CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

1.1. **Lý do chọn đề tài**

Việc chọn đề tài "**Thiết kế và xây dựng Website quyên góp và ủng hộ cộng đồng bằng ReactJS**" này có thể được thực hiện với nhiều lý do khác nhau, tùy thuộc vào mục tiêu và lợi ích phục vụ cộng đồng:

Thị trường cần thiết: Nhu cầu về thông tin luôn tồn tại trong mọi lĩnh vực. Dựa trên sự phát triển của Internet và sự phổ biến của thiết bị di động, người dùng ngày càng có thêm lựa chọn để truy cập thông tin. Tạo một Website này có thể đáp ứng nhu cầu này.

Phát triển kỹ năng: Xây dựng một Website đòi hỏi bạn phải học và phát triển nhiều kỹ năng khác nhau, bao gồm thiết kế web, lập trình, quản lý nội dung, SEO, và quảng cáo trực tuyến. Việc này có thể giúp bạn phát triển nhiều kỹ năng quan trọng trong thời đại số hóa.

Tạo nội dung giá trị: Website này có thể giúp bạn thể hiện kiến thức và quan điểm của mình về các sự kiện và chủ đề quan trọng. Bạn có thể tạo nội dung giúp người đọc hiểu rõ hơn về thế giới xung quanh và thúc đẩy thảo luận xã hội.

Tiềm năng kiếm tiền: Nếu bạn có một lượng lớn lượt truy cập, bạn có thể cân nhắc kiếm tiền từ Website thông qua quảng cáo, chương trình liên kết, hoặc thuê viết bài quảng cáo. Điều này có thể biến đề tài này thành một nguồn thu nhập thụ động.

Gắn kết cộng đồng: Một Website có thể giúp bạn xây dựng cộng đồng trực tuyến quan tâm đến các chủ đề cụ thể. Điều này có thể tạo ra cơ hội để giao tiếp và hợp tác với những người có cùng sở thích và quan điểm.

Đóng góp vào lĩnh vực bạn quan tâm: Nếu bạn có niềm đam mê về một chủ đề cụ thể, việc tạo ra một Website có thể giúp bạn đóng góp vào lĩnh vực này và chia sẻ thông tin hữu ích với mọi người.

- Tính cấp thiết của đề tài:
  + Giúp người đọc có thể tiếp cận được những bài viết
  + Có thể thoải mái Donate

Những bất cập, hạn chế của hệ thống cũ:

+ Giao diện chưa được ưa nhìn

1.2. **Mục tiêu của đề tài**

Mục tiêu của đề tài là Thiết kế và xây dựng website quyên góp và ủng hộ cộng đồng bằng ReactJS.

1.3. **Đối tượng nghiên cứu**

Các đối tượng của đề tài gồm có:

- Khách hàng
- Các loại quyên góp
- Các nhà hảo tâm

1.4. **Phạm vi đề tài**

Thiết kế và xây dựng website quyên góp và ủng hộ cộng đồng bằng ReactJS gồm có các chức năng như:

- Đăng nhập hệ thống, đăng ký tài khoản, đăng xuất tài khoản
- Lịch sử donate (Lịch Sử Quyên Góp)
- Donate
- Pagination
- Row per page
- Thêm, xóa, sửa, cập nhật tài khoản người dùng của trang admin

1.5. **Cấu trúc của đề tài**

Chương 1: Tổng quan về đề tài

Chương 2: Cơ sở lý thuyết

Chương 3: Phân tích và thiết kế hệ thống

Chương 4: Thực nghiệm và triển khai

Kết luận

Kết quả đạt được

Hướng phát triển

Tài liệu tham khảo

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. **Giới thiệu về JSX**

```
var Hello = React.createClass({
    render: function() {
        return (
            <div>
                <h1>Hello from Kode</h1>
                <p>Maybe you prefer a hello from Adele</p>
            </div>
            );
    }
});

ReactDOM.render(<Hello/>, document.getElementById('react-target'));
```

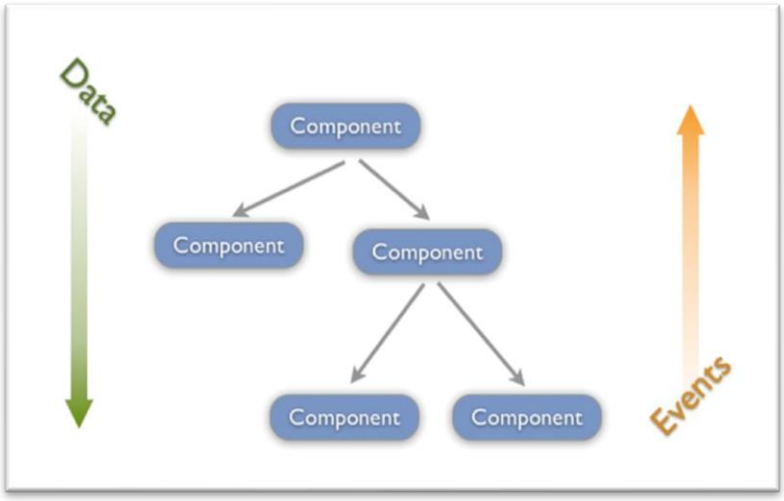
Hình 2.1: JSX

Trong React, thay vì thường xuyên sử dụng JavaScript để thiết kế trang web cục bộ thì sẽ sử dụng JSX. JSX được đánh giá là sử dụng JavaScript đơn giản hơn và cho phép trích dẫn HTML cũng như sử dụng các cú pháp thẻ HTML để hiển thị các thành phần phụ. JSX tối ưu hóa mã khi biên dịch, vì vậy nó chạy nhanh hơn so với mã JavaScript tương đương

Nó được gọi là JSX, là một cú pháp mở rộng cho JavaScript. Chúng tôi khuyến khích sử dụng JSX với React để mô tả giao diện (UI). JSX có thể trông giống Ngôn ngữ Khuôn mẫu (Template language), nhưng JSX đi kèm với toàn bộ tính năng của JavaScript.

React Không bắt buộc sử dụng JSX, nhưng phần lớn mọi người đều cho rằng nó hữu dụng khi làm việc với giao diện (UI) trong mã JavaScript. JSX cũng cho phép React hiển thị những thông báo lỗi và "lời cảnh báo" (warning) hữu ích hơn.

2.2. **Luồng dữ liệu một chiều**

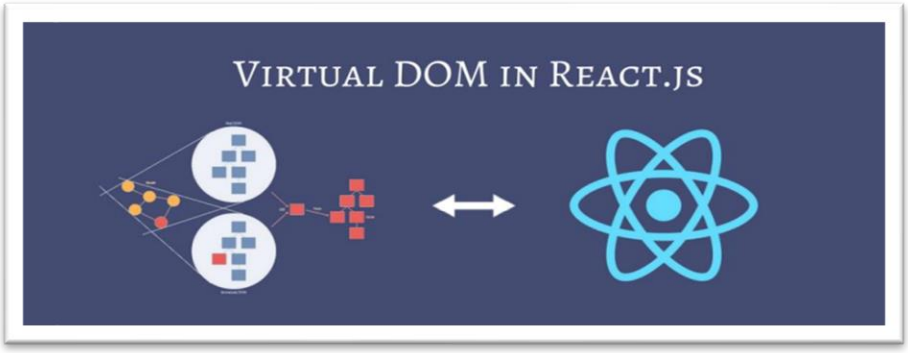


Hình 2.2: Luồng dữ liệu một chiều

ReactJS không có các mô-đun chuyên dụng để xử lý dữ liệu, vì vậy ReactJS chia nhỏ chế độ xem thành các thành phần nhỏ có mỗi hệ thống chặt chẽ với nhau. Tại sao chúng ta phải quan tâm đến cấu trúc và mối quan hệ giữa các thành phần trong ReactJS?

Câu trả lời chính là luồng truyền dữ liệu trong ReactJS: Luồng dữ liệu theo chiều từ cha sang con. Việc ReactJS sử dụng luồng dữ liệu một chiều có thể gây ra một chút khó khăn cho những người muốn tìm hiểu và ứng dụng vào trong các dự án. Tuy nhiên, cơ chế này sẽ phát huy được ưu điểm của mình khi cấu trúc cũng như chức năng của chế độ xem trở nên phức tạp thì ReactJS sẽ phát huy được vai trò của mình

2.3. **DOM ảo**

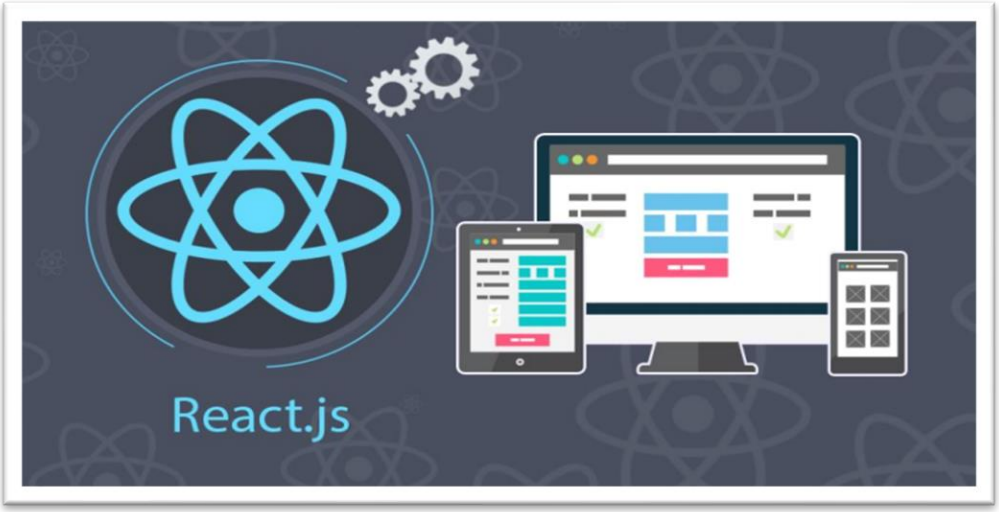


Hình 2.3: Dom ảo

Những Framework sử dụng Virtual-DOM như ReactJS khi Virtual-DOM thay đổi, chúng ta không cần thao tác trực tiếp với DOM trên View mà vẫn phản ánh được sự thay đổi đó. Do Virtual-DOM vừa đóng vai trò là Model, vừa đóng vai trò là View nên mọi sự thay đổi trên Model đã kéo theo sự thay đổi trên View và đảo ngược. Có nghĩa là mặc dù chúng ta không tác động trực tiếp vào các phần tử DOM trong Chế độ xem nhưng vẫn thực hiện được cơ chế ràng buộc dữ liệu. Điều này làm cho tốc độ ứng dụng tăng lên đáng kể – một lợi thế không thể tuyệt vời hơn khi sử dụng Virtual-DOM.

2.4. **Tại sao ReactJs lại lại sự chọn lựa hàng đầu cho doanh nghiệp**

Trong lĩnh vực phát triển công nghệ, các chủ doanh nghiệp và nhà phát triển luôn tìm kiếm những phương pháp tốt nhất để giúp doanh nghiệp của họ có được những lợi thế cạnh tranh tốt hơn. Và một trong những công nghệ tốt nhất có thể giúp những doanh nghiệp mặt đối thủ trong việc tạo ra những ứng dụng web chính là ReactJS



Hình 2.4: Hình minh họa ReactJS

ReactJS cho phép các doanh nghiệp tạo ra các ứng dụng web với giao diện người dùng tốt hơn để nâng cao trải nghiệm người dùng. Đây cũng chính là công nghệ mà các doanh nghiệp cần phải có được lượng tương tác của người dùng, tỷ lệ nhấp chuột cũng như chuyển đổi cao hơn. Hơn nữa, các doanh nghiệp sử dụng ReactJS chắc chắn có giao diện tốt hơn so với các doanh nghiệp sử dụng các framework khác nhờ ReactJS giúp ngăn chặn việc cập nhật của DOM giúp ứng dụng nhanh hơn và truyền tải UX tốt hơn.

