkbox" name="language">French`

☐ English ☐ French

CÁC THẺ NHẬP DỮ LIỆU TRONG FORM

■ Radio button

```
<input type="radio" name="name" value="value"  
checked="checked">
```

- Ví dụ:

```
<input type="radio" name="gioi_tinh">Nam
```



```
<input type="radio" name="gioi_tinh">Nữ
```

☐ Nam ☐ Nữ

■ Hidden fields

```
<input type="hidden" name="name" value="value">
```

- Ví dụ:

```
<input type="hidden" name="id" value="2">
```

CÁC THẺ NHẬP DỮ LIỆU TRONG FORM

■ Text area

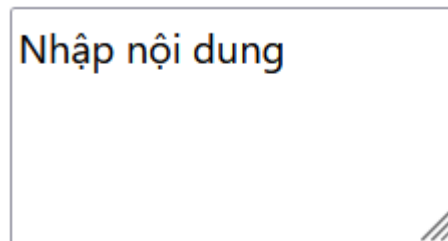
```
<textarea name="name" cols="width" rows="height"> </textarea>
```

- Ví dụ:

```
<textarea name="tom_tat" cols="15" rows="3"> </textarea>
```



```
<textarea name="tom_tat" cols="15" rows="3">Nhập nội dung</textarea>
```



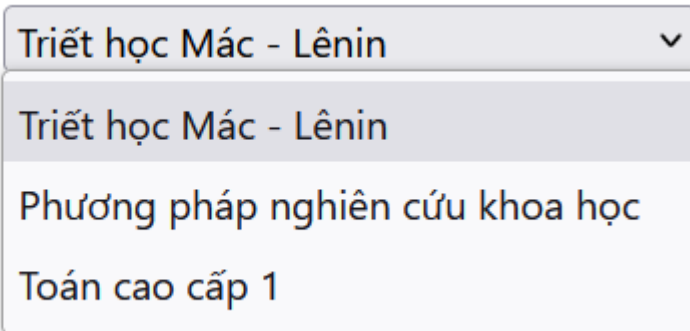
CÁC THẺ NHẬP DỮ LIỆU TRONG FORM

■ Combobox

- Ví dụ:

Chọn môn học:

```
<select name="mon_hoc">  
  <option value="1100001">Triết học Mác - Lênin</option>  
  <option value="1100002">Phương pháp nghiên cứu khoa học</option>  
  <option value="1100010">Toán cao cấp 1</option>  
</select>
```

Chọn môn học: 

GỬI DỮ LIỆU BẰNG PHƯƠNG THỨC POST

File (địa chỉ) xử lý dữ liệu,
có thể thay đổi tên

Phương thức gửi dữ liệu: **POST**

```
<form action="process_file.php" method="post">
```

Mã sinh viên:


```
</form>
```

Mã sinh viên:

Gửi

Tạo nút **Gửi**

Khi nhấn nút **Gửi**, dữ liệu trong form sẽ gửi đến file
process_file.php theo phương thức **POST**

GỬI DỮ LIỆU BẰNG PHƯƠNG THỨC **POST**

- **process_file.php** là nơi nhận dữ liệu và thực hiện xử lý theo một yêu cầu cụ thể.
- Khi dữ liệu được gửi lên theo phương thức POST → Tại **process_file.php**, các dữ liệu sẽ được lưu trong biến `$_POST`.
- `$_POST` là biến toàn cục, kiểu dữ liệu mảng. Sử dụng **`$_POST`**["giá trị thuộc tính **name**"] để lấy dữ liệu.

GỬI DỮ LIỆU BẰNG PHƯƠNG THỨC POST

```
<form action="process_file.php" method="post">
    Mã sinh viên: <input type="text" name="ma_sinh_vien"/>br>
    Họ tên: <input type="text" name="ho_ten"/><br>
    Ngày sinh: <input type="text" name="ngay_sinh"/><br>
    <input type="submit" value="Gửi">
</form>
```

Tại process_file.php

```
<?php
    var_dump($_POST);
?>
```



```
array (size=3)
    'ma_sinh_vien' => string '032000012' (length=9)
    'ngay_sinh' => string 'Test' (length=4)
    'ho_ten' => string '10/10/2001' (length=10)
```

- Để lấy dữ liệu mã sinh viên → **\$_POST**["ma_sinh_vien"]
- Sử dụng hàm **isset()** để kiểm tra dữ liệu có tồn tại?

```
if(isset($_POST["ma_sinh_vien"]))
    $maSV = $_POST["ma_sinh_vien"];
```

