Shape, square

Description automatically generated

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**🕯✡🕮🕮✡🕯**

|  |  |
| --- | --- |
|  | |
|  |  |

A picture containing text

Description automatically generatedLogo

Description automatically generated

**BÁO CÁO ĐỒ ÁN CUỐI KỲ**

**BẢO MẬT WEB**

**BẢO MẬT HỆ THỐNG WEBSITE KINH DOANH QUÀ LƯU NIỆM**

**GVHD: Ths. Lê Thị Minh Châu**

**Nhóm sinh viên thực hiện :**

Chu Trần Minh Đức 20110266

Lê Anh Kiệt 20110664

Huỳnh Công Quyền 20110258

Lương Xuân Thắng 20110724

**Tp. Hồ Chí Minh, tháng 5 năm 2023**

**Nhận xét của giảng viên**

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

Tháng 5 năm 2023

**Giáo viên chấm điểm**

# LỜI CẢM ƠN

Lời nói đầu tiên, nhóm thực hiện xin được gửi đến cô **Lê Thị Minh Châu** – giảng viên bộ môn lời cảm ơn chân thành và sâu sắc nhất.

Nhóm thực hiện xin cảm ơn sự quan tâm và giúp đỡ tận tình của cô trong suốt quá trình giảng dạy. Cảm ơn cô đã luôn giải đáp những thắc mắc cũng như đưa ra những nhận xét, góp ý giúp nhóm thực hiện cải thiện chất lượng công việc của nhóm.

Vì khả năng còn hạn chế nên trong quá trình thực hiện báo cáo không tránh khỏi sai sót, kính mong nhận được những ý kiến đóng góp từ cô để nhóm có thể cải thiện hơn sau này.

Nhóm thực hiện xin chân thành cảm ơn.

**MỤC LỤC**

# PHÂN CHIA CÔNG VIỆC

|  |  |  |  |
| --- | --- | --- | --- |
| Họ và tên | MSSV | Phân công | Mức độ hoàn thành |
| Chu Trần Minh Đức | 20110266 |  |  |
| Lê Anh Kiệt | 20110664 |  |  |
| Huỳnh Công Quyền | 20110258 |  |  |
| Lương Xuân Thắng | 20110724 |  |  |

# Phần 1: Giới thiệu project

## 1. Giới thiệu

### 1.1. Mục đích

MeowMeow shop là một hệ thống webside dùng để bán các loại mặt hàng lưu niệm cho phép khách hàng có thể thỏa thích lựa chọn và mua sắm online. MeowMeow shop sẽ cập nhật nhiều loại mặt hàng hot nhất, được ưa chuộng nhất và cập nhật nhiều thông tin mới về các mặt hàng sản phẩm để khách hàng nắm bắt đầy đủ nhất. MeowMeow shop cho phép khách hàng tạo tài khoản để thanh toán sản phẩm. Đối với khách hàng đã thanh toán xong MeowMeow shop sẽ cập nhật trạng thái đơn hàng cho khách hàng và liên hệ cho khách hàng.

### 1.2. Đối tượng của đề tài

* Khách hàng.
* Quản lý.
* Nhân viên.

### 1.3.Chức năng

#### 1.2.1. Khách hàng

* Đăng ký tài khoản: Người dùng tiến hành đăng ký tài khoản, thông tin cần cung cấp để đăng ký bao gồm: tên đăng nhập, mật khẩu, tên người dùng, gmail, số điện thoại, Địa chỉ, giới tính, ngày tháng năm sinh.
* Đăng nhập tài khoản: Người dùng nhập tên tài khoản và mật khẩu để đăng nhập vào tài khoản.
* Tìm kiếm sản phẩm: Người dùng có thể tìm kiếm các loại sản phẩm dựa vào tên sản phẩm.
* Cập nhập giỏ hàng: Người dùng có thể thêm, xóa, sửa các sản phẩm trong giỏ hàng.
* Mua hàng: Người dùng có thể chọn các sản phẩm trong giỏ hàng và hoàn thành
* Thay đổi thông tin tài khoản: Người dùng có thể thay đổi thông tin tài khoản cá nhân.
* Xem danh sách đơn hàng: người dùng có thể xem các đơn hàng đã đặt trên hệ thống website.
* Xem chi tiết sản phẩm: người dùng có thể xem thông tin chi tiết của sản phẩm.

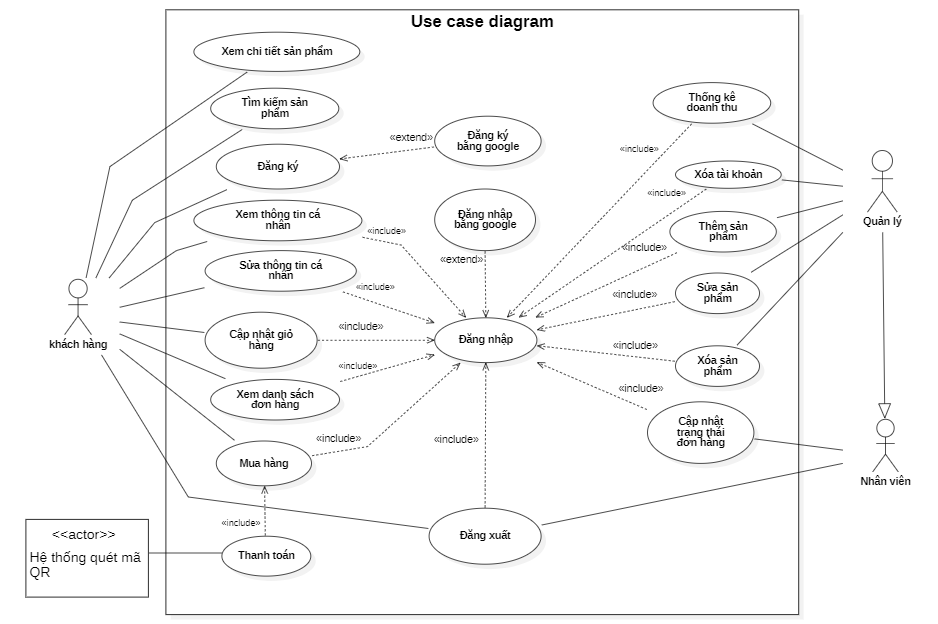
#### 1.2.2. Nhân viên

* Cập nhật trạng thái đơn hàng: Nhân viên có thể tiếp nhận đơn hàng, sửa đổi trạng thái đơn hàng, chuẩn bị hàng để gửi.

#### 1.2.3. Quản lý

* Quản lý tài khoản: Quản lý có thể xem, xóa các thông tin liên quan đến tài khoản.
* Quản lý sản phẩm: Quản lý có thể xem, thêm, xóa, sửa sản các thông tin liên quan đến sản phẩm.
* Thống kê doanh thu: Quản lý có thể thống kê doanh thu theo ngày, tháng, năm.

