

PHIẾU HỌC TẬP CHỦ ĐỘNG

Môn học: CSE485: Công nghệ Web

Họ và tên: Nguyễn Thị Yên – MSV: 2351160567

CHƯƠNG 9: BẢO MẬT ỦNG DỤNG WEB¹

1. Lý thuyết Cốt lõi (Khái niệm)

Trong chương này, chúng ta tập trung vào 4 khái niệm bảo mật chính:

- XSS (Cross-Site Scripting) (9.4):
 - **Ván đè:** Kẻ tấn công tìm cách "tiêm" (**inject**) một đoạn mã JavaScript độc hại (ví dụ: qua ô comment, ô nhập tên) vào CSDL của bạn. Khi một người dùng khác tải trang đó, mã JS độc hại sẽ chạy trên trình duyệt của họ và đánh cắp thông tin (như cookie đăng nhập).
 - **Giải pháp Laravel:** Cỗ máy Blade **tự động** bảo vệ bạn. Khi bạn dùng cú pháp `{{ $variable }}`, Laravel sẽ "thoát" (escape) tất cả các thẻ HTML, biến `<script>alert(1)</script>` thành text vô hại `<script>alert(1)</script;` (sẽ được trình duyệt in ra y hệt, chứ không chạy).
- CSRF (Cross-Site Request Forgery) (9.4):
 - **Ván đè:** Kẻ tấn công lừa trình duyệt của một người dùng (đang đăng nhập vào trang của bạn) gửi một yêu cầu đến trang của bạn mà người dùng không hề hay biết. Ví dụ: Kẻ tấn công đặt một ảnh "vô hình" trên trang web của hắn: ``. Khi người dùng (đã đăng nhập) truy cập trang của kẻ tấn công, trình duyệt sẽ tự động tải ảnh này, vô tình gửi yêu cầu xóa tài khoản.
 - **Giải pháp Laravel:** Laravel yêu cầu mọi form POST, PUT, PATCH, DELETE phải đính kèm một "**token**" (**mã bí mật**) duy nhất cho phiên đó.
 - Bạn chỉ cần thêm directive `@csrf` vào trong form.
 - Laravel sẽ tự động tạo thẻ `<input type="hidden" name="_token" value="...token...">`.

- Khi form được gửi, Laravel kiểm tra token này. Nếu token không khớp (hoặc không có), nó sẽ từ chối yêu cầu (lỗi 419 Page Expired).
- **Xác thực (Authentication) (9.3):**
 - Trả lời câu hỏi: "Bạn là ai?"
 - Đây là quá trình định danh người dùng, ví dụ: kiểm tra username/password (chức năng Đăng nhập, Đăng ký).
- **Phân quyền (Authorization) (9.3):**
 - Trả lời câu hỏi: "Bạn được phép làm gì?"
 - Xảy ra sau khi đã xác thực. Ví dụ: Người dùng A (đã đăng nhập) là "user" nên chỉ được xem bài viết, nhưng người dùng B (đã đăng nhập) là "admin" thì được quyền sửa và xóa bài viết.

2. Nhiệm vụ Thực hành (BẮT BUỘC)

Kịch bản: Chúng ta sẽ nâng cấp PHT Chương 8 (Quản lý sinh viên) để vá 2 lỗ hổng bảo mật lớn là CSRF và XSS.

Code Khởi đầu (Starter Code): Sử dụng dự án và các tệp của PHT Chương 8.

A. Chống tấn công CSRF (9.4)

1. // TODO 1: Mở tệp View resources/views/sinhvien/list.blade.php (từ PHT 8).
2. // TODO 2: Tìm thẻ <form ...> (dùng để thêm sinh viên).
3. // TODO 3: Thêm directive @csrf ngay bên dưới thẻ mở <form ...>: HTML

```

<form action="{{ route('sinhvien.store') }}"
method="POST">

@csrf

Tên sinh viên: <input type="text" name="ten_sinh_vien" required>
Email: <input type="email" name="email" required>
<button type="submit">Thêm</button>
</form>

```

B.Kiểm chứng khả năng chống XSS (9.4)

1. // **TODO 4:** Đảm bảo rằng trong vòng lặp @foreach của tệp list.blade.php, bạn đang dùng cú pháp {{ }} (hai dấu ngoặc nhọn) để in tên sinh viên, **không phải** {{ }} (một nhọn, hai chấm than). PHP

```

@foreach($danhSachSV as $sv)

<tr>
    <td>{{ $sv->id }}</td>
    <td>{{ $sv->ten_sinh_vien }}</td>
    <td>{{ $sv->email }}</td>
</tr>

@endforeach

```

2. // **TODO 5:** Chạy php artisan serve và mở trang /sinhvien.
3. // **TODO 6:** Sử dụng form, thêm một sinh viên mới với thông tin sau:
 - Tên sinh viên: <script>alert('Ban da bi XSS!');</script>
 - Email: hacker@email.com

4. // **TODO 7:** Nhấn "Thêm" và quan sát bảng danh sách sinh viên.