2.4.1 **Tương lai của ReactJs**

Facebook và toàn bộ nhóm phát triển ReactJS luôn cố gắng chứng minh trong việc cam kết nâng cao hiệu quả tính toán của ReactJS. Đây là vấn đề tiên quyết để vượt qua sự phát triển nhanh chóng của các framework khác như Vue.js. Một số kỳ vọng cập nhật của React được mong đợi trong tương lai có thể kể đến như: Sẽ có những loại kết xuất mới như việc thêm những pha cú pháp độc đáo vào JSX mà không cần đến các phím. Cải thiện trong công việc xử lý các lỗi phát sinh. Trước đây, các lỗi Javascript bên trong các Thành phần sẽ ảnh hưởng đến trạng thái của thành phần và cũng gây ra các lỗi trong quá trình kết xuất trong các thành phần khác. Những lỗi này được thông báo là rất khó hiểu gây ra khó khăn trong công việc giải quyết. Một vấn đề khác là trong các phiên bản trước đây, React không cung cấp cách thức để có thể bắt và xử lý lỗi và phục hồi khi lỗi xảy ra một cách rõ ràng trong Thành phần.

2.4.2 **Quy trình phát triển phần mềm**

Quy trình phát triển phần mềm là một cấu trúc bao gồm tập hợp các thao tác và các kết quả tương quan sử dụng trong việc phát triển để sản xuất ra một phần mềm.

Một cách đơn giản, chúng ta có thể chia quy trình phát triển phần mềm ra làm 4 giai đoạn:

Đặc tả phần mềm: là tiến trình để hiểu và xác định những dịch vụ nào cần có trong hệ thống, những ràng buộc đối với việc phát triển và chức năng của hệ thống. Tiến trình này sẽ sinh ra các tài liệu yêu cầu (bản đặc tả hệ thống).

Thiết kế và thực thi phần mềm: liên quan đến việc chuyển những yêu cầu phần mềm thành hệ thống có thể thực thi được

Thiết kế phần mềm: là việc mô tả cấu trúc phần mềm, dữ liệu của hệ thống, giao diện giao tiếp giữa các thành phần, thuật toán được sử dụng, …

Thực thi phần mềm: Các lập trình viên dùng các ngôn ngữ lập trình để viết lệnh (source) thực thi để tạo ra hệ thống dựa trên các bản đặc tả thiết kế chi tiết, đồng thời tiến hành các thử nghiệm (test case) với dữ liệu giả định.

Kiểm thử phần mềm: là quá trình vận hành chương trình để tìm ra lỗi

Cài đặt và bảo trì phần mềm:

Cài đặt và triển khai hệ thống vừa phát triển để người dùng có thể sử dụng được.

Bảo trì phần mềm: điều chỉnh các lỗi chưa được phát hiện trong các giai đoán trước, nâng cao tính năng sử dụng và an toàn vận hành của phần mềm, đảm bảo cho việc phần mềm được cập nhật khi môi trường và yêu cầu của người sử dụng thay đổi.

2.5. **Phương pháp phân tích thiết kế hướng đối tượng**

2.5.1 **Khái niệm về Phân tích và thiết kế hướng đối tượng (Object Oriented Analysis and Design: OOAD)**

Trong kỹ nghệ phần mềm để sản xuất được một sản phẩm phần mềm người ta chia quá trình phát triển sản phẩm ra nhiều giai đoạn như thu thập và phân tích yêu cầu, phân tích và thiết kế hệ thống, phát triển (coding), kiểm thử, triển khai và bảo trì. Trong đó, giai đoạn phân tích, thiết kế bao giờ cũng là giai đoạn khó khăn và phức tạp nhất. Giai đoạn này giúp chúng ta hiểu rõ yêu cầu đặt ra, xác định giải pháp, mô tả chi tiết giải pháp. Nó trả lời 2 câu hỏi What (phần mềm này làm cái gì?) và How (làm nó như thế nào?).

Để phân tích và thiết kế một phần mềm thì có nhiều cách làm, một trong những cách làm đó là xem hệ thống gồm những đối tượng sống trong đó và tương tác với nhau. Việc mô tả được tất cả các đối tượng và sự tương tác của chúng sẽ giúp chúng ta hiểu rõ hệ thống và cài đặt được nó. Phương thức này gọi là Phân tích thiết kế hướng đối tượng (OOAD).

2.5.2 **Giới thiệu về UML (Unified Modeling Language)**

UML (Unified Modeling Language) là ngôn ngữ dành cho việc đặc tả, hình dung, xây dựng và làm tài liệu của các hệ thống phần mềm.

UML tạo cơ hội để viết thiết kế hệ thống, bao gồm những khái niệm như tiến trình nghiệp vụ và các chức năng của hệ thống.

Cụ thể, nó hữu dụng cho những ngôn ngữ khai báo, giản đồ cơ sở dữ liệu, thành phần phần mềm có khả năng tái sử dụng.

UML được phát triển bởi Rational Rose và một số nhóm cộng tác, nó nhanh chóng trở thành một trong những ngôn ngữ chuẩn để xây dựng hệ thống phần mềm hướng đối tượng (Object-Oriented).

Đây là ngôn ngữ kế vị xứng đáng cho những ngôn ngữ mô hình hoá như Booch, OOSE/Jacobson, OMT và một số các phương thức khác.

UML cung cấp cho người dùng một ngôn ngữ mô hình hoá trực quan sẵn sàng để dùng và có ý nghĩa:

Cho phép phát triển và trao đổi những mô hình mang nhiều ý nghĩa.

Cung cấp khả năng mở rộng và chuyên môn hoá để mở rộng những khái niệm cốt lõi.

Độc lập với ngôn ngữ lập trình chuyên biệt và các tiến trình phát triển.

Cung cấp nền tảng về sự hiểu biết ngôn ngữ mô hình hoá.

Khuyến khích và hỗ trợ sự phát triển của các công cụ hướng đối tượng.

Hỗ trợ những khái niệm phát triển cấp độ cao như collaboration, framework, pattern and component.

Tích hợp một cách tốt nhất với thực tiễn.

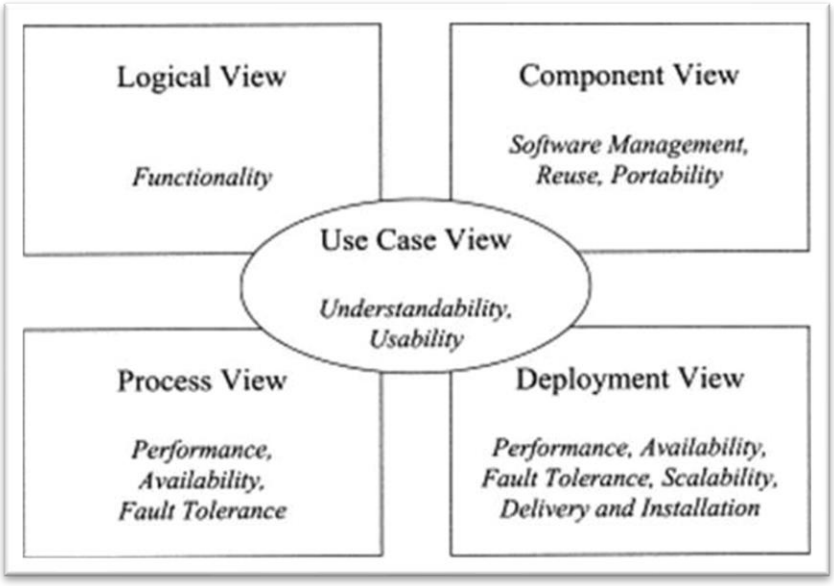
2.5.3 **Tại sao lại là OOAD và UML?**

OOAD cần các bản vẽ để mô tả hệ thống được thiết kế, còn UML là ngôn ngữ mô tả các bản vẽ nên cần nội dung thể hiện. Do vậy, chúng ta phân tích và thiết kế theo hướng đối tượng và sử dụng UML để biểu diễn các thiết kế đó nên chúng thường đi đôi với nhau.

2.5.4 **OOAD sử dụng UML**

UML sử dụng để vẽ cho nhiều lĩnh vực khác nhau như phần mềm, cơ khí, xây dựng v… trong phạm vi các bài viết này chúng ta chỉ nghiên cứu cách sử dụng UML cho phân tích và thiết kế hướng đối tượng trong ngành phần mềm. OOAD sử dụng UML bao gồm các thành phần sau:

View (góc nhìn): Mỗi góc nhìn như thầy bói xem voi, nó không thể hiện hết hệ thống nhưng thể hiện rõ hệ thống ở một khía cạnh. Chính vì thế trong xây dựng có bản vẽ kiến trúc (nhìn về mặt kiến trúc), bản vẽ kết cấu (nhìn về mặt kết cấu), bản vẽ thi công (nhìn về mặt thi công). Trong phần mềm cũng như vậy, OOAD sử dụng UML có các góc nhìn sau:



Hình 2.5: Các View trong OOAD sử dụng UML

Trong đó:

+ Use Case View: cung cấp góc nhìn về các ca sử dụng giúp chúng ta hiểu hệ thống có gì? ai dùng và dùng nó như thế nào.

+ Logical View: cung cấp góc nhìn về cấu trúc hệ thống, xem nó được tổ chức như thế nào. Bên trong nó có gì.

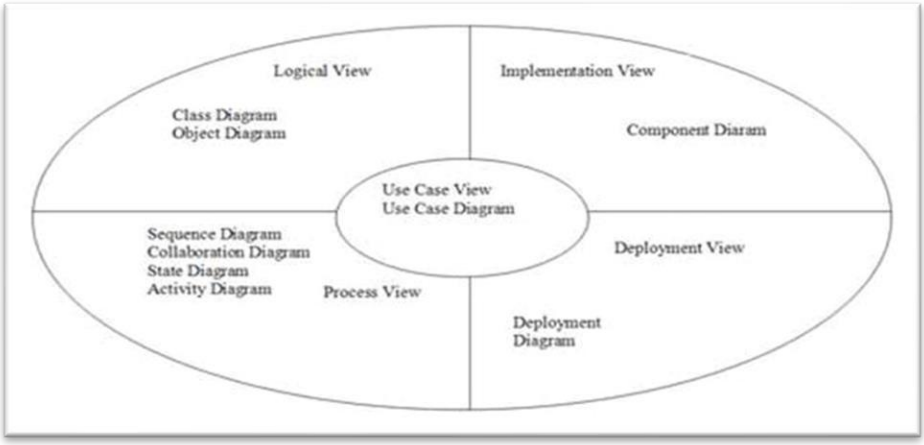
+ Process View: cung cấp góc nhìn động về hệ thống, xem các thành phần trong hệ thống tương tác với nhau như thế nào.

+ Component View: cũng là một góc nhìn về cấu trúc giúp chúng ta hiểu cách phân bổ và sử dụng lại các thành phần trong hệ thống ra sao.

+ Deployment View: cung cấp góc nhìn về triển khai hệ thống, nó cũng ảnh hưởng lớn đến kiến trúc hệ thống.

