KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO MÔN HỌC**

**PHÁT TRIỂN ỨNG DỤNG WEB VỚI MÃ NGUỒN MỞ**

**HỆ THỐNG QUẢN LÝ SINH VIÊN**

Giảng viên giảng dạy : **Nguyễn Ngọc Đan Thanh**

Sinh viên thực hiện:

Lớp: **DA21TTC**

**Nguyễn Hoàng Thương 110121224**

**Lê Thị Ngọc Hân 110121249**

**Ngô Thanh Quyền 110121223**

**Nguyễn Khánh Băng 110121248**

**Trà Vinh, tháng 10 năm 2024**

**MỤC LỤC**

Nội dung Trang

[MỤC LỤC 2](#_Toc180944310)

[CHƯƠNG 1: TẠO VÀ KẾT NỐI CSDL 3](#_Toc180944311)

[1.1 Tạo CSDL 3](#_Toc180944312)

[1.2 Kết nối cơ sở dữ liệu 4](#_Toc180944313)

[CHƯƠNG 2: ĐĂNG KÝ, ĐĂNG NHẬP VÀ PHÂN QUYỀN 5](#_Toc180944314)

[2.1 Đăng ký 5](#_Toc180944315)

[2.2 Đăng nhập 6](#_Toc180944316)

[2.3 Phân quyền 6](#_Toc180944317)

[CHƯƠNG 3: ĐĂNG NHẬP BẰNG TÀI KHOẢN GOOGLE 12](#_Toc180944318)

[CHƯƠNG 4: PHÂN TRANG 17](#_Toc180944319)

[4.1 Phân trang là gì? 17](#_Toc180944320)

[4.2 Các bước phân trang trong php 17](#_Toc180944321)

[CHƯƠNG 5: CÔNG NGHỆ AJAX TRONG PHP 19](#_Toc180944322)

[5.1 Khái niệm AJAX trong PHP 19](#_Toc180944323)

[5.2 Cách hoạt động 19](#_Toc180944324)

[5.3 Ứng dụng của AJAX 20](#_Toc180944325)

[5.3.1 Ứng dụng của AJAX trong tìm kiếm trực tiếp (AJAX Livesearch) 20](#_Toc180944326)

[5.3.2 Ứng dụng của AJAX trong lọc dữ liệu 22](#_Toc180944327)

[CHƯƠNG 6: REWIRITE 26](#_Toc180944328)

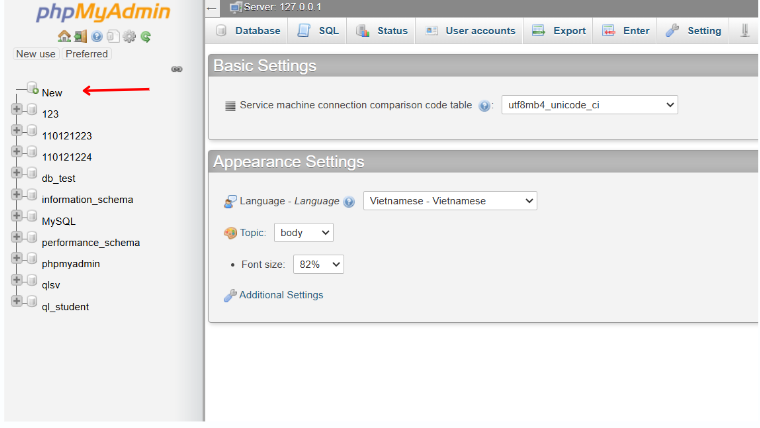
[CHƯƠNG 7: ỨNG DỤNG JAVASCRIPT GỌI API HIỂN THỊ ĐỊA LÝ 28](#_Toc180944329)

**CHƯƠNG 1: TẠO VÀ KẾT NỐI CSDL**

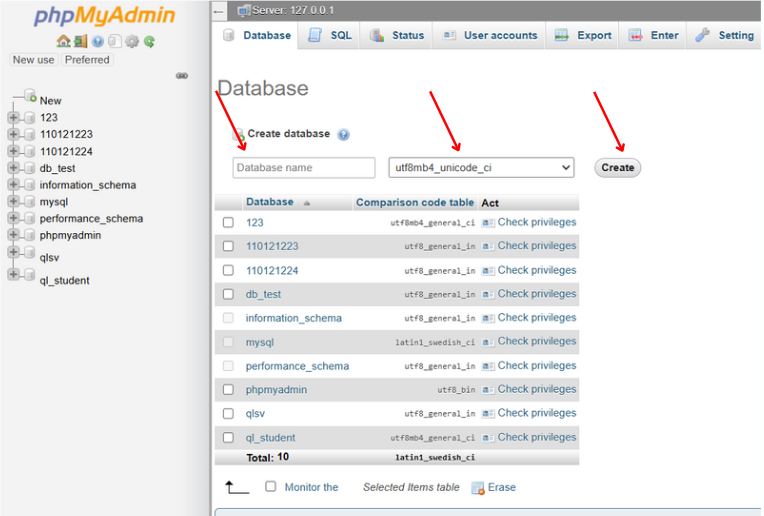
**1.1 Tạo CSDL**

1. **Tạo bằng thủ công**

* Chạy xampp vào trang phpMyAdmin, sau đó chọn new để tạo một csdl mới



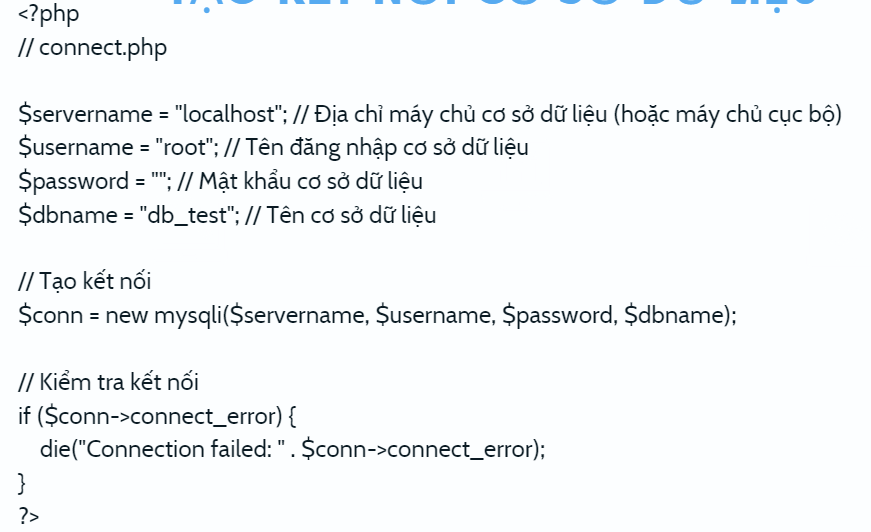
* Nhập tên csdl muốn tạo vào Database name và chọn ngôn ngữ sau đó bấm Create



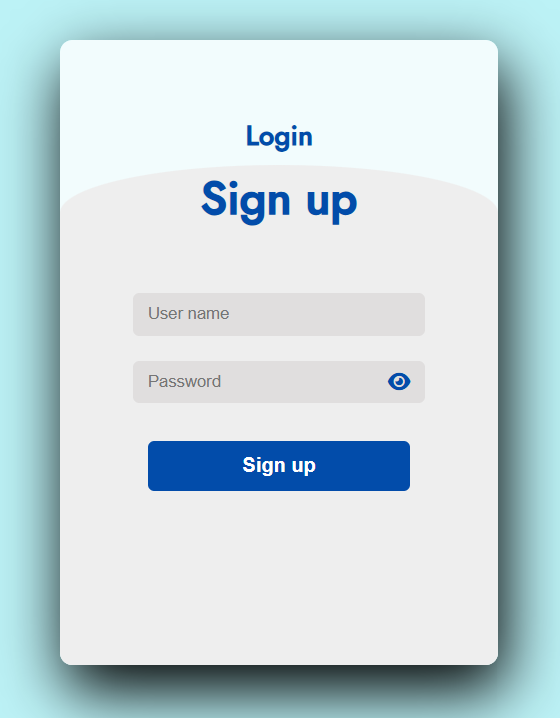
1. **Tạo bằng lệnh**



**1.2 Kết nối cơ sở dữ liệu**

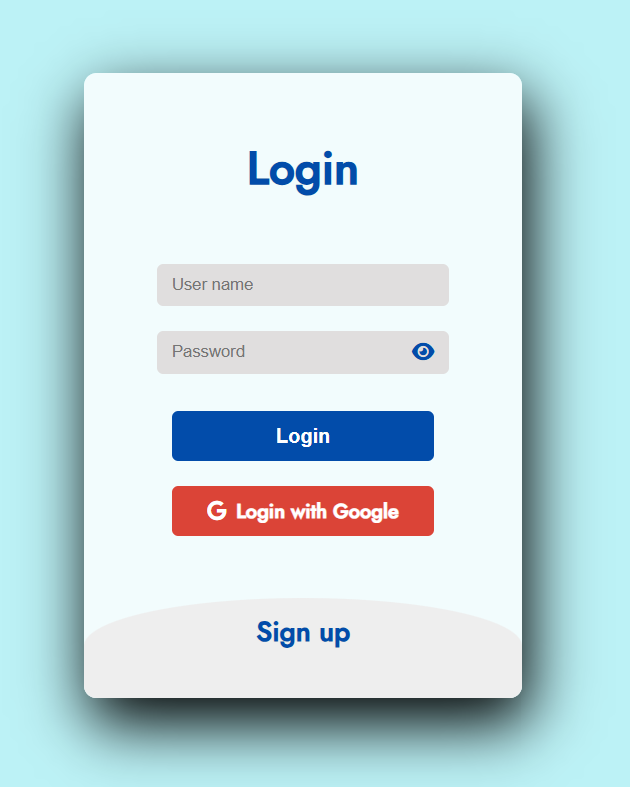
**** Kết nối csdl bằng các lệnh sau:

**CHƯƠNG 2: ĐĂNG KÝ, ĐĂNG NHẬP VÀ PHÂN QUYỀN**

**2.1 Đăng ký**

Hình 2. Đăng ký

Người dùng sẽ nhập tên tài khoản vào trường ***username*** và mật khẩu vào trường ***password*** trong form đăng ký. Sau khi nhập xong, hệ thống sẽ lưu trữ thông tin tên tài khoản và mật khẩu vào cơ sở dữ liệu để sử dụng cho quá trình đăng nhập sau này.

