TRƯỜNG CAO ĐẲNG CÔNG NGHỆ THỦ ĐỨC

**Khoa Công Nghệ Thông Tin**



**Chuyên đề phát triển web 1** | HKI – [2021 – 2022]

PRJ – REPORT

# Tìm Hiểu ReactJS Làm Demo Công Việc Trong Tuần

**Nhóm J:**

* Cao Trung Hiếu
* Trần Văn Lập
* Lê Trung Hiếu
* Võ Thành Đạt
* Nguyễn Khắc Danh

**Mục Lục**

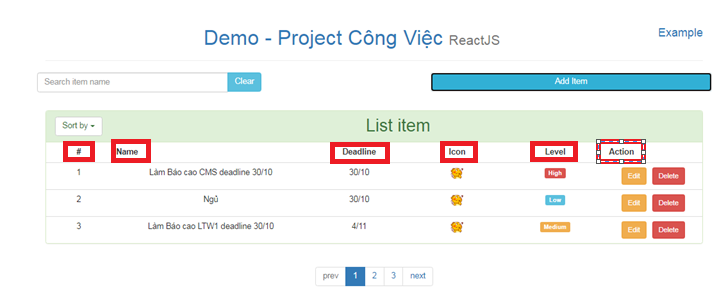
Mục Lục Hình

**BẢNG PHÂN CHIA CÔNG VIỆC**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Front End Web Client (Dùng ReactJs)** | | | | | | | | | |
| **STT** | **Trang** | **Khu Vực** | **Mô tả** | **Thành viên đảm nhận** | | | | | |
| **Front End** | **Deadline** | **Tình trạng** | **Back End** | **Deadline** | **Tình trạng** |
| 1 | Thành phần chung | Header | - Nút đăng ký, đăng nhập. Khi click vào sẽ dẫn về trang tương ứng | Hiếu Lê | 12h - 27/9 | Hoàn thành | Hiếu Lê | 12h 27/9 | Chưa làm |
| Footer |  | Hiếu Lê | 12h - 28/9 | Hoàn thành | Không | | |
| 2 | Trang danh sách bài viết | Khu vực bài viết nổi bật | - Hiển top 10 bài viết có lượt xem cao nhất, lượt tương tác cao nhất - Nội dung hiển thị bao gồm: Ảnh bìa bài viết, tiêu đề, nội dung tóm tắt, danh mục, người viết, thời gian được tạo - Khi click vào thì đi tới trang chi tiết của bài viết đó | Lập | 8h - 2/10 | Hoàn thành | Danh | 8h - 2/10 | Hoàn thành |
| Khu vực danh mục bài viết | Hiển thị các danh mục, khi click vào từng danh mục thì hiển thị danh sách bài viết thuộc danh mục đó | Lập | 8h - 3/10 | Hoàn thành | Lập | 8h - 3/10 | Hoàn thành |
| Danh sách bài viết | - Danh sách tất cả các bài viết của danh mục - Phân trang, mỗi trang 10 bài viết - Nội dung hiển thị bao gồm: Ảnh bìa bài viết, tiêu đề, nội dung tóm tắt, người viết, thời gian được tạo - Khi click vào thì đi tới trang chi tiết của bài viết đó | Lập | 8h - 4/10 | Hoàn thành | Lập | 8h - 4/10 | Hoàn thành |
| 3 | Trang đăng nhập | Đăng nhập | - Đăng nhập bằng email đăng ký - Người dùng nhập email và mật khẩu để đăng nhập | Danh | 6h - 8/10 | Hoàn thành | Hiếu Cao | 12h 8/10 | Hoàn thành |
| Đăng Đăng ký | - Link tới trang đăng ký | Danh | 1h - 8/10 | Hoàn thành | Không | | |
| Quên mật khẩu | - Link tới trang quên mật khẩu | Danh | 1h - 8/11 | Hoàn thành |
| 4 | Trang đăng ký | Đăng ký | - Người dùng nhập các thông tin để đăng ký: Email, Password, Họ tên, Giới tính,  - Phải tích chọn mục aggree policy thì mới cho đăng ký - User sau khi đăng ký xong, sẽ nhận được email để kích hoạt tài khoản, có thể sử dụng link hoặc mã xác nhận, có thời hạn, nếu quá thời hạn mà user chưa kích hoạt tài khoản thì tài khoản bị deactive | Danh | 7h - 9/10 | Hoàn thành | Hiếu Cao | 12h 9/10 | Hoàn thành |
| Nút Đăng nhập | - Link tới trang đăng nhập | Danh | 1h - 9/10 | Hoàn thành | Không | | |
| 5 | Trang quên mật khẩu | Nhập email để lấy mã reset | - User nhập email để nhận mã reset password | Danh | 4h - 10/10 | Hoàn thành | Hiếu Cao | 4h - 10/10 | Hoàn thành |
| Nhập mã reset và nhập mật khẩu mới | - User nhập mã reset (mã hoặc link reset có thời hạn) password nhận được và nhập mật khẩu mới | Danh | 4h - 10/10 | Hoàn thành | Hiếu Cao | 4h - 10/10 | Hoàn thành |
| 6 | Trang profile | Quản lý profile cá nhân | - User có thể thay đổi các thông tin cá nhân: Họ tên, Avatar, Giới tính. - User có thể thay đổi mật khẩu | Đạt | 8h - 11/10 4h - 12/10 | Đang làm | Hiếu Cao | 8h - 11/10 4h - 12/10 | Chưa làm |
| Quản lý bài viết | - List danh sách các bài viết của user đi kèm chức năng CRUD | Đạt | 4h - 12/10 8h - 13/10 | Đang làm | Hiếu Lê | 4h - 12/10 8h - 13/10 | Chưa làm |
| 7 | Trang tạo bài viết | Tạo bài viết mới | - User tạo dữ liệu cho bài viết mới - Nội dung bài viết sử dụng editor React-Quill | Đạt | 8h - 14/10 | Chưa làm | Hiếu Lê | 12h 14/10 | Chưa làm |
| **Web Quản trị và Back End(Dùng Botble CMS)** | | | | | | | | | |
| 1 | Quản lý danh sách User | Quản trị User | - Hiển thị list user của trang - Khi click vào một user thì xem thông tin chi tiết của user đó - Có thể deactive tài khoản của user, user khi bị deactive thì không thể đăng bài | Lập | 12h - 15/10 | Chưa làm | Lập | 12h - 15/10 | Chưa làm |
| 2 | Quản lý danh sách bài viết | Quản trị bài viết | - Hiển thị list bài viết của trang, - Khi click vào một bài viết thì xem thông tin chi tiết của bài viết đó - Đối với bài viết do user tạo thì admin phải duyệt qua bài viết, nếu bài viết chưa được duyệt qua thì chưa hiển thị trên trang client | Đạt | 12h - 15/10 | Chưa làm | Đạt | 12h - 15/10 | Chưa làm |
| 3 | Quản lý danh mục bài viết | Quản trị danh mục bài viết | - Hiển thị list danh mục bài viết - Khi click vào một danh mục thì xem thông tin chi tiết của danh mục đó | Hiếu Cao | 12h - 15/10 | Chưa làm | Hiếu Cao | 12h - 15/10 | Chưa làm |

## **Giới Thiệu**

* Giới thiệu môn học : Môn chuyên đề phát triển web-1 là một trong những môn quan trọng nhất để sinh viên có thể tiếp cận với những công việc trong tương lai, được tiếp xúc với các cách quản lí code, định hướng cho sinh viên tự chọn đề tài , được góp ý và bổ sung những kiến thức cần thiết để hoàn thành 1 đồ án hoàn chỉnh.
* Nhóm J chuyên đề phát triển web 1 sẽ hướng dẫn các bạn làm một web đơn giản bằng React kết hợp với Botble CMS và Tailwind CSS.
* **ReactJS** là thư viện do Facebook phát triển sau và được open source, vì trong quá trình phát triển xã hội lượng người sử dụng các phần mềm gia tăng nhiều nhu cầu khác nhau cho nên các framework cũ không đáp ứng được hết nhu cầu của người dùng. Và vì thế mà ReactJs đã được ra đời bởi các developer nghiên cứu và phát triển của Facebook. Hiện tại thì họ có 2 loại đó là **ReactJs** và **React Native** (phát triển bên điện thoại).
* **Botble** là một **CMS** dựa trên Laravel Framework cho phép bạn xây dựng các trang web cho bất kỳ mục đích nào. Nó có các công cụ mạnh mẽ cho các nhà phát triển để xây dựng bất kỳ loại trang web nào. Các chức năng trong bản **CMS** mặc định: Blog, page, menu, thư viện hình ảnh, custom fields.
* **Tailwind** là một tập hợp các lớp tiện ích (class) cấp thấp. Chúng có thể được sử dụng như những viên gạch lego để xây dựng bất kỳ loại thành phần nào. Framework bao gồm các thuộc tính **CSS** quan trọng nhất (nhưng có thể dễ dàng mở rộng bằng nhiều cách khác nhau). Với **Tailwind**, tùy biến trở nên dễ dàng hơn bao giờ hết.
* Trang website Blog có chức năng đăng các bài viết và quản lí các bài viết đó thông qua Botble CMS
* Và ứng dụng sẽ có những chức năng và giao diện trình duyệt sau đây.