Tập hợp các góc nhìn này sẽ giúp chúng ta hiểu rõ hệ thống cần phân tích, thiết kế. Trong hình 1 chúng ta thấy góc nhìn Use Case View nằm ở giữa và chi phối tất cả các góc nhìn còn lại. Chính vì thế chúng ta thường thấy các tài liệu nói về 4 view + 1 chứ không phải 5 view nhằm nhấn mạnh vai trò của Use Case View.

Diagram (bản vẽ): Diagram các bạn có thể dịch là sơ đồ. Tuy nhiên ở đây chúng ta sử dụng từ bản vẽ cho dễ hình dung. Các bản vẽ được dùng để thể hiện các góc nhìn của hệ thống.



Hình 2.6:Các bản vẽ trong OOAD sử dụng UML

Trong đó:

+ Use Case Diagram: bản vẽ mô tả về ca sử dụng của hệ thống. Bản vẽ này sẽ giúp chúng ta biết được ai sử dụng hệ thống, hệ thống có những chức năng gì. Lập được bản vẽ này bạn sẽ hiểu được yêu cầu của hệ thống cần xây dựng.

+ Class Diagram: bản vẽ này mô tả cấu trúc của hệ thống, tức hệ thống được cấu tạo từ những thành phần nào. Nó mô tả khía cạnh tĩnh của hệ thống.

+ Object Diagram: Tương tự như Class Diagram nhưng nó mô tả đến đối tượng thay vì lớp (Class).

+ Sequence Diagram: là bản vẽ mô tả sự tương tác của các đối tượng trong hệ thống với nhau được mô tả tuần tự các bước tương tác theo thời gian.

+ Collaboration Diagram: tương tự như sequence Diagram nhưng nhấn mạnh về sự tương tác thay vì tuần tự theo thời gian.

+ State Diagram: bản vẽ mô tả sự thay đổi trạng thái của một đối tượng. Nó được dùng để theo dõi các đối tượng có trạng thái thay đổi nhiều trong hệ thống.

+ Activity Diagram: bản vẽ mô tả các hoạt động của đối tượng, thường được sử dụng để hiểu về nghiệp vụ của hệ thống.

+ Component Diagram: bản vẽ mô tả về việc bố trí các thành phần của hệ thống cũng như việc sử dụng các thành phần đó.

+ Deployment Diagram: bản vẽ mô tả việc triển khai của hệ thống như việc kết nối, cài đặt, hiệu năng của hệ thống v.v…

+ Notations (ký hiệu): Notations là các ký hiệu để vẽ, nó như từ vựng trong ngôn ngữ tự nhiên. Bạn phải biết từ vựng thì mới ghép thành câu, thành bài được. Chúng ta sẽ tìm hiểu kỹ các notations trong từng bản vẽ sau này. Dưới đây là vài ví dụ về notation.

Hình 2.7: Kí hiệu về Use Case

Hình 2.8: Ký hiệu về Class

Hình 2.9: Kí hiệu về Actor

Mechanisms (quy tắc, cơ chế): Mechanisms là các quy tắc để lập nên bản vẽ, mỗi bản vẽ có quy tắc riêng và bạn phải nắm được để tạo nên các bản vẽ thiết kế đúng. Các quy tắc này chúng ta sẽ bàn kỹ trong các bài về các bản vẽ.

2.6. **Giới thiệu tổng quan về hệ quản trị Cơ sở dữ liệu MySQL**

MySQL là một hệ quản trị cơ sở dữ liệu (DBMS) phổ biến và mạnh mẽ, được sử dụng rộng rãi trên toàn thế giới. Dưới đây là một giới thiệu tổng quan về MySQL:

2.6.1 **MySQL là gì?**

MySQL là một hệ quản trị cơ sở dữ liệu mã nguồn mở (open source DBMS), nghĩa là bạn có thể sử dụng nó miễn phí và có thể tùy chỉnh mã nguồn theo nhu cầu. MySQL phát triển bởi Oracle Corporation và có một cộng đồng rộng lớn của các nhà phát triển và người dùng trên toàn thế giới.

2.6.2 **Đặc điểm chính của MySQL:**

Tốc độ: MySQL được thiết kế để xử lý các tải công việc lớn và cung cấp hiệu suất cao.

Khả năng mở rộng: MySQL cho phép bạn mở rộng cơ sở dữ liệu dễ dàng bằng cách sử dụng một hệ thống chia sẻ tải hoặc sao lưu đám mây.

Hỗ trợ đa nền tảng: MySQL có thể hoạt động trên nhiều hệ điều hành như Windows, Linux, macOS, và nhiều nền tảng khác.

Ngôn ngữ SQL: MySQL sử dụng ngôn ngữ SQL (Structured Query Language) để truy cập và quản lý cơ sở dữ liệu.

An toàn: MySQL hỗ trợ các tính năng bảo mật như xác thực, mã hóa dữ liệu, và quản lý quyền truy cập.

2.6.3 **Ứng dụng của MySQL:**

MySQL được sử dụng rộng rãi trong nhiều ứng dụng và lĩnh vực, bao gồm:

Phát triển web: MySQL thường được sử dụng để lưu trữ dữ liệu cho các trang web và ứng dụng web động như WordPress, Joomla, và Drupal.

Phân tích dữ liệu: Do hiệu suất cao và khả năng xử lý lớn, MySQL thường được sử dụng trong các hệ thống phân tích dữ liệu và báo cáo.

Ứng dụng di động: MySQL cung cấp hỗ trợ cho các ứng dụng di động thông qua API và thư viện kết nối dễ dàng.

Lưu trữ dữ liệu: MySQL có thể được sử dụng để lưu trữ dữ liệu của các ứng dụng doanh nghiệp và trang web thương mại điện tử.

2.6.4 **Các phiên bản của MySQL:**

MySQL có nhiều phiên bản khác nhau, bao gồm phiên bản mã nguồn mở (MySQL Community Server) và phiên bản thương mại (MySQL Enterprise Edition) với các tính năng và hỗ trợ mở rộng khác nhau. MySQL cũng có một số biến thể như MariaDB, Percona Server, và nhiều hệ thống quản lý cơ sở dữ liệu khác dựa trên mã nguồn của MySQL.

Hình 2.10: MySQL Workbench

2.7. **Giới thiệu tổng quan về ReactJS**

ReactJS là một thư viện JavaScript phát triển bởi Facebook để xây dựng giao diện người dùng (UI) cho các ứng dụng web và ứng dụng đơn trang (single-page applications). Dưới đây là một tổng quan về ReactJS:

2.7.1 **Phương pháp làm việc**

ReactJS tập trung vào việc xây dựng các thành phần giao diện tái sử dụng. Mỗi thành phần là một phần tử UI độc lập, có khả năng tự quản lý và tái sử dụng trong ứng dụng.

React sử dụng một cách tiếp cận "cập nhật giao diện dựa trên trạng thái" (UI updates based on state). Điều này có nghĩa là khi trạng thái (state) của một thành phần thay đổi, giao diện sẽ được cập nhật tự động để phản ánh trạng thái mới đó.

2.7.2 **Khả năng tái sử dụng**

React khuyến khích việc xây dựng các thành phần giao diện tái sử dụng. Điều này giúp tối ưu hóa việc quản lý mã nguồn, giảm lặp lại mã, và dễ dàng bảo trì ứng dụng.

2.7.3 **Virtual DOM:**

React sử dụng một cơ chế gọi là Virtual DOM (DOM ảo) để cải thiện hiệu suất. Thay vì cập nhật DOM trực tiếp mỗi khi có thay đổi trạng thái, React tạo ra một bản sao của DOM gọi là Virtual DOM và chỉ cập nhật các phần tử thực sự thay đổi. Sau đó, React so sánh Virtual DOM mới với Virtual DOM cũ để tìm ra các thay đổi cần áp dụng trên DOM thực tế, điều này giúp giảm thiểu overhead và tăng hiệu suất.

2.7.4 **Cộng đồng lớn và hệ sinh thái:**

React có một cộng đồng phát triển lớn và rất nhiều tài liệu hướng dẫn, ví dụ mẫu, và thư viện bổ sung được xây dựng bởi cộng đồng.

Ngoài ra, có nhiều công cụ và thư viện khác như React Router (để quản lý định tuyến), Redux (để quản lý trạng thái ứng dụng), và Material-UI (để xây dựng giao diện dựa trên nguyên tắc thiết kế vật liệu).

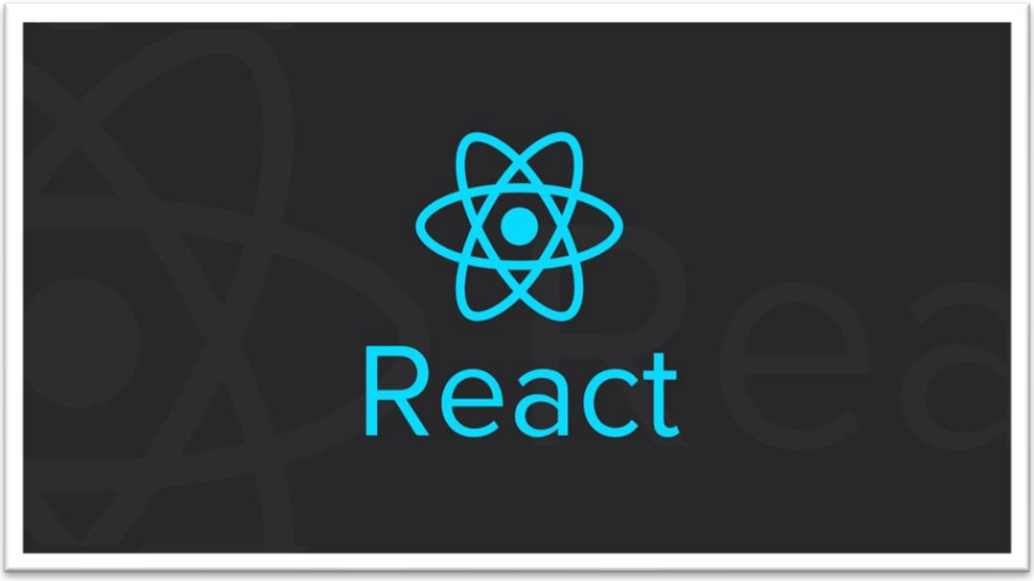
2.7.5 **Hỗ trợ dự án lớn:**

React thích hợp cho việc phát triển các ứng dụng lớn và phức tạp nhờ khả năng quản lý trạng thái tốt, khả năng tối ưu hóa hiệu suất và sự chia sẻ mã nguồn dễ dàng.

2.7.6 **Hỗ trợ cho ứng dụng di động:**

React Native là một framework được xây dựng dựa trên ReactJS, cho phép bạn phát triển ứng dụng di động đa nền tảng sử dụng JavaScript và React.

Các công ty sử dụng React:



Hình 2.11: Logo React

React được sử dụng rộng rãi bởi nhiều công ty lớn như Facebook, Instagram, Airbnb, Netflix, và nhiều ứng dụng web và di động khác.

Node.js là một môi trường chạy mã JavaScript phía máy chủ (server-side) được xây dựng trên JavaScript Engine V8 của Google Chrome. Nó cho phép bạn thực thi mã JavaScript trên máy chủ, cung cấp một cách linh hoạt và hiệu quả để phát triển các ứng dụng web và ứng dụng mạng.

2.8. **Giới thiệu tổng quan về NodeJS**

Dưới đây là một tổng quan về Node.js:

Node.js là một hệ thống phần mềm được thiết kế để viết các ứng dụng internet có khả năng mở rộng, đặc biệt là máy chủ web. Chương trình được viết bằng JavaScript, sử dụng kỹ thuật điều khiển theo sự kiện, nhập/xuất không đồng bộ để tối thiểu tổng chi phí và tối đa khả năng mở rộng.