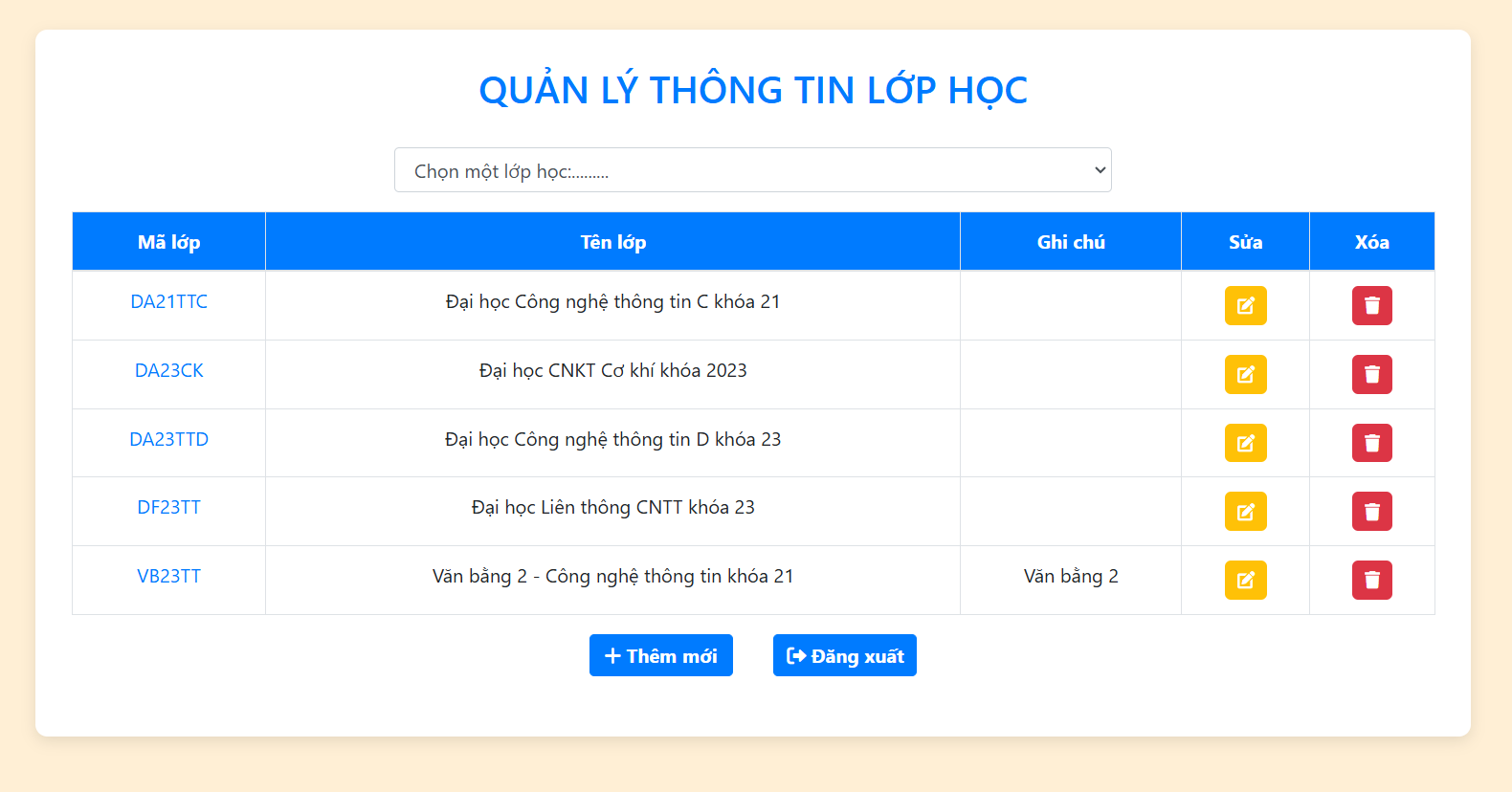
**2.2 Đăng nhập**

Hình 2. Đăng nhập

Khi đến với form đăng nhập, người dùng sẽ nhập tên tài khoản và mật khẩu đã sử dụng trong quá trình đăng ký. Hệ thống sẽ kiểm tra thông tin này với dữ liệu đã lưu trong cơ sở dữ liệu. Nếu tên tài khoản và mật khẩu khớp với thông tin đã đăng ký, người dùng sẽ đăng nhập thành công. Ngược lại, nếu thông tin không khớp, quá trình đăng nhập sẽ thất bại.

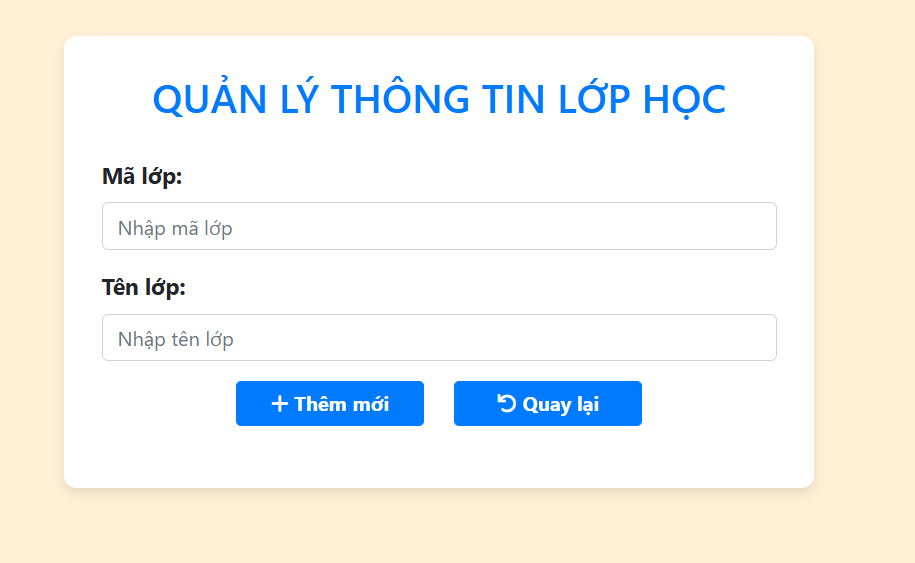
**2.3 Phân quyền**

Sau khi đăng nhập thành công, hệ thống sẽ kiểm tra vai trò (role) của người dùng. Nếu người dùng có vai trò là admin, họ sẽ được chuyển đến giao diện quản trị với các chức năng đầy đủ như sau:



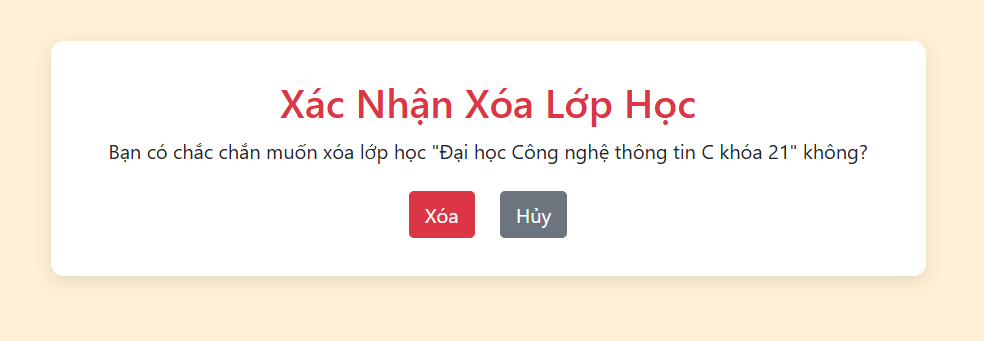
Hình 2. 3 Quyền admin

* Đối với quyền admin, họ sẽ có quyền truy cập vào các chức năng quản lý sau:
* Quản lý thông tin lớp học:
* Thêm lớp học: Cho phép admin tạo mới các lớp học với thông tin chi tiết như tên lớp, mã lớp.