## 2. Lược đồ usecase

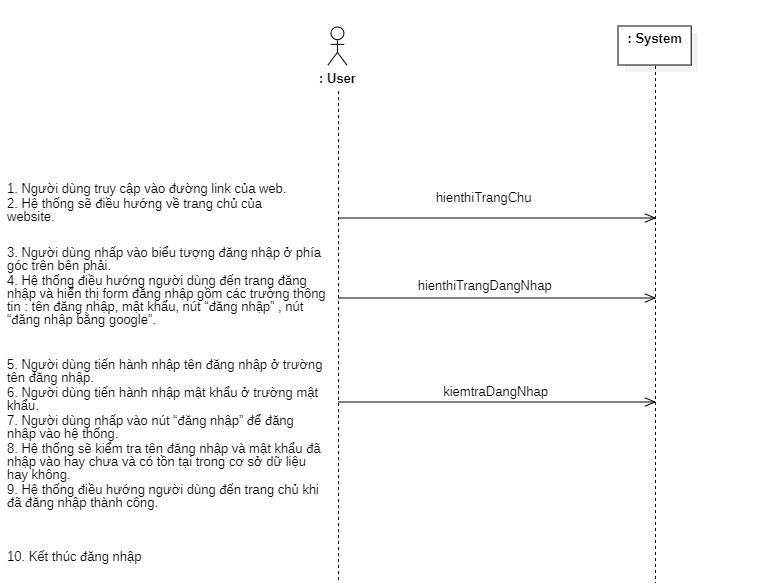


## 3. Lược đồ quan hệ ERD

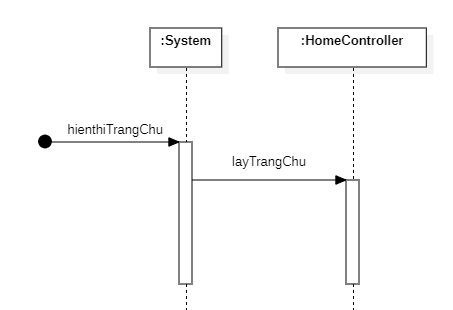


## 5. Lược đồ trình tự (Sequence Diagram – SD)

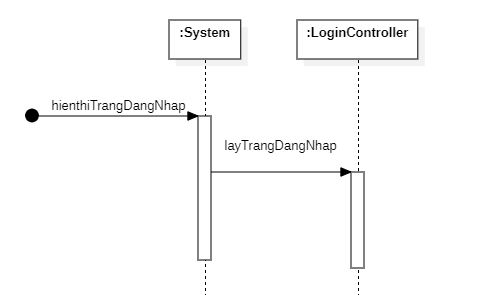
### 5.1. SSD đăng nhập



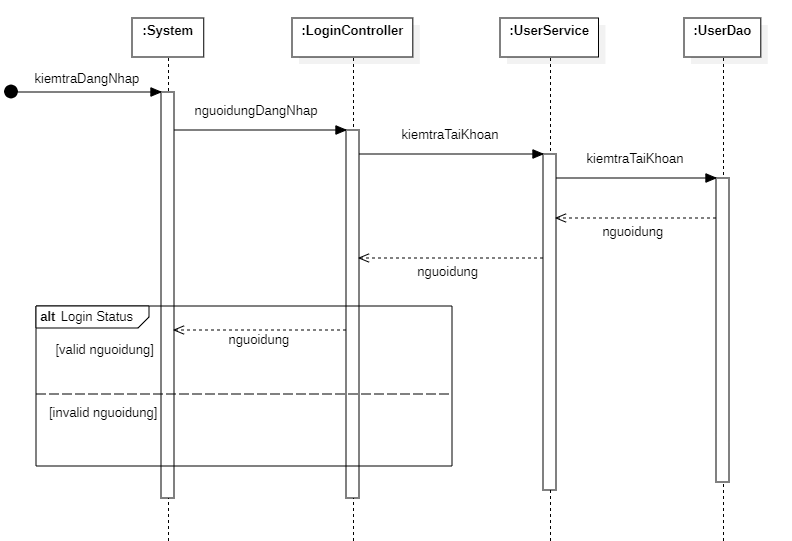
#### 5.1.1. SD hienthiTrangChu



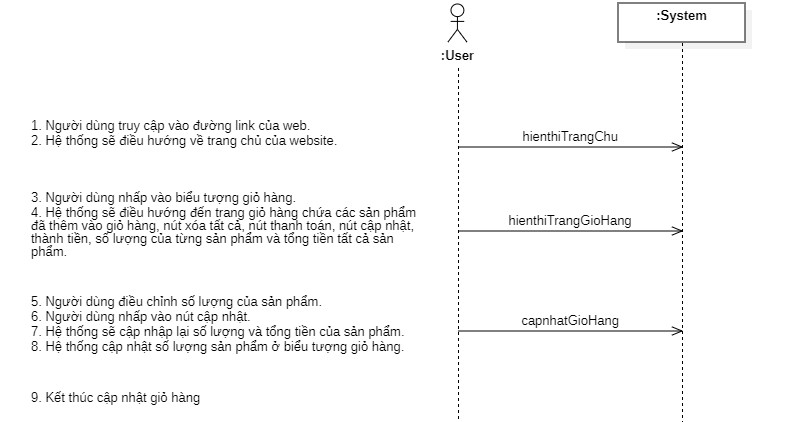
#### 5.1.2. SD hienthiTrangDangNhap



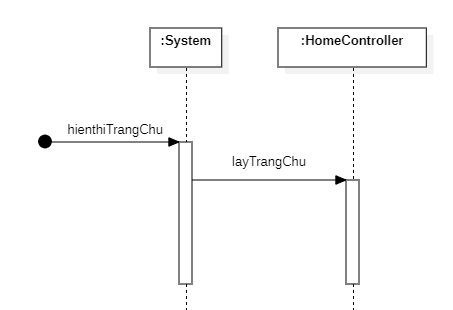
#### 5.1.3. SD kiemtraDangNhap



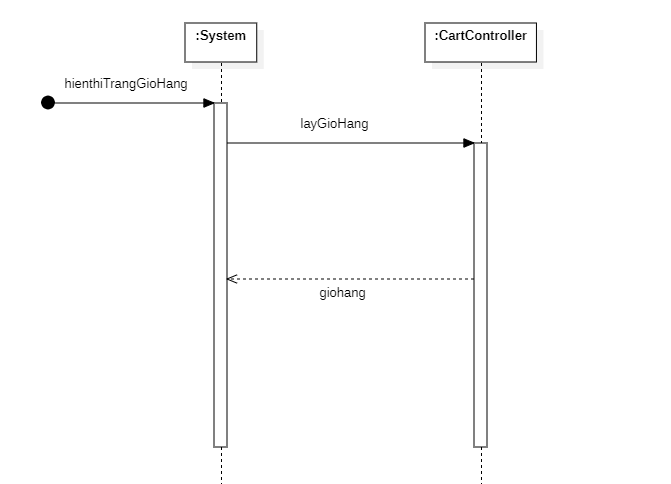
### 5.2. SSD cập nhật giỏ hàng



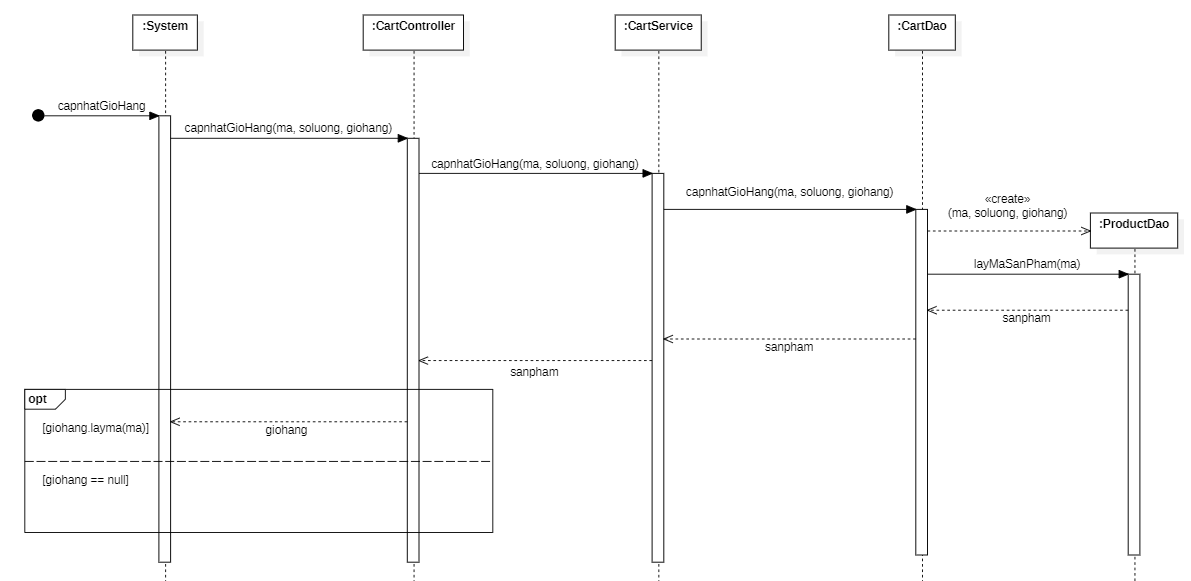
#### 5.2.1. SD hienthiTrangChu



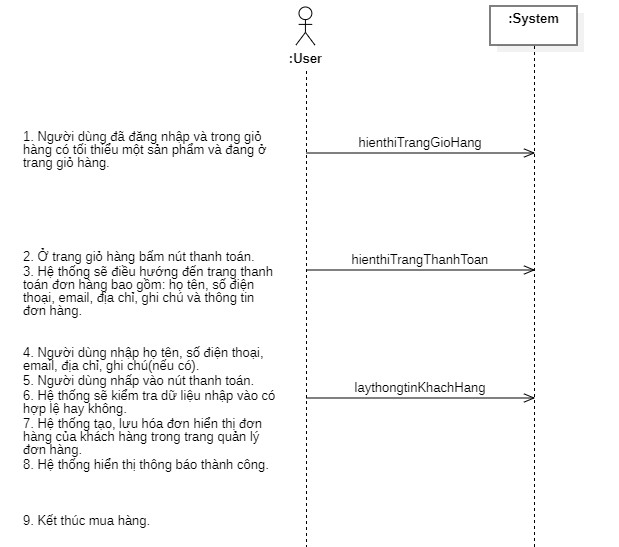
#### 5.2.2. SD hienthiTrangGioHang



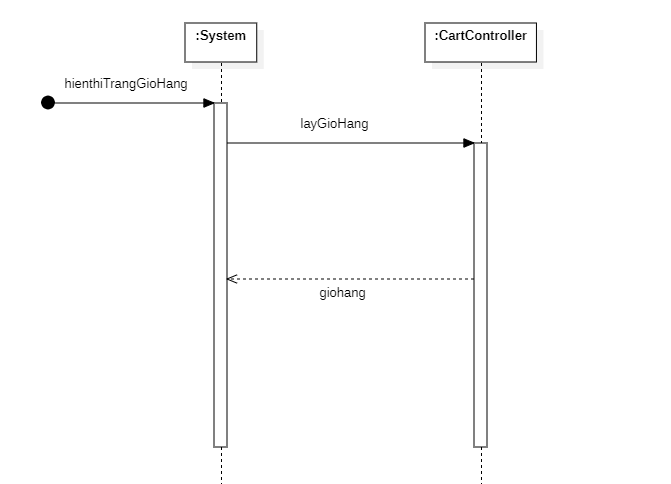
#### 5.2.3. SD capnhatGioHang



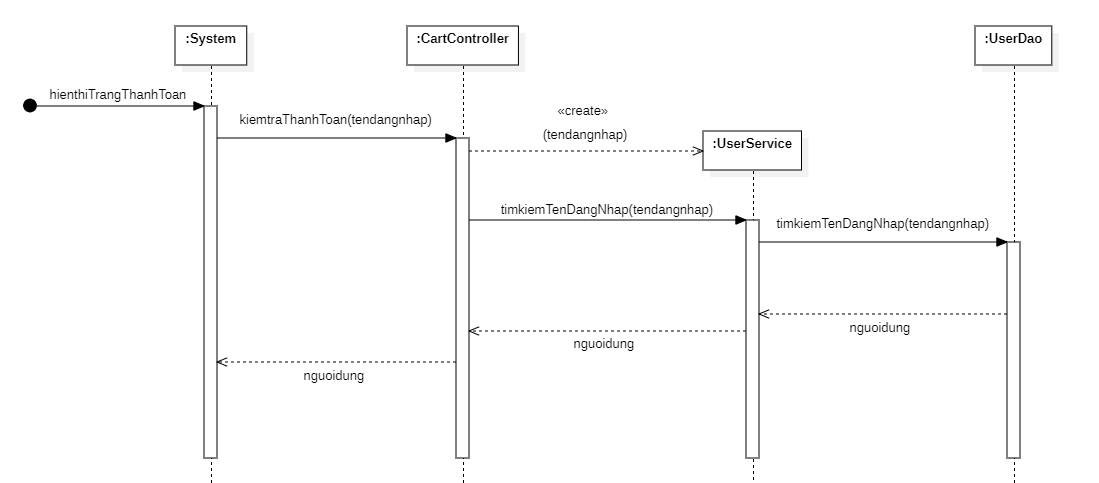
### 5.3. SSD mua hàng



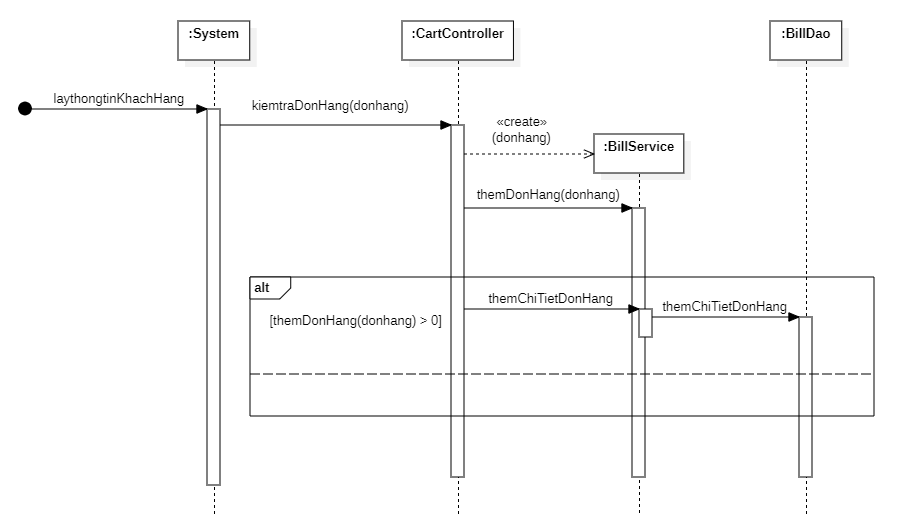
#### 5.3.1. SD hienthiTrangGioHang



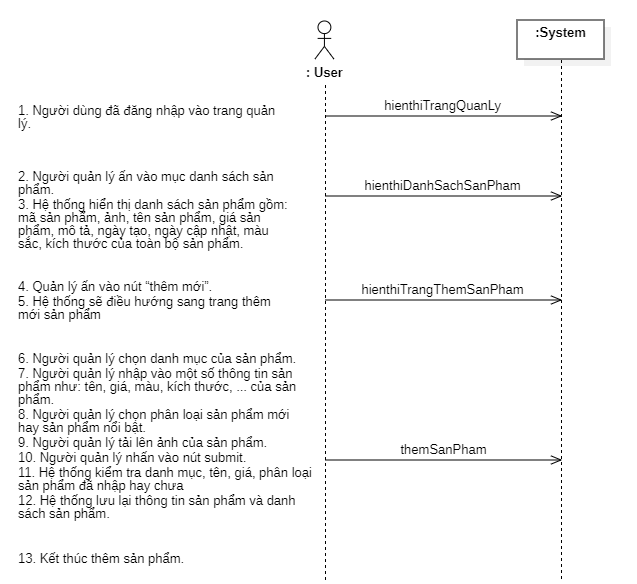
#### 5.3.2. SD hienthiTrangThanhToan



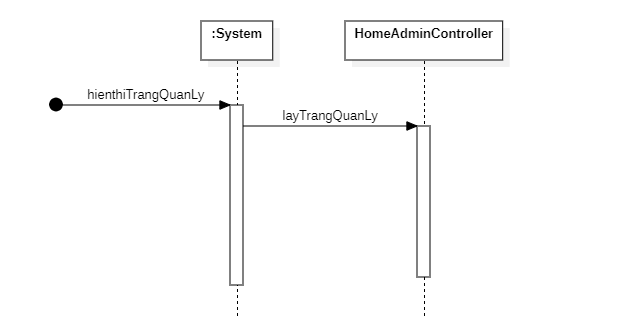
#### 5.3.3. SD laythongtinKhachHang



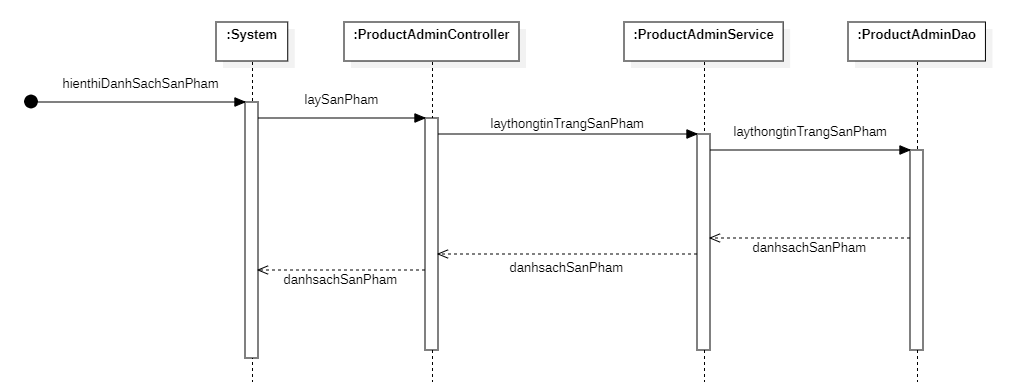
### 5.4. SSD thêm sản phẩm



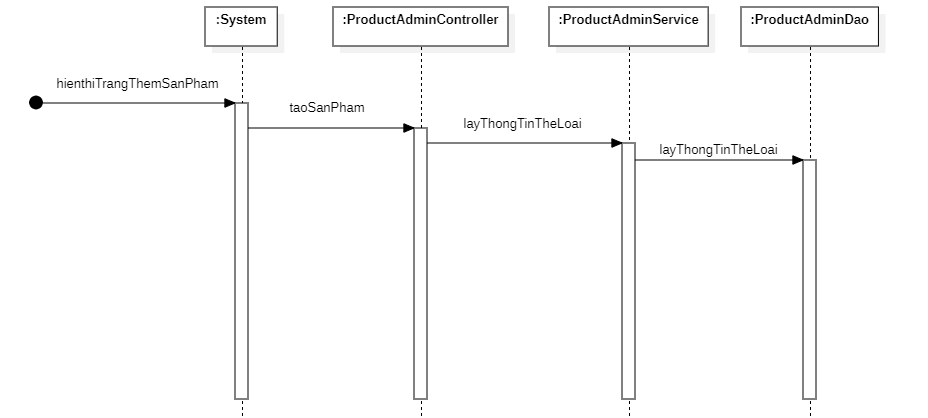
#### 5.4.1. SD hienthiTrangQuanLy



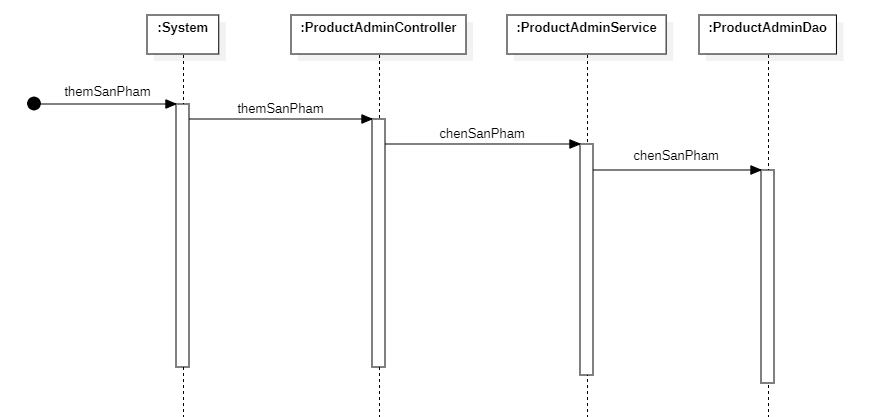
#### 5.4.2. SD hienthiDanhSachSanPham



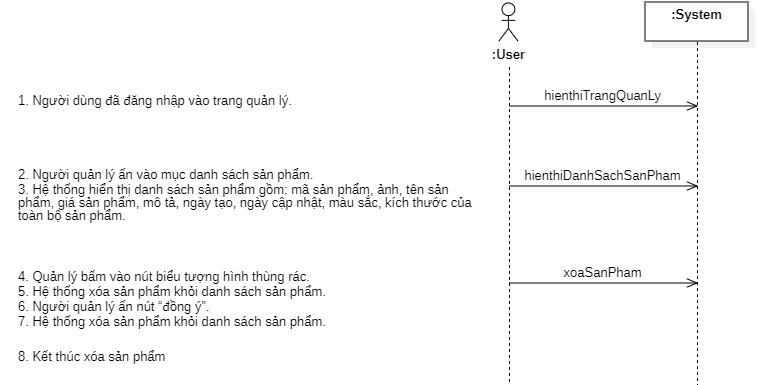
#### 5.4.3. SD hienthiTrangThemSanPham



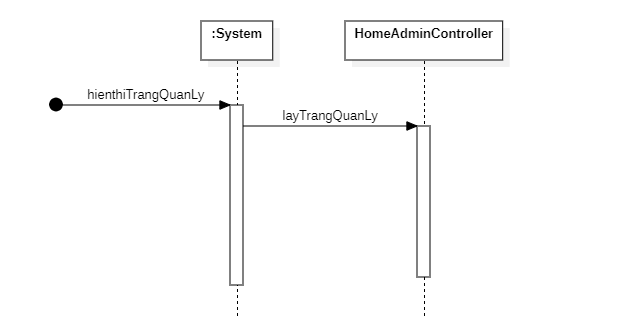
#### 5.4.4. SD themSanPham



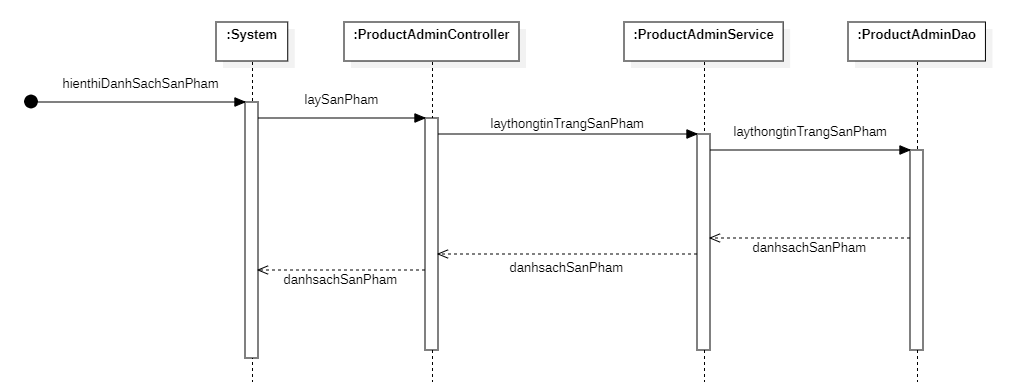
### 5.5. SSD xóa sản phẩm



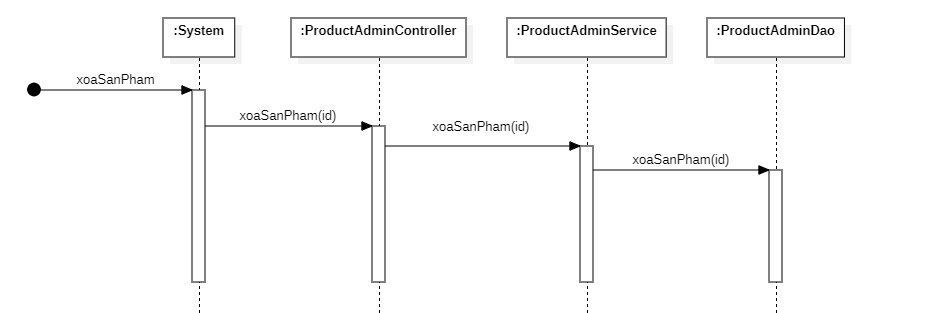
#### 5.5.1. SD hienthiTrangQuanLy



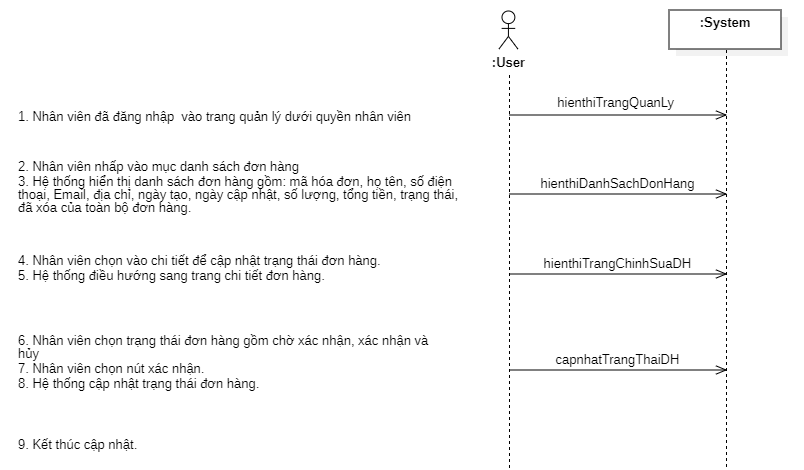
#### 5.5.2. SD hienthiDanhSachSanPham



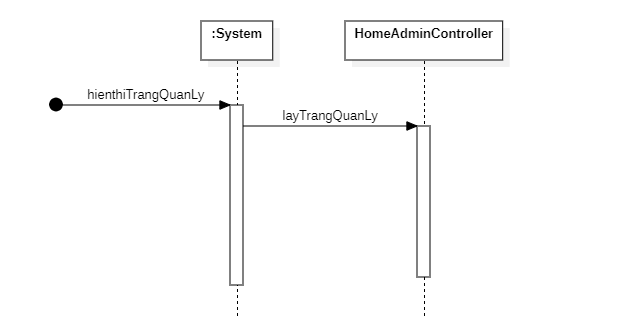
#### 5.5.3. SD xoaSanPham



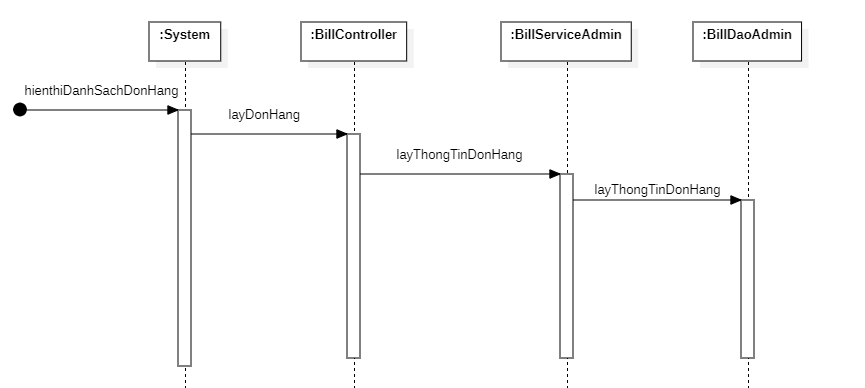
### 5.6. SSD cập nhật trạng thái đơn hàng