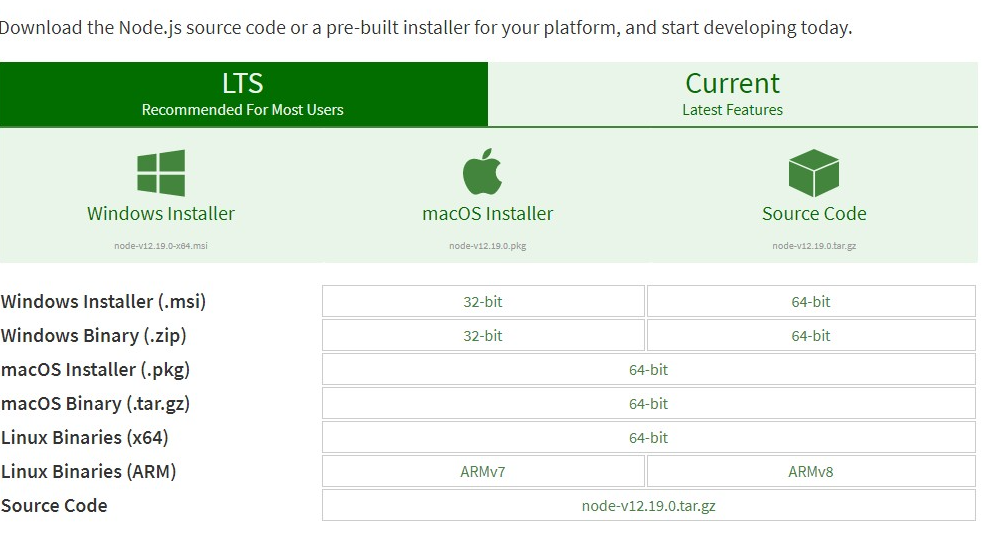
*Hình 1 Giao diện chính của app*

## **Chuẩn Bị Tài Nguyên**

* Ở đây mình sẽ cần 2 phần mềm giúp chúng ta code project reactjs đó là Node.js và Git-SCM ứng dụng giúp chúng ta tạo một project và cũng có thể sử dụng để úp code lên server.
* Đầu tiên các bạn cần cài đặt [Node.js](https://nodejs.org/en/) để tạo môi trường chạy React, việc cài đặt rất đơn giản và nhanh chóng.

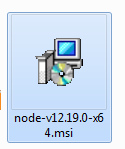
### **Tải nodejs**

Ta tải NodeJS tại link sau: <https://nodejs.org/en/download/>



*Hình 6 Tải nodeJS*

* Bạn hãy chọn phiên bản nodejs phù hợp với máy tính của mình và tải xuống máy tính của bạn .
* Sau khi download thành công bạn có một file trong thư mục (download) gần giống hình dưới đây.



*Hình 7 file nodeJS đã tải*

Giờ để cài đặt nó, ta chỉ cần click 2 lần chuột trái hoặc click chuột phải rồi chọn mục install

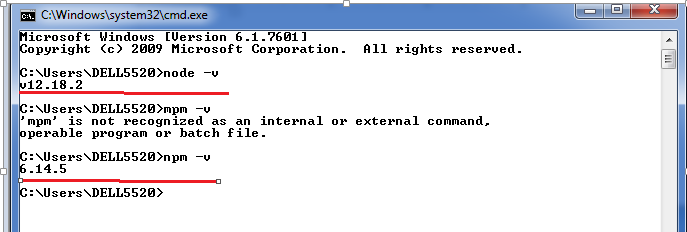
Ta Cài đặt NodeJS trên Windows rất đơn giản, chấp nhận các tùy chọn mặc định và nhấn "Next .. Next" cho tới bước cuối cùng.

**- Kiểm tra và cấu hình**

Hiện tại ở bước trên ta đã download và cài đặt thành công **NodeJS**. Và giờ bạn cần kiểm tra lại kết quả cài đặt và cấu hình **NodeJS**.

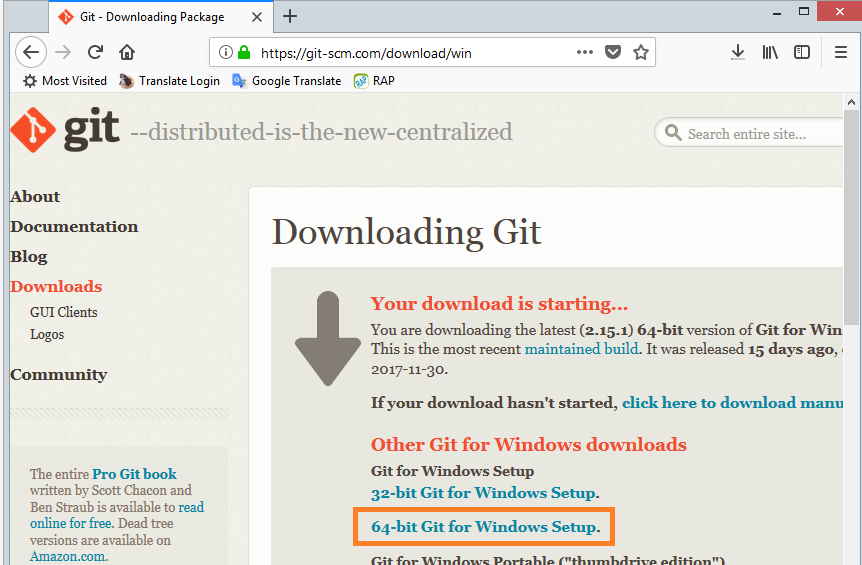
Mở cửa sổ **CMD**của máy tính và thực thi các lệnh sau để kiểm tra phiên bản của **NodeJS**và **NPM**:





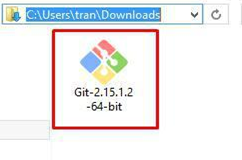
*Hình 9 Thực Hành*

### **Cài đặt Git-SCM**

* Điều thứ 2 để tạo project ReactJS là chúng ta cần phải tải GIT.
* Có nhiều cách để đến với trang chủ của Git, bạn có thể tìm kiếm trên trình duyệt hoạt bất cứ cách nào của bạn miễn sau có ứng dụng cho công việc làm demo.
* Link tải Git-scm: <https://git-scm.com/download/win>
* *Lưu ý các phiên bản sẽ có cập nhật các bạn nên xem tài liệu mới trên trình duyệt và mình chỉ giới thiệu cho bạn nguồn tài nguyên. Xem thêm ảnh minh họa bên dưới*.
* Ở đây cài theo hệ điều hành windows

*Hình 10 Trình duyệt chính của trang*

- Sau quá trình chờ tải về, ta sẽ có được một tệp thực thi trên máy tính có tên dạng  
**Git-2.15.12-64-bit.exe**, trong đó đây cũng là phiên bản có thể của bạn có thể xem hướng dẫn cài đặt ở trên trình duyệt để có hướng dẫn mới nhất:

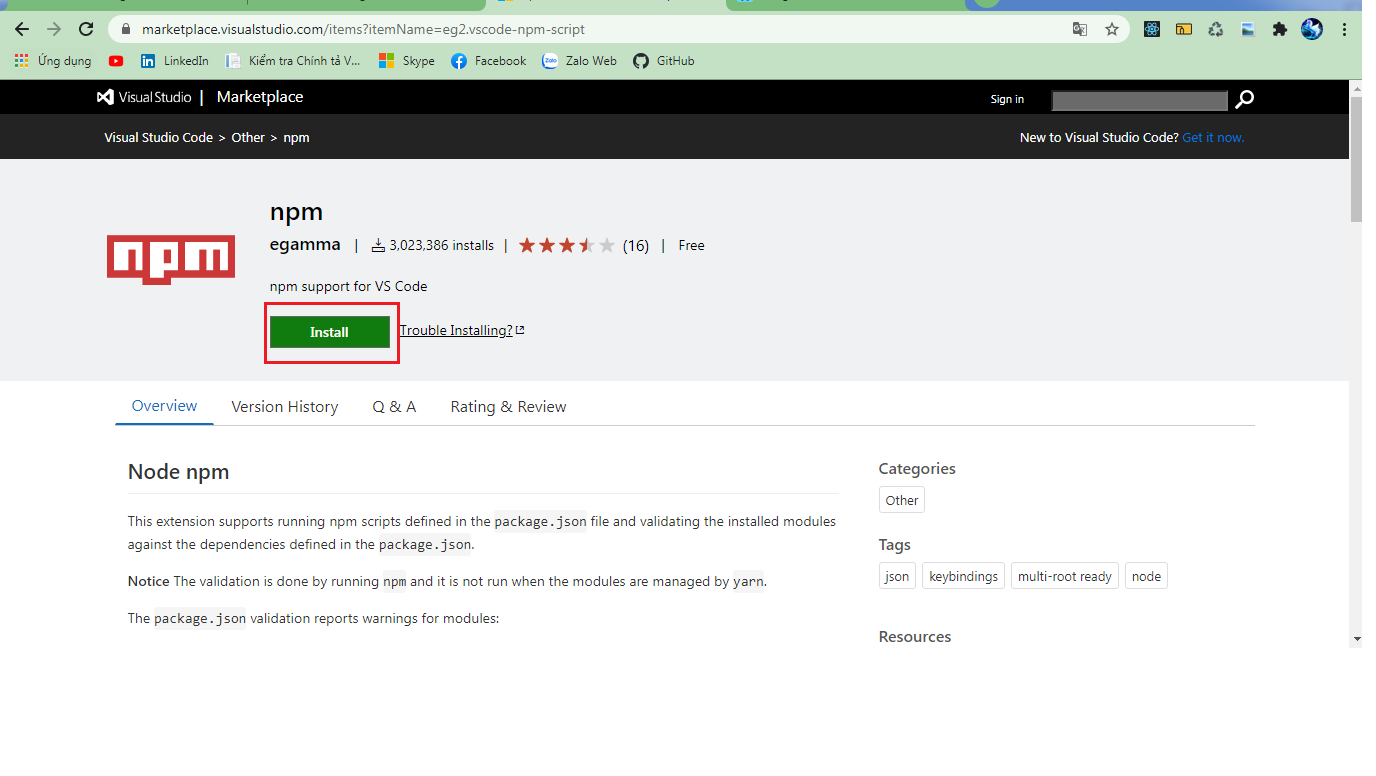
* Vào chỗ đã tải GIT-SCM xuống(Downloads) và cài đặt nó.

