**ĐỒ ÁN LẬP TRÌNH WEB – SHOP CẦU LÔNG**

1. INDEX.HTML VÀ INDEX.PHP

PHP và MYSQL giao tiếp được với nhau thông qua cái gì?

**ChatGPT đã nói:**

PHP và MySQL giao tiếp với nhau thông qua **các API (Application Programming Interface)**, cụ thể là các **thư viện hoặc extension PHP dùng để kết nối và tương tác với MySQL**. Các API phổ biến nhất để PHP giao tiếp với MySQL gồm:

**🔹 1. MySQLi (MySQL Improved)**

* **Giới thiệu:** Là extension cải tiến so với mysql cũ (đã bị loại bỏ từ PHP 7).
* **Hỗ trợ:** MySQL 4.1 trở lên.
* **Tính năng:** Hỗ trợ cả kết nối hướng thủ tục và hướng đối tượng, chuẩn bị truy vấn (prepared statements), giao dịch...
* **Ví dụ:**

php

Sao chépChỉnh sửa

$conn = new mysqli("localhost", "username", "password", "database");

$result = $conn->query("SELECT \* FROM users");

**🔹 2. PDO (PHP Data Objects)**

* **Giới thiệu:** Là một lớp trừu tượng cơ sở dữ liệu, hỗ trợ nhiều hệ quản trị khác nhau (MySQL, PostgreSQL, SQLite...).
* **Ưu điểm:** Code dễ bảo trì và có thể chuyển đổi giữa các CSDL dễ dàng.
* **Ví dụ:**

php

Sao chépChỉnh sửa

$pdo = new PDO("mysql:host=localhost;dbname=database", "username", "password");

$stmt = $pdo->query("SELECT \* FROM users");

**🧠 Tóm lại:**

* PHP giao tiếp với MySQL **thông qua các extension (API)** như MySQLi và PDO.
* Hai bên kết nối với nhau **qua giao thức TCP/IP** (thường là localhost:3306) hoặc qua **socket nội bộ** nếu chạy cùng máy chủ.

Nếu bạn muốn mình giải thích sâu hơn phần nào (như luồng xử lý, prepared statements, hay bảo mật), cứ hỏi nhé!

**Bạn đã nói:**

fetch và ajax là gì?

**ChatGPT đã nói:**

Rất hay! Cả **Fetch** và **AJAX** đều là **kỹ thuật dùng để gửi và nhận dữ liệu giữa trình duyệt và máy chủ (server) mà không cần reload lại trang web**. Tuy nhiên, chúng khác nhau ở cách thực hiện và thời điểm xuất hiện.

**✅ 1. AJAX (Asynchronous JavaScript and XML)**

**➤ Là gì?**

* Là một **kỹ thuật** chứ không phải là một ngôn ngữ hay công nghệ riêng lẻ.
* Sử dụng XMLHttpRequest (hoặc sau này là fetch) để giao tiếp bất đồng bộ với server.
* Dữ liệu thường được gửi/nhận là **JSON** (dù tên có chữ XML).

**➤ Ví dụ (AJAX thuần):**

javascript

Sao chépChỉnh sửa

var xhr = new XMLHttpRequest();

xhr.open("GET", "data.php", true);

xhr.onreadystatechange = function() {

if (xhr.readyState === 4 && xhr.status === 200) {

console.log(xhr.responseText);

}

};

xhr.send();

**✅ 2. Fetch API**

**➤ Là gì?**

* Là **API hiện đại hơn thay thế XMLHttpRequest**, được hỗ trợ rộng rãi từ các trình duyệt hiện nay.
* Viết ngắn gọn, dễ đọc, hỗ trợ Promise.

**➤ Ví dụ (Fetch):**

javascript

Sao chépChỉnh sửa

fetch('data.php')

.then(response => response.json())

.then(data => {

console.log(data);

})

.catch(error => console.error('Error:', error));

**⚖️ So sánh nhanh**

| **Tiêu chí** | **AJAX (XMLHttpRequest)** | **Fetch API** |
| --- | --- | --- |
| Ra đời | Cũ hơn (từ 2005) | Mới hơn (ES6+) |
| Viết code | Dài, phức tạp hơn | Ngắn gọn, dễ đọc |
| Hỗ trợ Promise | ❌ Không | ✅ Có |
| Xử lý lỗi | Khó | Dễ với .catch() |
| Dữ liệu trả về | JSON, XML, text... | JSON, text, blob... |
| Hỗ trợ rộng | Rất phổ biến | Hiện đại, đầy đủ |

**💡 Kết luận:**

* **AJAX** là kỹ thuật tổng thể.
* **Fetch** là **một cách hiện đại để thực hiện AJAX**.
* Nếu bạn viết web hiện đại, **nên dùng fetch** hoặc các thư viện như **Axios**.

Nếu bạn cần ví dụ PHP phía server để tương tác với Fetch/AJAX thì mình có thể cung cấp thêm nhé!

**Bạn đã nói:**

vẫn chưa hiểu

**ChatGPT đã nói:**

Không sao cả! Mình sẽ giải thích lại **đơn giản và dễ hiểu hơn**, bằng ví dụ thực tế.

**🔍 Vấn đề cần giải quyết:**

Giả sử bạn đang làm một trang web, và bạn muốn khi người dùng bấm nút **"Xem thông tin"**, trang web sẽ **lấy dữ liệu từ server (PHP)** và hiển thị kết quả **mà không cần tải lại trang**.