Node.JS là một trong những nền tảng phổ biến nhất hiện nay cho mục đích phát triển ứng dụng mạng phía server

Ưu điểm

- ✓ IO hướng sự kiện không đồng bộ, cho phép xử lý nhiều yêu cầu đồng thời.
- ✓ Sử dụng JavaScript – một ngôn ngữ lập trình dễ học.
- ✓ Chia sẻ cùng code ở cả phía client và server.
- ✓ NPM(Node Package Manager) và module Node đang ngày càng phát triển mạnh mẽ.
- ✓ Cộng đồng hỗ trợ tích cực.
- ✓ Cho phép stream các file có kích thước lớn.

Nhược điểm

- ✓ Không có khả năng mở rộng, vì vậy không thể tận dụng lợi thế mô hình đa lõi trong các phần cứng cấp server hiện nay.
- ✓ Khó thao tác với cơ sử dữ liệu quan hệ.
- ✓ Mỗi callback sẽ đi kèm với rất nhiều callback lồng nhau khác.
- ✓ Cần có kiến thức tốt về JavaScript.
- ✓ Không phù hợp với các tác vụ đòi hỏi nhiều CPU.

JavaScript Everywhere:

Node.js cho phép bạn sử dụng JavaScript cả phía máy chủ và phía trình duyệt. Điều này tạo ra một môi trường đồng nhất và giúp đơn giản hóa việc chia sẻ mã nguồn giữa phía máy chủ và phía khách hàng (client-side).

Kiến thức cơ bản về Node.js:

Hình 2.12: Logo Nodejs

Node.js sử dụng mô hình xử lý không đồng bộ (asynchronous) và sự kiện (event-driven) để xử lý các tác vụ I/O (đầu vào/đầu ra) một cách hiệu quả. Điều này giúp tránh tình trạng đứng chờ trong quá trình xử lý yêu cầu từ các khách hàng.

Nền tảng đa nhiệm (Multi-Threaded):

Node.js không sử dụng multi-threading truyền thống như nhiều môi trường máy chủ khác. Thay vào đó, nó sử dụng mô hình đa nhiệm một luồng đơn (single-threaded) và non-blocking I/O để xử lý nhiều yêu cầu cùng một lúc. Điều này giúp giảm tài nguyên và tối ưu hóa hiệu suất.

Cộng đồng lớn và Hệ sinh thái mạnh mẽ:

Node.js có một cộng đồng phát triển lớn và nhiều thư viện và framework bổ sung. Các module và gói (packages) có thể quản lý dễ dàng bằng npm (Node Package Manager), cho phép bạn sử dụng mã nguồn mở và công cụ của cộng đồng.

Phát triển ứng dụng real-time:

Node.js thường được sử dụng để phát triển các ứng dụng real-time như ứng dụng trò chuyện (chat applications), trò chơi trực tuyến, và ứng dụng phát sóng trực tiếp (live streaming) nhờ vào khả năng xử lý sự kiện và gửi dữ liệu một cách nhanh chóng và hiệu quả.

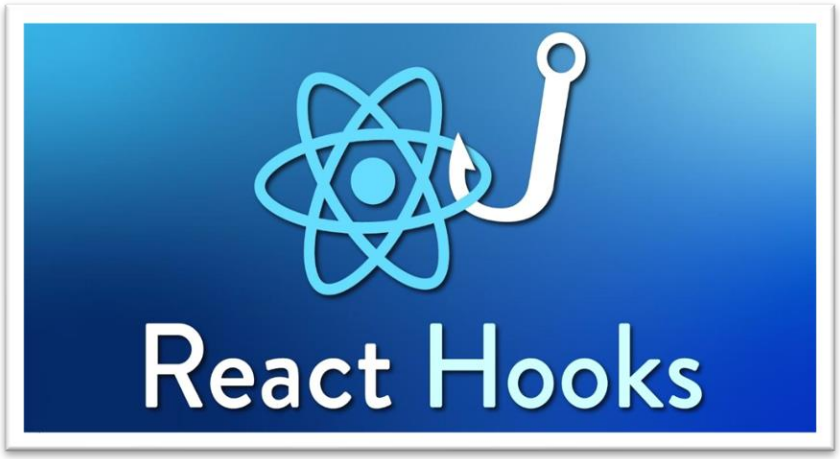
Hỗ trợ đa nền tảng:

Node.js hoạt động trên nhiều hệ điều hành như Windows, macOS và các phiên bản của Linux.

Sử dụng phổ biến:

Node.js được sử dụng rộng rãi bởi nhiều công ty lớn và dịch vụ nổi tiếng như Netflix, Uber, LinkedIn, và eBay.

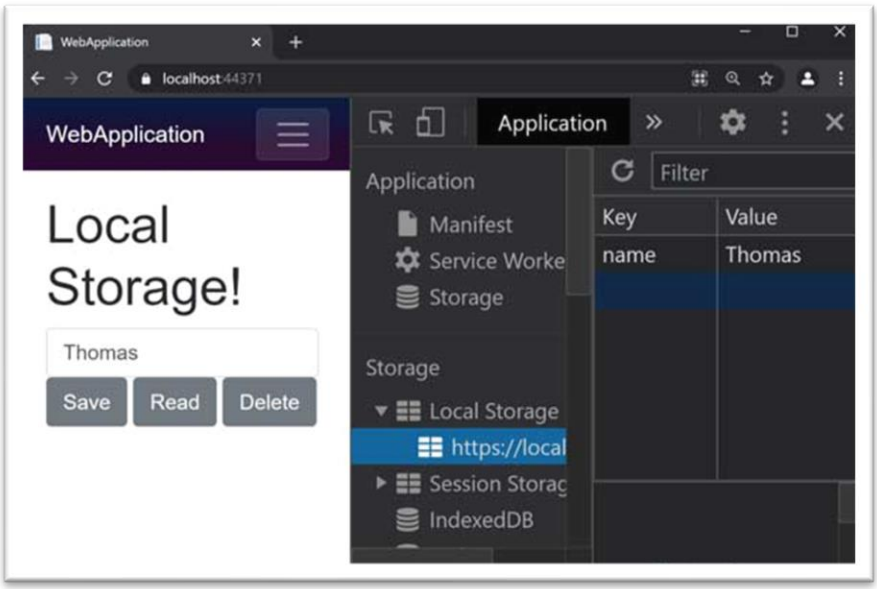
2.9. **Tìm hiểu React Hook**



Hình 2.13: React Hooks

Chúng ta đã có thể sử dụng state trong stateless (functional) component, việc mà từ trước tới nay ta bắt buộc phải khai báo Class. Có thể thấy, các nhà phát triển React họ đang muốn hướng đến 1 tương lai Functional Programming thay vì sử dụng những Class mà chỉ nghe cái tên thôi là ta đã nghĩ ngay đến OOP. Cộng với việc không sử dụng Class kế thừa từ React Component nữa nên giờ đây kích thước bundle sẽ được giảm đáng kể bởi code sử dụng Hooks.

2.10. **Tìm hiểu Local store**



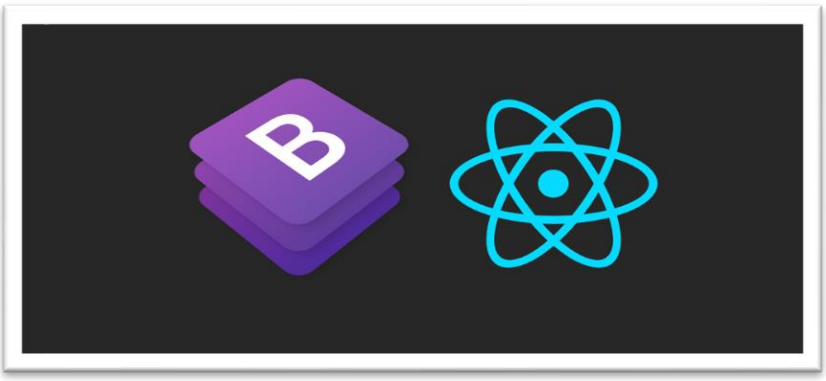
Hình 2.14 HTML5

Local storage là một feature mới của HTML5 cho phép bạn lưu trữ bất kì info nào bạn muốn trong browser dùng JavaScript.

Local Storage có tính đồng bộ (synchronous), nghĩa là không thể truy cập song song mà nó sẽ đáp ứng từng lời gọi thực thi theo thứ tự.

Một điểm trừ lớn khác của Local Storage là nó không có phương thức nào để bảo vệ dữ liệu lưu trên trình duyệt. Bất cứ đoạn mã javascript nào trong Website cũng có thể truy cập các giá trị lưu trữ trong Local Storage.

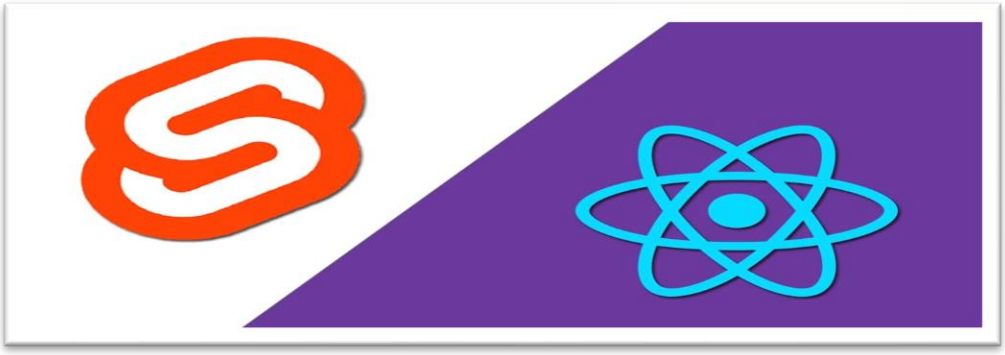
2.11. **Tìm hiểu React-Bootstrap**



Hình 2.15: Bootstrap

React-Bootstrap cung cấp sẵn một bộ các React component với look-and-feel của Twitter Bootstrap component từ đó giúp việc tạo dựng UI cho React app dễ dàng hơn bao giờ hết.

2.12. **Tìm hiểu SASS/SCSS**



Hình 2.16: Sass

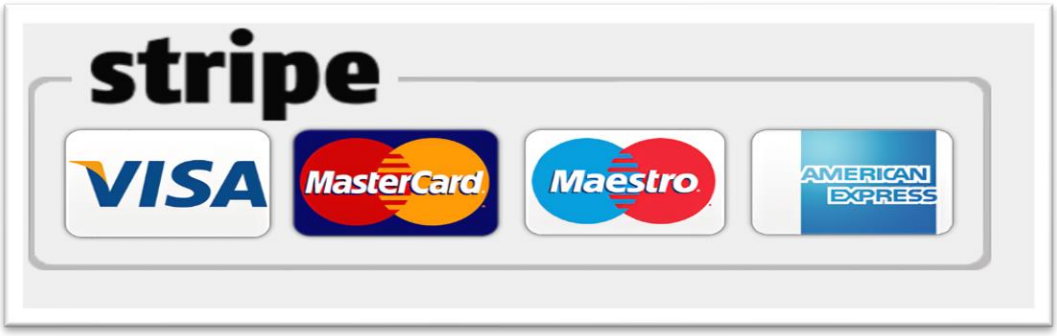
SASS (Syntactically Awesome StyleSheets) là một CSS Preprocessor giúp bạn viết CSS nhanh hơn và có cấu trúc rõ ràng hơn. Với SASS, bạn có thể viết CSS theo thứ tự rõ ràng, quản lý các biến đã được định nghĩa sẵn, các class dùng chung hay có thể tự động nén tập tin CSS lại để bạn tiết kiệm dung lượng.