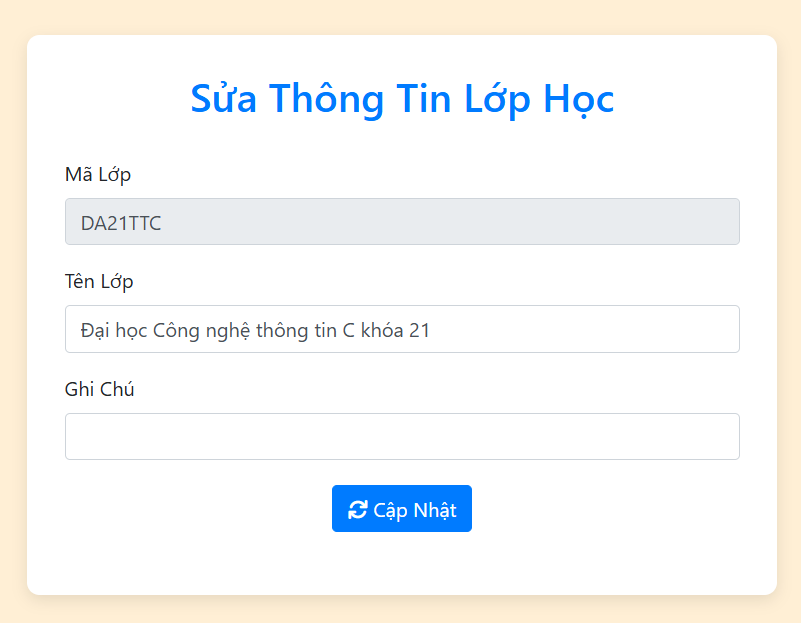


Hình 2. Thêm lớp học mới

* Xóa lớp học: Cho phép admin xóa các lớp học không còn hoạt động hoặc không cần thiết. Khi xóa lớp sẽ xóa hết tất cả sinh viên thuộc lớp đó

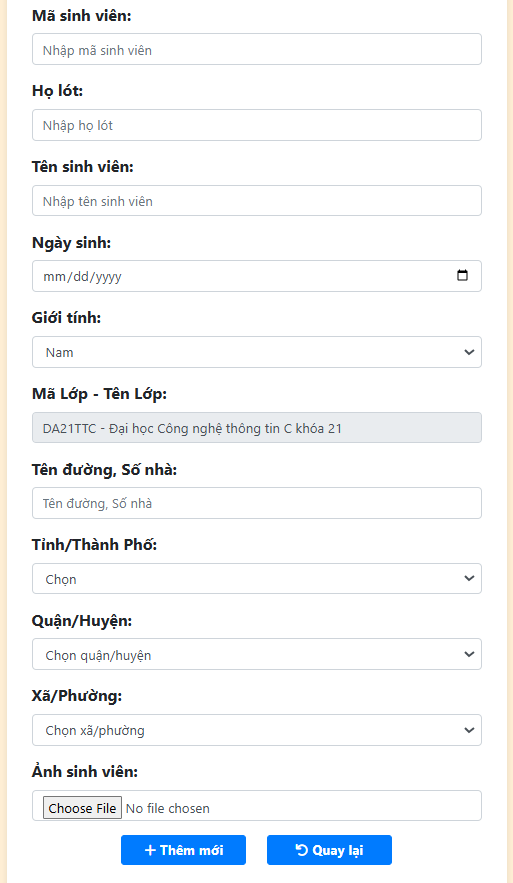


Hình 2. Xóa lớp học

* Sửa lớp học: Cho phép admin chỉnh sửa thông tin của các lớp học hiện có.

Hình 2. Sửa thông tin lớp học

* Quản lý thông tin sinh viên:
* Thêm sinh viên: Cho phép admin thêm mới thông tin sinh viên vào hệ thống, bao gồm tên, mã sinh viên, lớp học, và các thông tin khác. Khi thêm phải đảm bảo các điều kiện sau: mã sinh viên phải là số, ngày sinh phải đảm bảo sinh viên đủ 18 tuổi trở lên, ảnh sinh viên phải là file ảnh với định dạng JPG, JPEG, PNG & GIF, và không được vượt quá 500kb



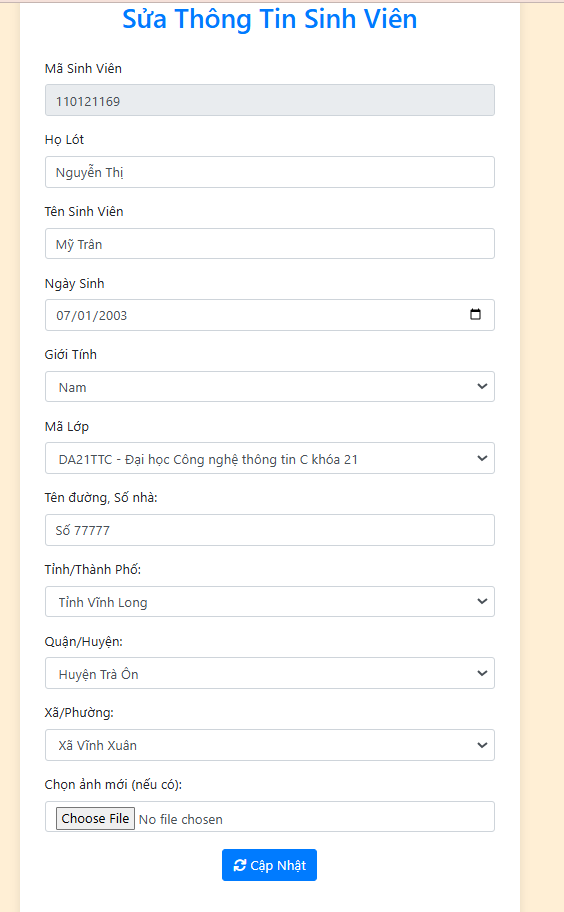
Hình 2. Thêm sinh viên

* Xóa sinh viên: Cho phép admin xóa thông tin sinh viên khi cần thiết, chẳng hạn như khi sinh viên tốt nghiệp hoặc rời khỏi trường.



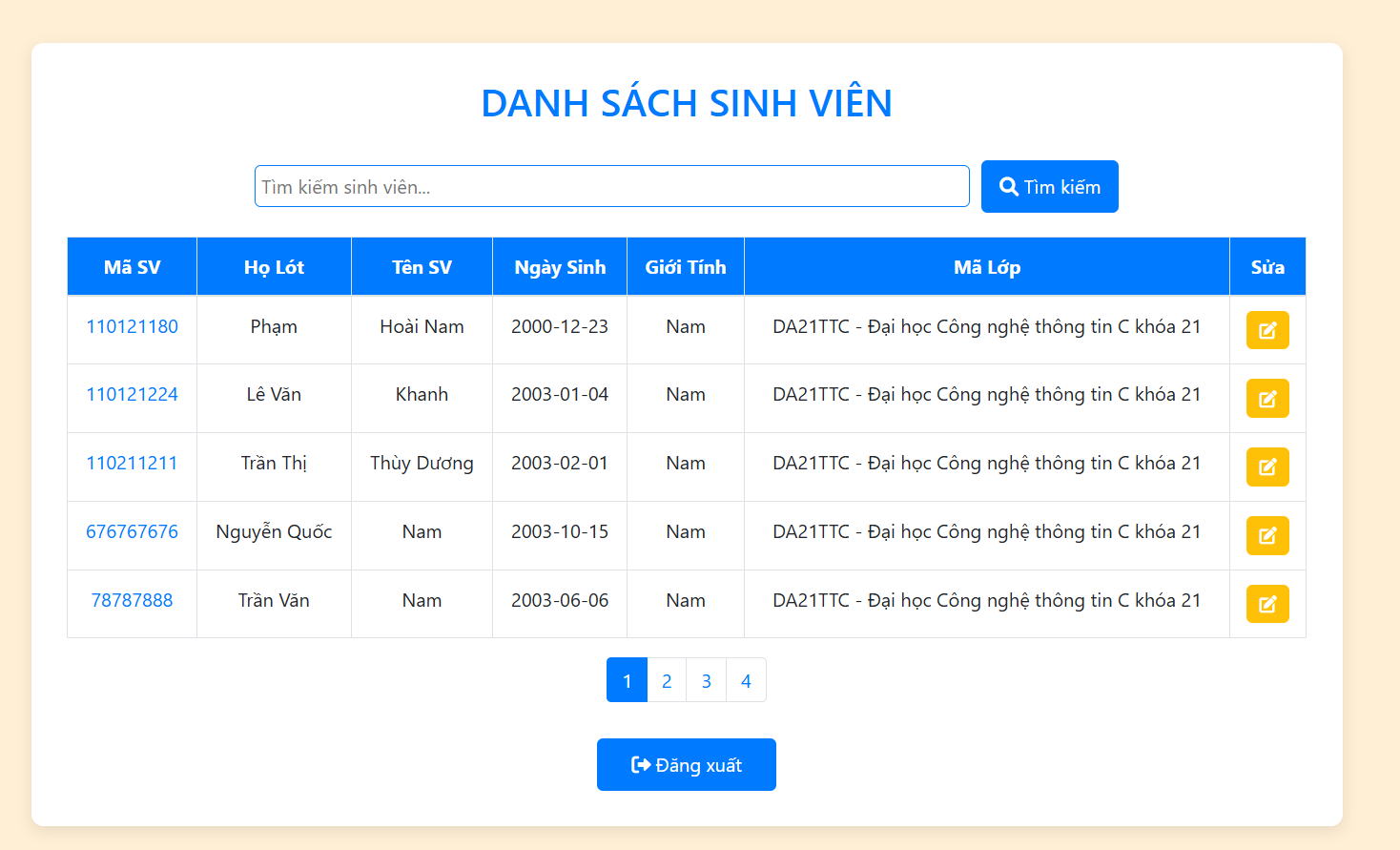
Hình 2. Xóa sinh viên

* Sửa sinh viên: Cho phép admin chỉnh sửa thông tin sinh viên hiện có.

****

Hình 2. Sửa thông tin sinh viên

* Đối với vai trò là user, giao diện sẽ được thiết kế đơn giản và tập trung vào các chức năng cơ bản là xem và sửa thông tin sinh viên.