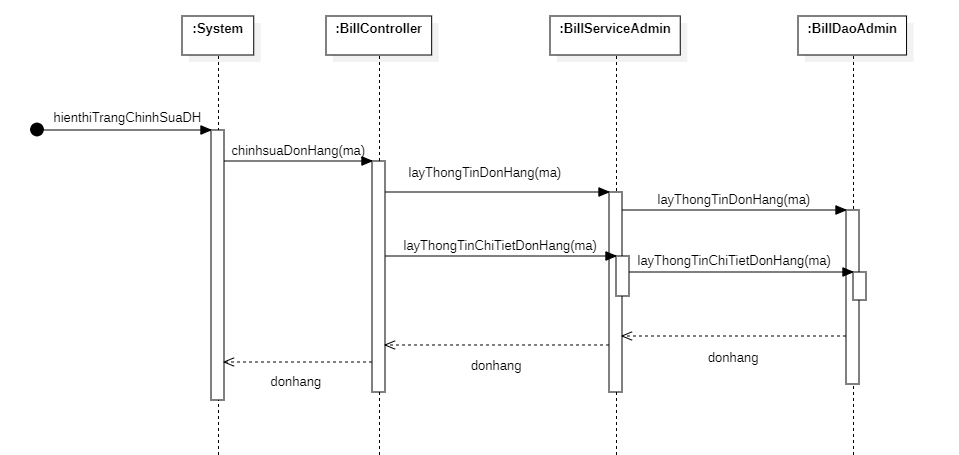
#### 5.6.1. SD hienthiTrangQuanLy



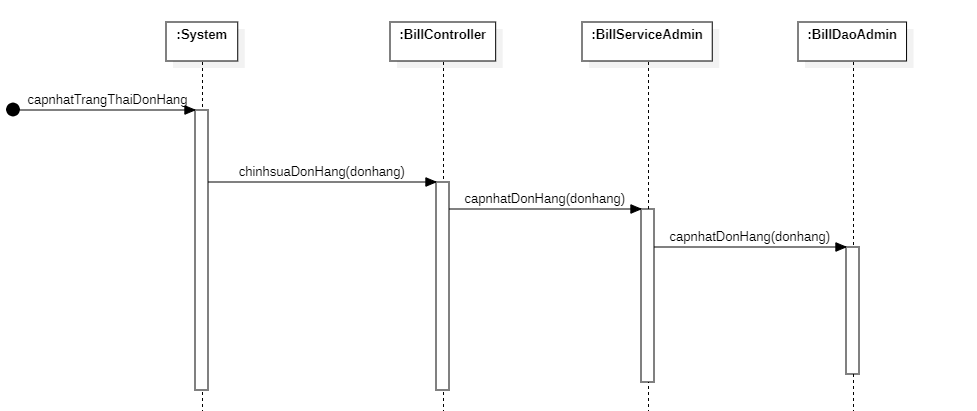
#### 5.6.2. SD hienthiDanhSachDonHang



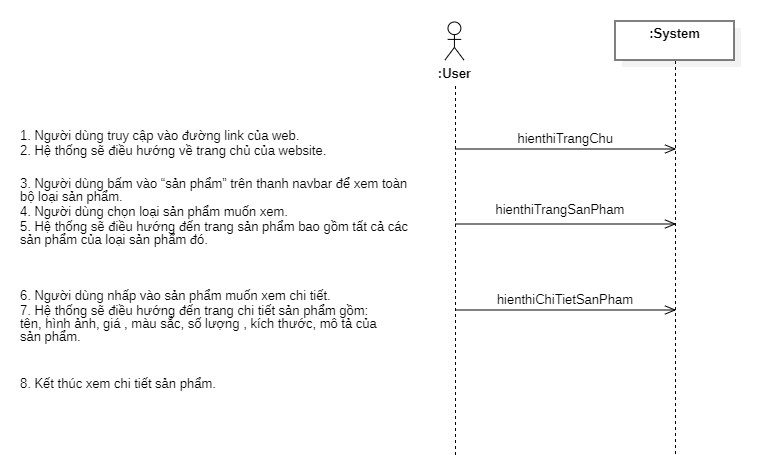
#### 5.6.3. SD hienthiTrangChinhSuaDH



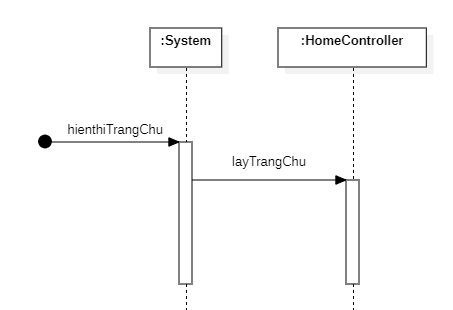
#### 5.6.4. SD capnhatTrangThaiDonHang



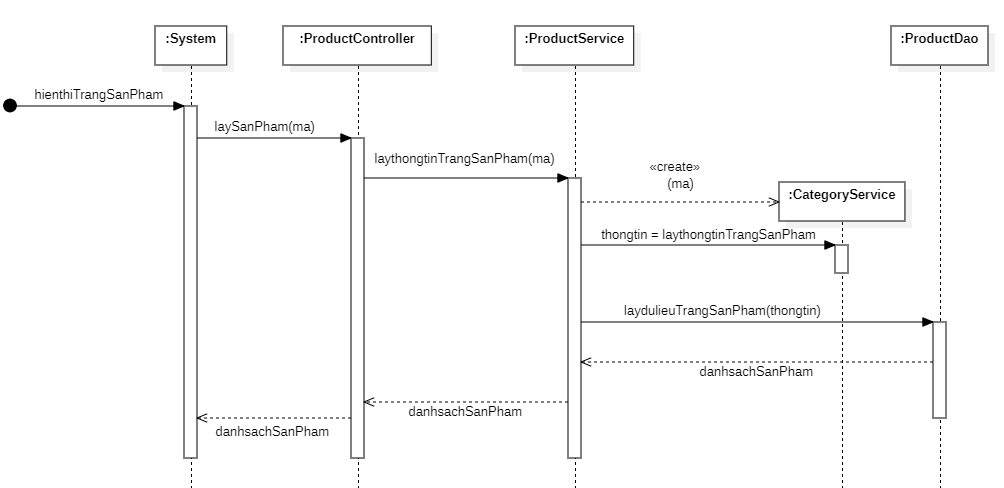
### 5.7. SSD xem chi tiết sản phẩm



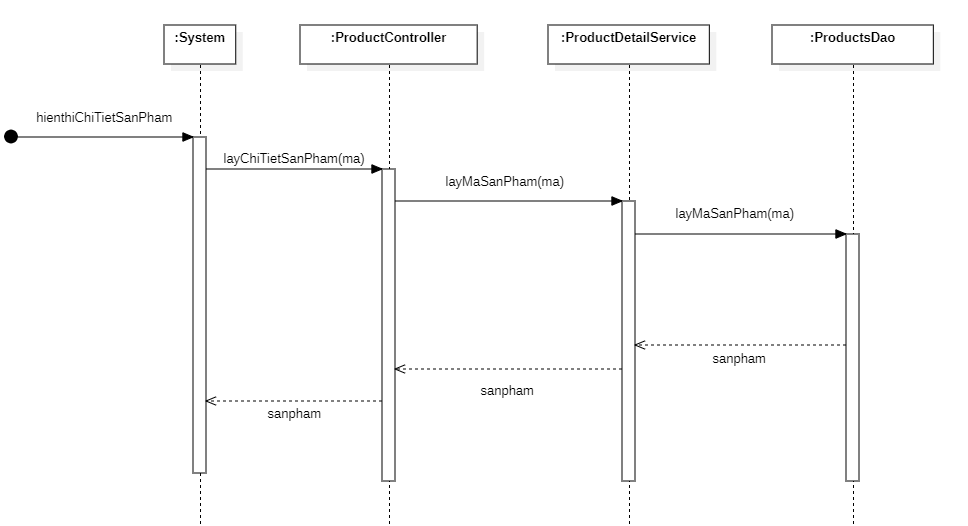
#### 5.7.1. SD hienthiTrangChu



#### 5.7.2. SD hienthiTrangSanPham



#### 5.7.3. SD hienthiChiTietSanPham



## 6. Hướng dẫn cài đặt và chạy

### 6.1. Sơ lược về project

- Link project: <https://github.com/CTminhduc/SouvenirShop_BMW.git>

- Ngôn ngữ lập trình: Java

- Framework: Spring MVC

- Cơ sở dữ liệu: MySQL

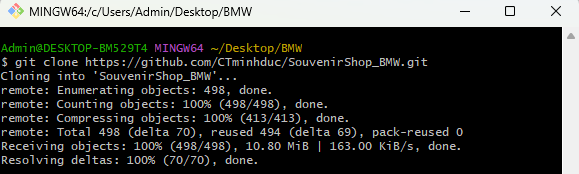
- Version control: GitHub

- IDE: Spring tool suite 4.0 for Eclipse

### 6.2. Hướng dẫn cài đặt

**Bước 1:** Tải project từ github về bằng câu lệnh:

git clone <https://github.com/CTminhduc/SouvenirShop_BMW.git>



**Bước 2:** Import project vào IDE

Mở 1 IDE bất kỳ hỗ trợ java ( Nên dùng STS cho eclipse ). Chọn file -> Open Project From file System -> Directory -> Chọn project -> Finish

# Phần 2: Bảo mật website

## Kiểm tra thủ công (manual)

### Insecure Direct Object Reference ( IDOR )

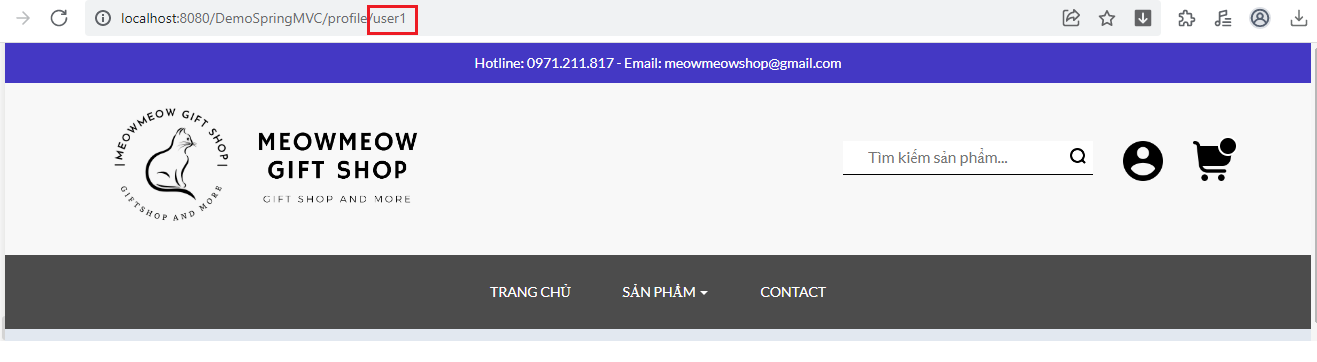
#### Khái niệm

IDOR là cụm từ viết tắt của Insecure Direct Object Reference (Tham chiếu đối tượng trực tiếp không an toàn). Lỗ hổng này nằm trong TOP 10 lỗ hổng nguy hiểm mà OWASP đã đưa ra IDOR. 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.

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.

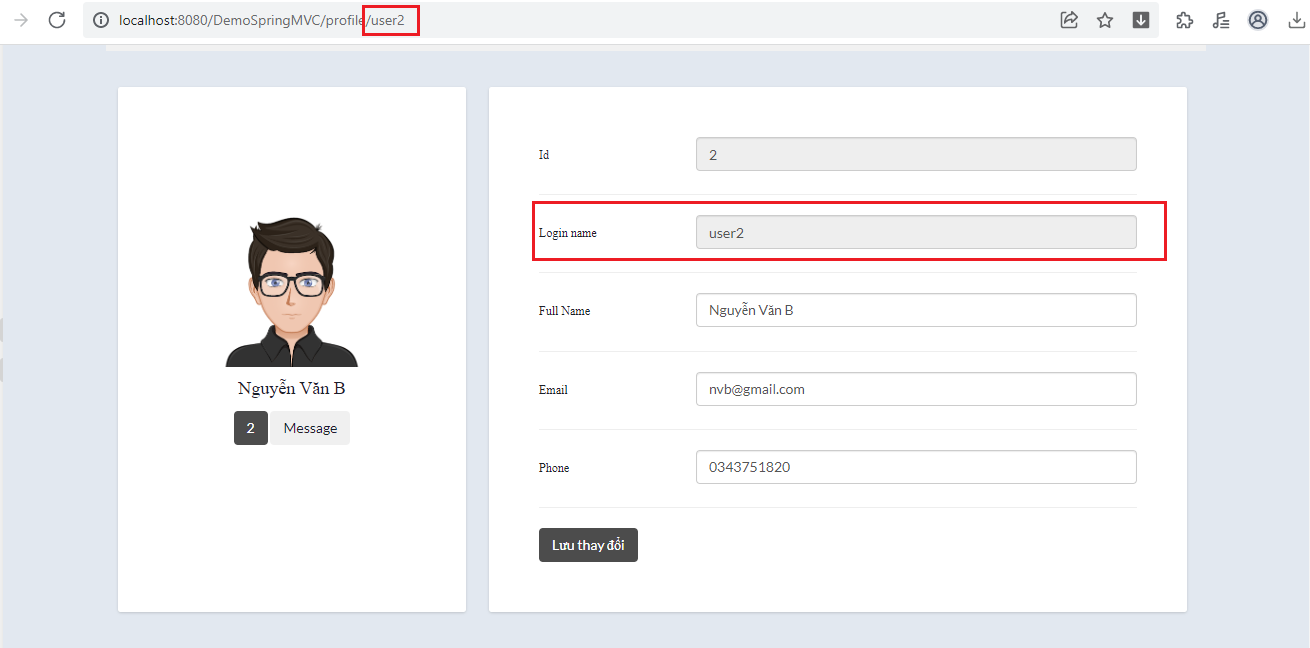
IDOR không chỉ cho phép truy cập thông tin trái phép của người dùng khác mà còn có thể cho phép kẻ tấn công thực hiện thay đổi hoặc xóa dữ liệu của người dùng khác. Vì vậy lỗ hổng này có mức độ nguy hiểm rất cao, gây hậu quả nghiêm trọng tới người dùng. Đặc biệt là các hệ thống có dữ liệu quan trọng của người dùng như: mạng xã hội, tài chính - ngân hàng, y tế..

#### Tấn công project

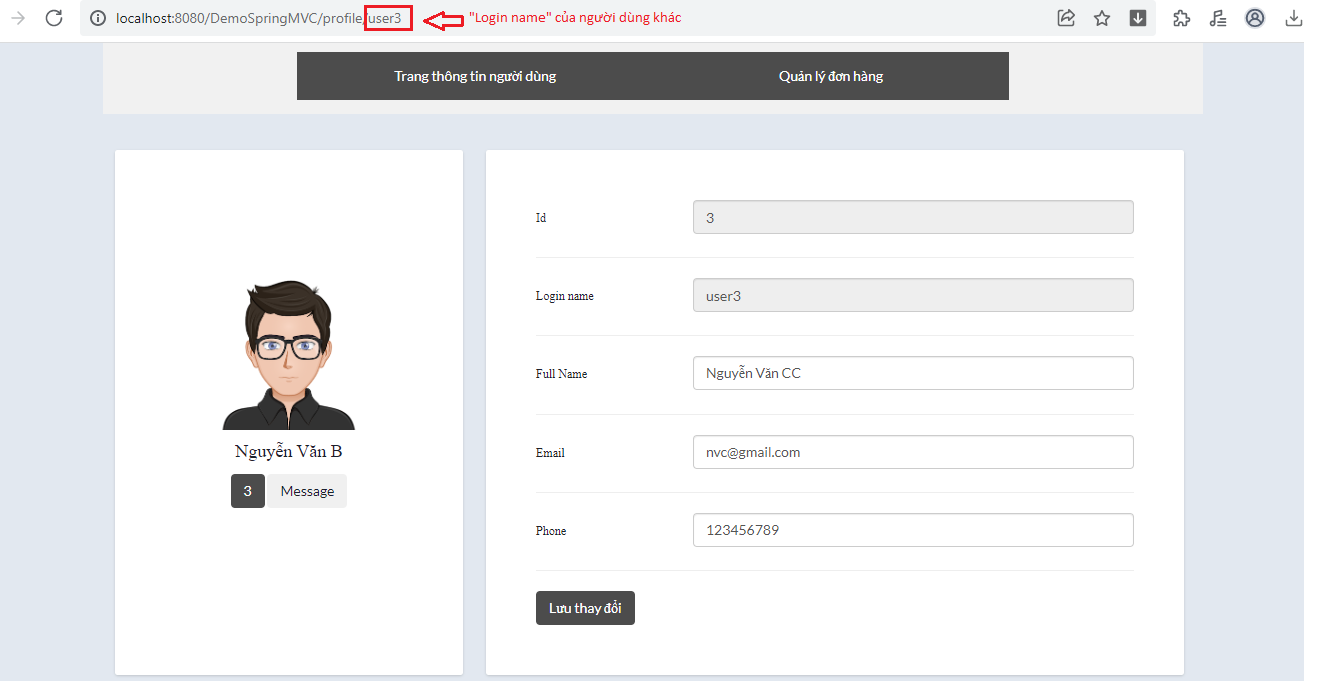
Trong project, tại trang xem thông tin cá nhân, để ý trên url người dùng sẽ thấy có thể tìm kiếm thông tin cá nhân theo “Login name” bằng cách thay đổi trực tiếp giá trị “user1” trên thanh url

Tuy nhiên vấn đề xảy ra khi người dùng thay đổi giá trị “user1” không phải của mình.

Lấy ví dụ tài khoản của người dùng với Login name: user2.



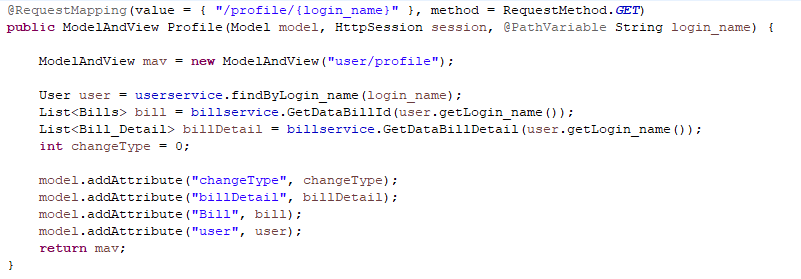
Người dùng với Login name: “user2”. Khi thay đổi giá trị “user2” trên url có thể truy cập và xem thông tin cá nhân của người dùng với Login name: “user3”



Như vậy, các thông tin cá nhân có thể bị đọc bởi người dùng khác.

#### Nguyên nhân

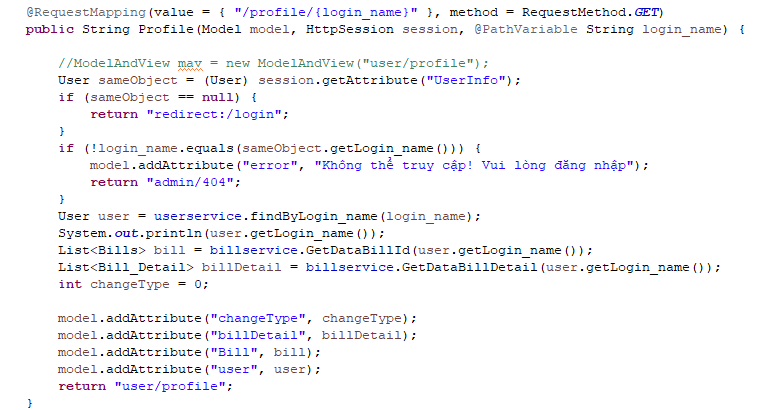
Nguyên nhân đến từ phía server đã không tiến hành kiểm tra trước khi người dùng tiến hành đăng nhập vào trang web.



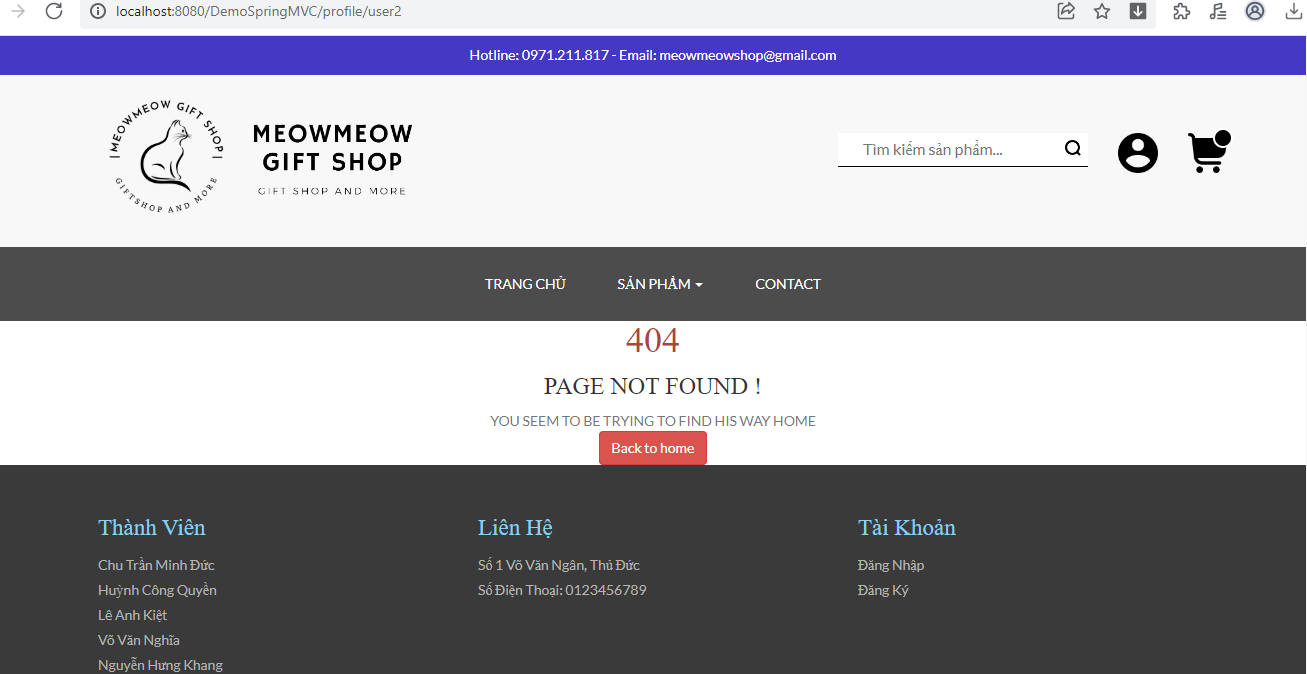
Việc giá trị Login name được lấy trực tiếp từ đường dẫn mà không qua bất kỳ kiểm tra nào đã khiến cho thông tin hóa cá nhân có thể dễ dàng bị truy cập bởi người dùng trái phép.

#### Khắc phục

Nên để khắc phục, ta sẽ tiến hành viết thêm code kiểm tra:



Khi người dùng cố ý thay đổi “Login name” để truy cập vào thông tin cá nhân của người khác, website sẽ dẫn đến màn hình lỗi:



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.