GỬI DỮ LIỆU BẰNG PHƯƠNG THỨC GET

File (địa chỉ) xử lý dữ liệu,
có thể thay đổi tên

Phương thức gửi dữ liệu: **GET**

```
<form action="process_file.php" method="get">  
Mã sinh viên: <input type="text" name="ma_sinh_vien"/><br>  
               <input type="submit" value="Gửi">  
</form>
```

Mã sinh viên: Tạo nút **Gửi**

localhost/laptrinhweb/process_file.php?ma_sinh_vien=032000012 → query string

Với phương thức GET, khi nhấn nút **Gửi** → dữ liệu trong form gửi đến file **process_file.php** thông qua URL

GỬI DỮ LIỆU BẰNG PHƯƠNG THỨC GET

- **process_file.php** là nơi nhận dữ liệu và thực hiện xử lý theo một yêu cầu cụ thể.
- Khi dữ liệu được gửi lên theo phương thức GET → Tại **process_file.php**, các dữ liệu sẽ được lưu trong biến `$_GET`.
- `$_GET` là biến toàn cục, kiểu dữ liệu mảng. Sử dụng **`$_GET`**["giá trị thuộc tính **name**"] để lấy dữ liệu.

GỬI DỮ LIỆU BẰNG PHƯƠNG THỨC GET

```
<form action="process_file.php" method="get">
    Mã sinh viên: <input type="text" name="ma_sinh_vien"/><br>
    Họ tên: <input type="text" name="ho_ten"/><br>
    Ngày sinh: <input type="text" name="ngay_sinh"/><br>
    <input type="submit" value="Gửi">
</form>
```

query string

/process_file.php?ma_sinh_vien=032000012&ngay_sinh=Test&ho_ten=10%2F10%2F2001

Tại process_file.php

```
<?php
    var_dump($_GET);
?>
```



```
array (size=3)
    'ma_sinh_vien' => string '032000012' (length=9)
    'ngay_sinh' => string 'Test' (length=4)
    'ho_ten' => string '10/10/2001' (length=10)
```

- Để lấy dữ liệu mã sinh viên → **`$_GET["ma_sinh_vien"]`**
- Sử dụng hàm **`isset()`** để kiểm tra dữ liệu có tồn tại?

```
if(isset($_GET["ma_sinh_vien"]))
    $maSV = $_GET["ma_sinh_vien"];
```

MỘT SỐ ĐIỂM KHÁC BIỆT GIỮA GET VÀ POST

Tiêu chí	GET	POST
Truyền dữ liệu	<p>Dữ liệu được gửi thông qua URL</p> <p>Kích thước dữ liệu phụ thuộc vào độ dài tối đa của URL</p> <p>Không gửi được dữ liệu dạng tập tin (hình ảnh, file văn bản, . . .)</p>	<p>Dữ liệu được gửi thông qua HTTP header</p> <p>Kích thước dữ liệu không phụ thuộc URL</p> <p>Gửi được dữ liệu dạng tập tin (hình ảnh, file văn bản, . . .)</p>
Bảo mật	Kém.	An toàn hơn.
Tốc độ	Tốc độ thực thi nhanh hơn POST.	Tốc độ thực thi chậm hơn GET.
Sử dụng	<p>Không nên sử dụng GET để gửi thông tin Password và các thông tin nhạy cảm.</p> <p>Thường sử dụng cho các yêu cầu đọc và tìm kiếm thông tin.</p>	<p>Được sử dụng khi gửi thông tin Password và các dữ liệu nhạy cảm.</p>

MỘT SỐ TRẠNG THÁI PHẢN HỒI TỪ MÁY CHỦ

Mã trạng thái (Status Code)	Tên trạng thái	Ý nghĩa
200	OK	Thành công.
403	Forbidden	Lỗi: Máy chủ từ chối yêu cầu truy cập của người dùng, xảy ra khi người dùng không đủ quyền để truy cập tài nguyên.
404	Not found	Lỗi: Khi người dùng truy cập tài nguyên không tồn tại trên máy chủ.
500	Internal Server Error	Lỗi: Xảy ra khi các đoạn mã xử lý tại máy chủ bị lỗi.