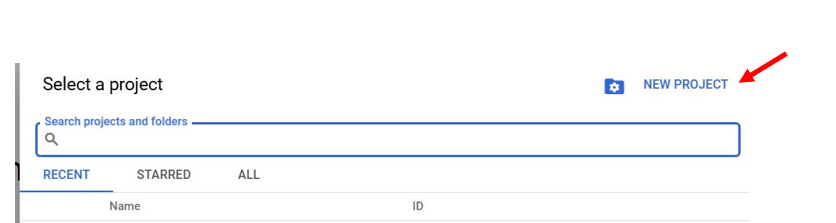
Hình 2. 10 Quyền user

****

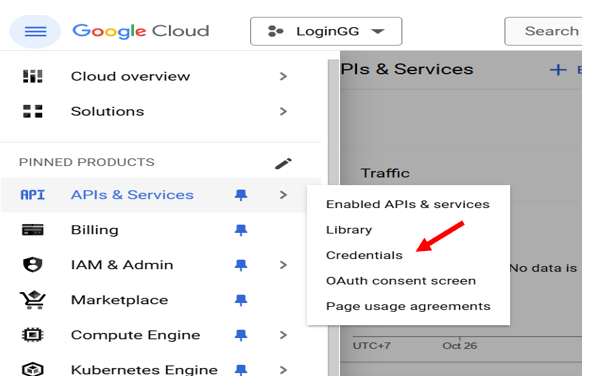
Hình 2. Xem thông tin sinh viên

**CHƯƠNG 3: ĐĂNG NHẬP BẰNG TÀI KHOẢN GOOGLE**

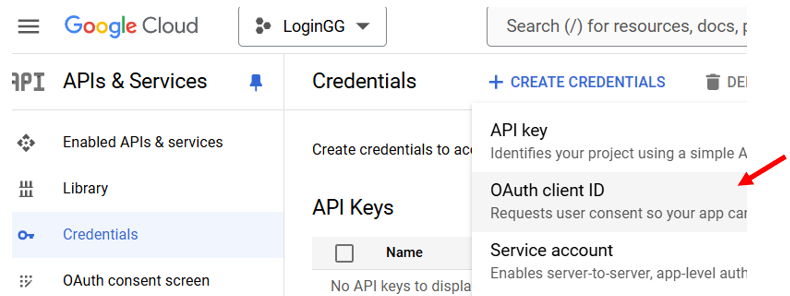
* **Bước 1: Tạo Dự Án Trên Google Developer Console**

1. Truy cập Google Developer Console:
2. Tạo một dự án mới:

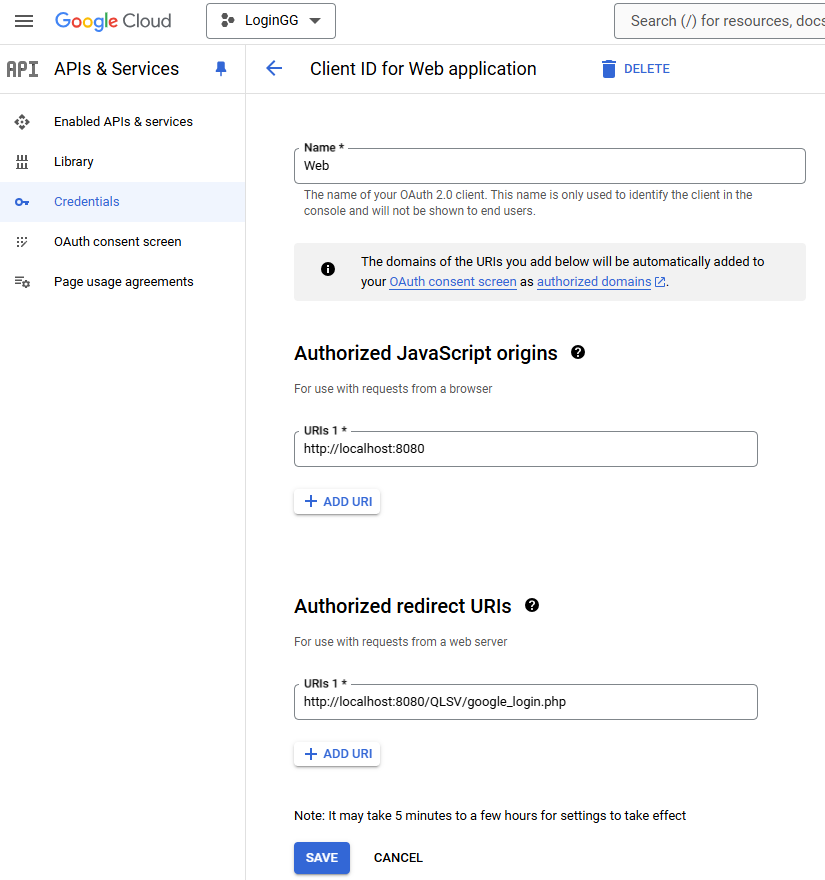
Hình 3. Create new project trên Google Developer Console.

* **Bước 2: Bật API và Tạo Thông Tin Xác Thực**
* Trong bảng điều khiển dự án, sẽ thấy một menu bên trái. Từ đây, đi đến phần “Library” (Thư viện). Tại đây, cần tìm kiếm và bật Google People API hoặc Google+ API. Việc bật API này là cần thiết để ứng dụng của bạn có thể truy cập vào dữ liệu người dùng.
* Tạo thông tin xác thực: Sau khi bật API, quay lại menu bên trái và chọn “Credentials” (Thông tin xác thực).

Hình 3. Create Credentials

* Nhấn vào “Create Credentials” (Tạo thông tin xác thực) và chọn “OAuth Client IDs”. Đây là bước quan trọng để tạo thông tin xác thực cho ứng dụng của bạn.

Hình 3. OAuth Client IDs

* Chọn “Web application” (Ứng dụng web) và điền các thông tin cần thiết như tên ứng dụng và URL chuyển hướng.

Hình 3. Client ID for Web application

* **Bước 3: Cài đặt Google Client Library**

Cài đặt Google Client Library là bước cần thiết để tích hợp các dịch vụ của Google vào ứng dụng .

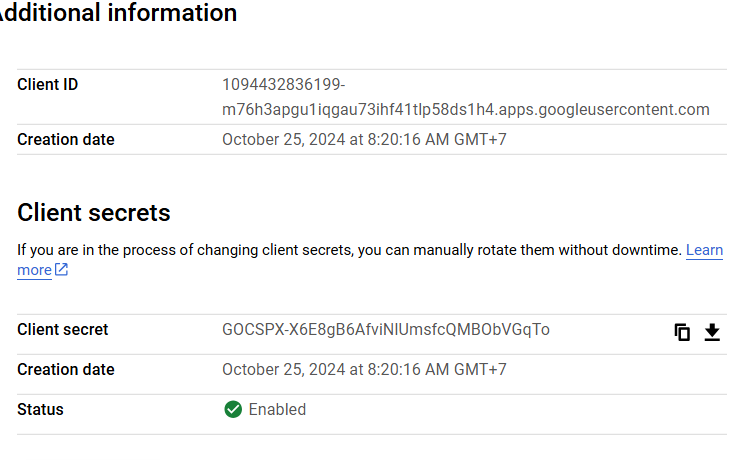
***composer require google/apiclient***

* **Bước 4: Cấu hình Google Client trong PHP**

|  |
| --- |
| <?php  require\_once 'vendor/autoload.php';  $client = new Google\_Client();  $client->setClientId('YOUR\_CLIENT\_ID');  $client->setClientSecret('YOUR\_CLIENT\_SECRET');  $client->setRedirectUri('YOUR\_REDIRECT\_URI');  $client->addScope("email");  $client->addScope("profile");  $googleLoginUrl = $client->createAuthUrl();  ?> |

Tệp ***autoload.php*** trong thư mục vendor của Composer, cho phép sử dụng các thư viện bên ngoài, cụ thể là thư viện Google API Client hỗ trợ cho quá trình đăng nhập.

Các phương thức ***setClientId, setClientSecret, và setRedirectUri*** được sử dụng để cấu hình client với thông tin từ Google Developer Console.



Hình 3. Thông tin Client Id, secret

***$client->addScope('email');***

***$client->addScope('profile');***

Các phương thức này thêm phạm vi quyền mà ứng dụng yêu cầu từ người dùng. Trong trường hợp này, ứng dụng yêu cầu quyền truy cập vào địa chỉ email và thông tin hồ sơ của người dùng.

***$googleLoginUrl = $client->createAuthUrl();***

Tiếp theo sẽ tạo ra một URL mà người dùng sẽ được chuyển hướng đến để đăng nhập vào tài khoản Google của họ và ủy quyền cho ứng dụng.

* **Bước 5: Tạo google\_login xử lý phản hồi từ Google**

Xử lý quá trình xác thực người dùng thông qua Google OAuth 2.0. Sau khi người dùng đăng nhập, mã sẽ lưu trữ thông tin người dùng vào session và cơ sở dữ liệu, đồng thời chuyển hướng người dùng đến trang phù hợp dựa trên vai trò (role) của họ.

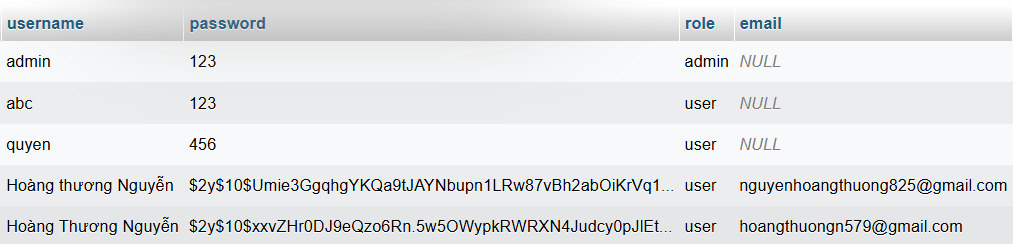
|  |
| --- |
| if (isset($\_GET['code'])) {  $token = $client->fetchAccessTokenWithAuthCode($\_GET['code']);  $client->setAccessToken($token['access\_token']);  } |

Kiểm tra xem có mã ủy quyền (code) từ Google không, được gửi qua URL sau khi người dùng đăng nhập thành công. Nếu có mã, nó sẽ được sử dụng để lấy token truy cập thông qua phương thức ***fetchAccessTokenWithAuthCode().*** Token truy cập này được thiết lập cho client, cho phép ứng dụng gửi yêu cầu đến Google API với quyền truy cập của người dùng.