**✅ AJAX và Fetch là cách để làm điều đó.**

Chúng cho phép **JavaScript trong trình duyệt** gửi yêu cầu (request) đến **PHP trên server**, và nhận lại dữ liệu.

**📦 Ví dụ dễ hiểu**

**🔹 Giao diện HTML:**

html

Sao chépChỉnh sửa

<button onclick="layThongTin()">Xem thông tin</button>

<div id="ketqua"></div>

**🔹 File PHP (server) - data.php:**

php

Sao chépChỉnh sửa

<?php

$data = [

'ten' => 'Nguyễn Văn A',

'tuoi' => 25

];

echo json\_encode($data);

**✅ Cách 1: Dùng AJAX (cũ)**

javascript

Sao chépChỉnh sửa

function layThongTin() {

var xhr = new XMLHttpRequest();

xhr.open("GET", "data.php", true);

xhr.onreadystatechange = function() {

if (xhr.readyState == 4 && xhr.status == 200) {

var duLieu = JSON.parse(xhr.responseText);

document.getElementById("ketqua").innerHTML =

"Tên: " + duLieu.ten + "<br>Tuổi: " + duLieu.tuoi;

}

};

xhr.send();

}

**✅ Cách 2: Dùng Fetch (hiện đại hơn)**

javascript

Sao chépChỉnh sửa

function layThongTin() {

fetch("data.php")

.then(response => response.json())

.then(duLieu => {

document.getElementById("ketqua").innerHTML =

"Tên: " + duLieu.ten + "<br>Tuổi: " + duLieu.tuoi;

})

.catch(error => console.error('Lỗi:', error));

}

**🧠 Tổng kết lại lần nữa:**

|  | **AJAX** | **Fetch** |
| --- | --- | --- |
| Là gì? | Kỹ thuật dùng XMLHttpRequest | Cách hiện đại hơn để gửi/nhận dữ liệu |
| Dùng làm gì? | Lấy/gửi dữ liệu từ server mà không cần tải lại trang |  |
| Ưu điểm | Phổ biến, dùng được ở nhiều nơi | Code ngắn gọn, dễ đọc hơn |
| Hiện nay nên dùng? | ❌ Hạn chế dùng | ✅ Nên dùng Fetch |

Nếu bạn muốn, mình có thể làm 1 ví dụ trọn bộ (HTML + JS + PHP) cho bạn copy chạy thử luôn. Muốn không?

**Bạn đã nói:**

JSON là gì?

**ChatGPT đã nói:**

Rất tốt, câu hỏi này rất quan trọng trong lập trình web hiện đại.

**🧩 JSON là gì?**

**✅ JSON (viết tắt của JavaScript Object Notation) là:**

* **Một định dạng dữ liệu nhẹ** để **trao đổi dữ liệu giữa máy khách (trình duyệt) và máy chủ (server)**.
* Dễ đọc, dễ viết cho con người.
* Dễ phân tích, xử lý bởi máy.

**📦 Cách JSON trông như thế nào?**

JSON rất giống với **đối tượng trong JavaScript**.

**🔹 Ví dụ:**

json

Sao chépChỉnh sửa

{

"ten": "Nguyễn Văn A",

"tuoi": 25,

"diaChi": {

"thanhPho": "Hà Nội",

"quocGia": "Việt Nam"

},

"soThich": ["bóng đá", "âm nhạc", "lập trình"]

}

Đây là một **đối tượng JSON** gồm:

* 3 cặp key-value đơn giản: ten, tuoi, soThich
* 1 đối tượng lồng bên trong: diaChi

**🔁 JSON dùng để làm gì?**

* Khi bạn dùng **AJAX hoặc Fetch**, bạn thường nhận dữ liệu từ server **dưới dạng JSON**.
* Trình duyệt sẽ chuyển JSON đó thành **đối tượng JavaScript** để bạn có thể dễ dàng xử lý.

**📥 JSON trong PHP và JavaScript**

**🔸 PHP → JSON**

Bạn có thể dùng json\_encode() để chuyển dữ liệu PHP thành JSON:

php

Sao chépChỉnh sửa

$data = ['ten' => 'Nguyễn Văn A', 'tuoi' => 25];

echo json\_encode($data);

Kết quả:

json

Sao chépChỉnh sửa

{"ten":"Nguyễn Văn A","tuoi":25}

**🔸 JavaScript → sử dụng JSON**

Sau khi Fetch dữ liệu:

javascript

Sao chépChỉnh sửa

fetch("data.php")

.then(response => response.json()) // chuyển JSON thành JS object

.then(data => {

console.log(data.ten); // Truy cập dữ liệu như JS object

});

**🧠 Tóm gọn:**

| **Đặc điểm** | **JSON** |
| --- | --- |
| Định dạng | Văn bản (text) |
| Dùng để | Gửi/nhận dữ liệu giữa client và server |
| Hình thức | Key - Value (giống JS object) |
| Đọc dễ không? | ✅ Dễ đọc và xử lý |

Nếu bạn cần một bài tập nhỏ để hiểu JSON rõ hơn, mình có thể đưa ví dụ thực hành cho bạn thử nhé?

**Bạn đã nói:**

vẫn chưa hiểu

**ChatGPT đã nói:**

Không sao cả! Mình sẽ giải thích lại **JSON** theo cách **rất đơn giản, dễ hình dung**, như bạn chưa từng lập trình cũng hiểu được.