SASS và SCSS về bản chất vấn đề là giống nhau, chỉ khác nhau ở cách viết

Sass là chữ viết tắt của Syntactically Awesome Style Sheets, chương trình tiền xử lý bằng ngôn ngữ kịch bản (Preprocessor Scripting Language), sẽ được biên dịch thành CSS. Nghĩa là, mình sẽ làm style bằng SASS, rồi SASS sẽ render việc mình làm thành file CSS.

2.13. **Giới thiệu tổng quan về Stripe**

Stripe là một công ty và nền tảng thanh toán trực tuyến được phát triển để giúp các doanh nghiệp và các nhà phát triển tích hợp dịch vụ thanh toán vào các ứng dụng và trang web của họ. Dưới đây là một tổng quan về Stripe:



Hình 2.17: Stripe Payment

2.13.1 **Lịch sử và Xuất phát điểm:**

Stripe được thành lập vào năm 2010 bởi hai anh em Patrick và John Collison. Họ tạo ra Stripe với mục tiêu cải thiện và đơn giản hóa quá trình thanh toán trực tuyến, đặc biệt là đối với các nhà phát triển.

Stripe bắt đầu như một công ty khởi nghiệp và sau đó đã phát triển thành một trong những nền tảng thanh toán phổ biến nhất trên thế giới.

2.13.2 **Dịch vụ Stripe:**

Stripe cung cấp nhiều dịch vụ và sản phẩm liên quan đến thanh toán, bao gồm:

Stripe Payments: Dịch vụ thanh toán cho phép doanh nghiệp chấp nhận thanh toán trực tuyến từ khách hàng thông qua thẻ tín dụng, thẻ ghi nợ và nhiều phương thức thanh toán khác.

Stripe Connect: Cho phép xây dựng nền tảng thị trường và kết nối với nhiều đối tác thanh toán khác nhau.

Stripe Billing: Giúp quản lý chu kỳ thanh toán, tạo và quản lý hoá đơn cho khách hàng, và tạo các kế hoạch thuê bao (subscription) linh hoạt.

Stripe Radar: Sử dụng trí tuệ nhân tạo để phát hiện và ngăn chặn gian lận trong các giao dịch thanh toán.

Stripe Sigma: Cung cấp công cụ để truy vấn và phân tích dữ liệu thanh toán.

Stripe Atlas: Hỗ trợ các công ty khởi nghiệp trong việc thành lập doanh nghiệp tại Hoa Kỳ.

Khả năng tích hợp:

Stripe có các thư viện và API mạnh mẽ, cho phép các nhà phát triển tích hợp dịch vụ thanh toán của họ vào các ứng dụng web và di động dễ dàng.

Hỗ trợ đa quốc gia:

Stripe hỗ trợ nhiều loại tiền tệ và có khả năng xử lý thanh toán từ khách hàng trên toàn thế giới.

Bảo mật:

Stripe tuân theo các tiêu chuẩn bảo mật cao và cung cấp các công cụ để giảm nguy cơ gian lận và bảo vệ dữ liệu của khách hàng.

Phí và Cước phí:

Stripe tính phí dựa trên từng giao dịch và không yêu cầu phí hàng tháng hoặc phí đăng ký.

2.14. **Lập luận về công nghệ**

Qua việc tổng quan và phân tích các công nghệ cũng như cở sở dữ liệu, đề tài lựa chọn các công nghệ sau:

- NodeJS

React là một thư viện JavaScript front-end mã nguồn mở và miễn phí để xây dựng giao diện người dùng dựa trên các thành phần UI riêng lẻ. Nó được phát triển và duy trì bởi Meta và cộng đồng các nhà phát triển và công ty cá nhân.

- Mysql

MySQL là hệ quản trị cơ sở dữ liệu tự do nguồn mở phổ biến nhất thế giới và được các nhà phát triển rất ưa chuộng trong quá trình phát triển ứng dụng.

- Stripe

Stripe là cổng thanh toán của Mỹ được thành lập năm 2010 cho phép các trang thương mại điện tử nhận thanh toán trên website bán hàng của mình.

Stripe cung cấp SDK để khách hàng có thể tích hợp ngay trên các thiết bị iOS và Android. Ngoài ra, nó còn cung cấp Stripe API để có thể được sử dụng bởi các ngôn ngữ lập trình khác nhau như: Ruby, Python, Java, GO… (Stripe API).

Tài khoản Stripe giống với tài khoản ngân hàng. Bạn có thể tạo nhiều tài khoản cho sản phẩm hoặc các trang bán hàng. Mỗi tài khoản có các khoản thanh toán, bảng điều khiển, người dùng được ủy quyền riêng mà bạn lựa chọn.

- Reactjs

React là một thư viện JavaScript front-end mã nguồn mở và miễn phí để xây dựng giao diện người dùng dựa trên các thành phần UI riêng lẻ. Nó được phát triển và duy trì bởi Meta và cộng đồng các nhà phát triển và công ty cá nhân.

CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

3.1. **Phân tích và đặc tả yêu cầu**

Website quản lý

- Thêm thông tin người dùng
- Quản lý người dùng
- Cập nhật người dùng
- Sửa thông tin người dùng
- Xóa thông tin người dùng
- Xóa lịch sử Donate

App người dùng

- Đăng ký
- Đăng nhập
- Xem thông tin bài viết
- Donate

Ứng dụng được viết cho 2 đối tượng sử dụng là admin, user với yêu cầu bảo mật như sau:

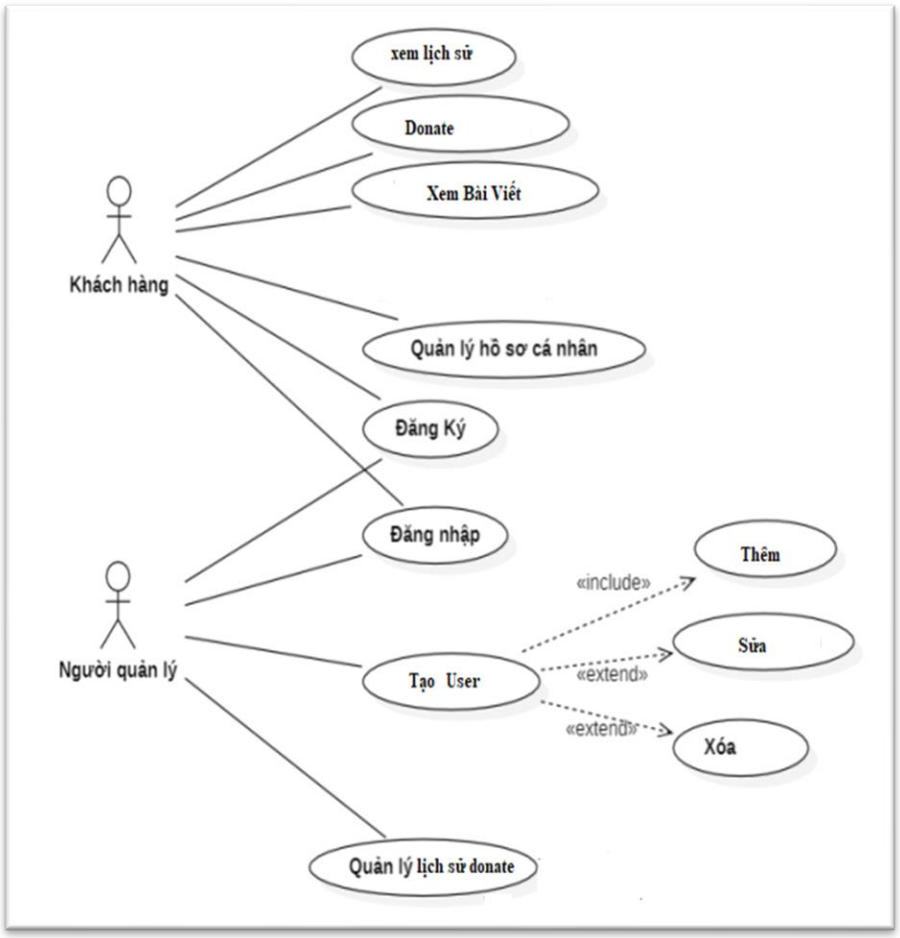
✓ Tất cả đối tượng phải đăng nhập mới được sử dụng ứng dụng

✓ Tài khoản người dùng chỉ có quyền thay đổi các thông tin cá nhân và đổi mật khẩu, đăng xuất

✓ Ứng dụng đoợc xây dựng bằng React js , Nodejs, Stripe

✓ Cơ sở dữ liệu lưu trữ trên mây MySQL

3.2. **Sơ đồ use-case tổng quát**



Hình 3.1: UseCase Tổng Quát

Tác nhân	Khách hàng, Admin
Mô tả	Tất cả đối tượng phải đăng nhập mới được sử dụng ứng dụng
Điều kiện kích hoạt	Người dùng bấm đăng nhập
Tiền điều kiện	Người dùng đã đăng ký tài khoản
Hậu điều kiện	Người dùng đăng nhập thành công

Luồng sự kiện chính	1. Hiển thị màn hình đăng nhập 2. Người dung nhập email và mật khẩu 3. Người dùng chọn tiếp tục đăng nhập hoặc đăng xuất 4. Kết thúc use-case
Luồng sự kiện phụ	Nếu người nhập sai email và mật khẩu hệ thống sẽ hiển thị Email hoặc mật khẩu không chính xác Use-case kết thúc

Bảng 3.1: Use Case Đăng nhập

Tác nhân	Khách hàng, Admin
Mô tả	Tất cả đối tượng phải đăng ký có tài khoản mới được sử dụng ứng dụng
Điều kiện kích hoạt	Người dùng chọn chức năng đăng ký từ hệ thống
Tiền điều kiện	Không có
Hậu điều kiện	Người dùng đăng ký thành công
Luồng sự kiện chính	1. Hiển thị màn hình đăng ký 2. Người dùng nhập: họ và tên, email, số điện thoại, mật khẩu, nhập lại mật khẩu, địa chỉ 3. Người dùng chọn đăng ký

Luồng sự kiện phụ	Nếu người dùng nhập sai thông tin : họ và tên, email, số điện thoại, mật khẩu, nhập lại mật khẩu, địa chỉ thì sẽ có thông báo hiển thị trên màn hình người dùng đã nhập sai và yêu cầu người dùng nhập lại

Bảng 3.2: Use Case Đăng Ký

Tác nhân	Khách hàng, Admin
Mô tả	Người dùng sau khi đăng nhập vào hệ thống sẽ được xem thông tin
Điều kiện kích hoạt	Người dùng đã đăng ký và đăng nhập
Tiền điều kiện	Người dùng đã đăng ký và đăng nhập
Hậu điều kiện	Nhân viên đăng nhập thành công vào hệ thống
Luồng sự kiện chính	1. Người dùng 2. Nhân viên điền thông tin và bấm đăng nhập 3. Hệ thống kiểm tra thông tin đăng nhập của nhân viên
Luồng sự kiện phụ	Không có

Bảng 3.3: Use Case Xem thông tin bài viết

Tác nhân	Khách hàng, Admin
Mô tả	Người dùng xem thông tin lịch sử Donate

Điều kiện kích hoạt	Không có
Tiền điều kiện	Không có
Hậu điều kiện	Không có
Luồng sự kiện chính	Người dùng nhấn vào lịch sử Donate, Trang Donate sẽ hiển thị ra tất cả lịch sử của tất cả người dùng đã donte
Luồng sự kiện phụ	Không có