### Hidden Field Manipulation

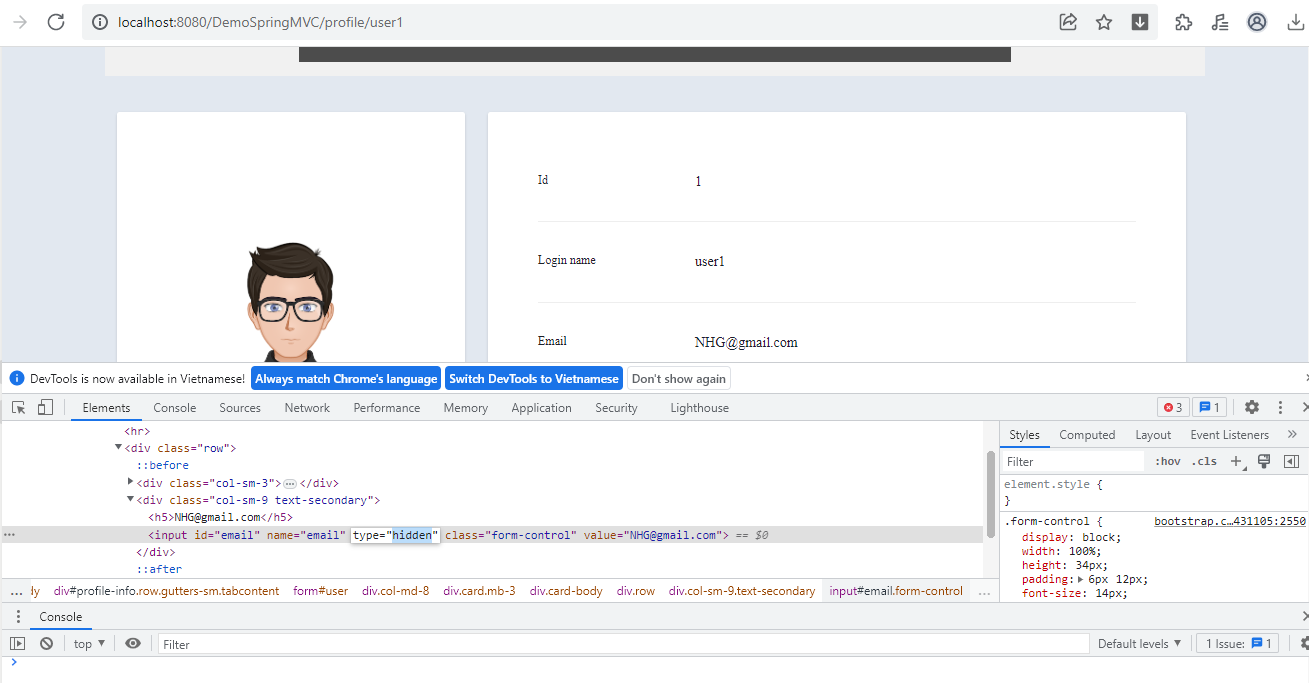
#### Khái niệm

Là thể loại tấn công web trong đó lợi dụng các ô input được ẩn đi bởi các lập trình viên. Các ô input này thường chứa giá trị như: id, session id,... Hacker sẽ sử thay đổi giá trị các ô input này để vượt qua cơ chế bảo mật của website và gửi yêu cầu đến phía server. Nếu server kiểm tra dữ liệu lỏng lẻo, không có cơ chế xác thực người dùng. Hacker có thể thay đổi được các dữ liệu quan trọng.

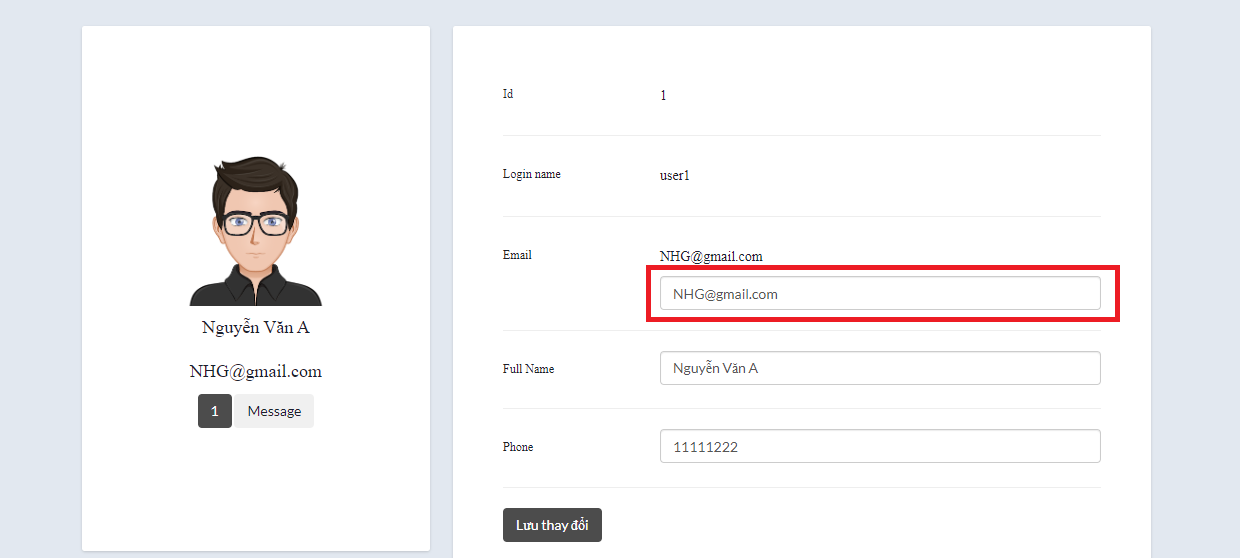
#### Tấn công project

Trong project, tại trang quản lý thông tin cá nhân, người dùng có thể tùy ý chỉnh sửa thông tin tài khoản của mình tại form chỉnh sửa.

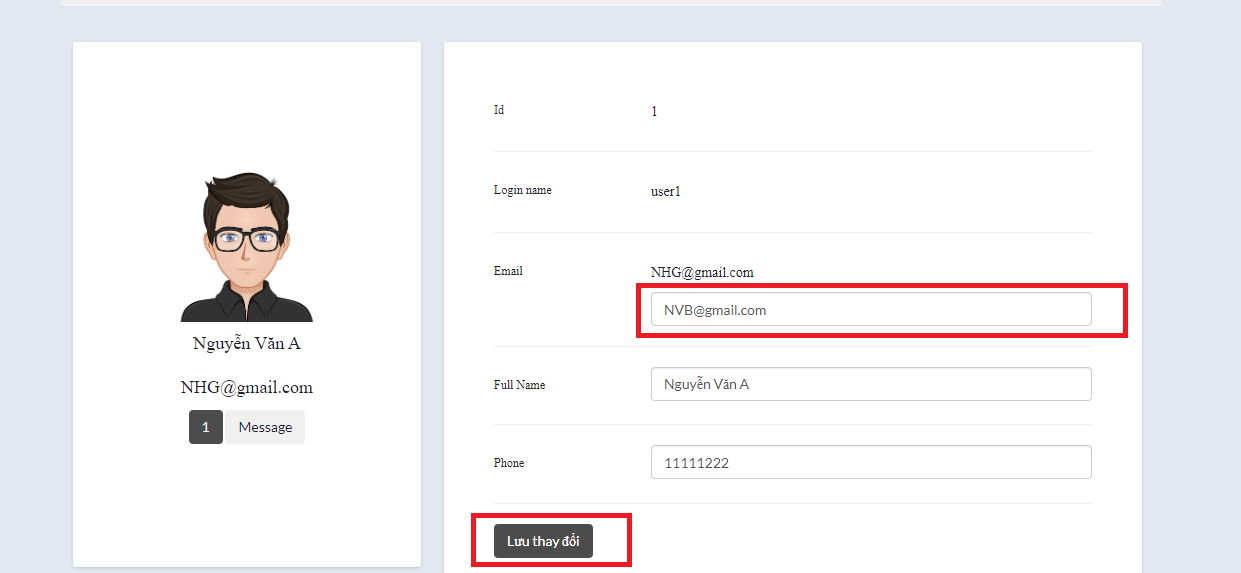
Tuy nhiên khi kiểm tra bằng F12, ta có thể nhận ra tồn tại thẻ input với thuộc tính là hidden, thẻ input này cho phép thay đổi email của yêu cầu chỉnh sửa.



Khi ta loại bỏ thuộc tính “hidden”, 1 ô input hiện ra cho phép ta chỉnh sửa email:



Sau đó, thay đổi email ở ô hidden thành email của người dùng khác, sau đó chỉnh sửa thông tin mình mong muốn ở form chỉnh và bấm nút “Lưu thay đổi”

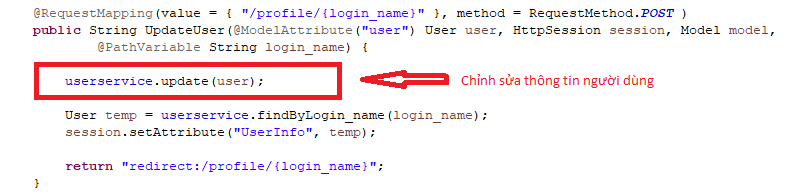


Ta kiểm tra bằng cách đăng nhập vào tài khoản nạn nhân ([NVB@gmail.com](mailto:NVB@gmail.com)) và thông tin của nạn nhân đã bị chỉnh sửa trái phép bởi người dùng khác

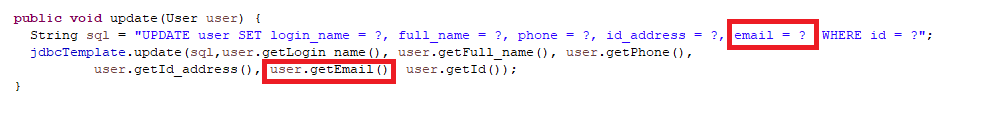


#### Nguyên nhân

Xem lại code, tại phần chỉnh sửa thông tin của người dùng, đồng thời lấy giá trị email từ ô input chứa thuộc tính hidden.

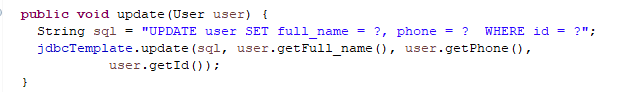


Trong lớp truy vấn dữ liệu có update trường thuộc tính “email”

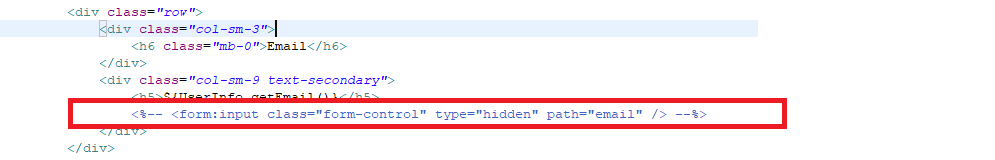


#### Khắc phục

Để khắc phục lỗi ta không cho phép chỉnh sửa “email” từ người dùng



Đồng thời loại bỏ hidden ở ô input “email”



### Brute Force Attack ( Admin Password )

#### Khái niệm

Tấn công brute-force là một phương pháp bẻ khóa phổ biến . Hacker sẽ 'đoán' tên người dùng và mật khẩu để truy cập trái phép vào hệ thống thông qua việc sử dụng phương pháp thử và sai để cố gắng đoán thông tin đăng nhập hợp lệ.

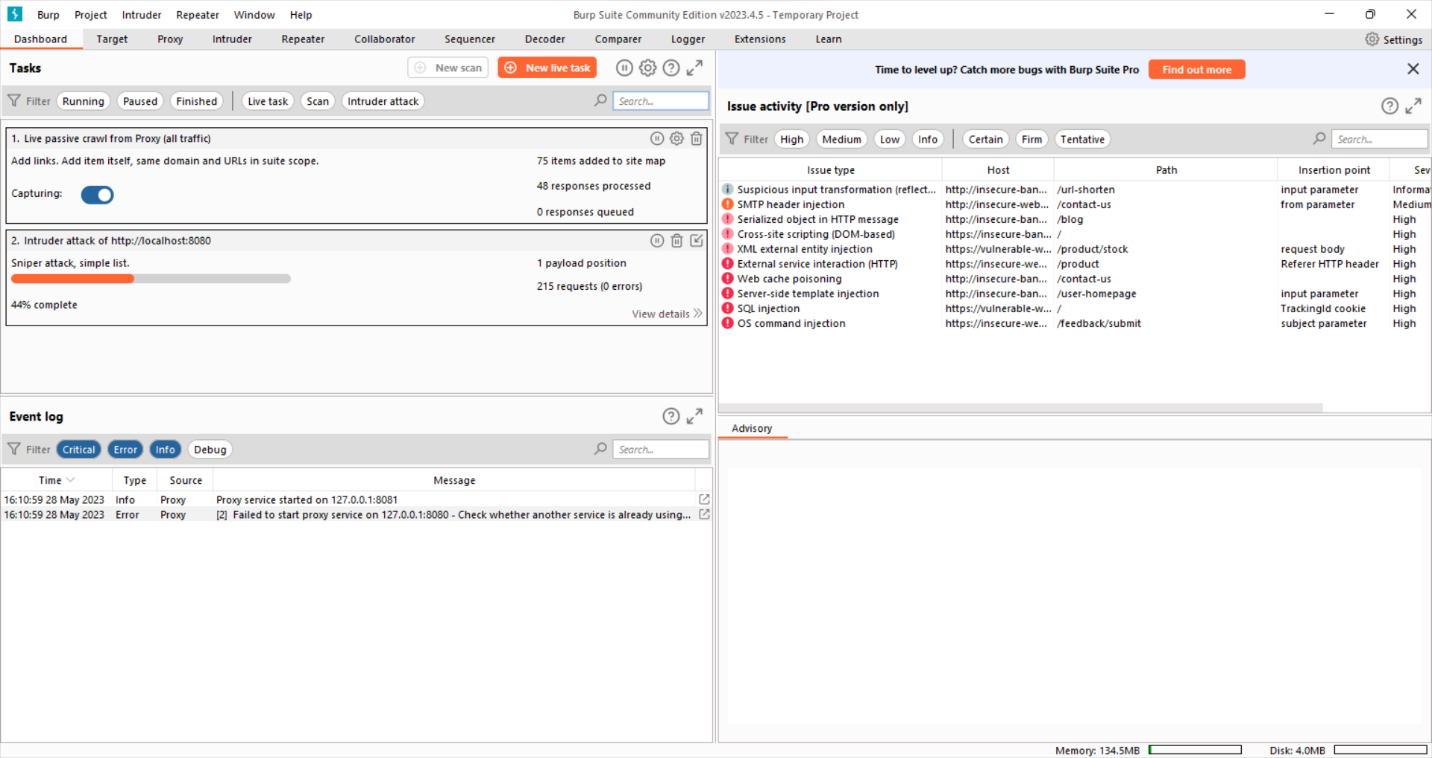
Các cuộc tấn công này thường được tự động hóa bằng cách sử dụng danh sách các mật khẩu thông dụng thường được gọi là “dictionary”. Một dictionary có thể bao gồm hàng triệu mật khẩu. Vì vậy, hacker sẽ sử dụng nhưng công cụ chuyên dụng có khả năng cho phép hacker thực hiện đăng nhập 1 cách tự động nhiều lần với tốc độ cao.

Nếu mật khẩu sử dụng tất cả các chữ cái thường và không có ký tự đặc biệt hoặc chữ số, chỉ mất 2-10 phút là một cuộc tấn công brute force có thể crack mật khẩu này. Ngược lại, một mật khẩu có sự kết hợp của cả chữ hoa và chữ thường cùng với một vài chữ số (giả sử có 8 chữ số) sẽ mất hơn 14-15 năm để bị crack.

Để kiểm tra độ mạnh của mật khẩu. Ta có thể kiểm tra tại các trang web như: <https://www.passwordmonster.com/>

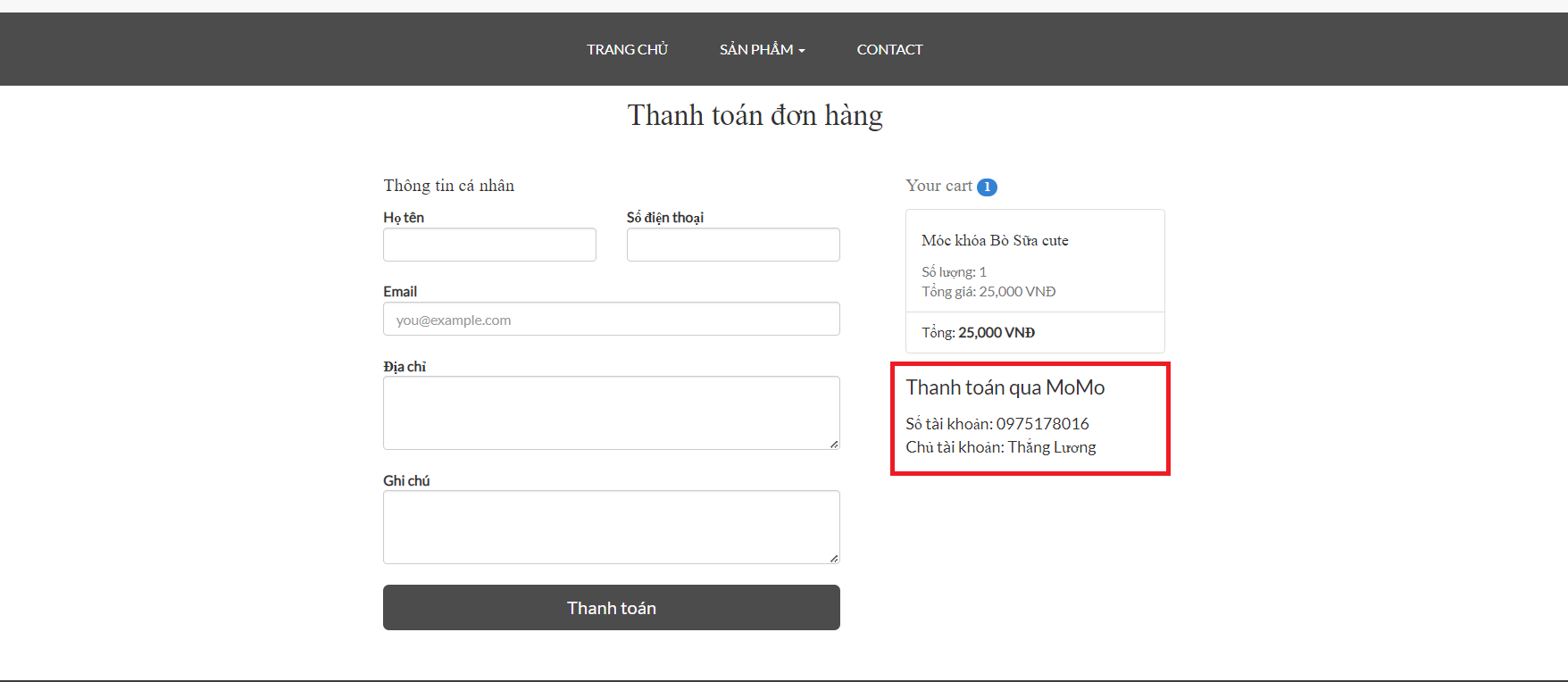
#### Tấn công project

Ta sẽ tiến giả làm hacker tiến hành tấn công bằng phân mềm burp suite. Đây là một công cụ cực kỳ mạnh mẽ trong việc kiểm thử và tấn công thông qua việc gửi và chỉnh sửa các yêu cầu ( request )