**🧠 JSON là gì?**

JSON giống như **một cách viết thông tin theo cấu trúc mà máy tính có thể hiểu**, và con người cũng dễ đọc.

**🧊 Ví dụ dễ hiểu nhất:**

**📝 Thông tin của một người:**

**Bạn viết bình thường (tiếng Việt):**

less

Sao chépChỉnh sửa

Tên: Nguyễn Văn A

Tuổi: 25

Sở thích: bóng đá, âm nhạc

**Viết theo kiểu JSON sẽ là:**

json

Sao chépChỉnh sửa

{

"ten": "Nguyễn Văn A",

"tuoi": 25,

"soThich": ["bóng đá", "âm nhạc"]

}

**👉 Nghĩa là:**

* {}: mở đầu và kết thúc một **đối tượng**
* "ten": tên của thuộc tính (key)
* "Nguyễn Văn A": giá trị (value) tương ứng với key
* Dấu : để nối key và value
* Dấu , để ngăn cách các cặp key-value

**🧑‍💻 Khi nào dùng JSON?**

Giả sử bạn có **trang web**, khi bạn nhấn nút “Lấy thông tin”, trình duyệt sẽ **gửi yêu cầu đến máy chủ**, máy chủ trả lại dữ liệu như sau (JSON):

json

Sao chépChỉnh sửa

{

"ten": "Nguyễn Văn A",

"tuoi": 25

}

Rồi JavaScript đọc JSON đó và hiển thị:

less

Sao chépChỉnh sửa

Tên: Nguyễn Văn A

Tuổi: 25

**📌 Vậy JSON là gì?**

✅ Là **cách viết dữ liệu có cấu trúc**, để **máy chủ và trình duyệt trao đổi thông tin với nhau**.

**🔄 JSON so với kiểu dữ liệu khác:**

| **Kiểu** | **Dễ đọc không?** | **Có cấu trúc không?** | **Máy có hiểu được không?** |
| --- | --- | --- | --- |
| Văn bản thường | Dễ | ❌ Không | ❌ Không rõ ràng |
| JSON | ✅ Rất dễ | ✅ Có | ✅ Có |

**📣 Tóm gọn lại (một câu):**

JSON là **cách viết dữ liệu giống kiểu JavaScript**, giúp **trình duyệt và server trao đổi thông tin với nhau dễ dàng**.

Bạn có muốn mình làm một ví dụ cực đơn giản mà bạn có thể **copy, dán vào máy và chạy thử** để tự thấy JSON hoạt động như thế nào không?

**Bạn đã nói:**

vẫn chưa hiểu API là gì?

**ChatGPT đã nói:**

Rất tốt, câu hỏi này **rất quan trọng trong lập trình và công nghệ**. Mình sẽ giải thích **API là gì** theo cách đơn giản, dễ hình dung, **không cần nhiều kiến thức kỹ thuật**.

**✅ API là gì?**

**👉 API là viết tắt của:**

**A**pplication **P**rogramming **I**nterface  
(tạm dịch: Giao diện lập trình ứng dụng)

**🧠 API giống như... menu ở nhà hàng**

**📌 Hãy tưởng tượng:**

* Bạn đi vào nhà hàng và gọi món.
* Bạn không vào bếp, không cần biết đầu bếp nấu thế nào.
* Bạn **chỉ nhìn vào menu**, chọn món, gọi phục vụ.
* Phục vụ sẽ mang thức ăn ra cho bạn.

👉 Trong ví dụ này:

| **Thực thể** | **Ý nghĩa** |
| --- | --- |
| **Menu** | API (giao diện để bạn gọi dịch vụ) |
| **Bạn** | Người dùng hoặc phần mềm |
| **Nhà bếp** | Máy chủ, nơi xử lý yêu cầu |
| **Phục vụ** | API giúp bạn nói chuyện với máy chủ |
| **Món ăn được mang ra** | Dữ liệu từ máy chủ gửi về |

**📱 Trong lập trình:**

**API là cách để một chương trình (hoặc trang web) giao tiếp với một chương trình khác.**

Ví dụ:

* Website của bạn muốn **lấy thời tiết**.
* Bạn không tự đo nhiệt độ ngoài trời.
* Bạn **gửi yêu cầu tới API thời tiết (như của Google hoặc OpenWeatherMap)**.
* API trả về kết quả như: {"nhietDo": 33, "trangThai": "nắng"}

**🧪 Ví dụ cực đơn giản về API (giống như nói chuyện):**

**Trình duyệt gửi:**

http

Sao chépChỉnh sửa

GET /api/nguoidung?id=1

**Server trả lại (dữ liệu JSON):**

json

Sao chépChỉnh sửa

{

"ten": "Nguyễn Văn A",

"tuoi": 25

}

**✅ Vậy API là gì? (tóm gọn):**

API là **cầu nối để các phần mềm giao tiếp với nhau**.  
Nó giống như **menu** mà bạn gọi món – bạn không cần biết hệ thống bên trong hoạt động thế nào, chỉ cần **gửi yêu cầu và nhận kết quả**.

**📌 Một số ví dụ về API trong thực tế:**

| **Bạn muốn làm gì?** | **API nào sẽ giúp bạn** |
| --- | --- |
| Lấy thời tiết | API thời tiết |
| Gửi tin nhắn | API của Zalo, Facebook |
| Đăng nhập Google | API của Google |
| Tìm địa điểm trên bản đồ | API Google Maps |
| Thanh toán online | API của ngân hàng, Momo, ZaloPay |

Nếu bạn vẫn chưa rõ, mình có thể cho bạn một **ví dụ thực tế hoàn chỉnh (trình duyệt gọi PHP API và trả JSON)** – bạn chỉ cần copy và thử.