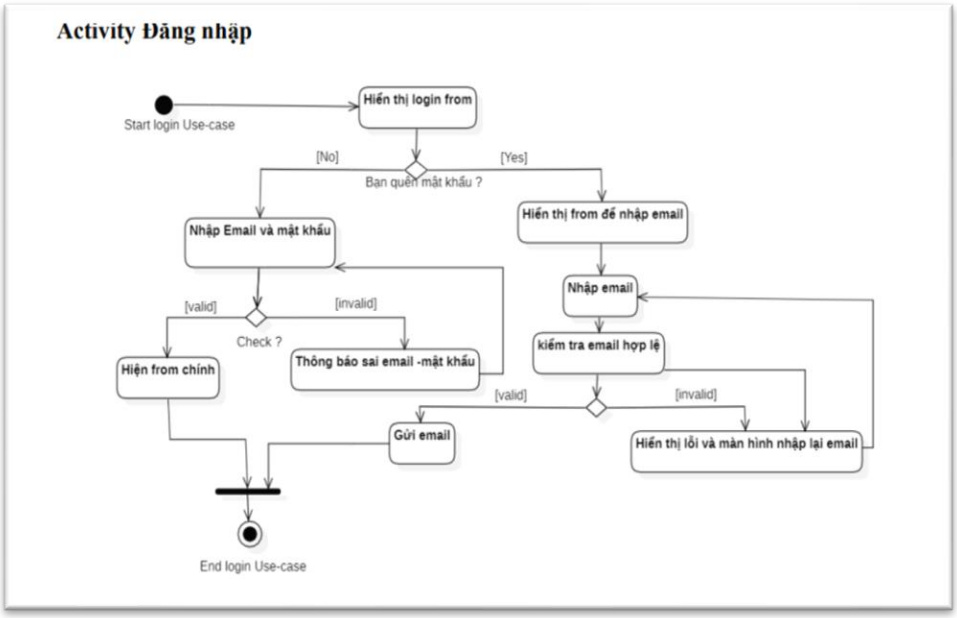
Bảng 3.4: Use Case Xem lịch sử Donate

Tên Use - Case	Donate
Tác nhân	Khách hàng
Mô tả	Người dùng nhấn vào Donate sẽ hiện thị ra 1 form để người dùng nhập thông tin vào
Điều kiện kích hoạt	Người dùng đã đăng nhập
Tiền điều kiện	Người dùng đã đăng ký tài khoản
Hậu điều kiện	Người dùng đăng ký thành công
Luồng sự kiện chính	Người dùng nhấn vào Donate sẽ hiện thị ra 1 form để người dùng nhập thông tin vào, thông tin Donate và thời gian của người Donate sẽ được lưu lại vào trang lịch sử Donate

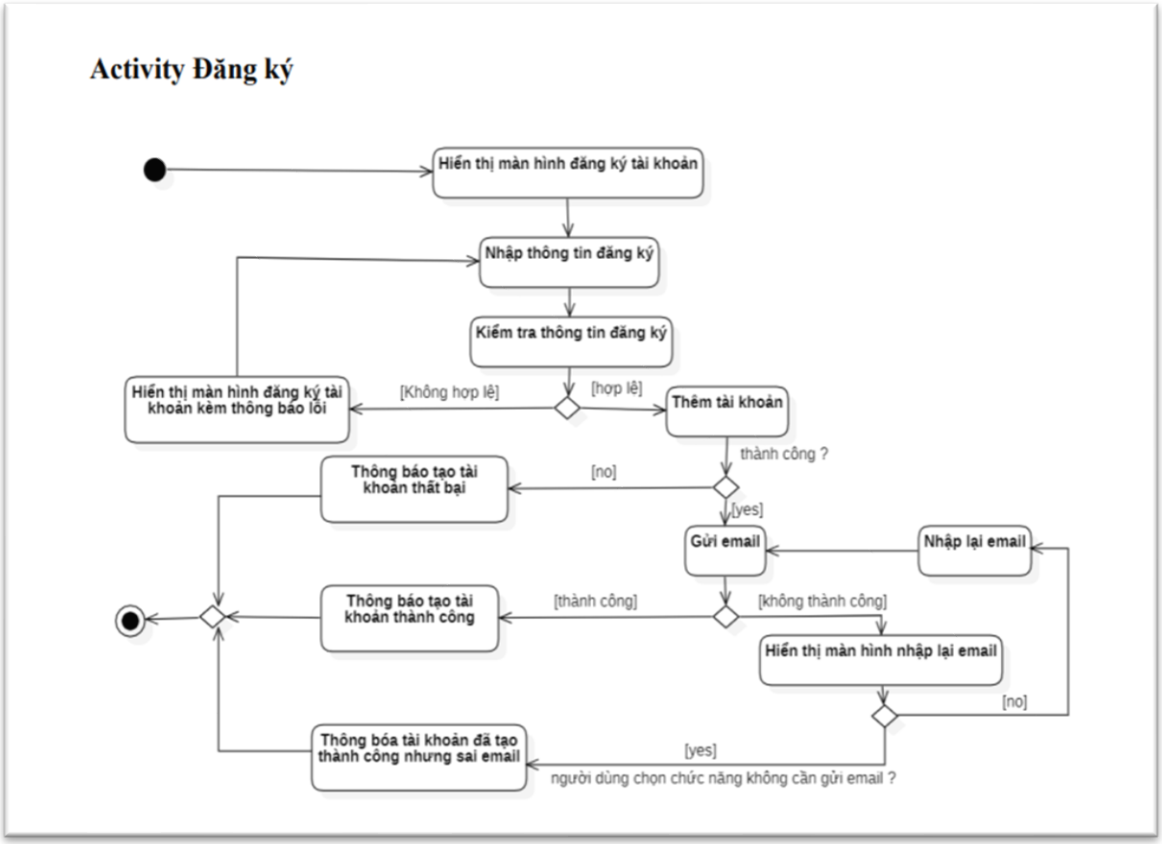
Luồng sự kiện phụ	Nếu người dùng nhập sai thông tin sẽ có thông báo hiển thị trên màn hình người dùng đã nhập sai và yêu cầu người dùng nhập lại

Bảng 3.5: Use Case Donate

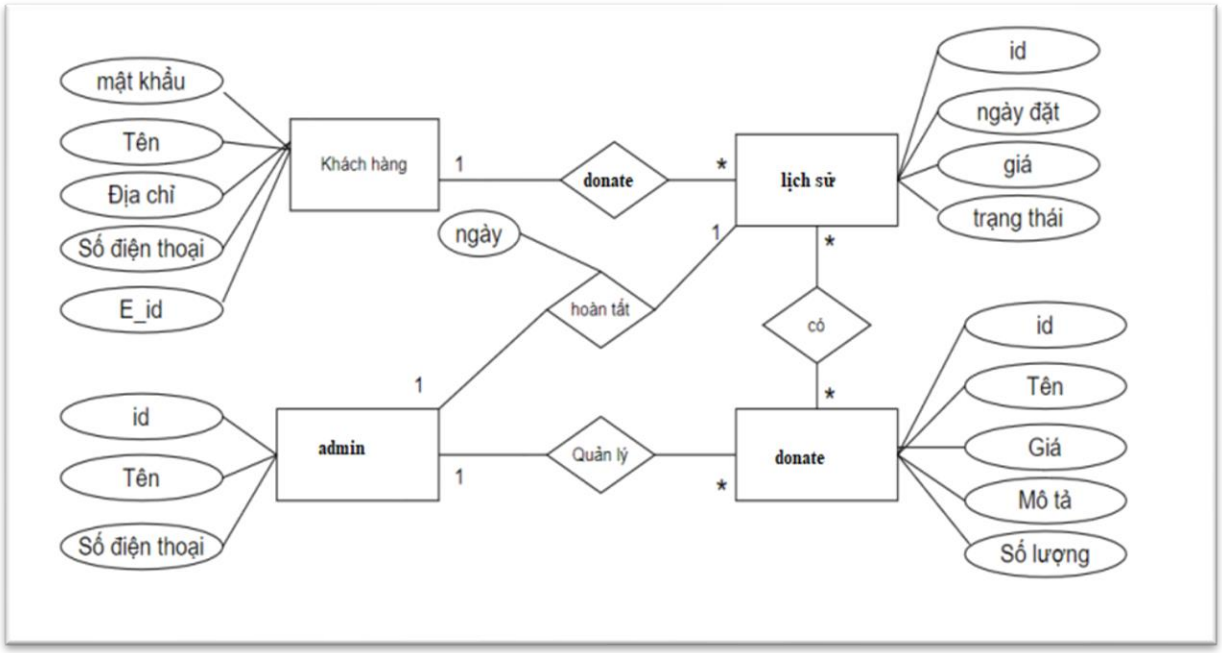
3.3. **Sơ đồ hoạt động**



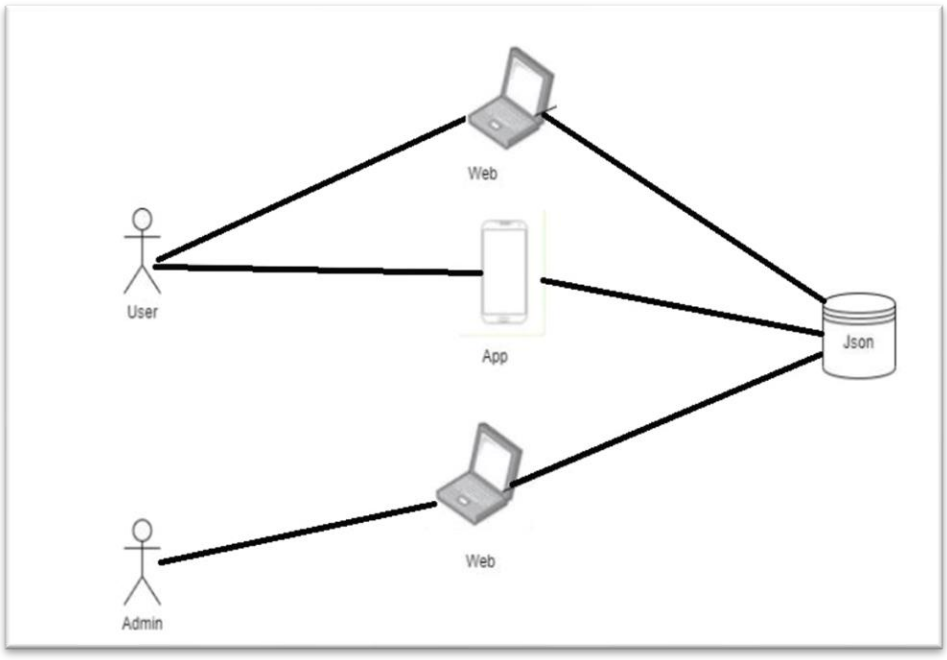
Hình 3.2: Activity Đăng nhập



Hình 3.3: Activity Đăng nhập



Hình 3.4: Mô hình thực thể kết hợp (ERD)