TẠO KẾT NỐI

```
<?php
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "qlsv";

    // Create connection
    $conn = mysqli_connect($servername, $username,
    $password,$dbname);
    mysqli_set_charset($conn , 'UTF8');
    // Check connection
    if (!$conn) {
        die("Connection failed: " . mysqli_connect_error());
    }
    echo "Connected successfully";
?>
```

- Tạo file kết nối với MySQL và lưu file: **config.php**

LẤY DỮ LIỆU (Select)

<?php

```
require_once "config.php";
$sql = "SELECT * FROM SinhVien";
$result = mysqli_query($conn, $sql);
if (mysqli_num_rows($result) > 0) {
    // output data of each row
    while($row =mysqli_fetch_assoc($result)){
        // xử lý hiển thị dữ liệu
    }
}
else {
    echo "0 results";
}
mysqli_close($conn);
```

mysqli_query: thực
thi câu truy vấn

mysqli_num_rows:
lấy số dòng dữ liệu
trả về

mysqli_fetch_assoc:
lấy dữ liệu từng
dòng và đưa vào
biến mảng

?>

THÊM DỮ LIỆU (Insert)

```
<?php
    require_once "config.php";
    $sql = "INSERT INTO SinhVien (MaSinhVien, Ho, Ten)
           values('03210002', 'Test', 'Test')";

    if (mysqli_query($conn, $sql) > 0) {
        echo "Thêm dữ liệu thành công";
    }
    else {
        echo "Lỗi: " . $sql . "<br>" . mysqli_error($conn);
    }
    mysqli_close($conn);

?>
```

THÊM NHIỀU DÒNG DỮ LIỆU (Insert Multiple Data)

```
<?php
```

```
    require_once "config.php";
    $sql = "INSERT INTO SinhVien (MaSinhVien, Ho, Ten)
           values('03210002', 'Test', 'Test');";
    $sql .= "INSERT INTO SinhVien (MaSinhVien, Ho, Ten)
           values('03210003', 'Test1', 'Test1');";

    if (mysqli_multi_query($conn, $sql) > 0) {
        echo "Thêm dữ liệu thành công";
    }
    else {
        echo "Lỗi: " . $sql . "<br>" . mysqli_error($conn);
    }
    mysqli_close($conn);
```

```
?>
```

CẬP NHẬT DỮ LIỆU (Update)

```
<?php
```

```
require_once "config.php";  
$sql = "UPDATE SinhVien set Ho = 'An'  
       where MaSinhVien = '03210002'";  
  
if (mysqli_query($conn, $sql) > 0) {  
    echo "Cập nhật dữ liệu thành công";  
}  
else {  
    echo "Lỗi: " . $sql . "<br>" . mysqli_error($conn);  
}  
mysqli_close($conn);
```

```
?>
```

XÓA DỮ LIỆU (Delete)

```
<?php
```

```
require_once "config.php";  
$sql = "DELETE FROM SinhVien  
       where MaSinhVien = '03210002'";  
  
if (mysqli_query($conn, $sql) > 0) {  
    echo "Xóa dữ liệu thành công";  
}  
else {  
    echo "Lỗi: ". $sql . "<br>".mysqli_error($conn);  
}  
mysqli_close($conn);
```

```
?>
```

PREPARED STATEMENTS

- Prepared Statement là một tính năng hỗ trợ thực thi hiệu quả các câu lệnh SQL (giống nhau về cấu trúc) lặp lại nhiều lần .
- Ngoài ra Prepared Statement rất hữu ích để chống lại kỹ thuật tấn công SQL injection

LẤY DỮ LIỆU (PREPARED STATEMENTS)

<?php

```
$sql = "SELECT * FROM SinhVien WHERE MaLop = ?";
$stmt = mysqli_prepare($conn,$sql);
```

```
$lop = 'CNTT08';
mysqli_stmt_bind_param($stmt,'s',$lop);
mysqli_stmt_execute($stmt);
```

```
$result = mysqli_stmt_get_result($stmt);
if (mysqli_num_rows($result) > 0){
    while($row = mysqli_fetch_assoc($result)){
        // Xử lý hiển thị dữ liệu
    }
}
else {
    echo "0 results";
}
mysqli_close($conn);
```

's': cho biết giá trị của biến **\$lop** là kiểu string.

'i': integer

'd': double

'b': BLOB

?>

THÊM DỮ LIỆU (PREPARED STATEMENTS)

<?php

```
require_once "config.php";
$sql = "INSERT INTO SinhVien (MaSinhVien, Ho, Ten)
      values(?, ?, ?)";
$stmt = mysqli_prepare($conn,$sql);

$maSV = '03200001';
$ho = 'Test';
$ten = 'Test';
mysqli_stmt_bind_param($stmt, 'sss', $maSV, $ho, $ten);
mysqli_stmt_execute($stmt);

$maSV = '03200002';
$ho = 'Test1';
$ten = 'Test1';
mysqli_stmt_bind_param($stmt, 'sss', $maSV, $ho, $ten);
mysqli_stmt_execute($stmt);
```

'sss': cho biết
giá trị của biến
\$maSV, \$ho, \$ten
là kiểu string.