*Hình 11 Icon sau khi tải trên mạng về*

* Cách cài cũng có phần tương tự như node.js cứ next tới bước cuối cùng là được hoạt bạn nên xem trên mạng để có hướng dẫn cách tải mới nhất.

**Cài đặt npm support for VS code hướng dẫn trên Visual Studio Code**

* Cài Đặt npm support for VS code dùng để hỗ trợ code trong việc xác thực các modules và các package.json nhầm nhắc lỗi không chú ý và một nhiều công dụng khác.
* Bạn lên trình duyệt tìm kiếm npm support for VS code hoạt bạn có thể truy cập link dưới đây.
* Link npm support for VS code: <https://marketplace.visualstudio.com/items?itemName=eg2.vscode-npm-script>
* Sau khi tìm kiếm song bạn chỉ cần Install nó là được.



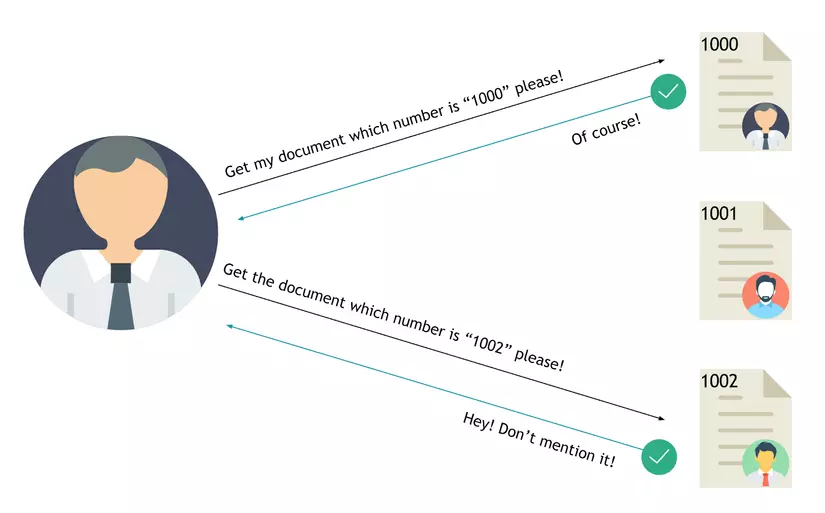
## *Hình 12 Trang chính npm support for VS code*

## **BÁO CÁO VỀ CÁC LỖI BẢO MẬT**

****I. IDOR****  (****Insecure Direct Object Reference****)

# 1. IDOR là gì?

- **IDOR** là cụm từ viết tắt của **Insecure Direct Object Reference** (Tham chiếu đối tượng trực tiếp không an toàn. Lỗ hổng này xảy ra khi chương trình cho phép người dùng truy cập tài nguyên (dữ liệu, file, thư mục, database..) một cách trực tiếp thông qua dữ liệu do người dùng cung cấp nhưng kém an toàn. Những dữ liệu này thường là những dữ liệu quan trọng, dữ liệu nhạy cảm hoặc không quyền sở hữu của hacker, những dữ liệu public và cho phép bất kì ai truy cập không thuộc phạm vi của lỗ hổng này.



**\* Ví dụ từ cuộc sống** : Phiếu gửi xe gắn máy giấy



- Có 2 xe máy là A và B, xe A có biển số là 1234, xe B có biển số là 12345, và cả 2 cùng gửi xe cùng 1 nơi, cả 2 người chủ xe đều được nhân viên phiếu giữ xe bằng giấy, phiếu A ghi mã số biển xe A (1234), và phiếu B ghi mã số biển xe B (12345). Sau 1 hồi thì người chủ xe A lấy phiếu gửi xe của mình ghi thêm vào từ 1234 thành 12345. Như vậy là chủ xe A có thể sử dụng phiếu đó để lấy xe B ra.

=> Người chủ xe B bị mất xe

\* **Ví dụ từ bảo mật Website :**

- Trong Website kinhtedothi.vn, ta vào 1 bài viết có đường dẫn là :

/kinhtedothi.vn/ngay-1310-chat-luong-khong-khi-tai-ha-noi-tiep-tuc-on-dinh-437875.html



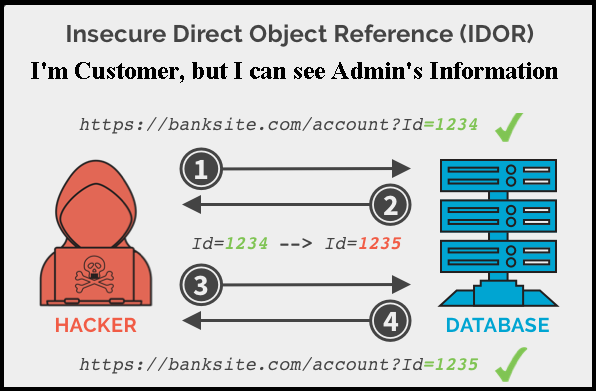
- Ta tiến hành sửa đoạn mã 437875 trong đường dẫn thành 437874 sau đó reload Website lại thì Website dẫn chúng ta quá 1 bài viết khác



=> Nếu có bài biết nào bị ẩn đi ta vẫn có thể vào xem được miễn là biết id của bài viết đó

# 2. Nguyên nhân

- Nguyên nhân sâu xa của lỗ hổng **IDOR** nằm ở cơ chế phân quyền và kiểm soát truy cập người dùng trên website. Việc website không phân quyền đúng cũng như kiểm soát chặt chẽ việc truy cập tài nguyên của user trên hệ thống dẫn đến việc truy cập trái phép từ kẻ tấn công. Việc sử dụng tham chiếu truy cập tới đối tượng là một cách làm tiện lợi nhưng nếu không phân quyền rõ ràng theo ma trận phân quyền thì việc này lại tiềm ẩn nguy cơ lộ dữ liệu rất nguy hiểm.



# 3. Hậu quả

- Người dùng có thể truy cập thông tin trái phép của người dùng khác

- Kẻ tấn công có thể thực hiện thay đổi hoặc xóa dữ liệu của người dùng khác.

=> Lỗ hổng này có mức độ nguy hiểm rất cao, gây hậu quả nghiêm trọng tới người dùng. Đặc biệt là các hệ thống có dữ liệu quan trọng của người dùng như: mạng xã hội, tài chính - ngân hàng, y tế..



# 4. Cách kiểm tra lỗi

 - Để kiểm tra xem ứng dụng của mình có bị lỗ hổng IDOR hay không các bạn có thể tham khảo phương pháp dưới đây.

- Kiểm tra tất cả các đoạn end point của website có truy cập tới tài nguyên, dữ liệu người dùng có dạng truy cập theo Object reference. Một số ví dụ như: user\_id=123, id=123, code=abc,profile\_id=123,folder\_id=111...

- Tiếp theo bạn thử thay đổi các giá trị truyền vào là các giá trị khác. Nếu truy cập được các thông tin nhạy cảm, thông tin quan trọng thì tức là Website đã bị lỗi IDOR.

# 5. Ngăn chặn

- Để ngăn chặn lỗi IDOR chúng ta chỉ cần ngăn chặn nguyên nhân gây ra lỗi:

## 5.1. Phân quyền chặt chẽ người dùng

- Nguyên nhân của lỗi **IDOR** nằm ở **cơ chế phân quyền**, vì vậy việc phát triển ứng dụng tuân thủ chặt chẽ việc phân quyền và kiểm tra quyền truy cập tới dữ liệu là điều quan trọng nhất.

- Luôn kiểm tra quyền truy cập của mỗi người dùng ở tất cả các request.

- Việc kiểm tra phân quyền luôn luôn cần thực hiện trên phía server, không kiểm tra ở phía client.

## 5.2. Hạn chế tham chiếu trực tiếp tới đối tượng

- Thay vì tham chiếu trực tiếp tới đối tượng như: **users?user\_id=123** chúng ta thực hiện hash giá trị id để thành một chuỗi ngẫu nhiên khó đoán :

**users?user\_id=F1244AD6A71A9C6C9E08BA6D819D119FBD7944D0.**

# 6. Tổng kết

- IDOR là một lỗ hổng nguy hiểm có thể đe dọa quyền riêng tư của người dùng và tính toàn vẹn của ứng dụng của bạn. Việc tuân thủ các cơ chế phân quyền chặt chẽ và kiểm tra ứng dụng kĩ sẽ giúp các bạn hạn chế được nguy cơ về lỗ hổng bảo mật IDOR, cũng như là tránh việc sai, mất dữ liệu của cá nhân

II. SQL Injection

# 1. SQL Injection là gì?

**SQL Injection** là một kỹ thuật lợi dụng những lỗ hổng về câu truy vấn của các ứng dụng. Được thực hiện bằng cách chèn thêm một đoạn [SQL](https://topdev.vn/blog/sql-la-gi/) để làm sai lệnh đi câu truy vấn ban đầu, từ đó có thể khai thác dữ liệu từ database. **SQL injection** có thể cho phép những kẻ tấn công thực hiện các thao tác như một người quản trị web, trên cơ sở dữ liệu của ứng dụng.