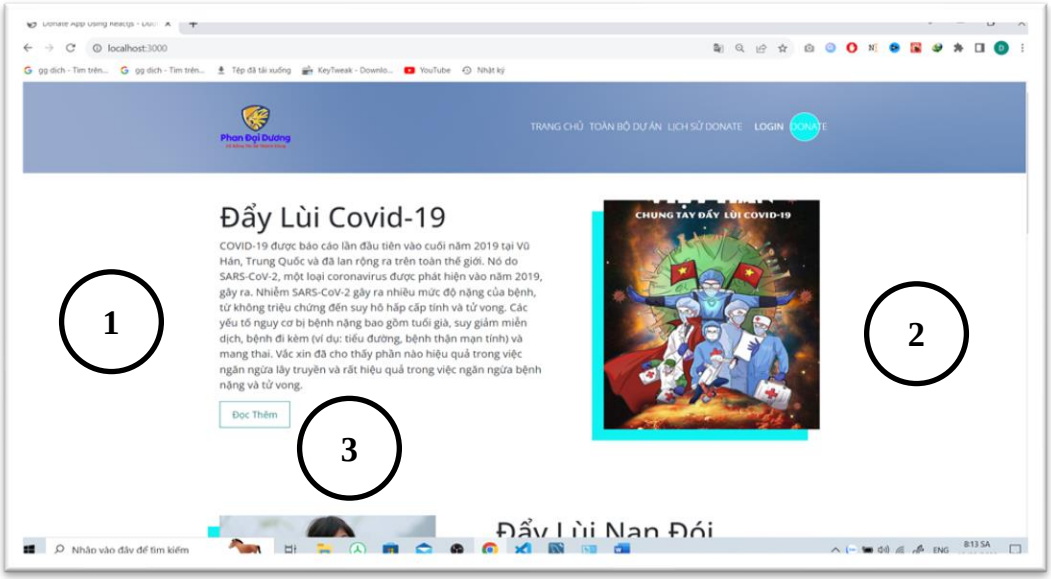
Hình 3.5: Sơ đồ chuyển khai

CHƯƠNG 4: THỰC NGHIỆM VÀ TRIỂN KHAI

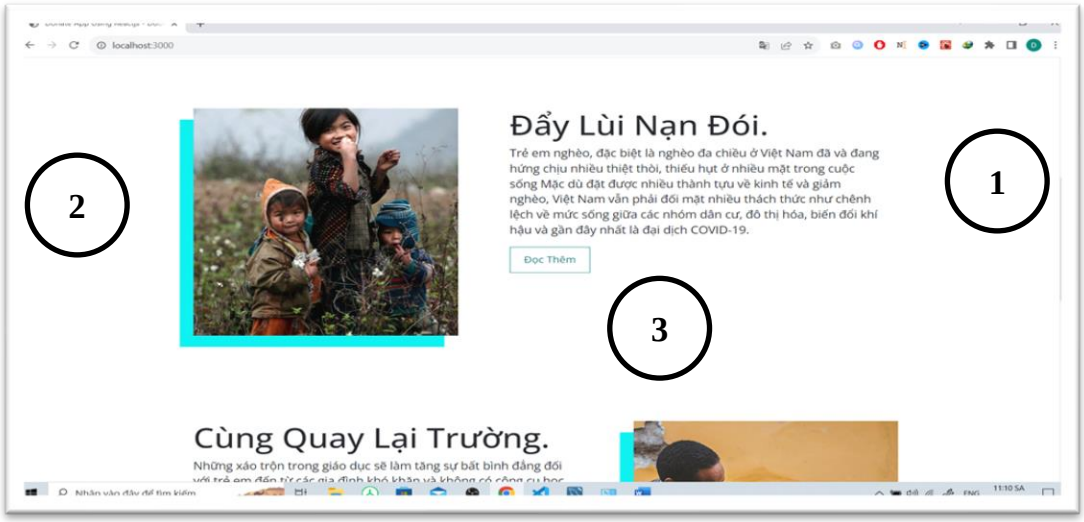
4.1. **Giới thiệu hệ thống**

Hệ thống Website quyên góp và ủng hộ cộng đồng là một hệ thống rất cần thiết ngày nay. Bởi vì ngày càng có các loại dịch bệnh, thiên tai do con người gây ra hoặc do tự nhiên gây ra. Hệ thống này được đề tài xây dựng gồm có một số tính năng sẽ được trình bày chi tiết bên dưới đây.