Muốn mình làm cho bạn thử không?

**Bạn đã nói:**

vẫn chưa hiểu

**ChatGPT đã nói:**

Không sao cả! Mình sẽ giải thích lại **API** theo cách **đời thường nhất, đơn giản nhất**, **không cần biết lập trình cũng hiểu được**.

**🧠 API là gì? – Giải thích như người không biết kỹ thuật**

**📌 API = cách 2 bên nói chuyện với nhau**

**🔧 Ví dụ 1: Gọi món ở quán ăn**

* Bạn đi vào quán ăn.
* Bạn không vào bếp để nấu ăn, bạn chỉ **nói với nhân viên**:  
  👉 *"Làm cho tôi 1 tô phở"*

➡️ Nhân viên đi vào bếp, nấu xong rồi mang ra cho bạn.

🟩 **Trong ví dụ này:**

| **Thứ gì** | **Giống gì trong công nghệ** |
| --- | --- |
| Bạn | Người dùng hoặc ứng dụng |
| Nhân viên phục vụ | ✅ API |
| Nhà bếp | Máy chủ hoặc hệ thống chính |
| Tô phở được mang ra | Dữ liệu hoặc kết quả bạn nhận được |

👉 API là **người trung gian** giúp bạn giao tiếp với hệ thống mà bạn **không tự mình truy cập trực tiếp** được.

**🔧 Ví dụ 2: Gọi Grab hoặc Shopee**

* Bạn mở app Grab, chọn "Gọi xe".
* Ứng dụng sẽ gửi thông tin lên hệ thống Grab:  
  📤 *"Người A đang ở đây, cần xe 4 chỗ"*
* Hệ thống Grab **tìm tài xế**, và gửi thông tin lại:  
  📥 *"Tài xế B đang đến"*

👉 Cách Grab gửi – nhận thông tin qua lại chính là dùng **API**.

**✅ Vậy API là gì?**

**API là cách để phần mềm giao tiếp với nhau**, giống như **gửi yêu cầu và nhận kết quả**.

**🤖 Trong lập trình thì sao?**

Khi bạn viết **trang web**, bạn có thể:

* Gửi yêu cầu đến API để **lấy thông tin người dùng**, **thời tiết**, **tin tức**, v.v.
* API sẽ trả lại dữ liệu, thường là **dạng JSON** như:

json

Sao chépChỉnh sửa

{

"ten": "Nguyễn Văn A",

"tuoi": 25

}

**📌 Kết luận siêu dễ nhớ:**

💬 **API = người trung gian** giữa bạn và hệ thống.  
Bạn nói gì → API chuyển đi → API mang kết quả về cho bạn.

**📍 Muốn hiểu tận tay?**

Mình có thể gửi bạn một ví dụ nhỏ:

* HTML + JavaScript gọi tới một API đơn giản do PHP tạo ra.
* Bạn chạy nó lên và xem kết quả.

👉 Bạn có muốn thử không? Mình sẽ viết cho bạn chạy ngay!

**Bạn đã nói:**

Giải thích chi tiết, rõ ràng, tường tận về đoạn mã php kết nối với hệ quản trị cơ sở dữ liệu mysql

**ChatGPT đã nói:**

Tuyệt vời! Đây là kiến thức **cốt lõi trong lập trình web với PHP**, và mình sẽ **giải thích chi tiết – từng dòng một cách rõ ràng, tường tận**, để bạn **hiểu chứ không chỉ nhớ**.

**🎯 Mục tiêu:**

Kết nối PHP với hệ quản trị cơ sở dữ liệu **MySQL** (cụ thể là **MySQL/MariaDB**) để thực hiện các thao tác như: lấy dữ liệu, thêm, sửa, xóa...

**📄 Đoạn mã mẫu PHP kết nối MySQL (dùng MySQLi - cách phổ biến)**

php

Sao chépChỉnh sửa

<?php

$host = "localhost"; // Địa chỉ máy chủ CSDL (thường là localhost)

$username = "root"; // Tên đăng nhập MySQL

$password = ""; // Mật khẩu MySQL (nếu có)

$database = "mydatabase"; // Tên cơ sở dữ liệu muốn kết nối

// 1. Tạo kết nối

$conn = new mysqli($host, $username, $password, $database);

// 2. Kiểm tra kết nối

if ($conn->connect\_error) {

die("Kết nối thất bại: " . $conn->connect\_error);

} else {

echo "Kết nối thành công!";

}

?>

**🧠 Phân tích chi tiết từng dòng**

**🔹 1. Khai báo thông tin kết nối**

php

Sao chépChỉnh sửa

$host = "localhost";

$username = "root";

$password = "";

$database = "mydatabase";

* localhost: máy chủ cơ sở dữ liệu nằm **trên cùng máy với server PHP**
* root: tài khoản mặc định trong MySQL
* "": mật khẩu trống (nếu MySQL không yêu cầu password)
* mydatabase: là **tên cơ sở dữ liệu** bạn đã tạo trong MySQL

👉 Bạn phải chắc chắn cơ sở dữ liệu đó **đã tồn tại**.