?>

CẬP NHẬT DỮ LIỆU (PREPARED STATEMENTS)

<?php

```
require_once "config.php";  
$sql = "UPDATE SinhVien set Ho = 'An'  
       where MaSinhVien = ?";  
$stmt = mysqli_prepare($conn,$sql);  
  
$maSV = '03200001';  
mysqli_stmt_bind_param($stmt,'s',$maSV);  
mysqli_stmt_execute($stmt);
```

?>

XÓA DỮ LIỆU (PREPARED STATEMENTS)

<?php

```
require_once "config.php";  
$sql = "DELETE FROM SinhVien  
        where MaSinhVien = ?";  
$stmt = mysqli_prepare($conn,$sql);  
  
$maSV = '03200001';  
mysqli_stmt_bind_param($stmt,'s',$maSV);  
mysqli_stmt_execute($stmt);
```

?>

- Khi truy cập vào các website, máy chủ không thể nhận biết được các thông tin người dùng truy cập.
- Làm sao website nhận biết được người dùng để có những tác vụ xử lý phù hợp?
- Session, Cookie giúp lưu trữ các thông tin của người dùng cho các mục đích cụ thể (duy trì đăng nhập, theo dõi trạng thái, hành vi của người dùng, . . .)

SESSION

- Một session dùng để lưu trữ tạm thời thông tin người dùng trên máy chủ.
- Các thông tin được lưu trữ trong các biến session với nhiều kiểu dữ liệu và có thể sử dụng ở tất cả các trang của một website (cùng domain).
- Session được lưu trữ trên máy chủ và được mã hóa → đảm bảo tính bảo mật.
- Các giá trị của session sẽ tự động xóa khi người dùng đăng xuất hoặc đóng trình duyệt.

SỬ DỤNG SESSION

- Để bắt đầu sử dụng session, cần sử dụng hàm:
`session_start();`
- Tất cả session được lưu trữ trong biến toàn cục `$_SESSION`.
- Khởi tạo một session:

```
$_SESSION['session_name'] = 'session_value';
```

SỬ DỤNG SESSION

- Xuất giá trị của session:

```
echo $_SESSION['session_name'];
```

- Hủy một session:

```
unset($_SESSION['session_name']);
```

- Hủy một session:

```
session_destroy()
```

BÀI TẬP SESSION

- Tạo một trang web với tên file là: page1.php và khởi tạo một biến session với session_name là 'username'.
- Tạo một trang web với tên file là: page2.php, (cùng thư mục với page1.php) và in ra giá trị của biến session được khởi tạo tại page1.php

COOKIE

- Cookie giúp website theo dõi được các hành vi của người dùng và lịch sử truy cập của trình duyệt người dùng.
- Cookie chỉ lưu trữ kiểu dữ liệu chuỗi.
- Cookie được lưu trữ trên máy tính người dùng dưới dạng các tập tin văn bản có thể đọc được → bảo mật kém.

SỬ DỤNG COOKIE

■ Tạo cookie:

```
setcookie(name, value, expire, path, domain, secure, httponly);
```

- **name:** Tên cookie.
- **value:** Giá trị cookie.
- **expire:** Thời gian tồn tại của cookie. Nếu bằng 0, cookie sẽ bị xóa cùng với session.
- **path:** Đường dẫn sử dụng cookie. Nếu bằng "/", cookie sử dụng trên toàn bộ domain.
- **domain:** Tên miền sử dụng cookie. Nếu bằng "buh.edu.vn", cookie chỉ sử dụng được trên toàn bộ tên miền con (subdomain) của "buh.edu.vn".
- **secure:** Nếu bằng True, cookie chỉ sử dụng cho các kết nối bảo mật.
- **HTTPOnly:** Nếu bằng True, cookie chỉ có thể sử dụng thông qua giao thức HTTP.

SỬ DỤNG COOKIE

- Ví dụ tạo cookie:

```
$cookie_name = "user";  
$cookie_value = "John Doe";  
Setcookie ($cookie_name, $cookie_value,  
           time() + (86400 * 30), "/"); // 86400 = 1 day
```

SỬ DỤNG COOKIE

- Xuất giá trị cookie:

```
echo $_COOKIE['name'];
```

- Thay đổi giá trị cookie:
 - Sử dụng lại setcookie với tham số name là tên cookie muốn thay đổi giá trị.
- Xóa cookie:

```
setcookie("user", "", time() - 3600, '/');
```

BÀI TẬP COOKIE

- Tạo một trang web với tên file là: page1.php và khởi tạo một biến cookie với tên là 'username' có thời gian tồn tại là 5 phút và xuất ra giá trị của biến cookie vừa tạo.
- Tắt trình duyệt và mở lại trang page1.php

HỎI & ĐÁP