# 2. Ví dụ thực tiễn SQL Injection

Ví dụ, trong form đăng nhập, người dùng nhập dữ liệu, trong trường tìm kiếm người dùng nhập văn bản tìm kiếm, trong biểu mẫu lưu dữ liệu, người dùng nhập dữ liệu cần lưu. Tất cả các dữ liệu được chỉ định này đều đi vào cơ sở dữ liệu.

Thay vì nhập dữ liệu đúng, kẻ tấn công lợi dụng lỗ hổng để insert và thực thi các câu lệnh SQL bất hợp pháp để lấy dữ liệu của người dùng… SQL Injection được thực hiện với ngôn ngữ lập trình SQL. SQL (Structured Query Language) được sử dụng để quản lý dữ liệu được lưu trữ trong toàn bộ cơ sở dữ liệu.

**Tuy nhiên** ngày nay chứng ta thường làm việc trên những [framework](https://topdev.vn/blog/framework-la-gi/) hiện đại. Các framework đều đã được test cẩn thận để phòng tránh các lỗi, trong đó có SQL Injection.

# 3. Sự nguy hiểm của SQL Injection

* Hack tài khoản cá nhân.
* Ăn cắp hoặc sao chép dữ liệu của trang web hoặc hệ thống.
* Thay đổi dữ liệu nhạy cảm của hệ thống.
* Xóa dữ liệu nhạy cảm và quan trọng của hệ thống.
* Người dùng có thể đăng nhập vào ứng dụng với tư cách người dùng khác, ngay cả với tư cách quản trị viên.
* Người dùng có thể xem thông tin cá nhân thuộc về những người dùng khác, ví dụ chi tiết hồ sơ của người dùng khác, chi tiết giao dịch của họ,…
* Người dùng có thể sửa đổi cấu trúc của cơ sở dữ liệu, thậm chí xóa các bảng trong cơ sở dữ liệu ứng dụng.
* Người dùng có thể kiểm soát máy chủ cơ sở dữ liệu và thực thi lệnh theo ý muốn.

# 4. Ví dụ tấn công

Việc kiểm tra lỗ hổng này có thể được thực hiện rất dễ dàng. Đôi khi ta chỉ cần nhập ký hiệu ' hoặc " vào các trường được kiểm tra. Nếu nó trả về bất kỳ thông báo bất ngờ hoặc bất thường, thì ta có thể chắc chắn rằng SQL Injection khả thi cho trường đó.

Ví dụ: một Form đăng nhập như sau



Và đoạn code server xử lý của bạn:

if(isset($\_POST['username']) && isset($\_POST['password'])){

$sql = "SELECT \* FROM tbl\_user WHERE username='". $\_POST['username'] . "' AND password = '" .$\_POST['password'] ."'";

}

Nếu như người dùng không nhập bình thường nữa mà chẳng hạn như họ có thêm một dấu nháy ' hoặc " vào thì dòng code của bạn sẽ bị lỗi ngay. Hoặc họ có thể sửa thành một câu truy vấn luôn luôn đúng như sau.

SELECT \* FROM tbl\_user WHERE username = '' OR '1' = '1' and password = '' OR '1' = '1'

Hoặc chèn thêm một câu lệnh truy vấn phía sau:

**VD**:

SELECT \* FROM tbl\_user WHERE username = 'admin' and password = 'admin'; Drop table users;

**Các phần dễ bị tấn công**

Các phần dễ bị tấn công bao gồm:

* Form đăng nhập
* Form tìm kiếm
* Form nhận xét
* Bất kì trường lưu hoặc trường đầu vào của dữ liệu
* Liên kết của website

Cần lưu ý là trong khi thử nghiệm chống lại tấn công này là không thể chỉ kiểm tra một hoặc một vài trường bởi vì một trường có thể được bảo vệ chống lại SQL Injection, nhưng một trường khác thì không. Do đó, điều quan trọng là đừng quên kiểm tra tất cả các trường của trang web.

# 5. Cách giảm thiểu và phòng ngừa SQL Injection

Luôn kiểm tra kỹ các trường nhập dữ liệu và các bạn cần ràng buộc thật kỹ dữ liệu người dùng nhập vào.

Ví dụ:

//Thông thường

$id = $\_GET['id'];

//Ràng buộc

$id = isset($\_GET['id'])?(string)(int)$\_GET['id']:false;

Dùng [Regular Expression](https://topdev.vn/blog/regular-expression-la-gi/) để loại bỏ đi các ký tự lạ hoặc các ký tự không phải là số.

$id = isset($\_GET['id']) ? $\_GET['id'] : false;

$id = str\_replace('/[^0-9]/', '', $id);

Hoặc dùng các hàm có sẵn để giảm thiểu lỗi. Mỗi khi truy vấn thì mọi người nên sử dụng thêm hàm mysqli\_real\_escape\_string để chuyển đổi một chuỗi thành một query an toàn.

$id = isset($\_GET['id'])?(string)(int)$\_GET['id']:false;

$sql= 'SELECT \* FROM tbl\_user WHERE id= ' . mysqli\_real\_escape\_string($id);

Và lời khuyên cuối cùng là chúng ta nên dùng các Framework và hạn chế dùng code thuần tối đa nếu có thể. Framework luôn có cộng đồng hoặc các chuyên gia bảo mật giúp tìm lỗi và update liên tục, từ đó chúng ta có thể giảm bớt thời gian xử lý lỗi để tăng thời gian làm sản phẩm cũng là một điều hay.

III. Optimistic Locking

# 1. Cơ Chế hoạt động :

Optimistic Locking giải quyết vấn đề xung đột dữ liệu, khi có 2 hoặc nhiều user muốn thay đổi hay update cùng một dữ liệu nào đó, thì người thay đổi đầu tiên mới có thể update, còn những người thay đổi tiếp theo sẽ không thể thay đổi.

***Ví dụ :***

Có 2 giáo viên cùng update điểm của 1 sinh viên.

+ Giáo viên đầu tiên, sửa điểm từ 5 => 7.

+ Giáo viên thứ 2, sửa điểm từ 5 =>8.

Thì lập tức như bình thường thì 2 người cùng update vào một dòng dữ liệu sẽ gây ra conflict. Khi chúng ra áp dụng Optimistic Locking thì trong trường hợp này, chỉ cho phép 1 ngừoi update đầu tiên sẽ update thành công, còn ngừoi update sau sẽ không thành công.

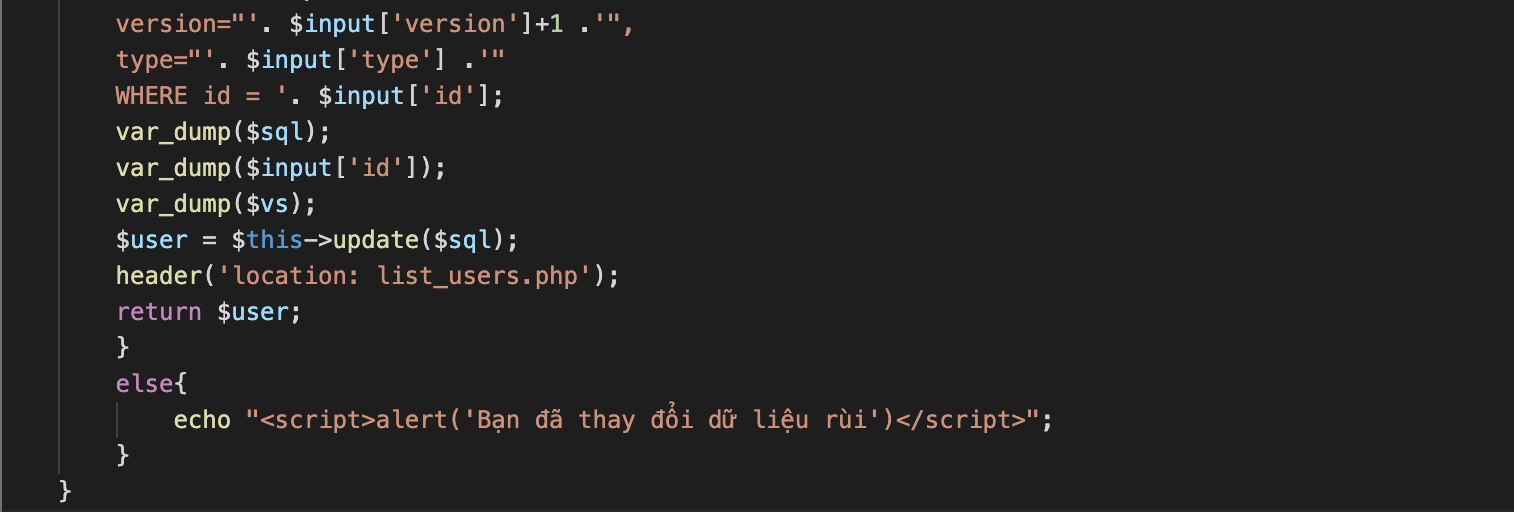
# 2. Giải Pháp :

Để giải quyết trường hợp conflict này thì trong mỗi trường dữ liệu em sẽ thêm một cột version, để kiểu INT.



Khi người dùng thay đổi dữ liệu thì em sẽ +1 cho version, và sau đó so sánh, nếu version hiện tại trùng với version trên database thì thay đổi dữ liệu thành công (Người update đầu tiên). Ngược lại là không thành công (Những người update sau) và trả về thông báo.