**🔹 2. Tạo kết nối với MySQL**

php

Sao chépChỉnh sửa

$conn = new mysqli($host, $username, $password, $database);

* new mysqli(...) là cách dùng **hướng đối tượng** để tạo kết nối.
* $conn là **biến kết nối**, dùng để làm mọi thứ sau này (chạy truy vấn, thêm dữ liệu,...).

**🔹 3. Kiểm tra kết nối có thành công không**

php

Sao chépChỉnh sửa

if ($conn->connect\_error) {

die("Kết nối thất bại: " . $conn->connect\_error);

}

* $conn->connect\_error: kiểm tra xem có lỗi trong quá trình kết nối không.
* die(...): dừng chương trình và in ra lỗi nếu kết nối thất bại.
* Nếu không lỗi, dòng echo "Kết nối thành công!" sẽ chạy.

**📝 Kết quả**

* Nếu thành công: in ra “Kết nối thành công!”
* Nếu sai mật khẩu, sai tên DB, hoặc máy chủ không chạy MySQL: in ra lỗi cụ thể.

**🧪 Ví dụ chạy thực tế**

**✅ Trước khi chạy:**

1. Bạn cần **cài đặt XAMPP hoặc Laragon** (gồm PHP + MySQL)
2. Tạo một CSDL tên mydatabase trong phpMyAdmin
3. Tạo file PHP, dán đoạn mã vào, rồi chạy qua trình duyệt

**🔐 Gợi ý nâng cao: dùng PDO (an toàn hơn)**

php

Sao chépChỉnh sửa

try {

$pdo = new PDO("mysql:host=localhost;dbname=mydatabase", "root", "");

echo "Kết nối thành công!";

} catch (PDOException $e) {

echo "Lỗi kết nối: " . $e->getMessage();

}

✅ PDO giúp bạn viết code bảo mật hơn, hỗ trợ nhiều hệ quản trị khác như PostgreSQL, SQLite,...

**📌 Kết luận:**

| **Thành phần** | **Vai trò** |
| --- | --- |
| $host, $username, $password, $database | Thông tin để kết nối |
| new mysqli(...) | Tạo kết nối |
| $conn->connect\_error | Kiểm tra lỗi kết nối |
| die(...) | Dừng nếu lỗi |
| PDO | Cách kết nối hiện đại và an toàn hơn |

Nếu bạn muốn mình viết tiếp phần **chạy truy vấn (SELECT, INSERT...)** sau khi kết nối thành công, mình có thể hướng dẫn luôn nhé! Chỉ cần nói 👍.

**Bạn đã nói:**

mysqli là gì?

**ChatGPT đã nói:**

Tuyệt vời! Câu hỏi này rất quan trọng nếu bạn đang học lập trình web với **PHP và MySQL**.

**✅ mysqli là gì?**

**👉 mysqli là viết tắt của:**

**MySQL Improved** (MySQL cải tiến)

**🧠 Nói đơn giản:**

mysqli là **một thư viện (extension)** có sẵn trong PHP, dùng để:

* Kết nối đến cơ sở dữ liệu MySQL
* Gửi truy vấn SQL (SELECT, INSERT, DELETE, UPDATE, ...)
* Nhận kết quả trả về từ MySQL
* Quản lý lỗi, bảo mật, và giao dịch

**📌 Vì sao có chữ “Improved”?**

Trước đây PHP có thư viện cũ tên là mysql, nhưng:

* Không an toàn
* Không hỗ trợ tính năng mới của MySQL
* Đã **bị loại bỏ từ PHP 7**

➡️ Vì vậy, mysqli ra đời để **thay thế và cải tiến hoàn toàn**.

**✅ mysqli có những điểm mạnh gì?**

| **Tính năng** | **Có trong mysqli** |
| --- | --- |
| Hỗ trợ **kết nối hướng đối tượng và thủ tục** | ✅ Có |
| Hỗ trợ **prepared statements** (bảo mật khỏi SQL injection) | ✅ Có |
| Hỗ trợ **giao dịch (transactions)** | ✅ Có |
| Hỗ trợ **multiple queries** (nhiều truy vấn 1 lúc) | ✅ Có |

**🛠 Cách dùng mysqli – 2 kiểu**

**🔹 1. Hướng thủ tục (procedural style)**

php

Sao chépChỉnh sửa

$conn = mysqli\_connect("localhost", "root", "", "mydatabase");

if (!$conn) {

die("Kết nối thất bại: " . mysqli\_connect\_error());

}

**🔹 2. Hướng đối tượng (object-oriented style) – được khuyên dùng**

php

Sao chépChỉnh sửa

$conn = new mysqli("localhost", "root", "", "mydatabase");

if ($conn->connect\_error) {

die("Lỗi kết nối: " . $conn->connect\_error);

}

👉 Cả hai cách **đều dùng mysqli**, chỉ khác nhau về cách viết.