Tiếp theo sẽ lấy thông tin hồ sơ người dùng

|  |
| --- |
| $google\_service = new Google\_Service\_Oauth2($client);  $user\_info = $google\_service->userinfo->get(); |

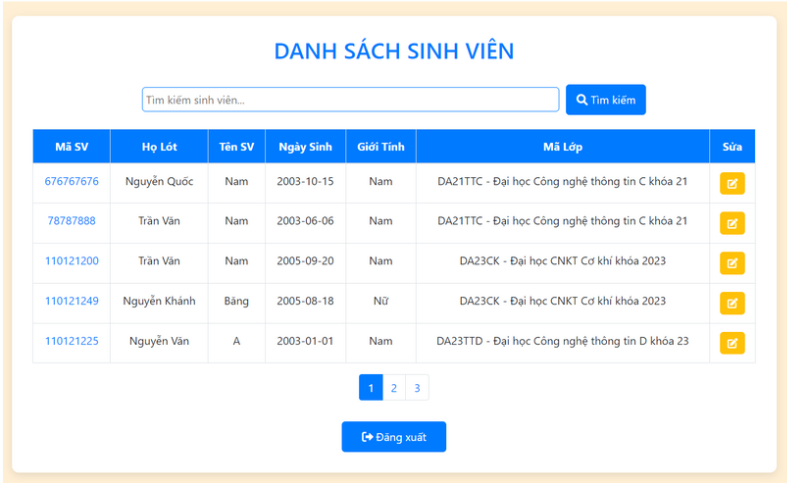
Tạo một đối tượng dịch vụ Google\_Service\_Oauth2 để tương tác với Google API. Sau đó gọi phương thức ***userinfo->get()*** để lấy thông tin hồ sơ của người dùng, bao gồm các thuộc tính như email, tên. Thông tin này sẽ được sử dụng để xác thực và quản lý người dùng trong ứng dụng. Sau khi lấy thông tin người dùng xong sẽ lưu vào cơ sở dữ liệu.



Hình 3. Lưu thông tin người dùng đăng nhập google vào cơ sở dữ liệu

**CHƯƠNG 4: PHÂN TRANG**

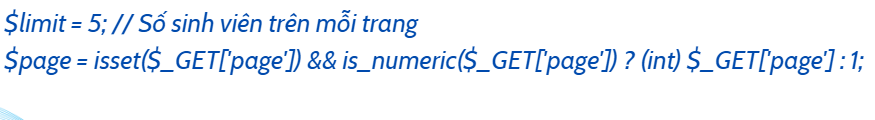
**4.1 Phân trang là gì?**

Phân trang cơ bản là quá trình lấy một tập hợp các kết quả và phân chia thành các trang để dễ xem hơn.Với những trang Web có dữ liệu lớn, việc phân trang là rất quan trọng, nó giúp trang web sẽ load nhanh hơn , giúp người dùng dễ dàng nhấn chọn trang mà mình mong muốn, đến được trang có nội dung mà mình cần, rất nhanh chóng, dễ dàng và chính xác.

Hình 4. Ví dụ về phân trang

**4.2 Các bước phân trang trong php**

* **Bước 1: Kết nối csdl**
* **Bước 2: Thiết lập số lượng sinh viên trên mỗi trang và xác định trang hiện tại**

****

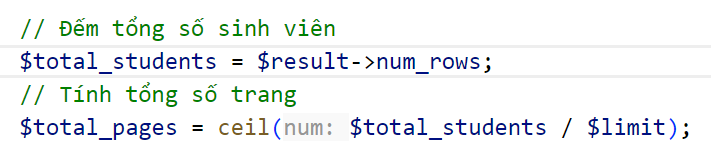
Đoạn mã trên dùng để thiết lập thông số (limit) cho việc phân trang, xác định số lượng record trên mỗi trang và xác định trang hiện tại dựa trên tham số trong URL, kiểm tra xem có tham số page trong URL không, nếu không, nó mặc định trang hiện tại là 1.

* **Bước 3: Tính toán chỉ số bắt đầu (offset)**

**$offset = ($page - 1) \* $limit;**

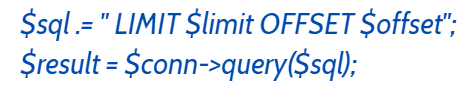
Đoạn mã trên sử dụng để tính toán số lượng trang cần thiết cho phân trang và xác định vị trí bắt đầu (offset) cho dữ liệu mà bạn sẽ lấy từ cơ sở dữ liệu.

* **Bước 4: Tìm tổng số sinh viên và tổng số trang**

****

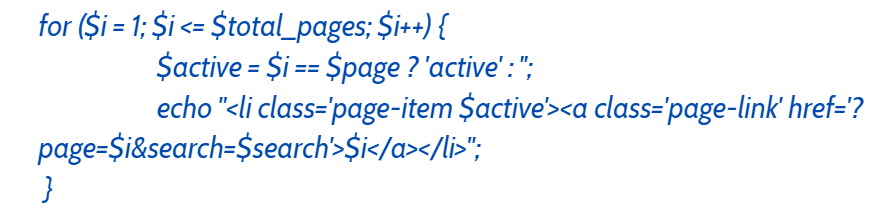
***num\_rows*** là thuộc tính của đối tượng ***$result*** dùng để lấy số lượng hàng (dòng) trong kết quả truy vấn. Điều này có nghĩa là ***$total\_students*** sẽ chứa tổng số sinh viên được truy vấn từ cơ sở dữ liệu. Sau đó sẽ tính tổng số trang bằng cách chia tổng số sinh viên ***($total\_students***) cho số lượng sinh viên trên mỗi trang ***($limit)*** để xác định số trang cần thiết

* **Bước 5: Truy vấn lấy danh sách sinh viên**

****

Thêm điều kiện ***LIMIT*** và ***OFFSET*** vào câu lệnh SQL để xác định số lượng record và vị trí bắt đầu của record cho truy vấn.

* **Bước 6: Hiển thị danh sách sinh viên và phân trang**

****

Tạo các liên kết phân trang trong một giao diện người dùng (UI) cho trang web. Đây là cách để người dùng có thể điều hướng qua các trang dữ liệu. Sử dụng vòng lặp for để lặp qua tất cả các số trang từ 1 đến tổng số trang ($total\_pages). Giá trị '**active**' có thể được sử dụng trong CSS để làm nổi bật trang hiện tại trong giao diện, giúp người dùng dễ nhận biết. Tạo một liên kết (<a>) với lớp CSS page-link cho phép người dùng chuyển đến trang $i.

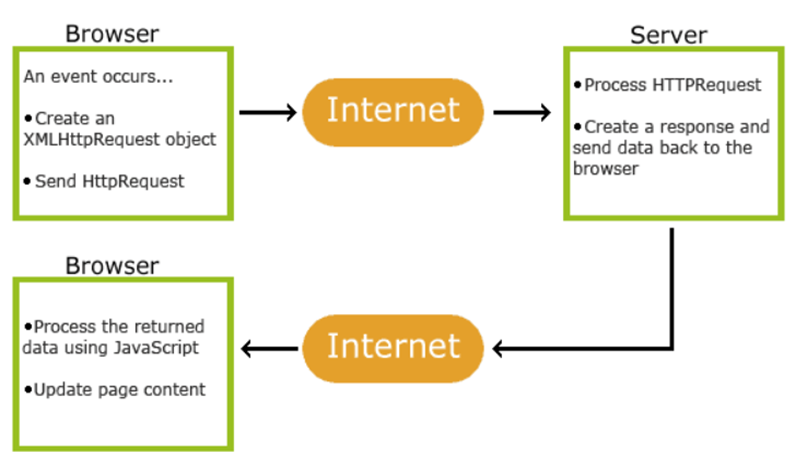
**CHƯƠNG 5: CÔNG NGHỆ AJAX TRONG PHP**

**5.1 Khái niệm AJAX trong PHP**

AJAX viết tắt từ Asynchronous JavaScript and XML, là bộ công nghệ giúp tạo ra các web động hay các ứng dụng giàu tính Internet, cho phép tăng tốc độ ứng dụng web bằng cách cắt nhỏ dữ liệu và chỉ hiển thị những gì cần thiết, thay vì tải đi tải lại toàn bộ trang web, làm như vậy trang web sẽ mượt và đẹp hơn. AJAX không phải một công nghệ đơn lẻ mà là sự kết hợp một nhóm công nghệ với nhau. Trong đó:

* HTML, CSS đóng vai trò hiển thị thông tin, dữ liệu
* Mô hình DOM (Document Object Model) được thực hiện thông qua JavaScript, nhằm hiển thị thông tin động và tương tác với những thông tin được hiển thị
* Đối tượng XMLHttpRequest để trao đổi dữ liệu một cách không đồng bộ với máy chủ web.
* XML thường là định dạng cho dữ liệu truyền.