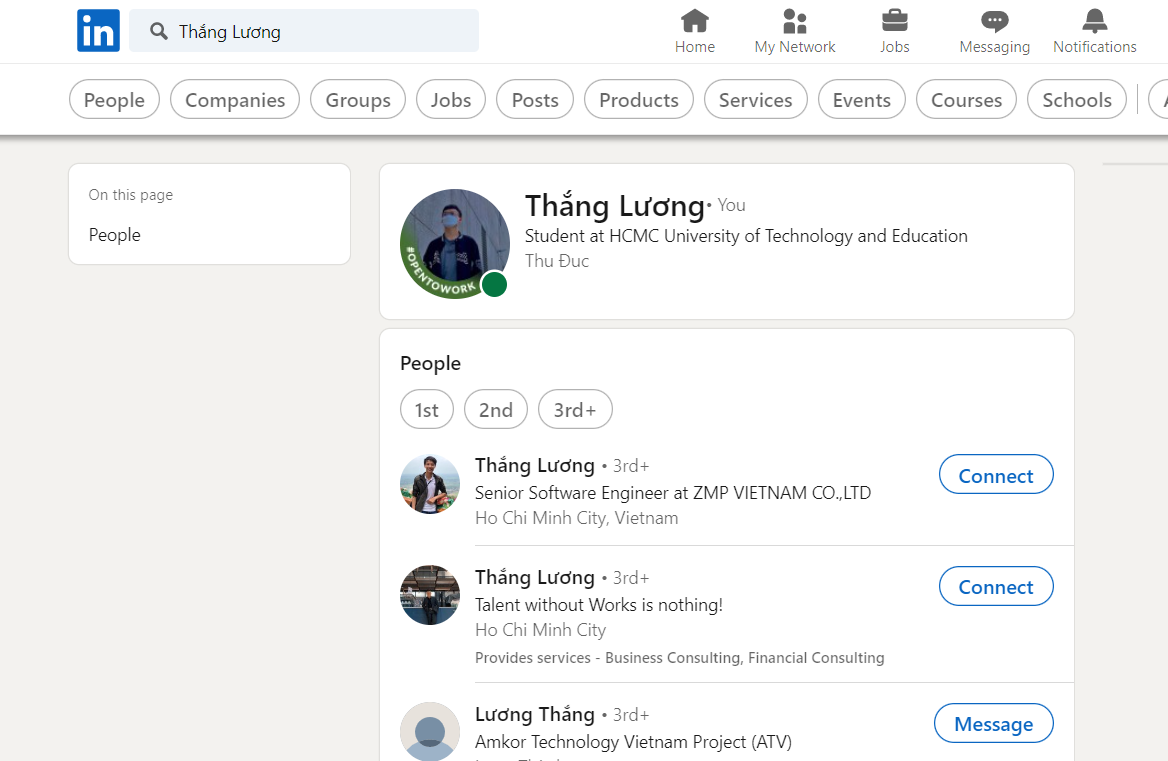


Đầu tiên, ta biết được trang đăng nhập của website chạy ở đường dẫn: <http://localhost:8080/login>Admin. Tiếp theo, cần tìm email của admin để tiến hành tấn công vét cạn mật khẩu.

Để ý thấy, khi người dùng đặt hàng thông qua phương thức thanh toán là momo. Số điện thoại và tên của admin sẽ được thông báo:

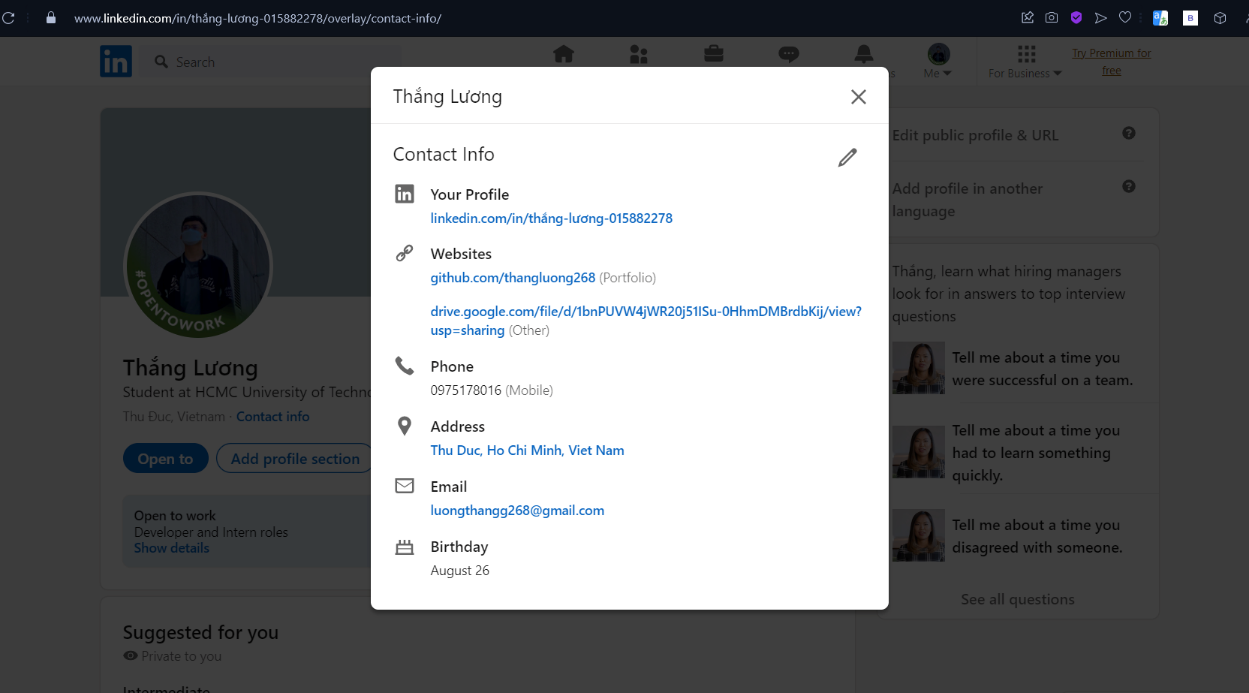


Ta đã biết được tên và số điện thoại của admin. Thông qua kỹ thuật OSINT, ta tìm được thông tin địa chỉ email của admin thông qua mạng xã hội linkedlink:



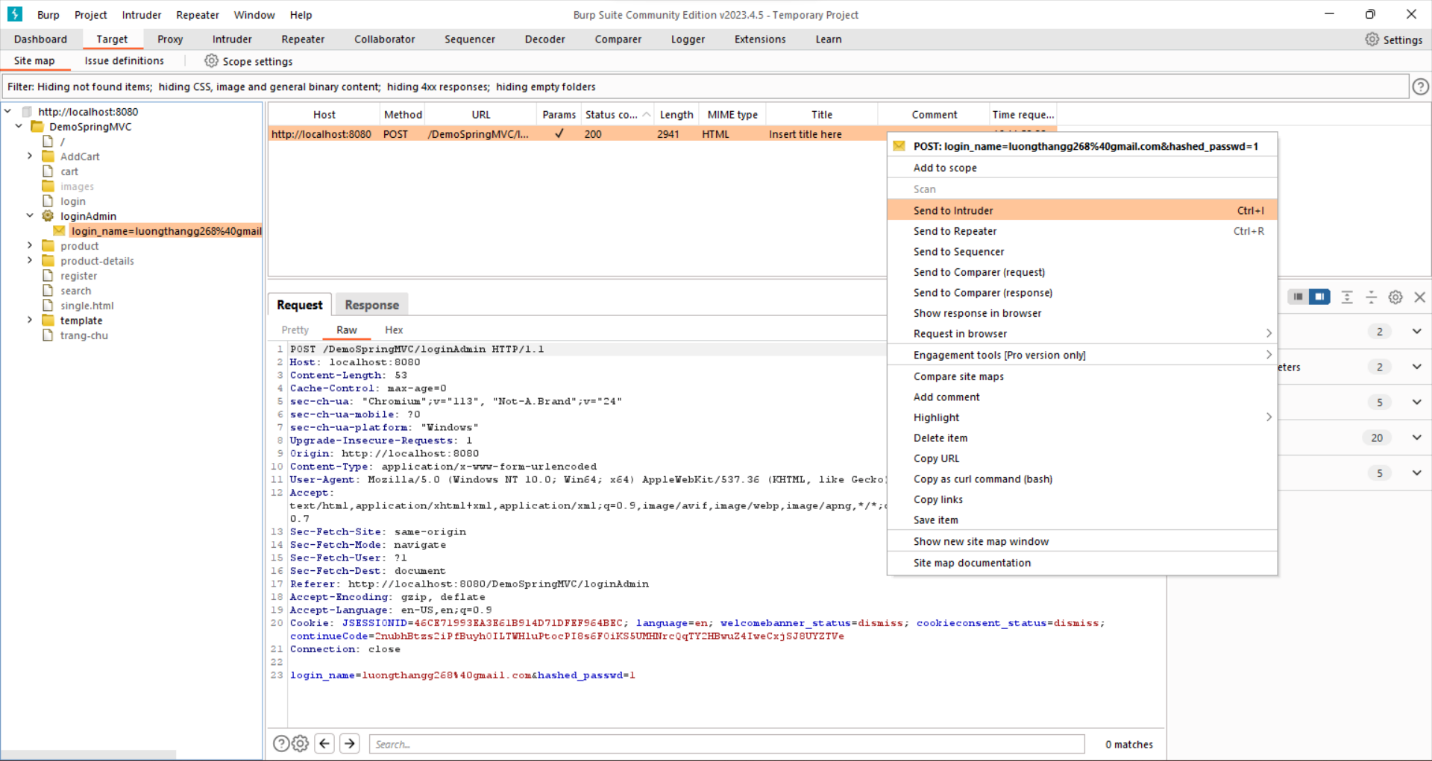
Khi vào xem trang cá nhân: [https://www.linkedin.com/in/thắng-lương-015882278/overlay/contact-info/](https://www.linkedin.com/in/th%E1%BA%AFng-l%C6%B0%C6%A1ng-015882278/overlay/contact-info/?fbclid=IwAR1h7LdPJmLvw-ElzIt9x9EpYaz6pW2i1B3gRbTvnTKAmL66SXVB6BhNXPQ)

Ta thấy được email của người dùng là: luongthangg268@gmail.com và khớp với số điện thoại ở trên website

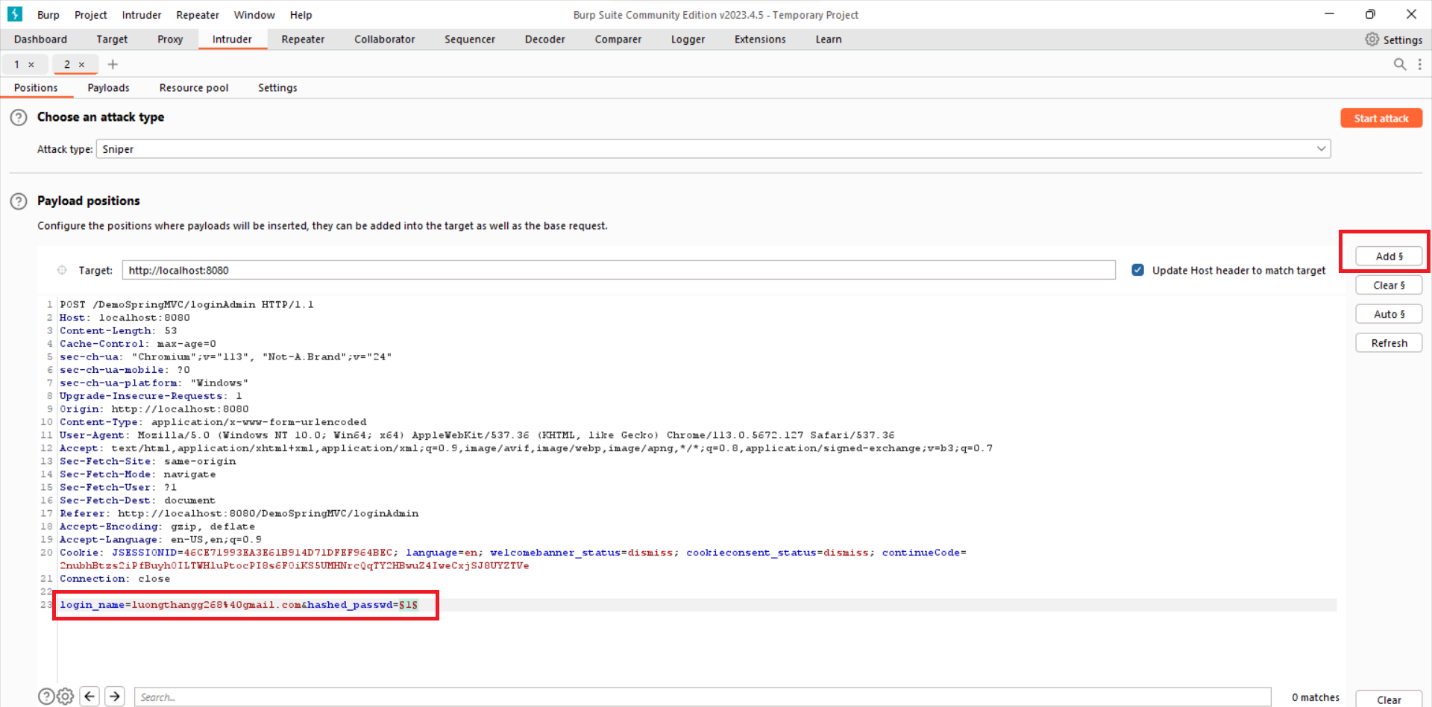


Vì vậy email của admin là : luongthangg268@gmail.com

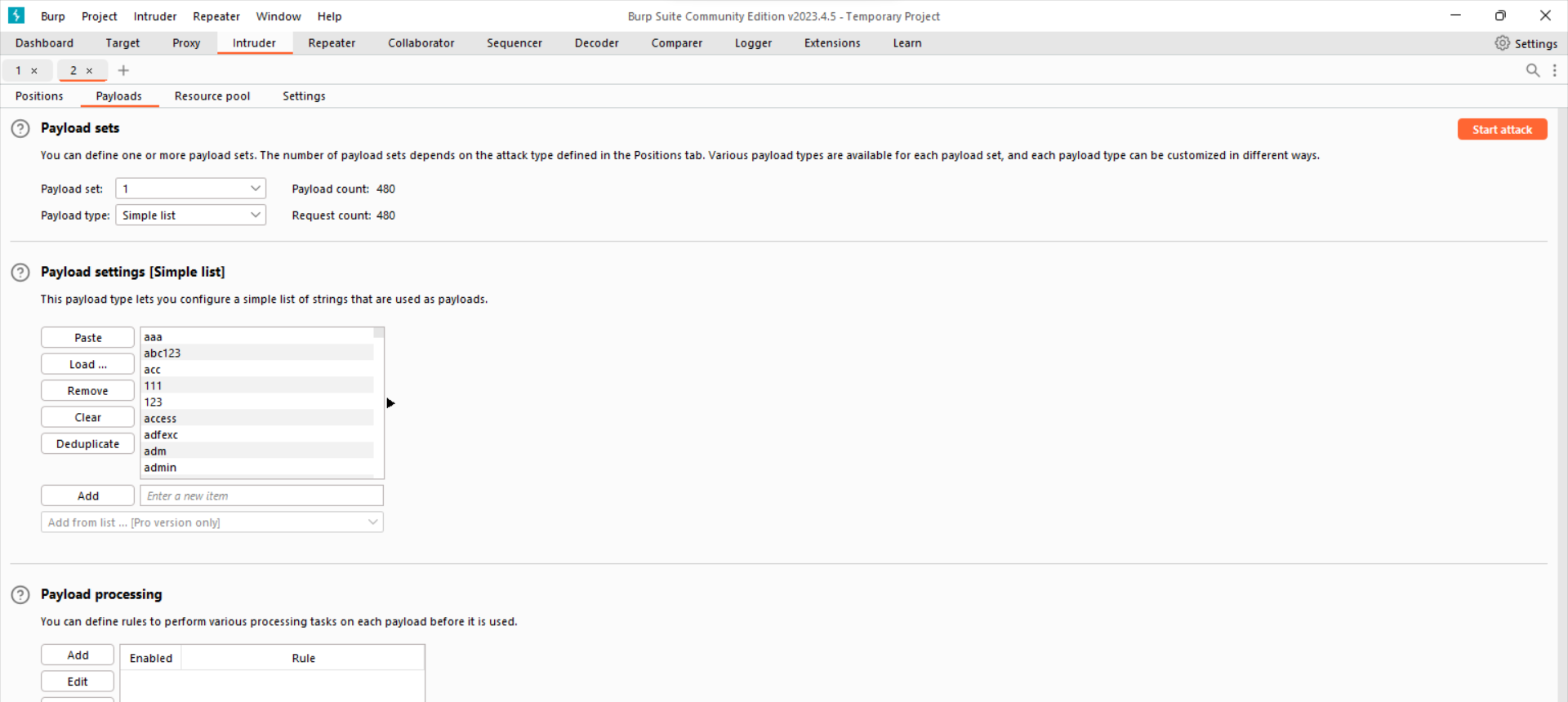
Tiếp theo, để có thể tấn công vét cạn ( Brute force ) mật khẩu. Ta sử dụng phần mềm burp suite. Gửi request đăng nhập với email là luongthangg268@gmail.com đến intruder:



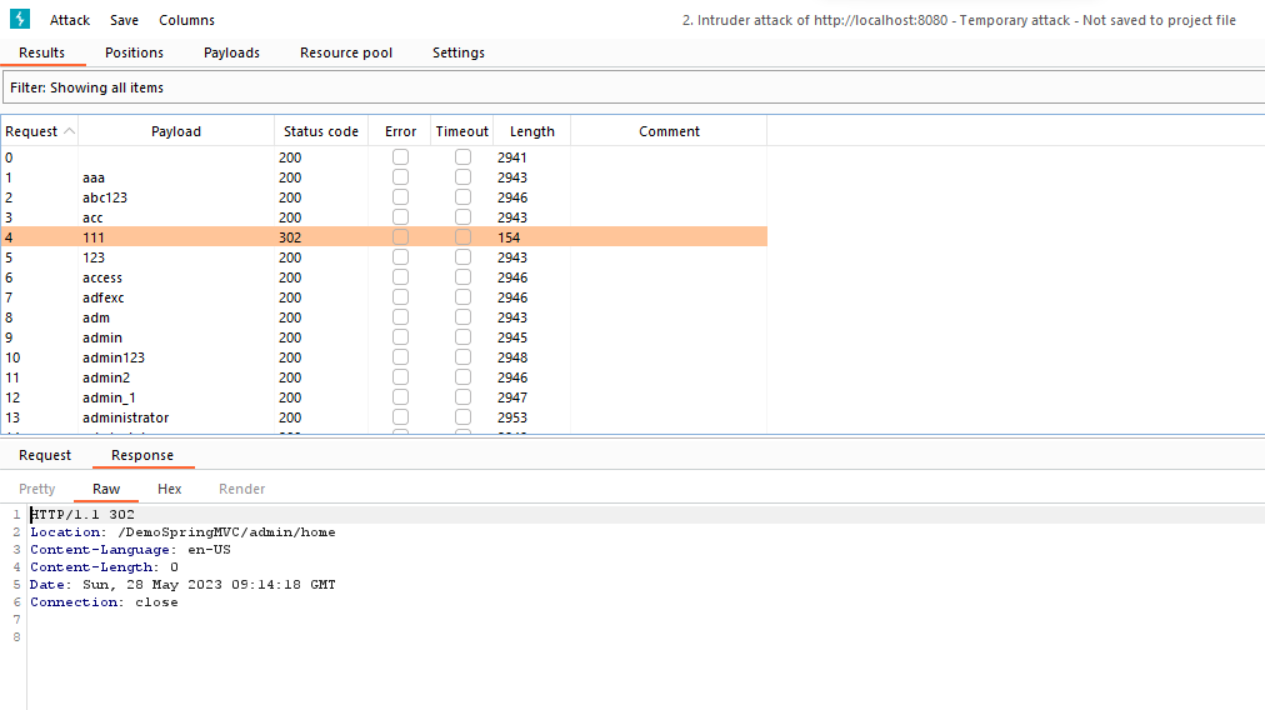
Vào Intruder -> Positions, sau đó bôi đen giá trị của mật khẩu và ấn Add $