**✅ mysqli dùng để làm gì?**

Sau khi kết nối, bạn có thể:

* 📥 Truy vấn lấy dữ liệu:

php

Sao chépChỉnh sửa

$result = $conn->query("SELECT \* FROM users");

* 🧾 Lặp qua kết quả:

php

Sao chépChỉnh sửa

while ($row = $result->fetch\_assoc()) {

echo $row["username"];

}

* 📤 Thêm dữ liệu:

php

Sao chépChỉnh sửa

$conn->query("INSERT INTO users (username) VALUES ('abc')");

**🔐 Bảo mật hơn với Prepared Statement trong mysqli**

php

Sao chépChỉnh sửa

$stmt = $conn->prepare("SELECT \* FROM users WHERE email = ?");

$stmt->bind\_param("s", $email); // "s" = string

$stmt->execute();

$result = $stmt->get\_result();

**🧩 Tổng kết ngắn gọn:**

| **mysqli là gì?** | **Thư viện PHP dùng để kết nối và làm việc với MySQL** |
| --- | --- |
| Thay thế gì? | mysql (cũ, lỗi thời, không còn dùng được từ PHP 7) |
| Ưu điểm | Bảo mật hơn, nhanh hơn, nhiều tính năng mới |
| Có nên dùng? | ✅ Có, nếu bạn chưa chuyển sang PDO |

Bạn có muốn mình làm **ví dụ đầy đủ dùng mysqli để SELECT dữ liệu từ bảng** và hiển thị ra HTML không? Chỉ cần nói **"có"** là mình viết ngay!

**Bạn đã nói:**

PDO là gì?

**ChatGPT đã nói:**

Rất hay! Bạn đang tìm hiểu đúng hướng rồi. Bây giờ mình sẽ giải thích **PDO** một cách **dễ hiểu, rõ ràng và đầy đủ** – để bạn hiểu thật chắc.

**✅ PDO là gì?**

**👉 PDO viết tắt là:**

**P**HP **D**ata **O**bjects

**🧠 Nói đơn giản:**

**PDO là một thư viện (extension) trong PHP** giúp bạn:

* Kết nối đến cơ sở dữ liệu
* Gửi truy vấn SQL (SELECT, INSERT, UPDATE, DELETE...)
* Nhận dữ liệu trả về
* Và làm tất cả điều đó một cách **bảo mật, hiện đại, dễ chuyển đổi** giữa các loại cơ sở dữ liệu khác nhau

**⚠️ Khác với mysqli như thế nào?**

| **Tiêu chí** | **mysqli** | **PDO** |
| --- | --- | --- |
| Hỗ trợ MySQL | ✅ Có | ✅ Có |
| Hỗ trợ nhiều loại CSDL khác | ❌ Không (chỉ MySQL) | ✅ Có (MySQL, PostgreSQL, SQLite...) |
| Giao diện lập trình | Hướng thủ tục + OOP | Chỉ hỗ trợ hướng đối tượng |
| Bảo mật qua Prepared Statement | ✅ Có | ✅ Có (rất dễ dùng) |
| Chuyển đổi CSDL dễ dàng | ❌ Phải viết lại nhiều | ✅ Chỉ đổi chuỗi kết nối |
| Code gọn và hiện đại | Trung bình | ✅ Gọn gàng, dễ bảo trì |

**📦 Cách dùng PDO – Đầy đủ & Dễ hiểu**

**🔹 1. Kết nối đến MySQL bằng PDO**

php

Sao chépChỉnh sửa

<?php

try {

$pdo = new PDO("mysql:host=localhost;dbname=mydatabase", "root", "");

// Thiết lập chế độ lỗi để dễ debug

$pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

echo "Kết nối thành công!";

} catch (PDOException $e) {

echo "Lỗi kết nối: " . $e->getMessage();

}

?>

**📌 Giải thích:**

* new PDO(...): Tạo đối tượng kết nối PDO
* "mysql:host=...;dbname=...": Đây là DSN (Data Source Name)
* setAttribute(...): Giúp bạn hiển thị lỗi rõ ràng hơn khi có vấn đề

**🔹 2. Truy vấn dữ liệu (SELECT)**

php

Sao chépChỉnh sửa

$sql = "SELECT \* FROM users";

$stmt = $pdo->query($sql);

while ($row = $stmt->fetch(PDO::FETCH\_ASSOC)) {

echo $row['username'] . "<br>";

}

**🔹 3. Truy vấn có tham số (Prepared Statement – an toàn)**

php

Sao chépChỉnh sửa

$sql = "SELECT \* FROM users WHERE email = :email";

$stmt = $pdo->prepare($sql);

$stmt->execute(['email' => 'abc@example.com']);

$user = $stmt->fetch(PDO::FETCH\_ASSOC);

echo $user['username'];

✔️ Cách này giúp bạn **chống lại SQL injection** – rất quan trọng về bảo mật.

**📌 Kết luận ngắn gọn:**

| **Khái niệm** | **Ý nghĩa** |
| --- | --- |
| **PDO là gì?** | Thư viện PHP giúp kết nối và làm việc với nhiều loại cơ sở dữ liệu |
| **Ưu điểm** | Bảo mật, gọn, hiện đại, dễ chuyển sang PostgreSQL, SQLite... |
| **Có nên dùng?** | ✅ Rất nên dùng (thay cho mysqli nếu bắt đầu dự án mới) |

Nếu bạn muốn mình **viết một ví dụ hoàn chỉnh**: kết nối bằng PDO, lấy dữ liệu từ bảng và hiển thị ra giao diện HTML – mình có thể viết ngay cho bạn chạy thử.

👉 Bạn muốn không?

**Bạn đã nói:**

Bây giờ tôi muốn bật xampp để chạy php, nhưng lại muốn mở riêng MYSQL đã tải sẵn về trong máy thì có được không? tôi sợ bị xung đột với mysql trên xampp

**ChatGPT đã nói:**

Đây là vấn đề rất thực tế khi bạn cài đặt **MySQL riêng** rồi lại muốn dùng XAMPP để chạy PHP mà không muốn xung đột giữa 2 MySQL.