C. Bảo vệ Route bằng Xác thực (9.3) (Giới thiệu)

1. // **TODO 8:** Mở file routes/web.php.
2. // **TODO 9:** Giả sử bạn đã cài đặt hệ thống đăng nhập (ví dụ: Laravel Breeze), bạn có thể "bọc" các route sinh viên bằng một middleware để bắt buộc người dùng phải đăng nhập mới được xem. PHP

```
// Gợi ý:  
  
// Route::middleware(['auth'])->group(function () {  
//     Route::get('/sinhvien', [SinhVienController::class, 'index'])->  
//     name('sinhvien.index');  
  
//     Route::post('/sinhvien', [SinhVienController::class, 'store'])->  
//     name('sinhvien.store');  
// });  
  
// 'auth' là middleware kiểm tra xác thực
```

(Đây là phần giới thiệu, bạn sẽ phải làm điều này trong BTL)

3. Yêu cầu Bằng chứng (Proof of Work)

Bạn phải nộp lại 2 bằng chứng sau:

A. Code đã hoàn thiện:

1. Dán (paste) code của khối <form> trong tệp list.blade.php (chứng minh bạn đã thêm @csrf).

```
<form action="{{ route('sinhvien.store') }}" method="POST">  
    @csrf  {{-- BẮT BUỘC với form POST của laravel--}}  
    <div>  
        <label for="ten_sinh_vien">Tên sinh viên:</label><br>
```

```

<input
    type="text"
    name="ten_sinh_vien"
    id="ten_sinh_vien"
    required>
</div>
<br>
<div>
    <label for="email">Email:</label><br>
    <input
        type="email"
        name="email"
        id="email"
        required>
</div>
<br>
<button type="submit">Thêm sinh viên</button>
</form>

```

2. Dán (paste) code của khối @foreach trong tệp list.blade.php (chứng minh bạn dùng {{ }}).

```

@forelse($danhSachSV as $index => $sv)
    <tr>
        <td>{{ $index + 1 }}</td>
        <td>{{ $sv->ten_sinh_vien }}</td>
        <td>{{ $sv->email }}</td>

```

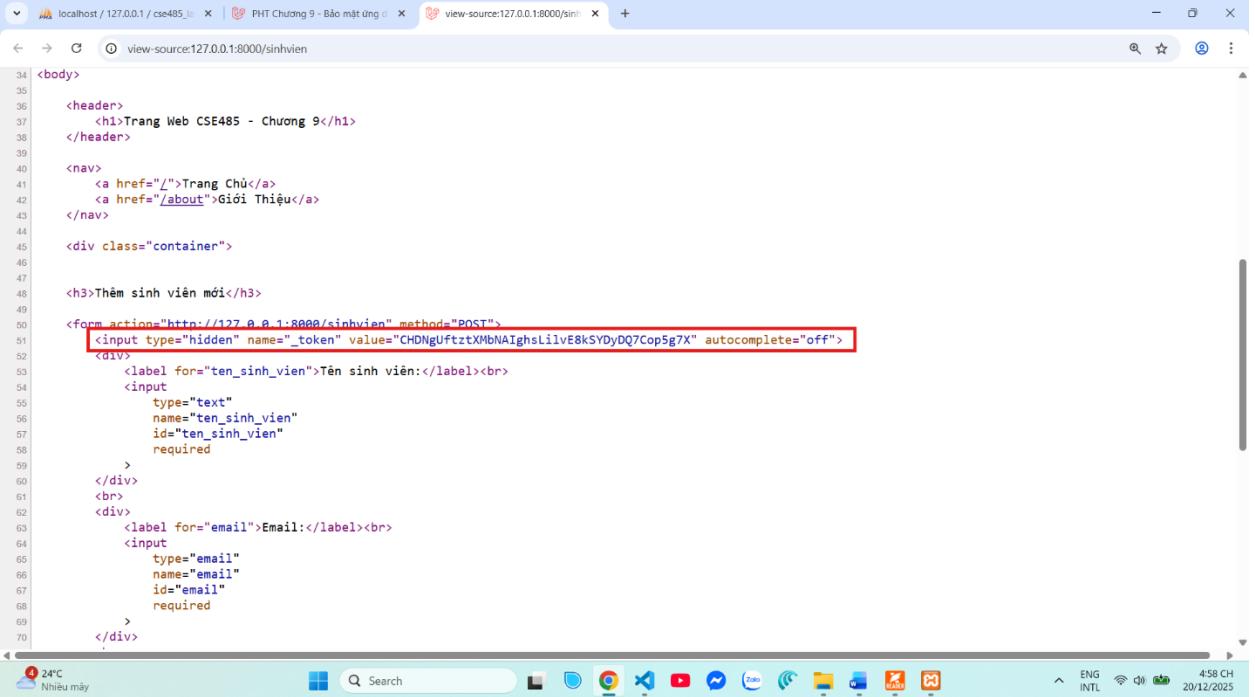
```

    </tr>
@empty
<tr>
    <td colspan="3" align="center">
        Chưa có sinh viên nào
    </td>
</tr>
@endforelse