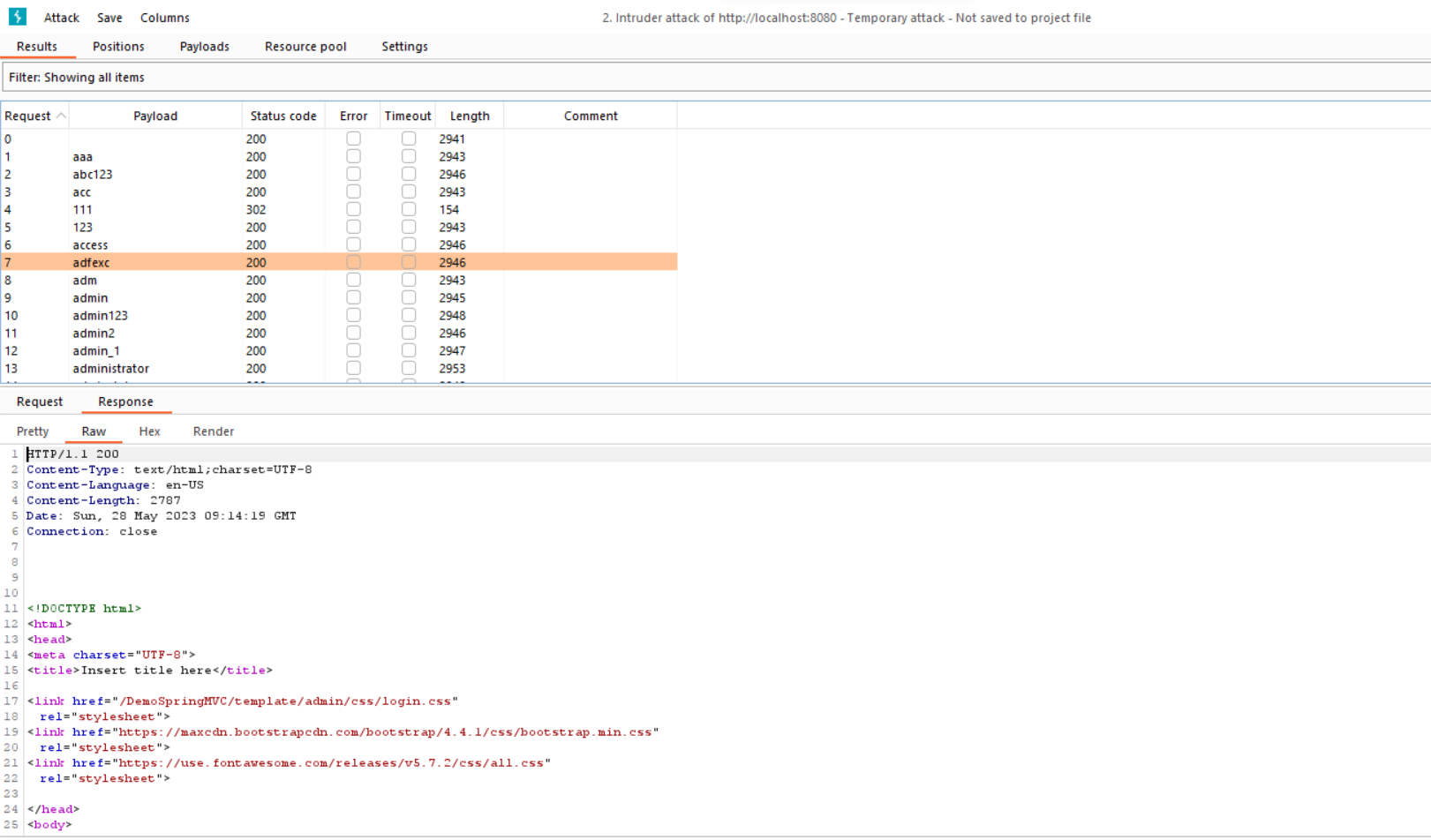
Tiếp theo, tiến hành đưa file dictionary chứa các mật khẩu thông dụng vào công cụ:



Tiến hành tấn công brute force. Để ý thấy tại mật khẩu là “111”, đường dẫn trả về là trang admin:

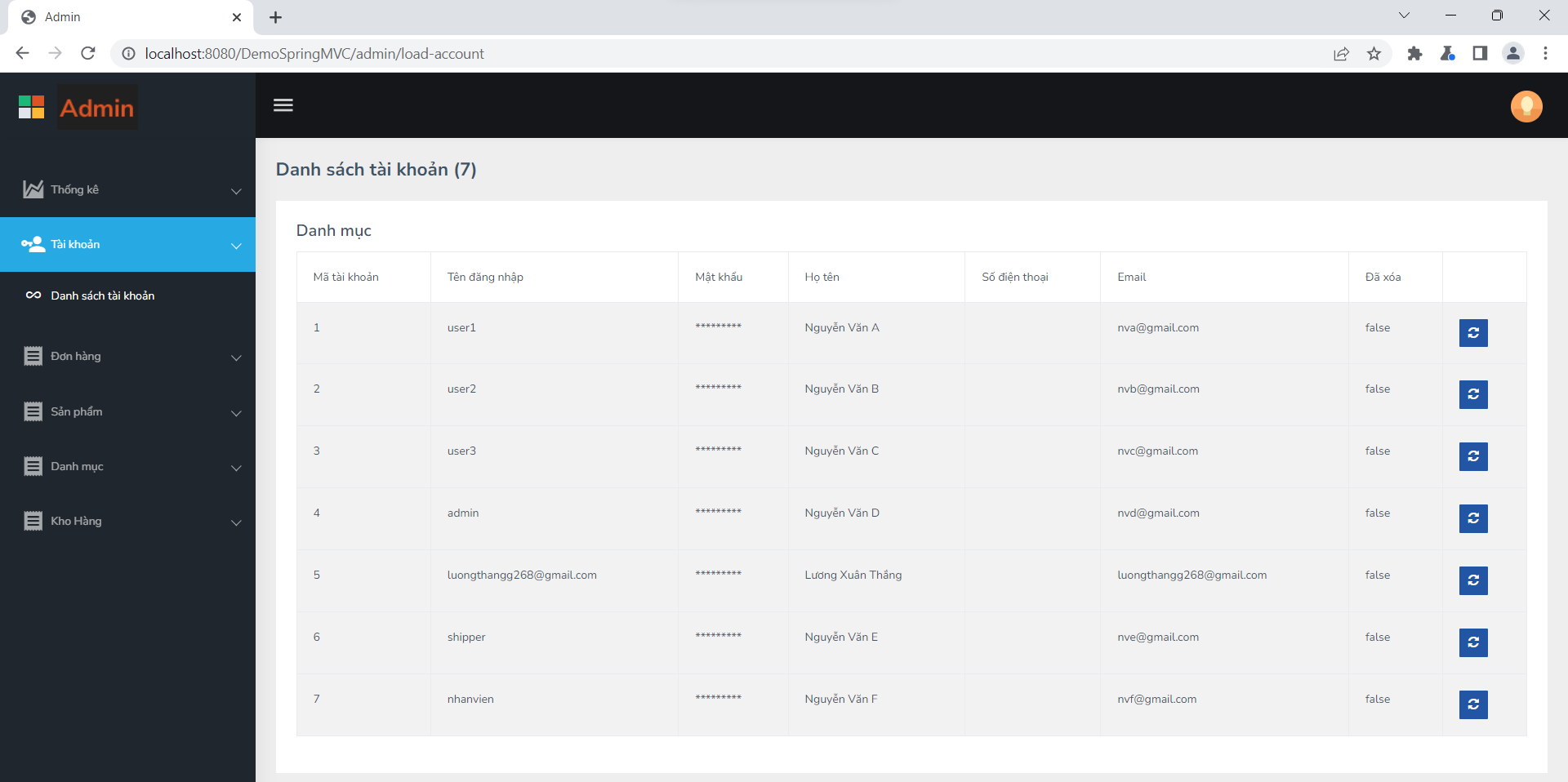


Còn các mật khẩu khác, đường dẫn trả về mã html hiển thị lại trang đăng nhập:



Vì vậy, ta biết được tài khoản và mật khẩu của admin là luongthangg268@gmail.com | 111

Tiến hành truy cập và đã vào được trang admin:



#### Nguyên nhân

Có 2 nguyên nhân chính dẫn đến việc trang web dễ dàng bị tấn công brute force.

*Nguyên nhân đầu tiên*, là việc email của admin bị lộ. Việc bảo vệ sự bí mật của tài khoản admin là rất quan trọng, bất kể là email hay mật khẩu. Trang web đã sử dụng email được công khai trên mạng xã hội để làm tài khoản admin. Việc này có thể bị hacker tìm thấy thông qua một số kỹ thuật OSINT cơ bản.

*Nguyên nhân thứ 2*, cũng là nguyên nhân quan trọng nhất là mật khẩu của admin quá yếu. Chỉ tốn 1-2ph là có thể tìm ra được mật khẩu thông qua tấn công vẹt cạn. “111” là mật khẩu nằm trong top các mật khẩu dễ phá và thường được sử dụng nhất mà nhóm thực hiện xin phép được demo.

Có thể thấy trong mã nguồn, mật khẩu mặc định cho admin là “111”

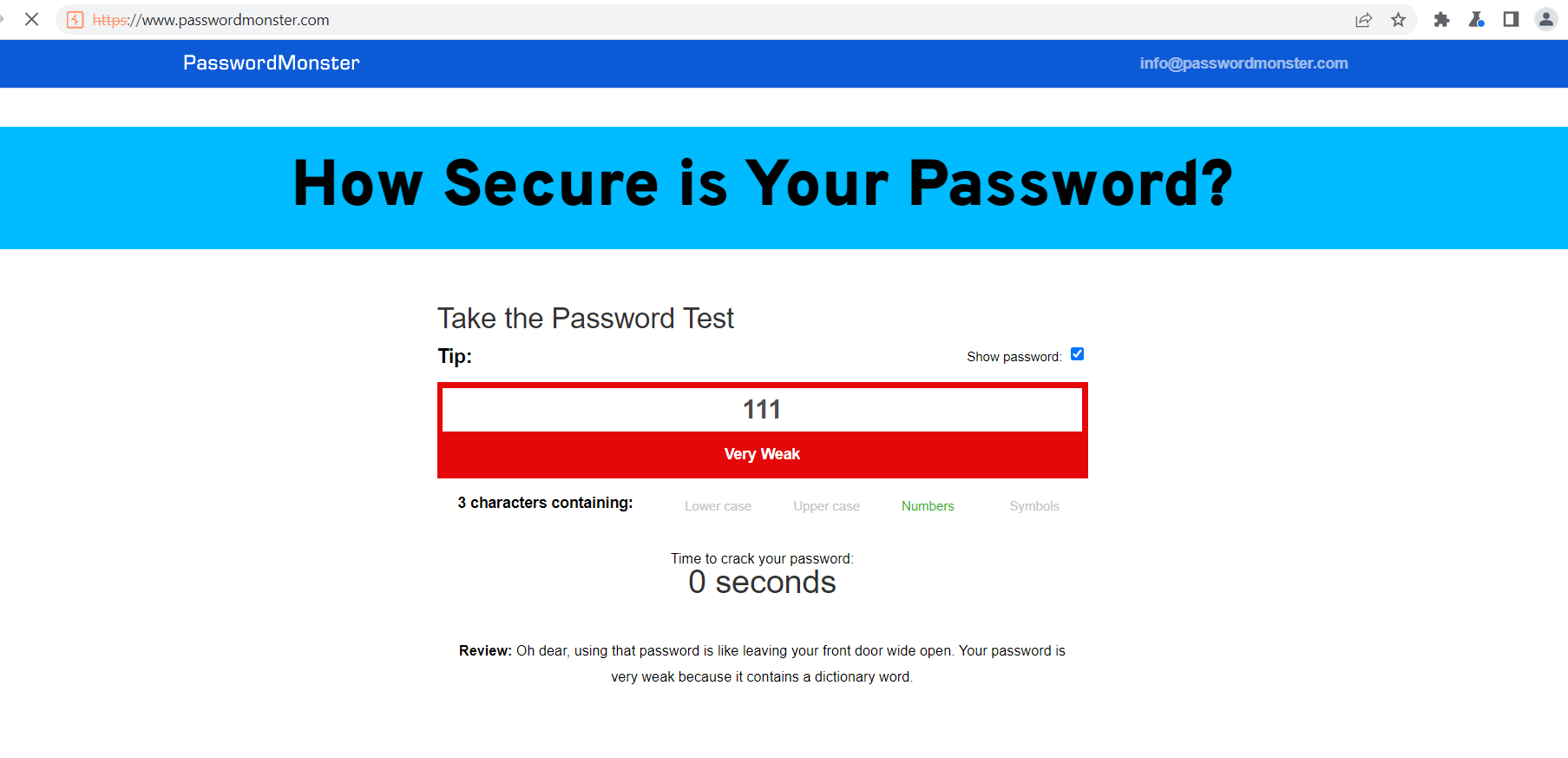


#### Khắc phục

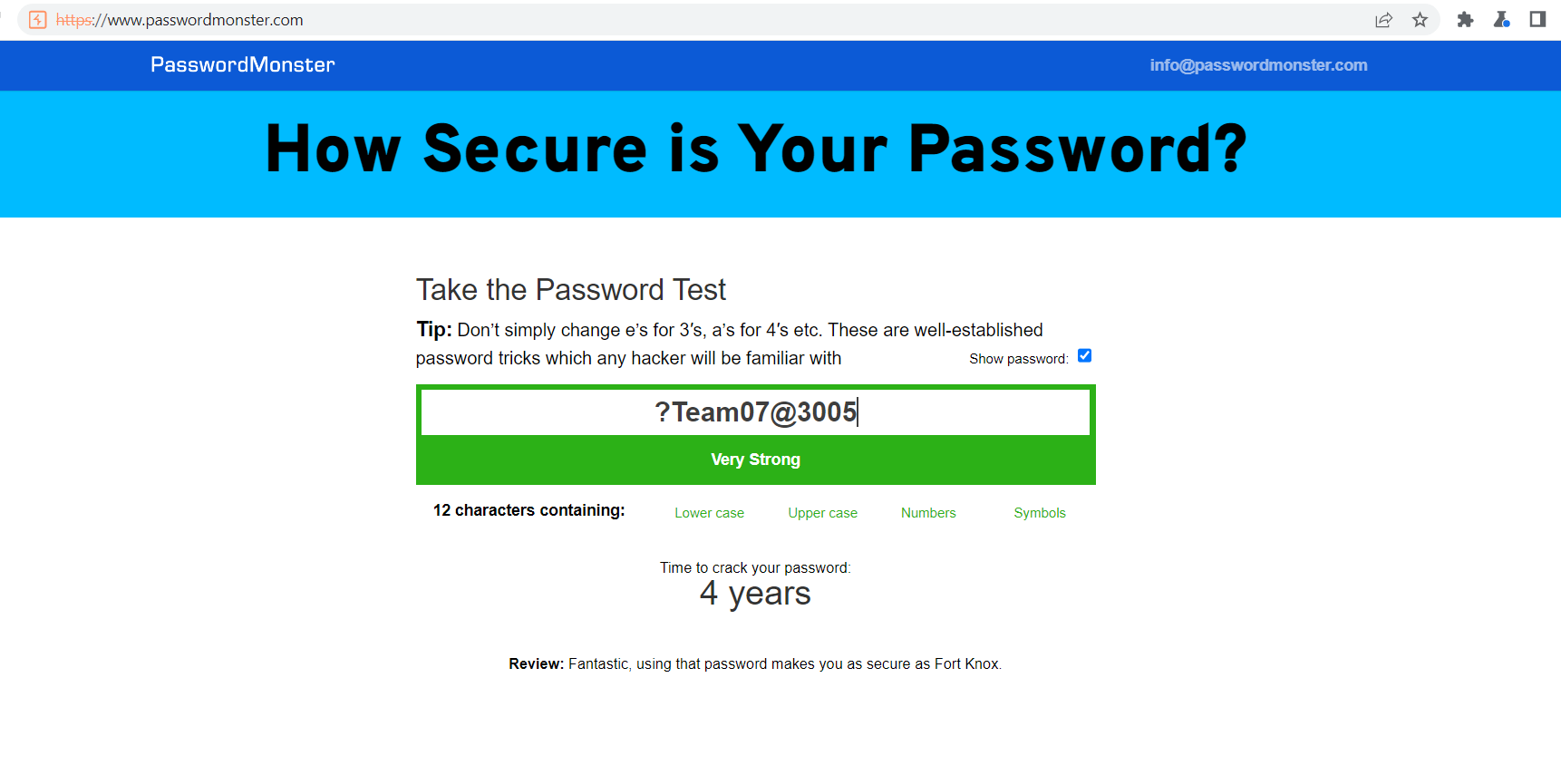
Đây là loại tấn công khai thác thông tin nhạy cảm của trang web. Hậu quả để lại rất nặng thế nhưng nhiều người dùng vẫn “Vô tư” đặc mật khẩu quá dễ đoán và dễ tấn công.

Để khắc phục được vấn đề này. Điều đầu tiên cần làm là nâng cao nhận thức của người dùng. Khiến họ thực sự cảm thấy việc đặc mật khẩu yếu kém là 1 mối đe dọa lớn đối với bản thân chính họ.

Tiếp theo, cần phải thay đổi mật khẩu mạnh hơn, ta có thể kiểm tra độ mạnh của mật khẩu ở một số trang web uy tín như: <https://www.passwordmonster.com/>:



Ta có thể thử đặt để có 1 mật khẩu đủ mạnh như:



Đồng thời đặt lại mật khẩu của admin:



Việc chọn 1 mật khẩu khó có thể tấn công rất quan trọng. Nó giúp phần ngăn chặn việc bị tấn công vét cạn và do ra mật khẩu.