**Câu trả lời ngắn gọn:**

**Bạn hoàn toàn có thể dùng XAMPP chỉ để chạy Apache + PHP, còn MySQL thì dùng MySQL đã cài riêng trên máy.** Nhưng cần làm đúng cách để tránh xung đột.

**Chi tiết và cách thực hiện:**

**1. Tắt MySQL của XAMPP (để không bị xung đột cổng)**

* Mặc định, MySQL trong XAMPP chạy trên **cổng 3306**.
* MySQL bạn cài sẵn trên máy cũng thường chạy trên cổng 3306.
* Nếu 2 dịch vụ chạy đồng thời trên cùng cổng thì sẽ bị xung đột.

**Bạn có thể:**

* Tắt MySQL trong XAMPP (mở XAMPP Control Panel, nhấn **Stop** ở dòng MySQL)
* Đảm bảo MySQL riêng của bạn đang chạy trên cổng 3306 (hoặc một cổng khác)

**2. Cấu hình PHP của XAMPP kết nối đến MySQL riêng của bạn**

* PHP trong XAMPP khi kết nối MySQL, bạn dùng thông tin kết nối:

php

Sao chépChỉnh sửa

$host = "127.0.0.1"; // hoặc "localhost"

$port = 3306; // Cổng MySQL của MySQL riêng bạn đang chạy

$username = "user"; // tài khoản MySQL riêng bạn

$password = "pass";

$database = "db\_name";

* Khi tạo kết nối, nếu dùng mysqli bạn có thể thêm cổng như:

php

Sao chépChỉnh sửa

$conn = new mysqli($host, $username, $password, $database, $port);

* Hoặc với PDO:

php

Sao chépChỉnh sửa

$pdo = new PDO("mysql:host=127.0.0.1;port=3306;dbname=db\_name", "user", "pass");

**3. Lưu ý quan trọng**

* Nếu MySQL riêng của bạn đang chạy trên **cổng khác 3306** (ví dụ 3307), hãy chắc chắn dùng đúng cổng trong kết nối.
* Bạn có thể kiểm tra cổng MySQL bằng cách mở file cấu hình my.ini (Windows) hoặc my.cnf (Linux) của MySQL riêng.
* Đảm bảo firewall không chặn cổng đó.

**4. Lợi ích khi dùng MySQL riêng**

* Bạn có thể dùng phiên bản MySQL mới hơn hoặc khác với MySQL của XAMPP.
* Quản lý MySQL độc lập, không phụ thuộc XAMPP.
* Tiện cho phát triển nâng cao.

**Tổng kết:**

| **Việc làm** | **Có được không?** | **Cần chú ý** |
| --- | --- | --- |
| Chạy PHP bằng XAMPP + dùng MySQL riêng | ✅ Có thể | Tắt MySQL XAMPP hoặc đổi cổng, cấu hình đúng cổng khi kết nối |
| Chạy đồng thời 2 MySQL trên cùng cổng 3306 | ❌ Không được | Phải đổi cổng cho một trong 2 dịch vụ |

Nếu bạn cần mình hướng dẫn chi tiết cách tắt MySQL trong XAMPP, hoặc cấu hình cổng MySQL riêng, hoặc cách viết đoạn PHP kết nối MySQL riêng đó, cứ nói nhé! Mình sẵn sàng giúp bạn.

**Bạn đã nói:**

.htaccess là gì?

**ChatGPT đã nói:**

Tuyệt vời! Đây là một câu hỏi rất quan trọng nếu bạn đang làm web với PHP và dùng **Apache (như trong XAMPP, Laragon, hoặc hosting cPanel)**.

**✅ .htaccess là gì?**

.htaccess là viết tắt của:

**HyperText Access**

**🧠 Nói đơn giản:**

.htaccess là **file cấu hình nhỏ của Apache**, cho phép bạn **thay đổi cách hoạt động của server** trong **một thư mục nhất định**, **mà không cần truy cập file cấu hình chính của Apache**.

**🔧 File .htaccess dùng để làm gì?**

Bạn có thể dùng .htaccess để:

| **Tác vụ** | **Ví dụ** |
| --- | --- |
| ✅ Viết lại URL (rewrite) | /tin-tuc/1 → news.php?id=1 |
| ✅ Chặn truy cập IP | Chặn IP xấu, spam |
| ✅ Đặt trang mặc định | Ví dụ: vào / thì tự mở home.php |
| ✅ Chống truy cập thư mục | Chặn người khác xem danh sách file |
| ✅ Bảo vệ thư mục bằng mật khẩu | Yêu cầu đăng nhập khi truy cập thư mục |
| ✅ Chuyển hướng 301, 302 | Từ domain cũ sang mới |
| ✅ Bật Gzip nén file | Tăng tốc tải trang |
| ✅ Cấu hình CORS, MIME, cache | Tối ưu hiệu năng và bảo mật |

**📄 Ví dụ cụ thể**

**🔹 1. Viết lại URL (Pretty URL)**

apache

Sao chépChỉnh sửa

RewriteEngine On

RewriteRule ^tin-tuc/([0-9]+)$ news.php?id=$1

➡️ Khi bạn truy cập example.com/tin-tuc/5, Apache sẽ **chuyển hướng ngầm sang news.php?id=5** mà người dùng không thấy.