```

B. Ảnh chụp màn hình Kết quả (BẮT BUỘC 2 ẢNH):

- Ảnh 1 (Bằng chứng Chống CSRF):** Tải trang /sinhvien, **nhấn chuột phải** \$\rightarrow\$ **View Page Source** (Xem nguồn trang). Chụp ảnh màn hình mã nguồn HTML, **khoanh tròn** vào thẻ `<input type="hidden" name="_token" ...>` mà @csrf đã tự động tạo ra.



The screenshot shows a browser window with the URL `view-source:127.0.0.1:8000/sinhvien`. The page content is displayed in a code editor-like interface. A specific line of code is highlighted with a red box:

```

<input type="hidden" name="_token" value="CHDNgUftztXMBNAIghsLilvE8kSYDyQ7Cop5g7X" autocomplete="off">

```

This line contains a randomly generated string of characters, which is the CSRF token.

- Ảnh 2 (Bằng chứng Chống XSS):** Chụp ảnh màn hình trang /sinhvien **sau khi** bạn đã thêm sinh viên ở (TODO 6 & 7). Ảnh phải cho thấy dòng chữ `<script>alert('Ban da bi XSS!');</script>` được **in ra dưới dạng text** trên bảng, **chứ KHÔNG CÓ** popup "alert" nào hiện lên.

The screenshot shows a web application titled "Trang Web CSE485 - Chương 9". The main content area has two sections: "Thêm sinh viên mới" (Add new student) with input fields for name and email, and a table titled "Danh sách sinh viên hiện có" (List of current students) showing three entries. The table has columns for #, Name, and Email. The third entry's name contains an XSS payload, and its email address is a hacker@example.com.

| # | Tên sinh viên | Email |
|---|---|---------------------|
| 1 | Nguyễn Văn A | nguyenvan@gmail.com |
| 2 | Nguyễn Thị B | nguyenth@gmail.com |
| 3 | <script>alert('Bạn đã bị XSS!');</script> | hacker@email.com |

(Dán Code A1, A2 và 2 Ánh B1, B2 của bạn vào đây)

4. Câu hỏi Phản biện (Bắt buộc)

Sau khi hoàn thành PHT, hãy đặt 01 câu hỏi tư duy.

(Gợi ý: "Sự khác biệt cơ bản giữa Xác thực (Authentication) và Phân quyền (Authorization) là gì? ² Trong Bài tập lớn, chức năng 'Đăng nhập' là Authentication hay Authorization? Chức năng 'Chỉ Admin mới thấy trang Quản trị' là gì?").

Câu hỏi của tôi là:

1. Nếu một lập trình viên dùng {{ }} ở hầu hết các chỗ, nhưng chỉ sai một chỗ duy nhất, XSS có thể gây hậu quả gì?
2. Vì sao Laravel không dùng cookie thông thường mà lại dùng CSRF token để chống giả mạo request?
3. Vì sao không nên phân quyền bằng cách kiểm tra role trong view (Blade)?

5. Kết nối Đánh giá (Rất quan trọng)

Kiến thức trong PHT này là **tối quan trọng**.

Áp dụng: Bảo mật (chống XSS, CSRF) và Xác thực/Phân quyền

(Authentication/Authorization) là các yêu cầu **BẮT BUỘC** và chiếm trọng số điểm cao trong **Bài tập lớn theo nhóm (50%)³**. Một dự án nộp cho giảng viên mà không có @csrf trong form, bị lỗi XSS, hoặc không bảo vệ các trang quản trị (cho phép truy cập mà không cần đăng nhập) sẽ bị xem là lỗi nghiêm trọng và bị trừ điểm rất nặng.