**5.2 Cách hoạt động**



Hình 5. 1 Cách hoạt động của AJAX

1. Một sự kiện xảy ra trong một trang web (trang được tải, một nút được nhấp)

2. Đối tượng XMLHttpRequest được tạo bởi JavaScript

3. Đối tượng XMLHttpRequest gửi yêu cầu đến máy chủ web

4. Máy chủ xử lý yêu cầu

5. Máy chủ gửi phản hồi trở lại trang web

6. Phản hồi được đọc bởi JavaScript

7. Hành động thích hợp (như cập nhật trang) được thực hiện bởi JavaScript.

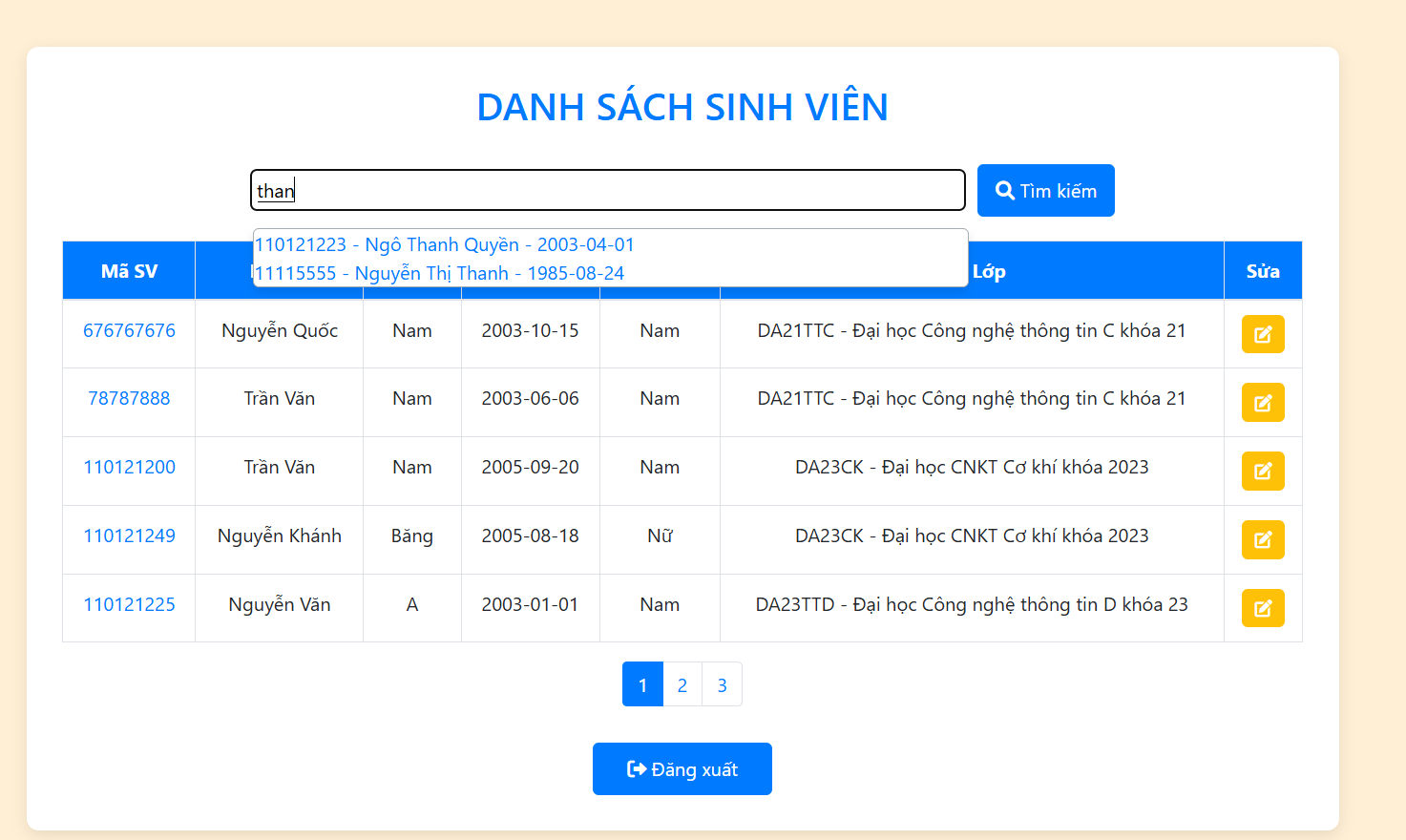
**5.3 Ứng dụng của AJAX**

**5.3.1 Ứng dụng của AJAX trong tìm kiếm trực tiếp (AJAX Livesearch)**

Ví dụ sau đây sẽ minh họa cách tìm kiếm trực tiếp, nơi nhận được kết quả tìm kiếm khi mình nhập.

Tìm kiếm trực tiếp có nhiều lợi ích so với tìm kiếm truyền thống:

* Kết quả được hiển thị khi người dùng nhập
* Kết quả tìm kiếm sẽ thu hẹp khi người dùng tiếp tục nhập
* Nếu kết quả tìm kiếm trở nên quá ít, hãy xóa các ký tự để xem kết quả rộng hơn.

****

Hình 5. Ứng dụng AJAX tìm kiếm sinh viên

* **Bước 1: Tạo trang HTML với ô tìm kiếm**

Trước tiên, cần một trang HTML với một ô nhập liệu để người dùng nhập từ khóa tìm kiếm:

<input type="text" name="search" class="search-input" placeholder="Tìm kiếm sinh viên..." onkeyup="showResult(this.value)">

<div id="livesearch" class="livesearch"></div>

***onkeyup="showResult(this.value)":*** Đây là sự kiện quan trọng trong tìm kiếm trực tiếp. ***onkeyup*** là một sự kiện JavaScript được kích hoạt mỗi khi người dùng nhấn phím lên sau khi gõ một ký tự trong ô tìm kiếm.

Khi sự kiện này được kích hoạt, hàm JavaScript ***showResult()*** sẽ được gọi và nhận giá trị của ô nhập liệu (giá trị này là từ khóa mà người dùng vừa nhập) thông qua ***this.value.***

Ví dụ: Khi người dùng nhập "A", hàm showResult("A") sẽ được gọi. Khi tiếp tục nhập "Ab", hàm showResult("Ab") sẽ được gọi để cập nhật kết quả tìm kiếm.

* **Bước 2: Tạo hàm JavaScript để thực hiện tìm kiếm trực tiếp**

<script>  
function showResult(str) {  
  if (str.length==0){  
    document.getElementById("livesearch").innerHTML="";  
   document.getElementById("livesearch").style.border="0px";  
    return;  
 }  
  var xmlhttp=new XMLHttpRequest();  
 xmlhttp.onreadystatechange=function(){  
    if (this.readyState==4 && this.status==200){  
      document.getElementById("livesearch").innerHTML=this.responseText;  
    document.getElementById("livesearch").style.border="1px solid #A5ACB2";  
   }  
 }  
 xmlhttp.open("GET","livesearch.php?q="+str,true);  
 xmlhttp.send();  
}

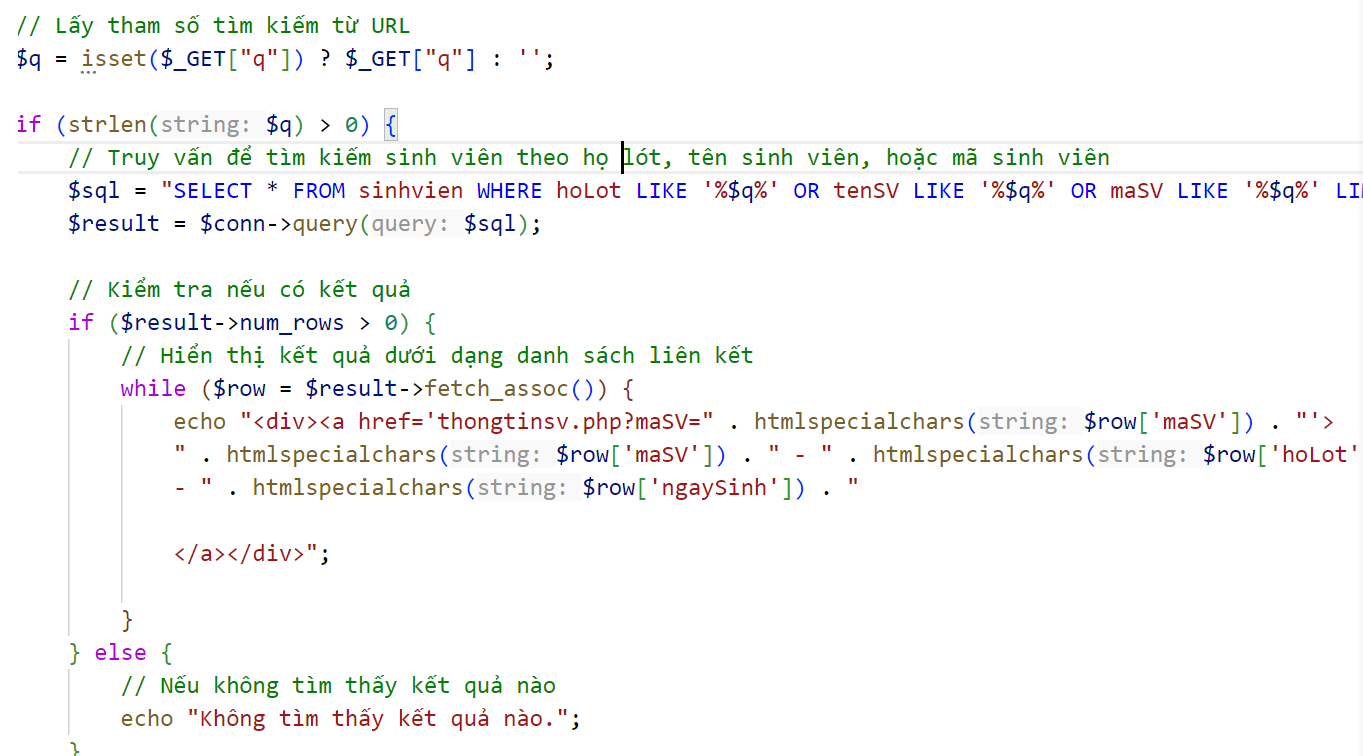
</script>

* Đầu tiên hàm sẽ kiểm tra độ dài của chuỗi nhập liệu (***str.length == 0***): Nếu chuỗi nhập rỗng, sẽ xóa kết quả tìm kiếm và không thực hiện yêu cầu.
* Kế tiếp tạo đối tượng ***XMLHttpRequest***: Đối tượng này cho phép thực hiện yêu cầu HTTP đến máy chủ mà không cần tải lại trang.