**🔹 2. Chuyển hướng vĩnh viễn (301 redirect)**

apache

Sao chépChỉnh sửa

Redirect 301 /cu-trang.html https://example.com/trang-moi

**🔹 3. Chặn IP**

apache

Sao chépChỉnh sửa

Order Deny,Allow

Deny from 192.168.1.100

**🔹 4. Bật Gzip (nén file HTML, CSS, JS để tải nhanh hơn)**

apache

Sao chépChỉnh sửa

<IfModule mod\_deflate.c>

AddOutputFilterByType DEFLATE text/html text/css application/javascript

</IfModule>

**📝 Vị trí và cách dùng .htaccess**

* Đặt file .htaccess vào **thư mục gốc của website** (nơi chứa index.php)
* **Tên file bắt buộc là .htaccess**, không có tên mở rộng
* File này hoạt động nếu **Apache bật module mod\_rewrite** (mặc định XAMPP/Laragon có bật)

**⚠️ Lưu ý:**

| **Vấn đề** | **Lưu ý** |
| --- | --- |
| Không thấy .htaccess | Do bị ẩn – cần bật "hiện file ẩn" trong File Explorer hoặc IDE |
| Sửa sai .htaccess | Có thể làm trang web **bị lỗi 500 Internal Server Error** |
| Dùng với Nginx | ❌ .htaccess chỉ dùng được với **Apache**, không có tác dụng với Nginx |

**✅ Tổng kết**

| **Mục** | **Nội dung** |
| --- | --- |
| .htaccess là gì? | File cấu hình của Apache cho từng thư mục |
| Dùng để làm gì? | Viết lại URL, chặn IP, redirect, nén file, bảo mật... |
| Cần Apache không? | ✅ Có, chỉ dùng được với Apache |
| Có nên học? | ✅ Rất nên, nếu bạn làm web PHP hoặc dùng hosting |

Bạn có muốn mình **viết mẫu file .htaccess để hỗ trợ rewrite URL đẹp** cho project PHP của bạn không? Mình có thể làm ngay!

**Bạn đã nói:**

vẫn chưa hiểu

**ChatGPT đã nói:**

Không sao cả! Mình sẽ giải thích lại **.htaccess** một cách **siêu đơn giản – ví dụ đời thường – dễ hiểu nhất có thể**, giống như bạn chưa biết gì về server cũng hiểu được.

**💡 .htaccess là gì? – Hình dung như bảo vệ chung cư**

Hãy tưởng tượng bạn sống trong **chung cư**:

* Mỗi căn hộ (website) có thể có **nội quy riêng**
* Nhưng chung cư có **bảo vệ** gác ở cửa, kiểm tra mọi người ra vào
* Bạn dán bảng ghi chú trước cửa nhà bạn:

👉 "Không cho người lạ vào"  
👉 "Nếu ai gõ cửa /tin-tuc/1, hãy dẫn họ tới news.php?id=1"

🟩 Bảng ghi chú đó chính là **file .htaccess**

**🧠 Hiểu đơn giản hơn:**

.htaccess là **một tờ giấy hướng dẫn** đặt trong **thư mục web**, để **chỉ đạo cách Apache server xử lý các yêu cầu truy cập.**

Nó giống như nói với server:

"Ê Apache, ai vào thư mục này, thì xử lý theo các quy tắc sau nhé!"

**📦 Bạn dùng .htaccess để làm gì?**

| **Mục đích** | **Giải thích dễ hiểu** |
| --- | --- |
| ✅ Đổi đường dẫn xấu → đẹp | /product.php?id=1 → /san-pham/1 |
| ✅ Chuyển hướng người dùng | Truy cập trang cũ → chuyển sang trang mới |
| ✅ Chặn truy cập | Chặn IP xấu, cấm ai đó vào |
| ✅ Bảo vệ thư mục | Đặt mật khẩu khi vào 1 folder |
| ✅ Tăng tốc tải trang | Bật nén file (gzip), cache lâu hơn |

**🧪 Ví dụ đơn giản nhất**

**🎯 Mục tiêu:**

Truy cập example.com/tin-tuc/5  
→ Server tự hiểu là news.php?id=5

**📝 Viết vào file .htaccess:**

apache

Sao chépChỉnh sửa

RewriteEngine On

RewriteRule ^tin-tuc/([0-9]+)$ news.php?id=$1

👉 Nghĩa là:

* Bật chế độ rewrite (RewriteEngine On)
* Nếu URL có dạng tin-tuc/1 → thì nội dung sẽ lấy từ news.php?id=1

**✅ Khi nào bạn cần .htaccess?**

* Bạn làm PHP và muốn URL đẹp như WordPress
* Bạn muốn chặn truy cập trái phép vào thư mục
* Bạn đang dùng **hosting (cPanel)** hoặc **XAMPP / Apache server**

**🔧 Làm sao để thử .htaccess?**

1. Mở thư mục dự án PHP của bạn
2. Tạo file tên là .htaccess (nhớ có dấu chấm đầu!)
3. Dán quy tắc vào
4. Đảm bảo Apache đã bật mod\_rewrite
5. Chạy thử trên trình duyệt

**📌 Tóm gọn siêu dễ nhớ:**

🔹 .htaccess = “sổ tay hướng dẫn” dành cho **server Apache**  
🔹 Giúp server biết: **ai truy cập thì phải xử lý thế nào**

Nếu bạn muốn mình **tạo một ví dụ hoàn chỉnh** gồm:

* File .htaccess
* File PHP (news.php)
* Và chỉ bạn cách chạy thử bằng XAMPP

👉 Chỉ cần nói: **"cho mình ví dụ hoàn chỉnh"**, mình sẽ viết từng bước cho bạn làm theo luôn nhé!