## Quét bằng OWAS ZAP

### Các lỗi OWASP ZAP quét được



### Vulnerable Library

#### Khái niệm

Vulnerabe library là các thành phần thư viện chứa các lỗi bảo mật có thể bị tấn công bởi hacker. Ngày nay, các trang web đều sử dụng rất nhiều các gói thư viện được xây dựng sẵn từ các bên thứ 3. Nếu lập trình viên không chú ý kỹ khi sử dụng các thư viện này. Rất có thể các thư viện chứa các lỗ hổng bảo mật chưa được phát hiện hoặc được phát hiện và được vá ở các phiên bản tiếp theo. Nhưng trang web vẫn sử dụng các gói thư viện cũ và dẫn đến các lỗi không đáng có.

#### Lỗ hổng trong project

ZAP đã quét được 1 số thư viện chứa lỗ hổng bảo mật trong trang web như:



Lỗi thư viện boostrap-3.1.1.min.js

* Phiên bản hiện tại là 3.1.1, phiên bản này có thể bị tấn công XSS thông qua các:

+ Plugin: tooltip, collapse, scrollspy.

+ Attribute: data-target, data-viewport, data-template, data-content, data-title.

* Link tham khảo: <https://security.snyk.io/package/npm/bootstrap/3.1.1>

Lỗi thư viện jquery.min.js

* Phiên bản này chứa các lỗ hổng có thể bị tấn công qua:

+ Cross-Site Scripting (XSS): kẻ tấn công chèn mã độc hoặc mã độc dạng script vào trang web, khiến người dùng khác thực thi mã độc đó trong trình duyệt của họ.

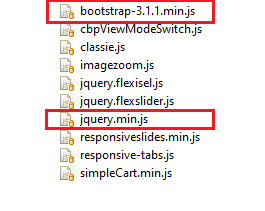
+ Denial-of-Service (DoS): Một lỗ hổng bảo mật trong jQuery có thể bị khai thác để gửi các yêu cầu không hợp lệ hoặc yêu cầu quá tải tới máy chủ, gây ra sự cố với dịch vụ hoặc hệ thống không thể sử dụng được.

+ Remote Code Execution (RCE): Nếu ứng dụng web chấp nhận và thực thi các lệnh trên máy chủ từ đầu vào không an toàn, kẻ tấn công có thể khai thác lỗ hổng này để thực thi mã nguy hiểm hoặc điều khiển máy chủ từ xa.

+ Prototype Polution: kẻ tấn công có thể sửa đổi Object.prototype

#### Khắc phục

Trang web sử dụng các thư viện cũ, lỗi thời, chứa các lỗ hổng đã được thông báo và sửa lỗi ở các phiên bản tiếp theo. Vì vậy, việc cập nhật phiên bản thư viện là cần thiết đối với 2 thư viện này.



Link tải phiên bản boostrap mới nhất: <https://getbootstrap.com/docs/5.3/getting-started/download/>

Link tải phiên bản jquery mới nhất: <https://code.jquery.com/jquery-3.7.0.min.js>

### Cross-Domain JavaScript Source File Inclusion

#### Khái niệm

Cross-Domain JavaScript Source File Inclusion là cảnh báo bảo mật có thể xảy ra khi trang web bao gồm và có khả năng chạy một hoặc nhiều tệp Javascript từ miền của bên thứ ba.

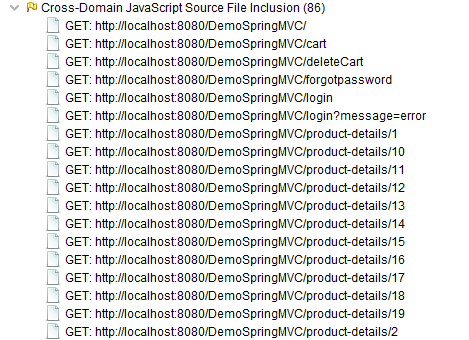
Nếu một ứng dụng web với các thẻ <script> lấy dữ liệu từ một miền khác, dữ liệu được lấy sẽ được chạy trên trang web như là một phần của trang web đó.

Trong một số trường hợp, kẻ tấn công có thể áp đặt mã hóa UTF-16 mặc dù bộ ký tự của thẻ <script> đã được đặt. Điều này giúp kẻ tấn công làm rò rỉ dữ liệu ở các định dạng khác nhau như JSON, XML, v.v. Khi người dùng gửi yêu cầu, tập lệnh sẽ được cập nhật cùng với thông báo phản hồi. Nếu phản hồi được lưu trữ trong các biến toàn cục, mọi người đều có thể đọc được. Nếu thông tin nhạy cảm được bao gồm trong phản hồi JSONP, thì hàm được thực thi có thể bị ghi đè để lấy thông tin nhạy cảm. Thủ thuật này cũng có thể được sử dụng cho các chức năng toàn cục. Thay vì ghi đè các hàm đã thực thi, chúng có thể sử dụng các hàm gọi lại được mã hóa tùy chỉnh cho các hàm toàn cục.

Khi một số tệp JavaScript ứng dụng của bạn nằm trên miền của bên thứ ba và không do bạn quản lý, kẻ tấn công có thể cố chiếm đoạt miền đó hoặc truy cập vào máy chủ của bên thứ ba đó để sửa đổi các tệp này và các tệp này có thể thực thi người dùng truy cập vào trang web.

#### Lỗ hổng trong project

ZAP đã quét được lỗ hổng bảo mật trong trang web như:



Nhóm đã sử dụng thẻ <script /> để lấy dữ liệu từ nguồn thứ 3 (google)



Ảnh hưởng của lỗ hổng:

* Có thể thực thi javascript độc hại.
* Thao tác và rò rỉ dữ liệu người dùng có thể xảy ra.
* Chức năng có thể thay đổi và chuyển hướng dữ liệu.
* Nhiễm phần mềm độc hại.

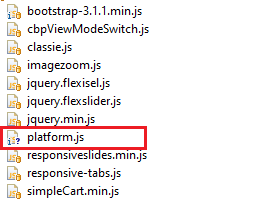
#### Khắc phục

Nguyên tắc chung để tránh khỏi tấn công từ lỗ hỏng này là luôn lưu trữ tất cả các tệp ứng dụng trên các máy chủ thuộc quyền quản lý hoặc dịch vụ bên thứ ba được công nhận và đáng tin cậy, công khai.

Tránh đặt thông tin nhạy cảm bên trong tệp javascript hoặc JSONP.

Sử dụng Content Security Policy (CSP): Đặt header CSP trong phản hồi HTTP để giới hạn nguồn và loại tệp JavaScript được chấp nhận trên trang web. CSP sẽ chỉ cho phép tải và thực thi tệp JavaScript từ các nguồn được xác định và từ các loại tệp cụ thể.

Nhóm đã sử dụng các phiên bản tệp JavaScript cục bộ: Thay vì nhúng tệp JavaScript từ nguồn bên thứ ba, nhóm đã tải tệp JavaScript cần thiết về máy chủ và nhúng nó từ nguồn cục bộ. Điều này đảm bảo rằng nhóm sẽ kiểm soát hoàn toàn tệp JavaScript và tránh tiềm năng của lỗ hổng Cross-Domain JavaScript Source File Inclusion.



Thay đổi đường dẫn tệp



### Absence of Anti-CSRF Tokens

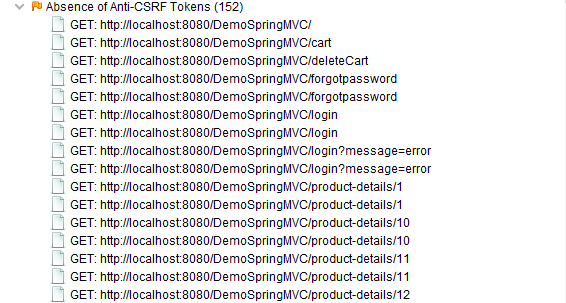
#### Khái niệm

Thiếu token xác thực CSRF là một lỗ hổng trong đó form với phương thức POST không được cài đặt các token xác thực CSRF. Điều này có thể dẫn đến việc hệ thống bị tấn công CSRF.

CSRF ( Cross Site Request Forgery) là kỹ thuật tấn công bằng cách sử dụng quyền chứng thực của người dùng đối với một website. CSRF là kỹ thuật tấn công vào người dùng, dựa vào đó hacker có thể thực thi những thao tác phải yêu cầu sự chứng thực. Hiểu một cách nôm na, đây là kỹ thuật tấn công dựa vào mượn quyền trái phép.

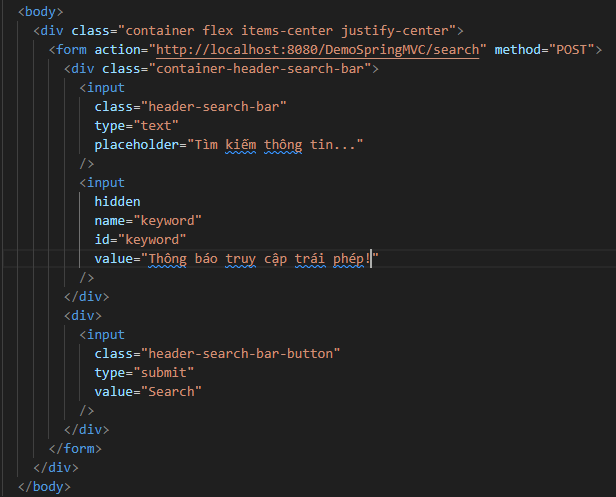
#### Tấn công project

Sử dụng công cụ ZAP, ta biết được rất nhiều request có thể tấn công CSRF. Ta sẽ tiến hành thử tấn công với request <http://localhost:8080/DemoSpringMVC/search>



Nhóm sẽ thực hiện demo để tấn công CSRF

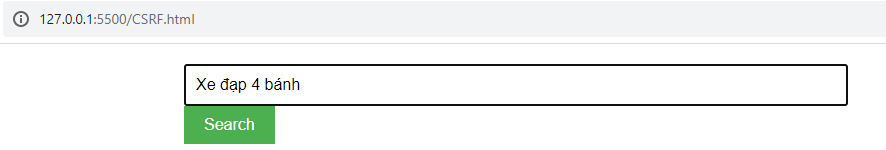
Đầu tiên, thực hiện tạo một trang web giả mạo chứa 1 form search: <http://127.0.0.1:5500/CSRF.html>



Trong form này, khi người dùng thực hiện tìm kiếm, hệ thống sẽ gửi 1 post request truyền keywork đến trang web với url: <http://localhost:8080/DemoSpringMVC/search>

Giả sử người dùng thực hiện tìm kiếm ở Website lừa đảo. Trang web sẽ tiến hành gửi request đó đến MeoMeo shop.

Người dùng thực hiện nhập nội dung tìm kiếm, đồng thời thực hiện click ô input “Search”



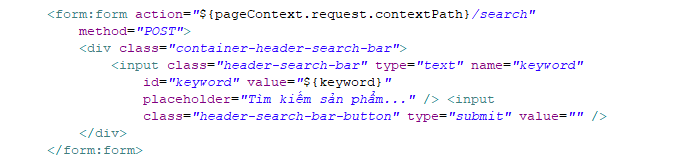
Trang web sẽ điều hướng đến trang search của website MeoMeo



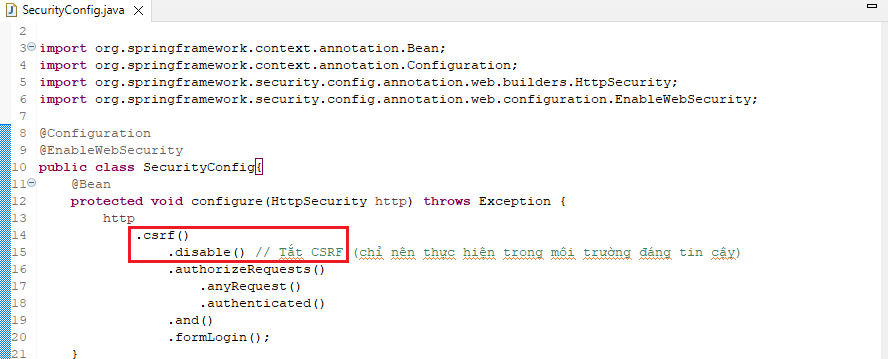
#### Nguyên nhân

Nguyên nhân xuất phát từ project chứa các form với phương thức là post không có CSRF token để phòng chống tấn công CSRF.

Form search không hề có CSRF token:

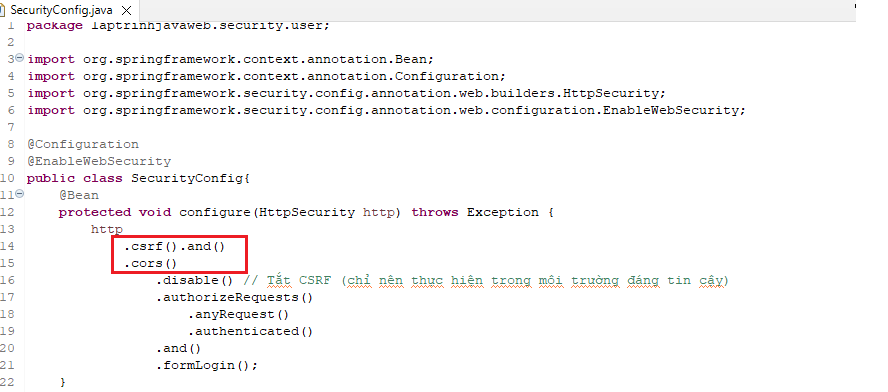


Trong file cấu hình bảo mật, cơ chế CSRF cũng bị tắt đi



#### Khắc phục

Để trang web không bị tấn công CSRF, ta cần bật chế độ bảo mật CSRF ( tắt disable ) ở trong file cài đặt:



### Cross-Site Scripting

#### Khái niệm

Cross-site scripting là một lỗ hổng phổ biến trong ứng dụng web. Để khai thác một lỗ hổng XSS, hacker sẽ chèn mã độc thông qua các đoạn script để thực thi chúng ở phía client. Có 3 loại Reflected XSS, Stored XSS và DOM-based XSS. Tấn công Cross Site Scripting nghĩa là gửi và chèn lệnh và script độc hại, những mã độc này thường được viết với ngôn ngữ lập trình phía client như Javascript, HTML, VBScript, Flash… Tuy nhiên, cách tấn công này thông thường sử dụng Javascript và HTML.

Nguyên nhân chính của loại tấn công này là xác thực đầu vào dữ liệu người dùng không phù hợp, dữ liệu độc hại từ đầu vào có thể xâm nhập vào dữ liệu đầu ra. Mã độc có thể nhập một script và được chèn vào mã nguồn của website. Khi đó trình duyệt không thể biết mã thực thi có phải độc hại hay không. Do đó mã độc hại có thể đang được thực thi trên trình duyệt của nạn nhận hoặc bất kỳ hình thức giả nào đang được hiển thị cho người sử dụng.

#### Các loại tấn công XSS

Có ba loại chính của tấn công Cross-Site Scripting (XSS), đó là:

Stored XSS (XSS lưu trữ): Đây là loại tấn công XSS phổ biến nhất. Trong trường hợp này, mã độc được chèn vào cơ sở dữ liệu hoặc hệ thống và được hiển thị cho người dùng khi truy cập vào trang web hoặc ứng dụng. Điều này xảy ra khi dữ liệu độc hại được lưu trữ và đưa ra cho người dùng mỗi khi trang web được tải lại hoặc hiển thị.

Reflected XSS (XSS phản ánh): Loại tấn công này xảy ra khi dữ liệu độc hại được chèn vào đường dẫn URL hoặc các thông số của biểu mẫu và được phản ánh trực tiếp trở lại người dùng. Đường dẫn URL hoặc thông số độc hại này sau đó được nhúng và thực thi trong mã HTML, khiến cho trình duyệt hiểu sai và thực thi mã độc.

DOM-based XSS: Đây là loại tấn công XSS xảy ra trong phần mã lưu trữ và thực thi trên phía máy khách (client-side). Thay vì chèn mã độc vào mã HTML phản ánh hoặc lưu trữ, loại tấn công này sẽ tác động trực tiếp vào cây DOM (Document Object Model) của trang web và thay đổi cấu trúc DOM để thực thi mã độc.

Đối với cả ba loại tấn công XSS này, mục tiêu chính của kẻ tấn công là chèn mã JavaScript độc hại hoặc các đoạn mã khác vào trang web hoặc ứng dụng để thực hiện các hành động độc hại, như đánh cắp thông tin người dùng, chuyển hướng trang, hoặc thực thi các hành động không mong muốn trên trình duyệt của người dùng.

#### Giải pháp ngăn chặn

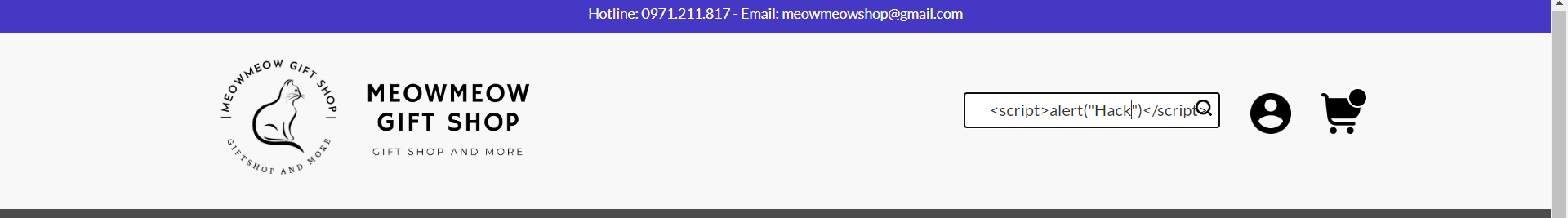
Để ngăn chặn tấn công Cross-Site Scripting (XSS), bạn có thể thực hiện các biện pháp bảo mật sau đây:

1. Validation (Xác thực dữ liệu): Kiểm tra và xác thực dữ liệu đầu vào từ người dùng trước khi chấp nhận nó. Hãy đảm bảo rằng chỉ chấp nhận dữ liệu hợp lệ và loại bỏ hoặc mã hóa các ký tự đặc biệt, như dấu ngoặc kép, dấu gạch chéo ngược, và ký tự đặc biệt HTML/XML khác.
2. Escape (Tránh thoát ký tự): Mã hóa hoặc thoát các ký tự đặc biệt trong dữ liệu trước khi hiển thị nó trong trình duyệt. Điều này đảm bảo rằng các ký tự đặc biệt sẽ không được hiểu là mã HTML hoặc JavaScript và sẽ không thực thi mã độc hại.
3. Content Security Policy (CSP): Đặt chính sách bảo mật nội dung (Content Security Policy) để hạn chế hoặc ngăn chặn việc tải và thực thi các tài nguyên không an toàn từ các nguồn không đáng tin cậy. CSP cho phép bạn kiểm soát chính xác loại tài nguyên (như script, font, hình ảnh) được phép tải xuống và từ đâu.
4. Input Sanitization (Lọc dữ liệu đầu vào): Xóa hoặc mã hóa các ký tự đặc biệt trong dữ liệu đầu vào để ngăn chặn việc chèn mã độc. Có thể sử dụng thư viện bảo mật đã được xây dựng sẵn để thực hiện việc lọc dữ liệu.
5. Sử dụng hàm Escape khi xuất dữ liệu: Trước khi hiển thị dữ liệu người dùng lên trình duyệt, hãy sử dụng hàm escape/encode thích hợp (như htmlspecialchars trong PHP) để đảm bảo rằng dữ liệu được hiển thị mà không thực thi các mã độc hại.
6. HTTPS: Sử dụng kết nối HTTPS an toàn để truyền tải dữ liệu giữa máy khách và máy chủ. Điều này giúp ngăn chặn tấn công Man-in-the-Middle và bảo vệ dữ liệu truyền qua mạng.
7. Regular Expression (Regex): Sử dụng biểu thức chính quy (regex) để xác thực và kiểm tra tính hợp lệ của dữ liệu đầu vào. Điều này giúp ngăn chặn việc chèn mã độc qua các chuỗi không hợp lệ.
8. Sử dụng Web Application Firewall (WAF): Đặt một tường lửa ứng dụng web để giám sát và ngăn chặn các tấn công XSS. WAF có thể xác định và chặn các mẫu tấn công phổ biến dựa trên quy tắc được cấu hình.