var xmlhttp=new XMLHttpRequest();

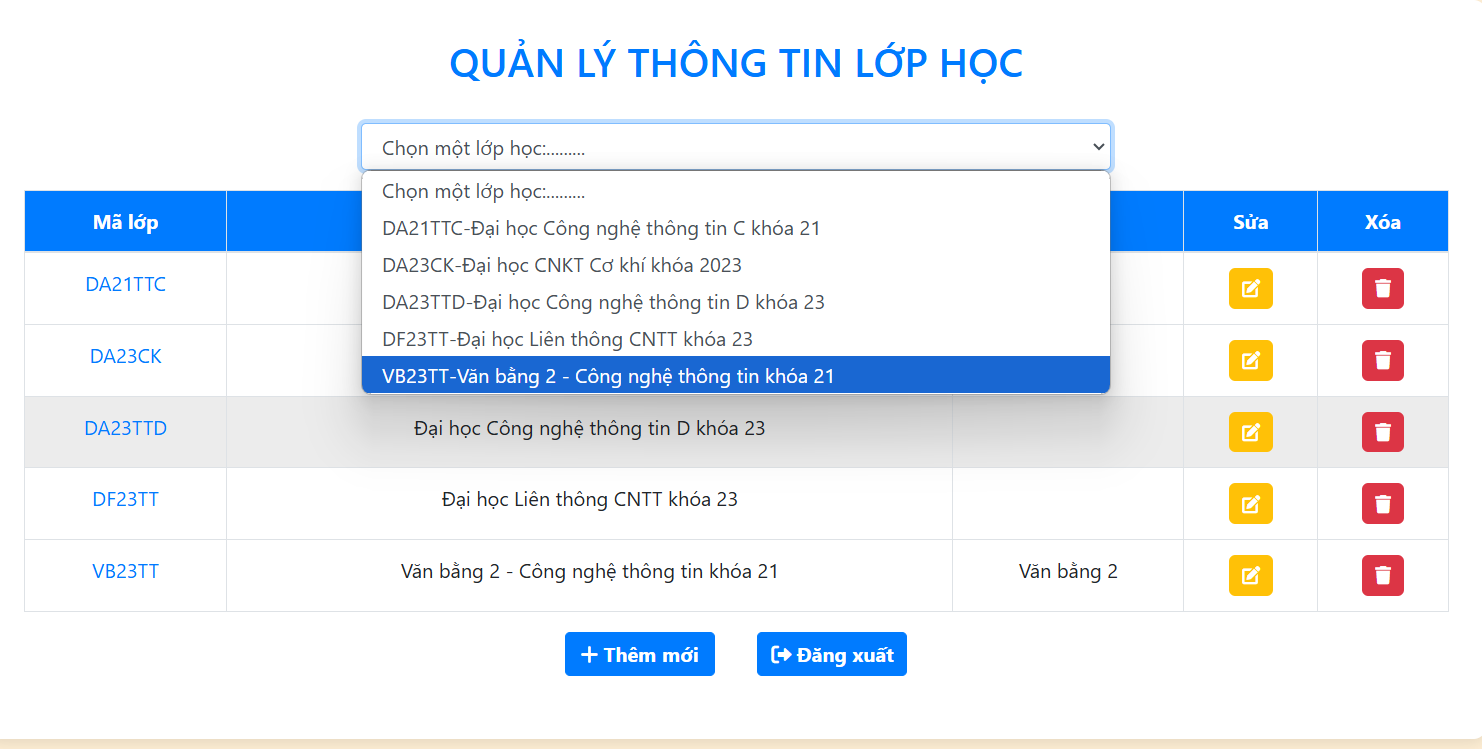
* Tiếp theo sự kiện onreadystatechange:
* Khi trạng thái của ***XMLHttpRequest*** thay đổi, hàm này sẽ kiểm tra xem yêu cầu đã hoàn thành ***(readyState == 4)*** và thành công ***(status == 200).***
* Nếu yêu cầu thành công, nội dung của **<*div id="livesearch">*** trong trang HTML sẽ được cập nhật với phản hồi từ máy chủ ***(this.responseText).***

- Cuối cùng là gửi yêu cầu đến máy chủ:

* xmlhttp.open("GET","livesearch.php?q="+str,true); mở một yêu cầu GET tới tệp ***livesearch.php*** với tham số truy vấn ***q*** là từ khóa tìm kiếm.
* xmlhttp.send() gửi yêu cầu.
* **Bước 3: Tạo tệp PHP (livesearch.php) để xử lý yêu cầu tìm kiếm**

Ví dụ trên sẽ thực hiện tìm kiếm cơ bản trên cơ sở dữ liệu để trả về danh sách sinh viên dựa trên tham số ***q*** từ URL. Nó tìm kiếm thông tin sinh viên dựa trên các trường: họ lót (hoLot), tên sinh viên (tenSV), mã sinh viên (maSV), giới tính (gioiTinh), và ngày sinh (ngaySinh),.... Kết quả được giới hạn ở 5 sinh viên đầu tiên phù hợp.

**5.3.2 Ứng dụng của AJAX trong lọc dữ liệu**

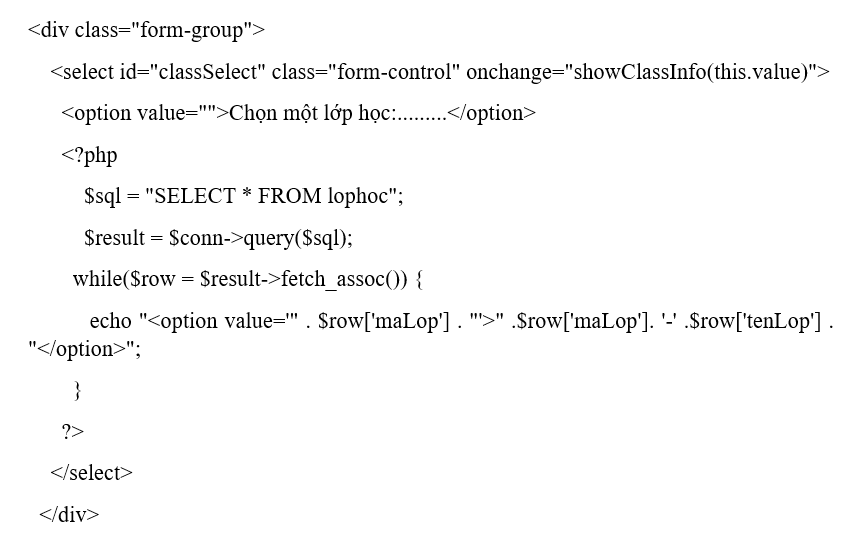
****Ứng dụng của AJAX trong lọc dữ liệu là một cách hiệu quả để cải thiện trải nghiệm người dùng bằng cách tải và hiển thị thông tin mà không cần phải tải lại toàn bộ trang. Dưới đây là ví dụ triển khai AJAX trong một ứng dụng lọc dữ liệu.

Hình 5. Ứng dụng AJAX lọc dữ liệu

* **Bước 1: Xác định yêu cầu**

Xác định dữ liệu cần lọc như: danh sách sinh viên, lớp học, sản phẩm, v.v.

Tạo giao diện người dùng: Thiết kế giao diện với các thành phần như dropdown, input textbox, hoặc checkbox,... để người dùng có thể chọn hoặc nhập thông tin cần lọc.

**Ví dụ:**

Trong ví dụ trên, khi người dùng chọn một người trong danh sách thả xuống ở trên, một hàm có tên ***showClassInfo()*** là sẽ được thực thi. Chức năng này được kích hoạt bởi sự kiện ***onchange.***

* **Bước 2: Viết hàm JavaScript để xử lý sự kiện**

<script>  
function showClassInfo(str) {

if (str == "") {

document.getElementById("classInfo").innerHTML = "";

return;

} else {

var xmlhttp = new XMLHttpRequest();

xmlhttp.onreadystatechange = function() {

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

document.getElementById("classInfo").innerHTML = this.responseText;

}

};

xmlhttp.open("GET", "ajax\_get\_class\_info.php?q=" + str, true); // Thay đổi đường dẫn tới file xử lý

xmlhttp.send();

}

}</script>

* Tương như trong AJAX livesearch, đầu tiên hàm sẽ kiểm tra xem người đó có được chọn không. Nếu không có người nào được chọn ***(str == "")***, hãy xóa nội dung txtHintvà thoát khỏi hàm.
* Kế tiếp tạo đối tượng ***XMLHttpRequest:*** Đối tượng này cho phép thực hiện yêu cầu HTTP đến máy chủ mà không cần tải lại trang.

var xmlhttp=new XMLHttpRequest();

* Tiếp theo sự kiện onreadystatechange:
* Khi trạng thái của ***XMLHttpRequest*** thay đổi, hàm này sẽ kiểm tra xem yêu cầu đã hoàn thành ***(readyState == 4)*** và thành công ***(status == 200).***
* Nếu yêu cầu thành công, nội dung của ***<div id="classInfo" class="mt-3"></div>*** trong trang HTML sẽ được cập nhật với phản hồi từ máy chủ ***(this.responseText).***

- Cuối cùng là gửi yêu cầu đến máy chủ:

* xmlhttp.open("GET", "ajax\_get\_class\_info.php?q=" + str, true); mở một yêu cầu GET tới tệp ***ajax\_get\_class\_info.php*** với tham số truy vấn ***q*** là giá trị đã chọn
* xmlhttp.send() gửi yêu cầu.
* **Bước 3: Tạo tệp PHP *ajax\_get\_class\_info.php*** **để xử lý yêu cầu lọc dữ liệu**

Tương tự như AJAX livesearch, chúng ta sẽ thực hiện những truy vấn cơ bản trên cơ sở dữ liệu để trả về dữ liệu mà ta mong muốn dựa trên tham số ***q*** từ URL.