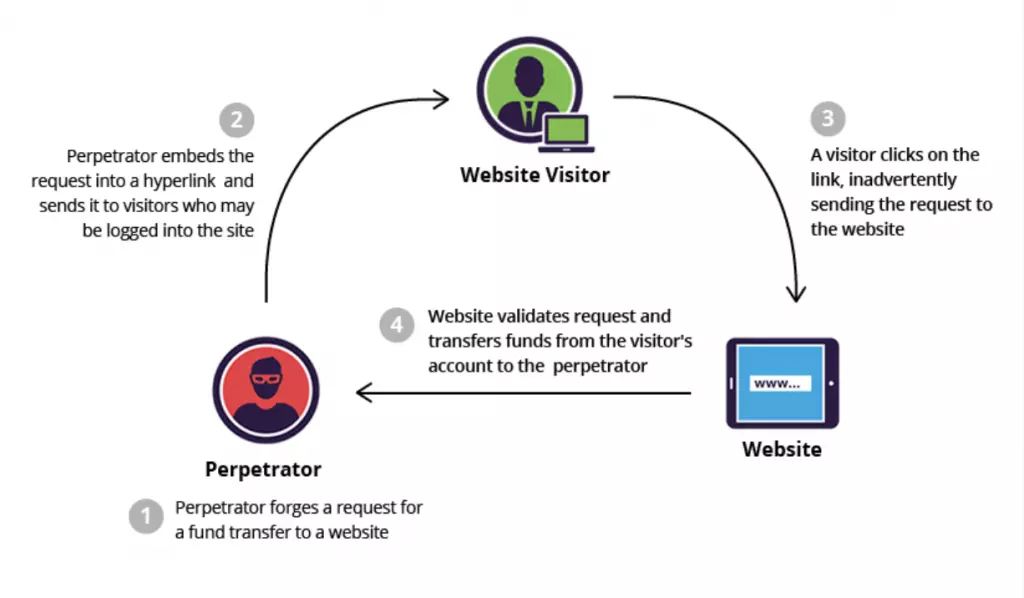
IV. CSRF (Cross-site request forgery)

# 1. Sự ra đời của CSRF

Năm 2001, Peter Watkins trong bài viết: "The Dangers of Allowing Users to Post Images" "trên bugtrag đã lần đầu tiên sử dụng thuật ngữ CSRF.

# 2. CSRF là gì?

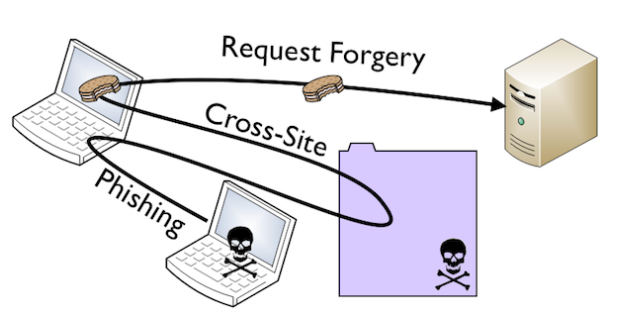
Cross-Site Request Forgery (CSRF / XSRF), là kiểu tấn công lừa người sử dụng thực hiện một hành động mà họ không mong muốn lên ứng dụng web, bằng chính quyền của người dùng đó. Sử dụng một số thủ thuật social engineering đơn giản (như gửi link qua email, chát), hacker có thể lừa người dùng thực hiện một số tác vụ lên ứng dụng web bị lỗi CSRF như: xóa bài, thêm người dùng, thay đổi email, thay đổi mật khẩu của victim ... Nếu người bị lừa là Admin, thì hacker hoàn toàn có thể chiếm quyền điều khiển ứng dụng web đó. Nói đơn giản là hacker có thể ngồi tại websiteAmà dụ dỗ người dùng tấn công site B và site C khác.



*Hình 1: Khái niệm CSRF*

# 3. Nguyên tắc hoạt động

1. Đầu tiên, người dùng phải đăng nhập vào trang mình cần (Tạm gọi là trang A).
2. Để dụ dỗ người dùng, hacker sẽ tạo ra một trang web độc.
3. Khi người dùng truy cập vào web độc này, một request sẽ được gửi đến trang A mà hacker muốn tấn công (thông qua form, img, …).
4. Do trong request này có đính kèm cookie của người dùng, trang web A đích sẽ **nhầm rằng đây là request do người dùng thực hiện**.
5. Hacker có thể mạo danh người dùng để làm các hành động như đổi mật khẩu, chuyển tiền, ….



*Hình 2: Nguyên tắc hoạt động*

* Ví dụ: Trong giao diện quản trị admin control panel của một ứng dụng web cho phép admin xóa bài viết bằng request sau?

Chuyện gì sẽ xảy ra nếu admin nhận được 1 email, đọc một comment có chèn đoạn code HTML sau:



Khi kết nối tới server để load ảnh tại link trên, trình duyệt đã tự động điền cookie, session vào request đó.

* Ví dụ: bạn chạy một dịch vụ blog vi mô cho phép người dùng của bạn tweet ý kiến của họ với nhau theo từng đoạn có kích thước 140 ký tự.

**Mal** là một tin tặc đã nhận thấy rằng các bài đăng trên dịch vụ của bạn được tạo bằng các yêu cầu GET. Điều này có nghĩa là tất cả thông tin được chuyển trong URL của yêu cầu HTTP.

**Mal** sửa đổi URL sau khi tạo để bao gồm một tải trọng độc hại. Bay giờ anh ta phải tìm cách nào đó để khiến nạn nhân truy cập vào URL trong trình duyệt của họ.



*Hình 3: URL của hacker*

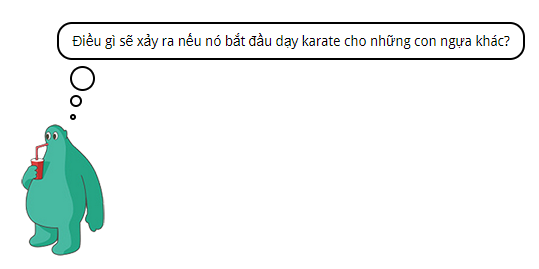
**Vic** là một trong những người dùng.

**Mal** có thể đoán địa chỉ email của **Vic** và gửi cho **Vic** một email với một liên kết rất hấp dẫn, trỏ đến URL được tạo.



*Hình 4: URL của hacker ở email người dùng*

**Vic** nhấp vào liên kết.



*Hình 5: Người dùng suy nghĩ*

Máy chủ của bạn diễn giải yêu cầu khi **Vic** đang viết một bài đăng và tạo một mục mới trên dòng thời gian của anh ấy. Đây không phải là hành động mà **Vic** dự định, nhưng có thể anh ấy không hoàn toàn để ý đến chuyện vừa xảy ra.



*Hình 6: Người dùng mắc bẫy*

Bài đăng được thiết kế đủ hấp dẫn để những người dùng khác trên trang web của bạn sẽ nhấp vào bài đăng đó. Khi họ làm như vậy, họ sẽ bị lừa giống như **Vic** đã từng.

Bây giờ bạn có một **con sâu** trên trang web của mình, vì mỗi người dùng nhấp vào liên kết sẽ mở ra một nhóm nạn nhân tiềm năng mới. Tin xấu!



*Hình 7: Sâu bị phát tán*

# 4. Các cách khai thác CSRF phổ biến

## 4.1. Khai thác qua các thẻ HTML

* IMG SRC



* SCRIPT SRC



* IFRAME SRC



## 4.2. Khai thác qua JavaScript (Thường dùng cho các request dạng POST)

* ‘Image’ Object



* ‘XMLHTTP’ Object

IE



* Mozilla



Ngoài ra còn rất nhiều kiểu khai thác khác nữa như sử dụng VBScript, Action Script, hay dùng các ngôn ngữ XML để thực hiện request.

CSRF ngoài việc khai thác trực tiếp qua trình duyệt, còn có thể khai thác qua Flash, Document file (PDF, DOC, EXEL ...), movie (WMA, AVI..), qua RSS, atom.

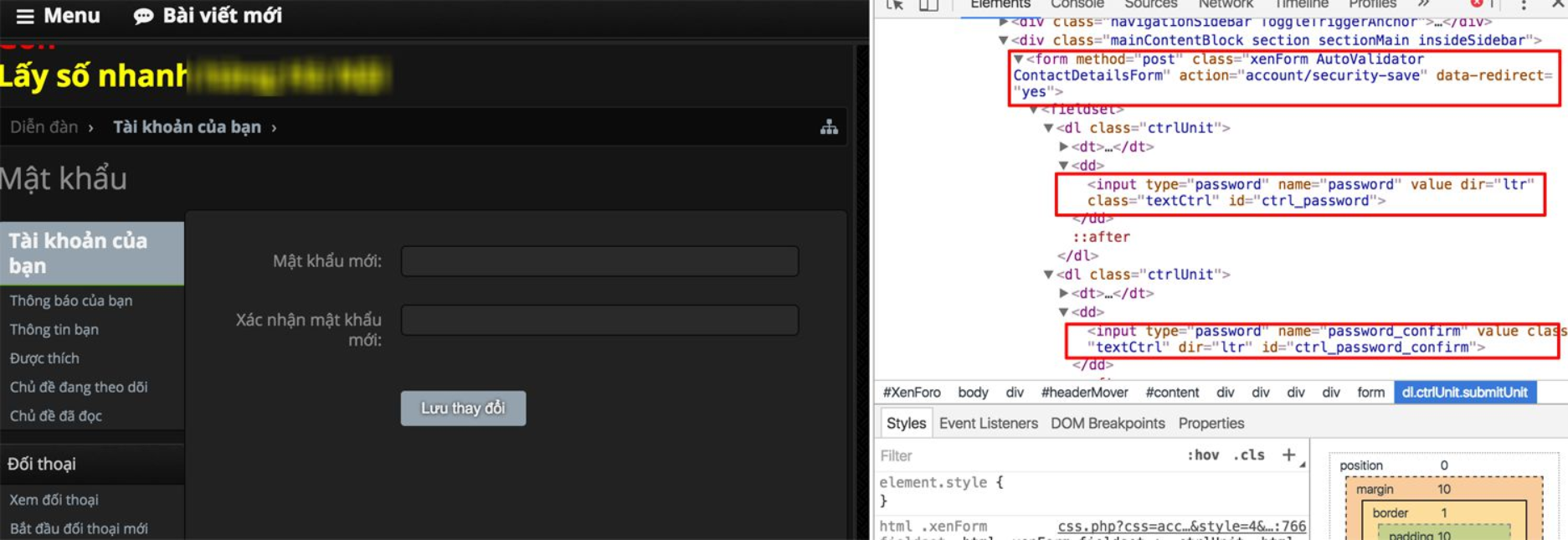
Trên thực tế, hacker sẽ sử dụng nhiều kỹ thuật tấn công kết hợp với nhau khiến hắn khó bị theo dấu hơn.