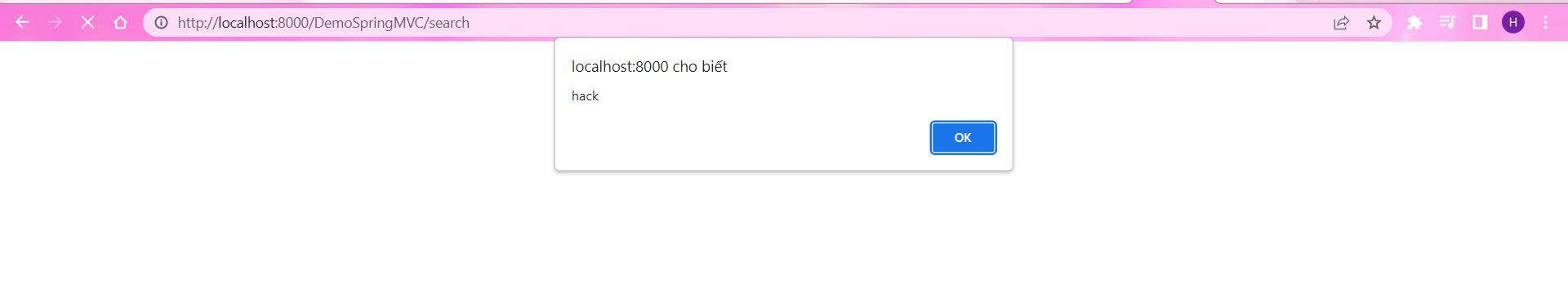
#### Tấn công project với XSS

Chèn các đoạn script vào những nơi có nhập input người dùng như thanh tìm kiếm sản phẩm, hệ thống web đều thực thi những đoạn lệnh này do chưa có kiểm tra lọc đầu vào hợp lệ. Như vậy hệ thống bị tấn công XSS , những đoạn mã được lưu dưới như câu lệnh script . Như hình dưới đây:

Chèn <script>alert("hack")</script> vào thanh kiếm



Nhấn nút tìm kiếm, ta sẽ thấy xuất hiện alert 1 thông tin nào đó kẻ tấn công muốn đánh cắp .

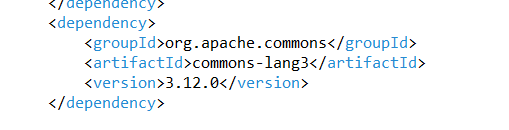


Hay chèn một form lên thanh tìm kiếm : chèn <input></input>

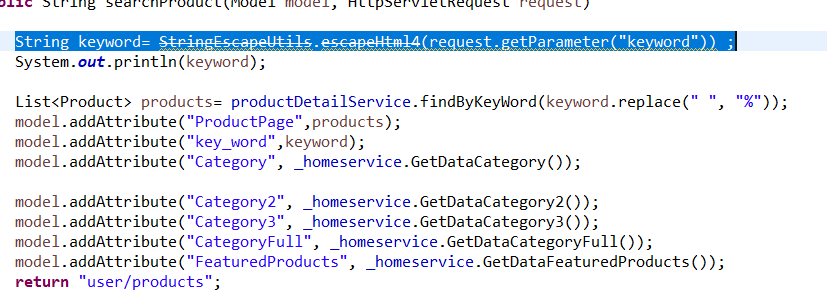


- Khắc phục :

Phía server : sử dụng StringEscapeUtils từ Apache Commons Lang, với hàm escapeHtml4(String) sẽ lọc những kí tự < , > , “ , ”, ‘ ,’,… và thay thế bằng các kí tự khác như &lt; , &quot; ,…



Thêm escapeHtml() khi lấy parameter nội dung search từ phía client trong class ProductController



### 2.6. Cookie without SameSite Attribute

#### 2.5.1. Khái niệm

SameSite Cookie là thuộc tính thêm vào trong cookie để yêu cầu chủ sở hữu trang web ghi rõ ràng nhãn cookie của Web khác, nhờ đó chỉ có thể chia sẻ Cookie với các trang web này. Như vậy một website lừa đảo không thể giả mạo người dùng vì không thể lấy được Cookie.

Lợi ích của SameSite giúp cho việc chống tấn công CSFR dễ dàng và hiệu quả hơn. Người làm hệ thống sẽ giảm chi phí do không cần bổ sung tính năng tạo và tương tác thông qua token ở client và server. Hiệu năng của hệ thống được nâng cao khi không phải sinh và đối chiếu token.

SameSite là một thuộc tính của Set-Cookie trong Http response header. Có 3 giá trị:

* SameSite = Strict

Giá trị của Strict phòng ngừa các web giả mạo mạnh mẽ. Trình duyệt sẽ ngăn cản dữ liệu cookie chuyển giữa các tên miền chéo khác trong mọi trường hợp. Cookie sẽ không được gửi cùng với các request được bắt đầu với các trang web thứ 3. Đặt cookie là Strict có thể ảnh hưởng tiêu cực đến trải nghiệm duyệt web. Ví dụ: nếu bạn nhấp vào 1 liên kết dẫn đến trang profile của Facebook, và Facebook.com đặt cookie của nó là SameSite=Strict thì bạn không thể tiếp tục redirect trên Facebook trừ khi bạn đăng nhập lại vào Facebook. Lý do là vì Cookie của Facebook không được gửi kèm với request này.

* SameSite = Lax

Trình duyệt cho phép chia sẻ cookie giữa các trang web có cùng miền, cookie sẽ được gửi cùng với GET request được tạo bởi bên thứ 3.

* SameSite = None

#### 2.5.2. Khắc phục

Phần này tạo thêm 1 class SameSiteCookieInterceptor

**public** **class** SameSiteCookieInterceptor **implements** HandlerInterceptor {

@Override

**public** **boolean** preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) **throws** Exception {

Cookie[] cookies = request.getCookies();

**if** (cookies != **null**) {

**for** (Cookie cookie : cookies) {

**if** (cookie != **null**) {

cookie.setSecure(**true**);

cookie.setHttpOnly(**true**);

// Thêm SameSite attribute vào Cookie

cookie.setPath("/\*");

//cookie.setValue(cookie.getValue() + "; SameSite=Strict");

response.addCookie(cookie);

response.setHeader("Set-Cookie", cookie.getName() + "=" + cookie.getValue() + "; SameSite=Strict");

}

}

}

**return** **true**;

}

@Override

**public** **void** postHandle(HttpServletRequest request, HttpServletResponse response, Object handler,

ModelAndView modelAndView) **throws** Exception {

// Xử lý logic SameSite Cookie tại đây

Cookie[] cookies = request.getCookies();

**if** (cookies != **null**) {

**for** (Cookie cookie : cookies) {

**if** (cookie != **null**) {

cookie.setSecure(**true**);

cookie.setHttpOnly(**true**);

// Thêm SameSite attribute vào Cookie

cookie.setPath("/\*");

//cookie.setValue(cookie.getValue() + "; SameSite=None");

response.addCookie(cookie);

response.setHeader("Set-Cookie", cookie.getName() + "=" + cookie.getValue() + "; SameSite=Strict");

}

}

}

}

@Override

**public** **void** afterCompletion(HttpServletRequest request, HttpServletResponse response, Object handler,

Exception ex) **throws** Exception {

// Xử lý logic SameSite Cookie tại đây

Cookie[] cookies = request.getCookies();

**if** (cookies != **null**) {

**for** (Cookie cookie : cookies) {

**if** (cookie != **null**) {

cookie.setSecure(**true**);

cookie.setHttpOnly(**true**);

// Thêm SameSite attribute vào Cookie

cookie.setPath("/\*");

response.addCookie(cookie);

response.setHeader("Set-Cookie", cookie.getName() + "=" + cookie.getValue() + "; SameSite=Strict");

}

}

}

}

}

Mục đích của class này là tạo chuỗi path thêm vào cookie, thay vì những cách khác là set thẳng từng giá trị vào response thì class SameSiteCookieInterceptor giúp thêm giá trị của SameSite vào một đối tượng cookie đã được tạo.

## Các lỗ hổng không tìm thấy

### SQL Injection

#### Khái niệm

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 để làm sai lệnh đi câu truy vấn ban đầu, từ đó có thể khai thác dữ liệu từ database.

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.

SQL injection có thể gây ra những tác hại như:

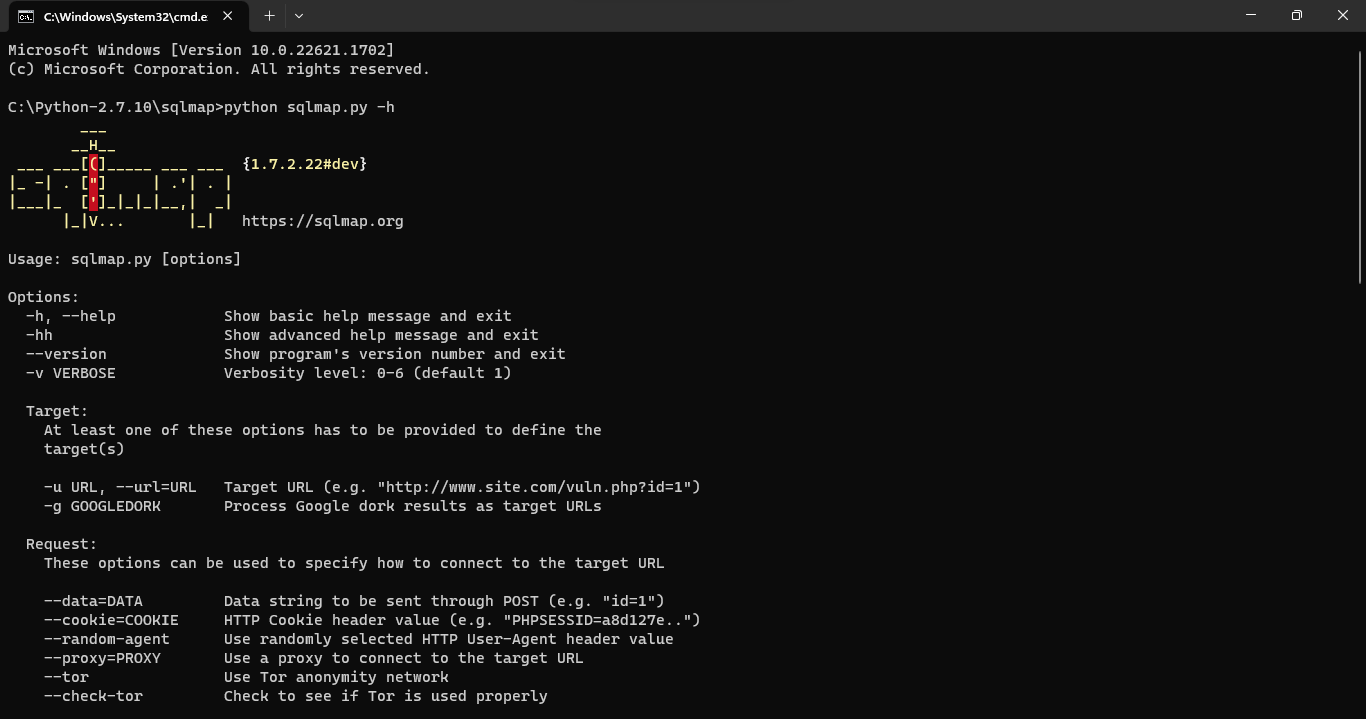
* Hack tài khoản cá nhân.
* Ăn cắp hoặc sao chép dữ liệu của trang web hoặc hệ thống.
* Thay đổi dữ liệu nhạy cảm của hệ thống.
* Xóa dữ liệu nhạy cảm và quan trọng của hệ thống.
* Người dùng có thể đăng nhập vào ứng dụng với tư cách người dùng khác, ngay cả với tư cách quản trị viên.

#### Quét bằng SQLMap

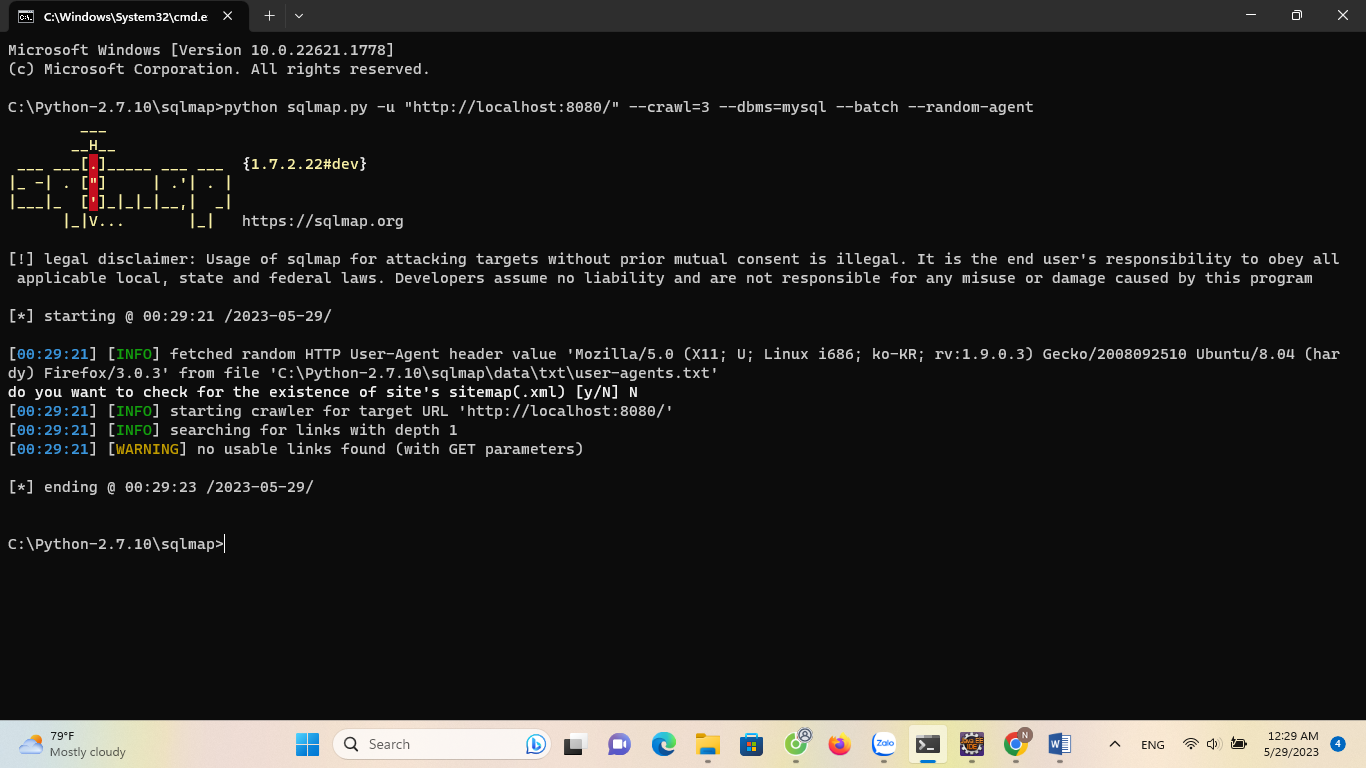
SQLMAP là công cụ khai thác những lổ hỏng của cơ sở dữ liệu SQL. Công cụ này được xem là công cụ khai thác SQL tốt nhất hiện nay. Được giới bảo mật và giới hacker sử dụng thường xuyên.

Đây là công cụ mã nguồn mở, tự động hóa quá trình phát hiện và khai thác lỗ hổng SQL. Nó đi kèm với một công cụ phát hiện mạnh mẽ, nhiều tính năng thích hợp cho trình kiểm tra thâm nhập cuối cùng.

Để xem các lệnh có thể sử dụng dùng câu lệnh: python sqlmap.py -h



Để tiến hành quét tất cả URL có trên đường dẫn là <http://localhost:8080/>, ta sử dụng câu lệnh: python sqlmap.py -u "http://localhost:8080/" --crawl=3 --dbms=mysql --batch --random-agent



SqlMap không quét ra đường dẫn nào để thực hiên quét lỗi sql ịnection

#### Sự bảo vệ của Spring Framework

Spring Framework cung cấp nhiều cơ chế bảo vệ để phòng ngừa các cuộc tấn công SQL Injection.

Prepared Statements: Spring JDBC hỗ trợ việc sử dụng Prepared Statements để thực hiện các truy vấn SQL. Prepared Statements sử dụng các tham số được truyền vào truy vấn SQL, giúp tránh việc nối chuỗi các tham số vào truy vấn SQL một cách rõ ràng. Điều này giúp ngăn chặn việc nhập các giá trị nguy hiểm vào câu truy vấn SQL.

SQL Injection Escape Functions: Spring Framework cung cấp các hàm escape để xử lý chuỗi đầu vào từ người dùng trước khi nó được sử dụng trong câu truy vấn SQL. Ví dụ, JdbcTemplate trong Spring cung cấp các phương thức như queryForList, queryForObject, và update mà tự động áp dụng các quy tắc escape để ngăn chặn SQL Injection.

Validation và Binding: Spring hỗ trợ các cơ chế validation và binding để đảm bảo dữ liệu được kiểm tra và ràng buộc trước khi được sử dụng trong câu truy vấn SQL. Các công cụ như Spring Data JPA cung cấp khả năng kiểm tra tích hợp và ràng buộc dữ liệu từ người dùng trước khi nó được sử dụng để tạo truy vấn SQL.

Security Framework: Spring Security là một phần mở rộng quan trọng của Spring Framework, cung cấp cơ chế bảo mật mạnh mẽ cho ứng dụng. Nó hỗ trợ các tính năng như kiểm tra xác thực, quản lý phiên, phân quyền và phòng ngừa tấn công CSRF (Cross-Site Request Forgery), giúp ngăn chặn các cuộc tấn công SQL Injection và các cuộc tấn công khác.

# TÀI LIỆU THAM KHẢO

1. Cách fix lỗi thư viện boostrap danh cho các phiên bản cũ tại: <https://security.snyk.io/package/npm/bootstrap/3.1.1>
2. Cross Domain JavaScript Source File Inclusion tại: <https://beaglesecurity.com/blog/vulnerability/cross-domain-javascript-source-file-inclusion.html>
3. Kỹ thuật tấn công XSS và cách ngăn chặn tại:

<https://viblo.asia/p/ky-thuat-tan-cong-xss-va-cach-ngan-chan-YWOZr0Py5Q0>

1. Cross Site Request Forgery (CSRF) tại:

<https://viblo.asia/p/spring-security-cross-site-request-forgery-csrf-vyDZOyjG5wj>

1. Brute Force, Brute Force Attack Là Gì? tại:

<https://hostingviet.vn/brute-force-brute-force-attack-la-gi>