**CHƯƠNG 6: REWIRITE**

Việc sử dụng rewrite trong Apache thông qua file ***.htaccess*** có nhiều ý nghĩa quan trọng trong phát triển và quản lý ứng dụng web. Dưới đây là một số lợi ích và ý nghĩa chính của việc sử dụng quy tắc rewrite:

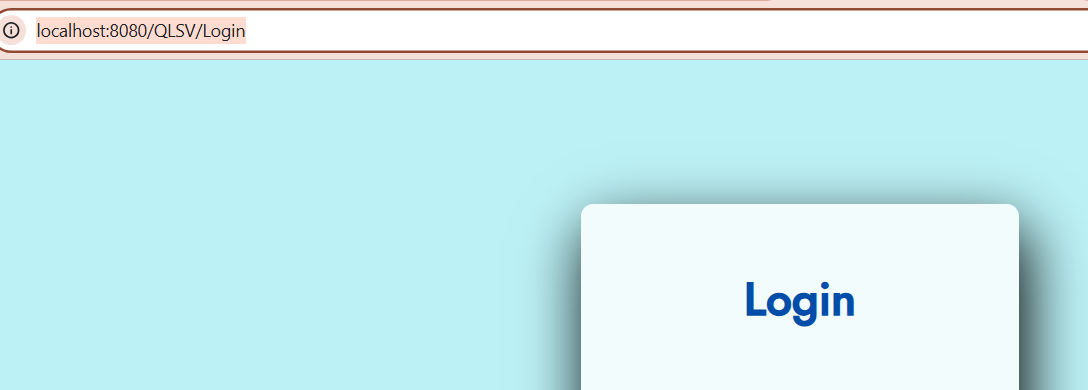
* Các URL được rewrite trở nên dễ đọc và dễ nhớ hơn.
* Bảo mật: Ẩn phần mở rộng file (như .php, .html) giúp giảm thiểu thông tin mà kẻ tấn công có thể thu thập về công nghệ mà bạn đang sử dụng, từ đó bảo vệ ứng dụng tốt hơn.
* Tạo Đường dẫn Động: Với các ứng dụng web động (như PHP), có thể tạo các URL động mà không cần phải chỉ định phần mở rộng file. Điều này giúp cải thiện trải nghiệm người dùng và làm cho ứng dụng trông chuyên nghiệp hơn.

Để sử dụng file ***.htaccess*** với XAMPP nhằm mã hóa URL như trong ví dụ sau, cần thực hiện một số bước thiết lập.

* **Bước 1: Kích hoạt mô-đun mod\_rewrite trong XAMPP**
* Khởi động XAMPP: Nhấp vào nút Config bên cạnh Apache và chọn ***httpd.conf*** để mở file cấu hình.
* Tìm dòng sau: #LoadModule rewrite\_module modules/mod\_rewrite.so. Sau đó bỏ dấu # ở đầu dòng để kích hoạt mô-đun mod\_rewrite
* Thêm hoặc sửa dòng sau: AllowOverride All
* **Bước 2: Tạo file .htaccess**

|  |
| --- |
| # Bật chế độ viết lại  RewriteEngine On  # Đặt đường dẫn cơ sở  RewriteBase /QLSV/  # Ẩn phần mở rộng .php cho trang Login  RewriteRule ^Login$ Login.php [L]  # Ẩn phần mở rộng .php cho trang lop  RewriteRule ^lophoc$ lophoc.php [L] |

Sau khi cấu hình xong, ta có thể chạy mà không cần phần đuôi mở rộng (.php)

****

Hình 6. Rewirite trang login.php

**CHƯƠNG 7: ỨNG DỤNG JAVASCRIPT GỌI API HIỂN THỊ ĐỊA LÝ**

* **Ưu điểm**
* Trải nghiệm người dùng tốt: Việc tải dữ liệu động giúp người dùng có thể chọn địa chỉ một cách dễ dàng và tiện lợi, mà không cần phải tải lại toàn bộ trang.
* Hiệu suất cao: Ứng dụng chỉ tải dữ liệu cần thiết theo yêu cầu, giúp giảm bớt tải cho server và cải thiện tốc độ phản hồi.
* **Hạn chế**
* Phụ thuộc vào API bên ngoài: Nếu API ngừng hoạt động hoặc có thay đổi về cấu trúc dữ liệu, ứng dụng có thể gặp lỗi.
* Xử lý lỗi đơn giản: Hiện tại, ứng dụng chỉ hiển thị thông báo lỗi cơ bản khi không tải được dữ liệu. Nên thêm tính năng thông báo chi tiết hơn cho người dùng để cải thiện trải nghiệm khi gặp lỗi.
* **Gọi API tải danh sách địa lý Việt Nam**

Ứng dụng này sử dụng API từ địa chỉ ***"https://provinces.open-api.vn/api/"*** để lấy thông tin về các địa danh. Mỗi loại địa danh được tải bằng các hàm khác nhau:

|  |
| --- |
| const host = "https://provinces.open-api.vn/api/"; |

**callAPI():** Gọi API để tải danh sách các tỉnh/thành phố.

**callApiDistrict():** Gọi API để tải danh sách các quận/huyện dựa trên mã tỉnh/thành phố được chọn.

**callApiWard():** Gọi API để tải danh sách các phường/xã dựa trên mã quận/huyện được chọn.

|  |
| --- |
| Ví dụ:  // Gọi API để tải danh sách tỉnh/thành phố  const callAPI = (api) => {  return axios.get(api)  .then((response) => {  renderData(response.data, "province");  })  .catch((error) => {  console.error("Error loading provinces:", error);  alert("Có lỗi khi tải thông tin tỉnh/thành phố.");  });  }; |

Mỗi hàm này đều sử dụng thư viện Axios để gửi yêu cầu GET tới API và xử lý kết quả trả về. Khi dữ liệu tải thành công, chúng sẽ được chuyển đến hàm ***renderData()*** để hiển thị vào các dropdown (ô chọn).

|  |
| --- |
| // Hàm để hiển thị dữ liệu vào dropdown  const renderData = (array, select) => {  let row = '<option disabled selected value="">Chọn</option>';  array.forEach(element => {  row += `<option value="${element.code}">${element.name}</option>`;  });  const selectElement = document.getElementById(select);  selectElement.innerHTML = row;  selectElement.disabled = false; // Kích hoạt dropdown khi có dữ liệu  };  // Sự kiện khi chọn tỉnh/thành phố  document.getElementById("province").addEventListener("change", function() {  callApiDistrict(host + "p/" + this.value + "?depth=2");  document.getElementById("district").disabled = true; // Vô hiệu hóa dropdown quận/huyện  document.getElementById("ward").disabled = true; // Vô hiệu hóa dropdown phường/xã  }); |

Hàm ***renderData()*** có nhiệm vụ hiển thị dữ liệu vào các dropdown tương ứng (tỉnh/thành phố, quận/huyện, hoặc phường/xã). Nó nhận vào hai tham số:

* array: Mảng dữ liệu từ API trả về.
* select: Tên của dropdown cần hiển thị dữ liệu.
* **Tạo nội dung cho dropdown:**

Biến ***row*** chứa một ***<option>*** mặc định với giá trị rỗng, để người dùng có thể chọn. Sử dụng ***forEach*** để duyệt qua từng phần tử trong mảng array, tạo một <option> cho mỗi phần tử với giá trị code và nội dung hiển thị là name.

* **Cập nhật nội dung của dropdown**

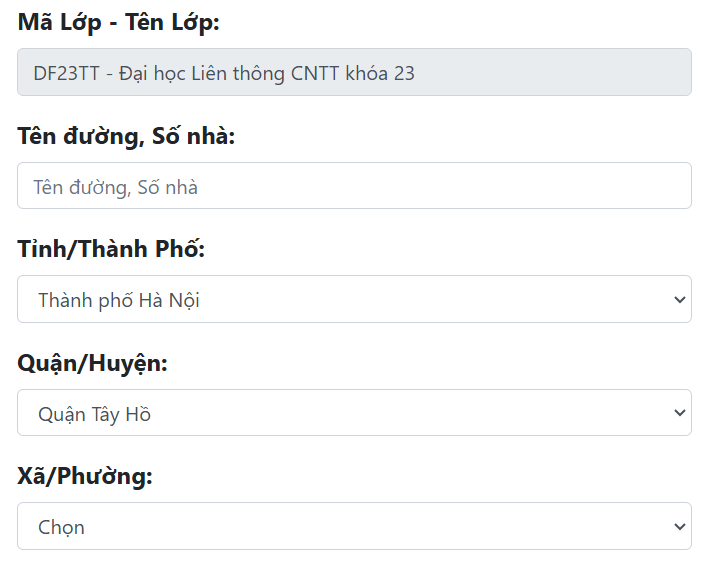
Tìm phần tử HTML bằng ***document.getElementById(select)*** và cập nhật nội dung của nó bằng ***row***. Kích hoạt ***dropdown (disabled = false)*** để có thể chọn được.

* **Sự kiện khi chọn tỉnh/thành phố**

Phần này lắng nghe sự kiện "change" khi người dùng chọn một tỉnh/thành phố:

* Gọi API để lấy dữ liệu quận/huyện: Sử dụng ***callApiDistrict*** để gọi API với URL chứa mã của tỉnh/thành phố đã chọn và độ sâu dữ liệu là 2 (bao gồm cả quận/huyện).
* Vô hiệu hóa dropdown quận/huyện và phường/xã:

Trước khi có dữ liệu mới, hai dropdown này sẽ bị vô hiệu hóa (disabled = true) để tránh việc chọn sai dữ liệu.



Hình 7. Thêm địa chỉ cho sinh viên sủ dụng API