📄 **Cách Pentest CSRF**

Nếu website cho phép thực hiện các chức năng thông qua các request GET hoặc POST cố định thì có khả năng mắc lỗi CSRF. Tức là nếu mình replay lại được request POST hoặc GET trên thì có khả năng là website sẽ mắc lỗi.

## 4.3 Dùng form

Ta có HTML của form đổi mật khẩu thiên địa. Form này gồm 2 field là *password* và *password\_confirm*, submit tới [*http://thiendia.com/account/security-save*](http://thiendia.com/account/security-save)



*Hình 8: Form đổi mật khẩu*

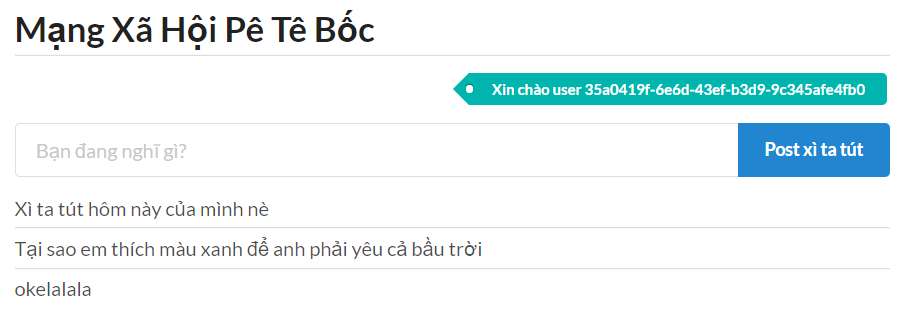
Hacker giả một trang web độc, giả vờ gửi cho người dùng xấu số. Trong trang web có một form ẩn với các giá trị tương tự form trên



*Hình 9: Hình ảnh khi ấn vào button*

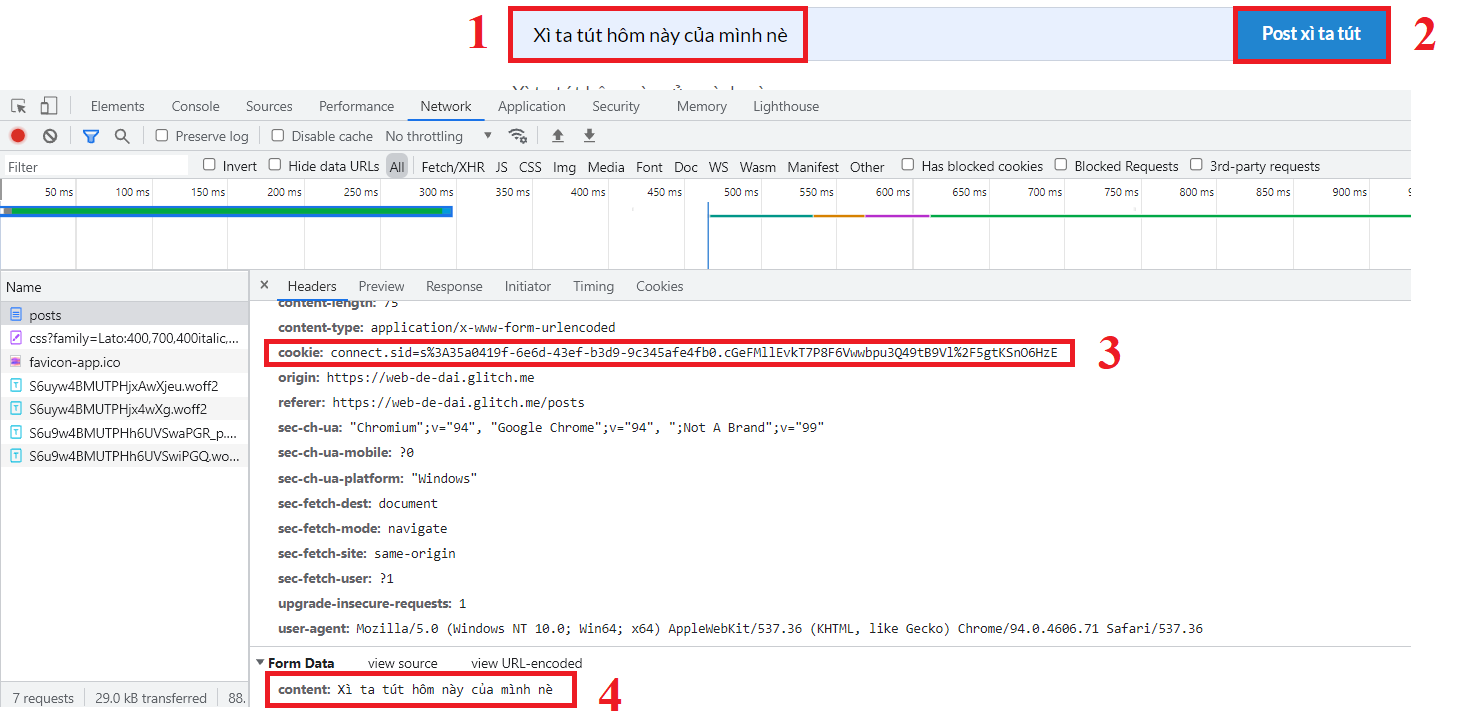
Người dùng ngây thơ mù IT, lỡ tay vào link và bấm vào button. Một request đổi password được gửi đến thiendia, kèm theo cookie account của người dùng. Thế là xong! hacker chỉ cần dùng email + mật khẩu mới là 123456 để đăng nhập vào account của người dùng xấu số.

Ví dụ: Sau đây là một trang mạng xã hội Pê Tê Bốc minh hoạ việc đăng status



*Hình 10: Hình ảnh minh hoạ mạng xả hội*

Sau khi nhập thông tin status và ấn post thì nó sẽ gửi 1 request về server và trong đó có một cookie để biết bạn là ai, và dựa vào cookie và content thì server sẽ ghi nội dung vào database

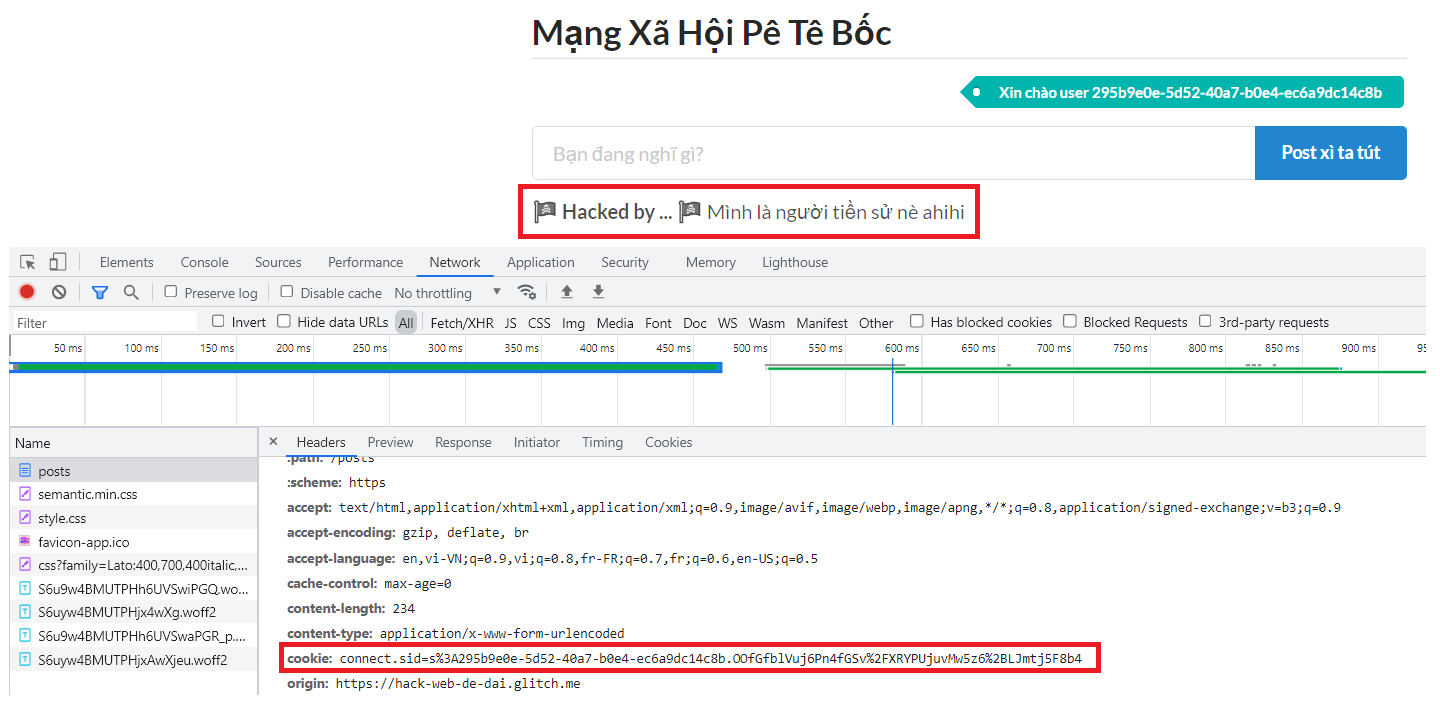


Giả sử hacker biết bạn sử dụng mạng xã hội này, và có một trang web được hacker tạo ra và gửi mail cho bạn. Và trang web có một form ẩn bên trong bạn không thể thấy.



*Hình 11: Hình ảnh minh hoạ form ẩn*

Khi người dùng submit thì bên trang mạng xã hội của bạn sẽ tự động đăng bài theo content mà hacker đã làm trước trong khi bạn không hề đăng tin đó. Khi submit thì nó sẽ gửi request giả và có cookie của bạn nên hacker có thể giả mạo bạn đăng bài.



*Hình 12: Hình ảnh minh hoạ nạn nhân bị hacker giả mạo đăng post*

Và trên thực tế có rất nhiều website cũng đang dính lỗ hổng bảo mật csrf.

# 5. Rủi ro



*Hình 13: Rủi ro*

Bất kỳ chức năng nào mà người dùng của bạn có thể cố tình thực hiện đều có thể bị lừa thực hiện một cách vô tình bằng cách sử dụng CSRF. Như chúng ta đã thấy trong ví dụ, trong hầu hết các trường hợp ác ý, các cuộc tấn công CSRF có thể tự lây lan như một con sâu.

Các cuộc tấn công CSRF trong quá khứ đã được sử dụng để:

* Đánh cắp dữ liệu bí mật
* Phát tán sâu trên phương tiện truyền thông xã hội
* Cài đặt phần mềm độc hại trên điện thoại di động

# 6. Phòng chống CSRF

## 6.1. Phía User

Để phòng tránh trở thành nạn nhân của các cuộc tấn công CSRF, người dùng internet nên thực hiện một số lưu ý sau:

* Nên thoát khỏi các website quan trong: Tài khoản ngân hàng, thanh toán trực tuyến, các mạng xã hội, gmail, yahoo… khi đã thực hiện xong giao dịch hay các công việc cần làm. (Check – email, checkin… )
* Không nên click vào các đường dẫn mà bạn nhận được qua email, qua facebook… khi bạn đưa chuột qua 1 đường dẫn, phía dưới bên trái của trình duyệt thường có địa chỉ website đích, bạn nên lưu ý để đến đúng trang mình muốn.
* Không lưu các thông tin về mật khẩu tại trình duyệt của mình (không nên chọn các phương thức “đăng nhập lần sau”, “lưu mật khẩu”…)
* Trong quá trình thực hiện giao dịch hay vào các website quan trọng không nên vào các website khác, có thể chứa các mã khai thác của kẻ tấn công.

## 6.2. Phía server

### 6.2.1. Synchronizer Token Pattern:

Tạo thêm 1 trường ẩn có giá trị là token



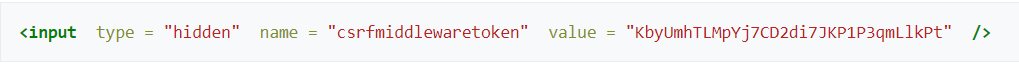
### 6.2.2. Disclosure of Token in URL

Cách này đưa token lên URL, bạn vào các website mà có đường dẫn loằng ngoằng toàn số, chính là loại này. Kiểu này có đặc điểm là không đẹp, khó SEO.

### 6.2.3. Mẫu mã thông báo đồng bộ hóa

[Mẫu mã thông báo đồng bộ hóa](https://en.wikipedia.org/w/index.php?title=Synchronizer_token_pattern&action=edit&redlink=1) (STP) là một kỹ thuật trong đó mã thông báo, giá trị bí mật và duy nhất cho mỗi yêu cầu, được ứng dụng web nhúng vào tất cả các biểu mẫu HTML và được xác minh ở phía máy chủ. Mã thông báo có thể được tạo bằng bất kỳ phương pháp nào đảm bảo tính không thể đoán trước và tính duy nhất (ví dụ: sử dụng [chuỗi băm](https://en.wikipedia.org/wiki/Hash_chain) của hạt giống ngẫu nhiên). Do đó, kẻ tấn công không thể đặt mã thông báo chính xác trong các yêu cầu của họ để xác thực chúng.

Ví dụ về STP  ở dạng HTML:



STP là tương thích nhất vì nó chỉ dựa trên HTML, nhưng có một số phức tạp ở phía máy chủ, do gánh nặng liên quan đến việc kiểm tra tính hợp lệ của mã thông báo trên mỗi yêu cầu. Vì mã thông báo là duy nhất và không thể đoán trước, nó cũng thực thi chuỗi sự kiện thích hợp (ví dụ: màn hình 1, sau đó 2, rồi 3) làm nảy sinh vấn đề về khả năng sử dụng (ví dụ: người dùng mở nhiều tab). Nó có thể được thư giãn bằng cách sử dụng mã thông báo CSRF mỗi phiên thay vì mã thông báo CSRF theo yêu cầu.

### 6.2.4. Mã thông báo cookie-to-header

Các ứng dụng web sử dụng [JavaScript](https://en.wikipedia.org/wiki/JavaScript) cho phần lớn các hoạt động của chúng có thể sử dụng kỹ thuật chống CSRF sau:

* Trong lần truy cập đầu tiên mà không có phiên máy chủ được liên kết, ứng dụng web đặt một cookie được xác định phạm vi phù hợp để nó không được cung cấp trong các yêu cầu nguồn gốc chéo. Cookie thường chứa một mã thông báo ngẫu nhiên có thể giữ nguyên cho đến thời gian tồn tại của phiên web

Set-Cookie: \_\_Host-csrf\_token = i8XNjC4b8KVok4uw5RftR38Wgp2BFwql; Expires = Thu, 23-07-2015 10:25:33 GMT; Tuổi tối đa = 31449600; Đường dẫn = /; SameSite = Lax; Chắc chắn

* [JavaScript](https://en.wikipedia.org/wiki/JavaScript) hoạt động ở phía máy khách đọc giá trị của nó và sao chép nó vào một [tiêu đề HTTP](https://en.wikipedia.org/wiki/HTTP_header) tùy chỉnh được gửi cùng với mỗi yêu cầu giao dịch

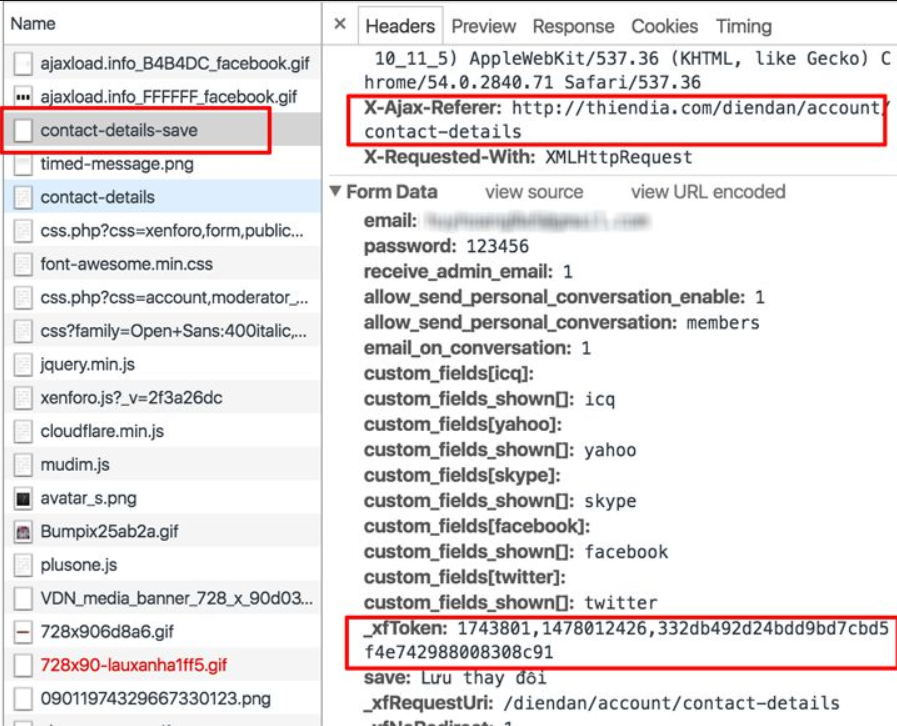
X-Csrf-Token: i8XNjC4b8KVok4uw5RftR38Wgp2BFwql

* Máy chủ xác nhận sự hiện diện và tính toàn vẹn của mã thông báo

Tính bảo mật của kỹ thuật này dựa trên giả định rằng chỉ [JavaScript](https://en.wikipedia.org/wiki/JavaScript) chạy ở phía máy khách của kết nối HTTPS tới máy chủ đã đặt cookie ban đầu mới có thể đọc giá trị của cookie. JavaScript chạy từ một tệp hoặc email giả mạo sẽ không thể đọc thành công giá trị cookie để sao chép vào tiêu đề tùy chỉnh. Mặc dù cookiecsrf-token sẽ được tự động gửi cùng với yêu cầu giả mạo, máy chủ vẫn mong đợi một tiêu đềX-Csrf-Token hợp lệ.

Bản thân mã thông báo CSRF phải là duy nhất và không thể đoán trước. Nó có thể được tạo ngẫu nhiên hoặc nó có thể được lấy từ [mã thông báo phiên](https://en.wikipedia.org/wiki/Session_cookie) sử dụng [HMAC](https://en.wikipedia.org/wiki/HMAC):

csrf\_token = HMAC (session\_token, application\_secret)



*Hình 14: Ảnh minh hoạ của thiên địa, trang này có CSRF token*

Cookie mã thông báo CSRF không được có cờ [httpOnly](https://en.wikipedia.org/wiki/HttpOnly) , vì nó được thiết kế để [JavaScript](https://en.wikipedia.org/wiki/JavaScript) đọc .

Kỹ thuật này được thực hiện bởi nhiều framework hiện đại, chẳng hạn như [Django](https://en.wikipedia.org/wiki/Django_(web_framework)) và [AngularJS](https://en.wikipedia.org/wiki/AngularJS) . Vì mã thông báo không đổi trong toàn bộ phiên người dùng, nó hoạt động tốt với [các](https://en.wikipedia.org/wiki/AJAX) ứng dụng [AJAX](https://en.wikipedia.org/wiki/AJAX) , nhưng không thực thi chuỗi sự kiện trong ứng dụng web.

Sự bảo vệ được cung cấp bởi kỹ thuật này có thể bị cản trở nếu trang web mục tiêu vô hiệu hóa[chính sách cùng nguồn gốc](https://en.wikipedia.org/wiki/Same-origin_policy) của nó bằng cách sử dụng một trong các kỹ thuật sau:

* tệp clientaccesspolicy.xml cấp quyền truy cập ngoài ý muốn vào các điều khiển Silverlight
* tệp crossdomain.xml cấp quyền truy cập ngoài ý muốn vào phim Flash

### 6.2.5. Gửi hai lần Cookie

Tương tự như phương pháp tiếp cận cookie đến tiêu đề, nhưng không liên quan đến JavaScript, một trang web có thể đặt mã thông báo CSRF làm cookie và cũng chèn nó dưới dạng trường ẩn trong mỗi biểu mẫu HTML. Khi biểu mẫu được gửi, trang web có thể kiểm tra xem mã thông báo cookie có khớp với mã thông báo biểu mẫu hay không. Chính sách cùng nguồn gốc ngăn kẻ tấn công đọc hoặc đặt cookie trên miền đích, vì vậy chúng không thể đặt mã thông báo hợp lệ ở dạng được tạo thủ công.

Ưu điểm của kỹ thuật này so với mẫu Synchronizer là mã thông báo không cần phải được lưu trữ trên máy chủ.

### 6.2.6. Thuộc tính cookie SameSite

Một thuộc tính "SameSite" bổ sung có thể được đưa vào khi máy chủ đặt cookie, hướng dẫn trình duyệt về việc có nên đính kèm cookie vào các yêu cầu trên nhiều trang web hay không. Nếu thuộc tính này được đặt thành "nghiêm ngặt", thì cookie sẽ chỉ được gửi theo các yêu cầu trên cùng một trang web, làm cho CSRF không hiệu quả. Tuy nhiên, điều này yêu cầu trình duyệt nhận ra và triển khai chính xác thuộc tính, đồng thời yêu cầu cookie phải có cờ "Bảo mật".

# 7. Lưu ý

Tất nhiên, trong bài chỉ là ví dụ. Theo nguyên tắc, request GET chỉ được dùng để truy cập dữ liệu, không được dùng để thực hiện các hoạt động thay đổi dữ liệu như edit/delete.

Các ngân hàng thường bảo mật rất kỹ bằng cách set cookie có thời gian sống khá ngắn, không cho phép chuyển tiền mà không có code OTP v…v. Ngoài ra, thiendia cũng có các biện pháp bảo mật khá tốt (xem phía dưới) nên các bạn không dùng cách này để chôm account của bạn bè được đâu, đừng thử nhé!



*Hình 15: HÌnh ảnh minh hoạ bảo mật của thiendia*

Tuy nhiên, ngày xưa, khi các lỗ hổng bảo mật còn chưa phổ biến thì đây là chính là cách mà hacker sử dụng. Chỉ cần post 1 tấm ảnh chứa đường dẫn như trên lên 1 forum nào đó, sẽ có vô số người dính bẫy khi truy cập vào forum đó.

V. Lỗi bảo Mật XSS (Cross Site Scripting)

# 1. Khái Niệm về Cross-site scripting (XSS)

XSS là gì: Cross-site scripting (XSS) là một kiểu tấn công bảo mật trong đó kẻ tấn công đưa các tập lệnh độc hại vào phần nội dung của các trang web đáng tin cậy khác. Tấn công Cross-site scripting xảy ra khi một nguồn không đáng tin cậy được phép đưa code của chính nó vào một ứng dụng web và mã độc đó được bao gồm trong nội dung gửi đến trình duyệt của nạn nhân.

XSS là một trong những lỗ hổng ứng dụng web phổ biến nhất và xảy ra khi một ứng dụng web sử dụng đầu vào từ người dùng không được xác thực hoặc không được mã hóa trong đầu ra mà nó tạo ra.

Bằng cách tận dụng XSS, kẻ tấn công không nhắm trực tiếp vào nạn nhân. Thay vào đó, kẻ tấn công sẽ khai thác lỗ hổng trong trang web hoặc ứng dụng web mà nạn nhân sẽ truy cập.

Mặc dù XSS có thể tận dụng lợi thế trong VBScript, ActiveX và Flash (hiện được coi là cũ hoặc thậm chí lỗi thời), nhưng phổ biến nhất vẫn là JavaScript - chủ yếu vì JavaScript là nền tảng cho hầu hết các trải nghiệm duyệt web.

Cross-Site Scripting (XSS) là một trong những kĩ thuật tấn công phổ biến nhất hiện nay, được liệt vào danh sách những kỹ thuật tấn công nguy hiểm nhất với ứng dụng web.

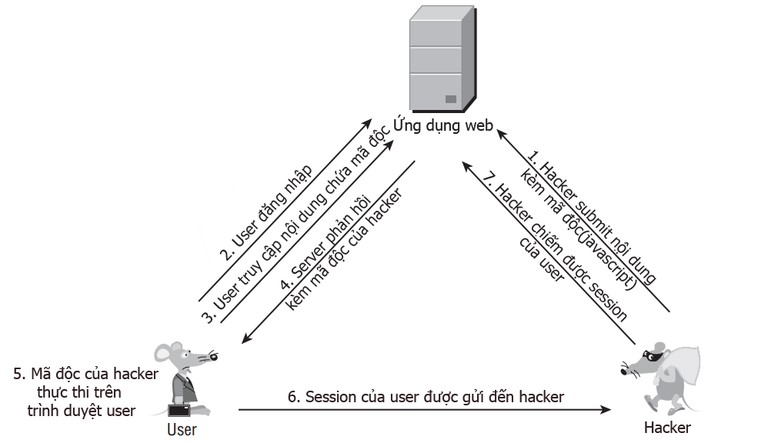
# 2. Mô tả ví dụ thực tiễn, nguy hiểm của lỗi bảo mật phụ trách:

# http://securitydaily.net/wp-content/uploads/2014/04/dombased-xss-3.png

Được thực hiện trên máy của nạn nhân. Nạn nhân chính là người dùng

Các lệnh độc hại sẽ được thực thi mỗi khi người dùng gọi chức năng đã bị tấn công

**Nguy hiểm** : Khi bị tấn công bằng cách này người bị hại có thể mất những tài khoản, những thông tin, thậm chí là những tài liệu đang nắm giữ



# 3. Một số hình thức của Cross Site Scripting như sau:

1. Cross Site Scripting sẽ diễn ra tại tập tin độc hại trên máy khách.
2. Giả mạo, làm sai form khi người dùng cần nhập thông tin (ví dụ như nhập thông tin đăng nhập hoặc nhấp vào liên kết độc hại).
3. Trên các vùng có hiển thị banner quảng cáo.
4. Gửi một email giả mạo đến người dùng yêu cầu nhấn vào link để xác thực,…

# 4. Ngăn ngừa, lọc và vá các lỗ hổng XSS

Đối với người dùng thì ta cần phải cân nhắc khi click vào link, kiểm tra các link thật kĩ trước khi click. Đặc biệt trên mạng xã hội. Cần cảnh giác trước khi click vào xem 1 link nào đó được chia sẻ.

Đối với người thiết kế và phát triển ứng dụng web Với những dữ liệu, thông tin nhập của người dùng, người thiết kế và phát triển ứng dụng web cần thực hiện vài bước cơ bản sau:

- Chỉ chấp nhận những dữ liệu hợp lệ.

- Từ chối nhận các dữ liệu hỏng.

- Liên tục kiểm tra và lọc dữ liệu.

- Tạo ra danh sách những thẻ HTML được phép sử dụng, xóa bỏ các thẻ, coi đoạn script đó như là đoạn trích dẫn lỗi.

Lọc dấu nháy đơn hay nháy kép

Xóa các kí tự“>”, “<” hoặc Output Endcoding các kí tự đó

Vẫn cho phép nhập dữ liệu đặc biệt nhưng chúng sẽ đc mã hóa theo chuẩn riêng

VD: sử dụng hàm Strip\_tags hoặc htmlentities để chuyển tất cả những ký tự nguy hiểm thành các ký tự HTML an toàn..

Ví dụ: $keyword = "<script>alert('XSS attack!');</script>";

// DO NOT USE $keyword without using htmlentities()

echo "Not found any result for keyword: " . htmlentities($str);