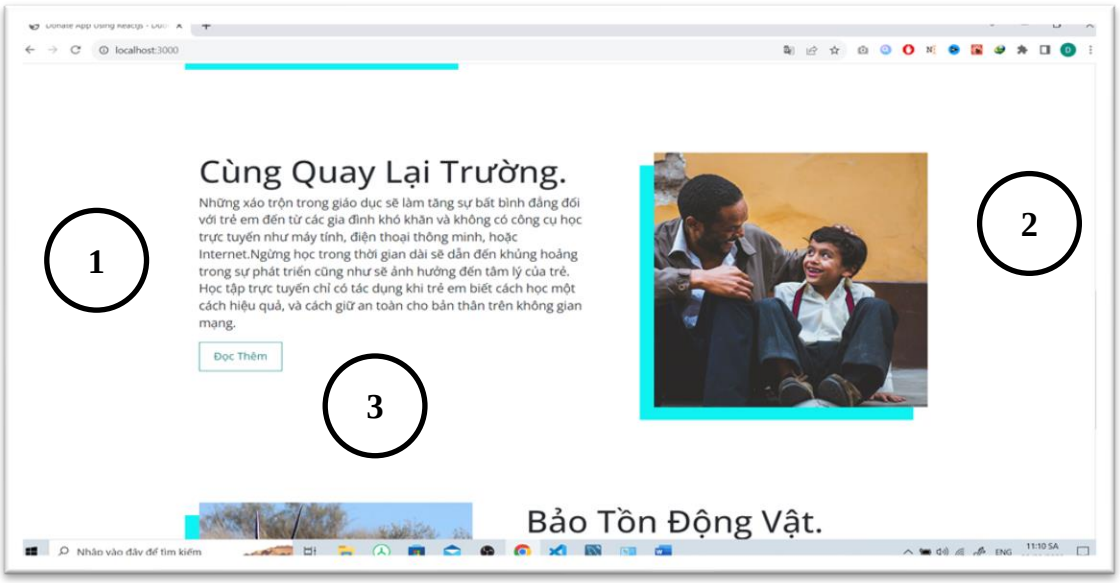
4.2. **Trang chủ**



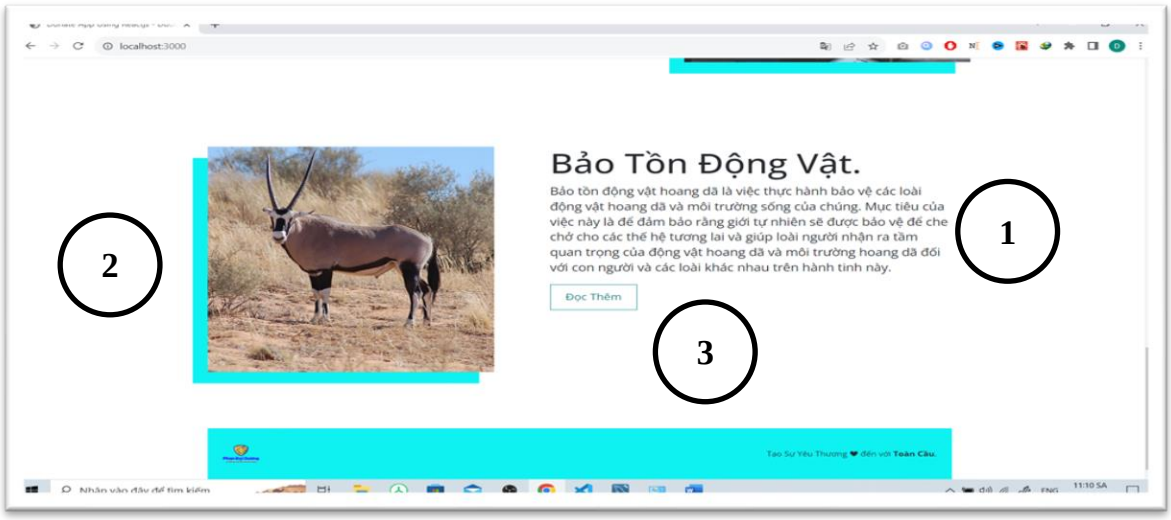
Hình 4.1: Trang chủ



Hình 4.2: Trang chủ



Hình 4.3: Trang chủ

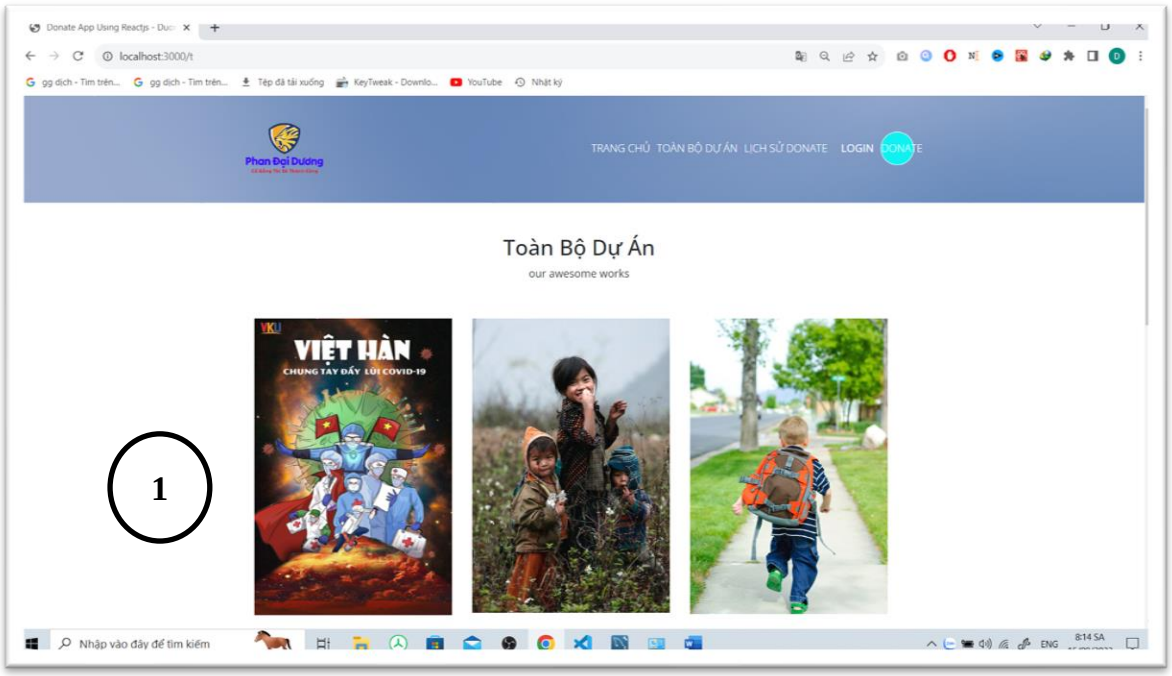


Hình 4.4: Trang chủ

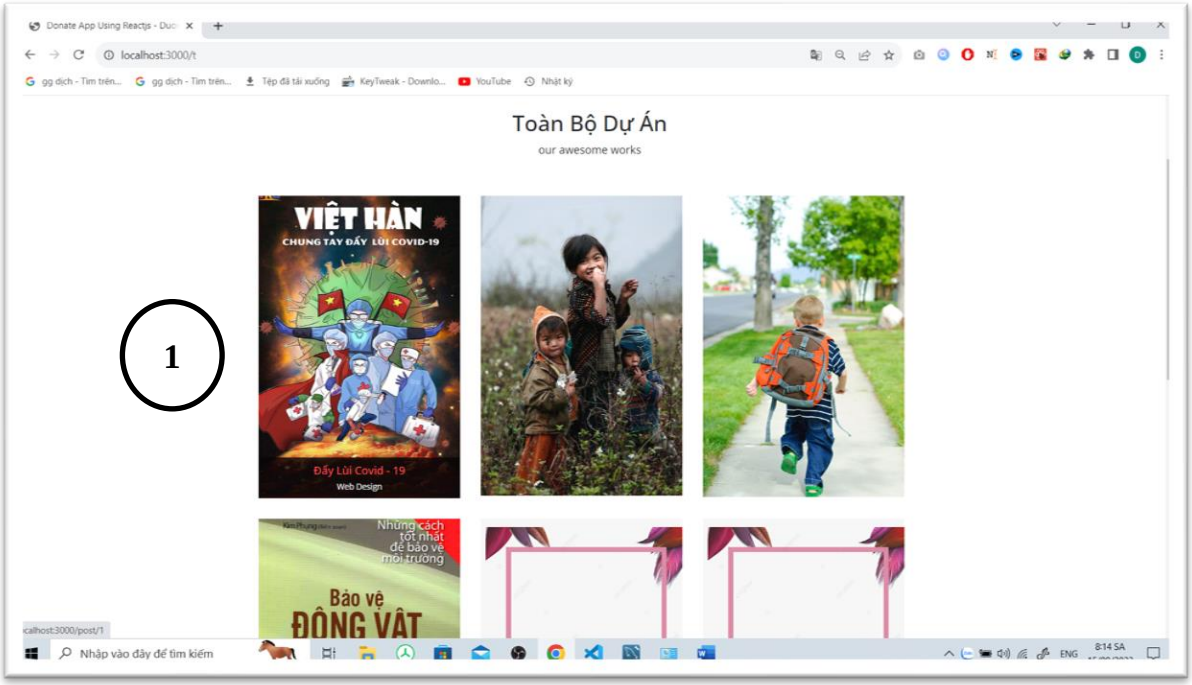
Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Thông tin về Covid-19	Mô tả ngắn gọn thông tin về Covid-19
2	Ảnh Minh Họa về Covid-19	Mô tả chi tiết về Covid-19 bằng hình ảnh
3	Chức Năng Đọc Thêm	Mở ra một trang bài viết chi tiết Covid -19

Bảng 4.1: Bảng mô tả chi tiết Trang chủ

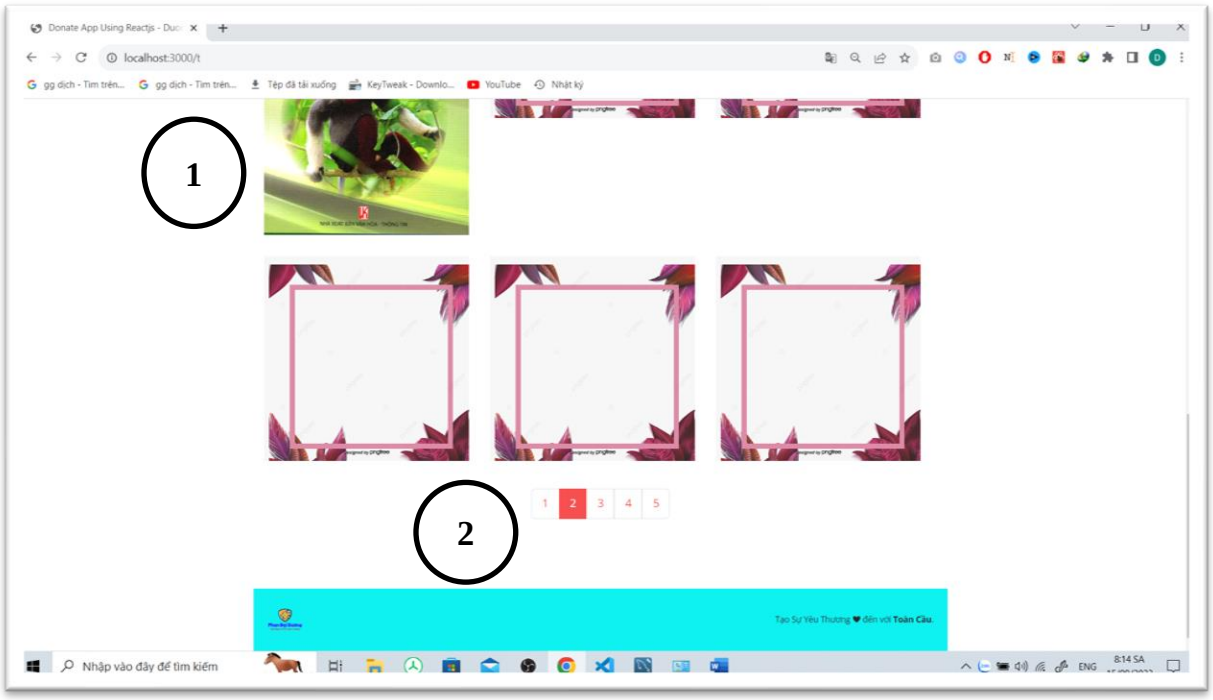
4.3. **Trang bài viết**



Hình 4.5: Trang bài viết



Hình 4.6: Trang bài viết

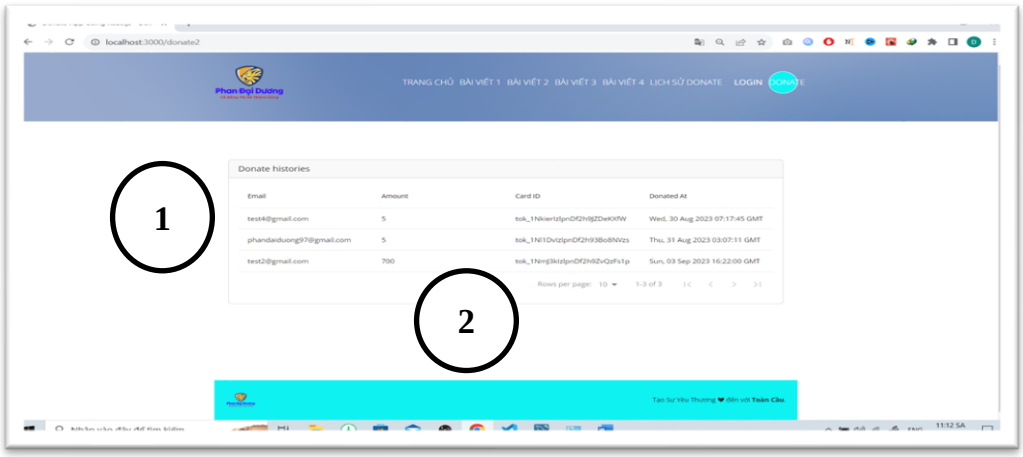


Hình 4.7: Trang bài viết

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Card Grid Link	Liên Kết đến các trang thông tin liên quan
2	Pagination	Phân trang là quá trình chia tài liệu thành các trang riêng biệt

Bảng 4.2: Bảng mô tả Trang bài viết

4.4. **Lịch sử Donate**

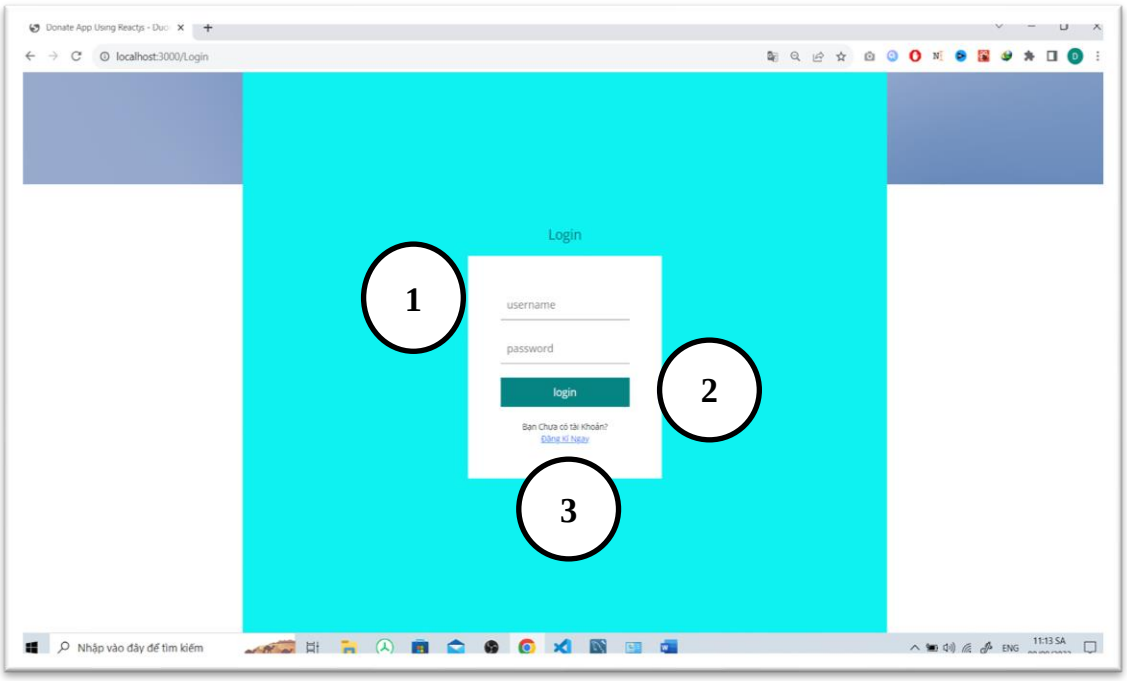


Hình 4.8: Lịch sử Donate

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Table dữ liệu	Chứa thông tin lịch sử người dùng donate
2	Row per page	Hàng trên mỗi trang, dùng thu nhỏ hoặc phóng to các hàng dữ liệu

Bảng 4.3: Bảng mô tả lịch sử donate

4.5. **Login**

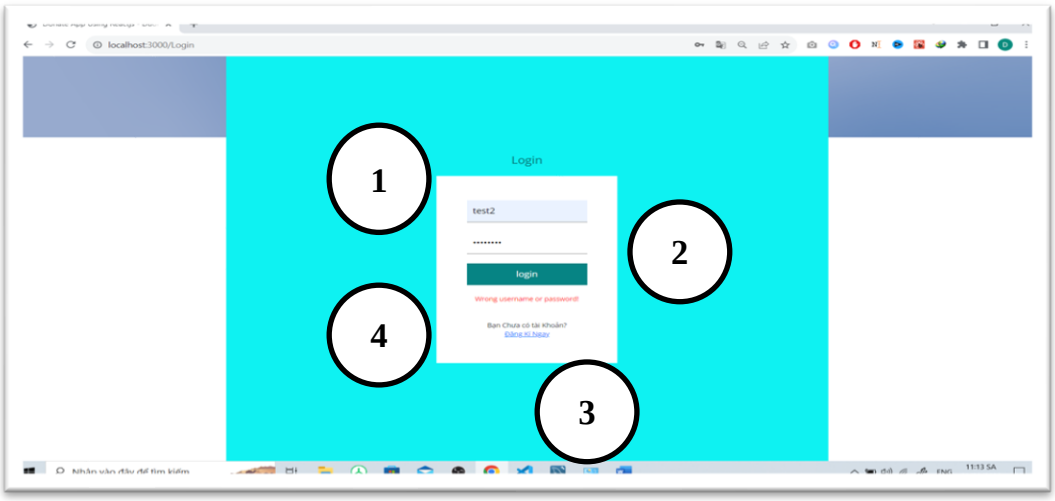


Hình 4.9: Trang Login

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Form nhập thông tin	Biểu mẫu nhập thông tin đăng nhập
2	Button link	Nút liên kết đến trang website
3	Link	Link liên kết đến trang web

Bảng 4.4: Bảng mô tả chi tiết Trang Login

4.4.1 Trang Lỗi Login

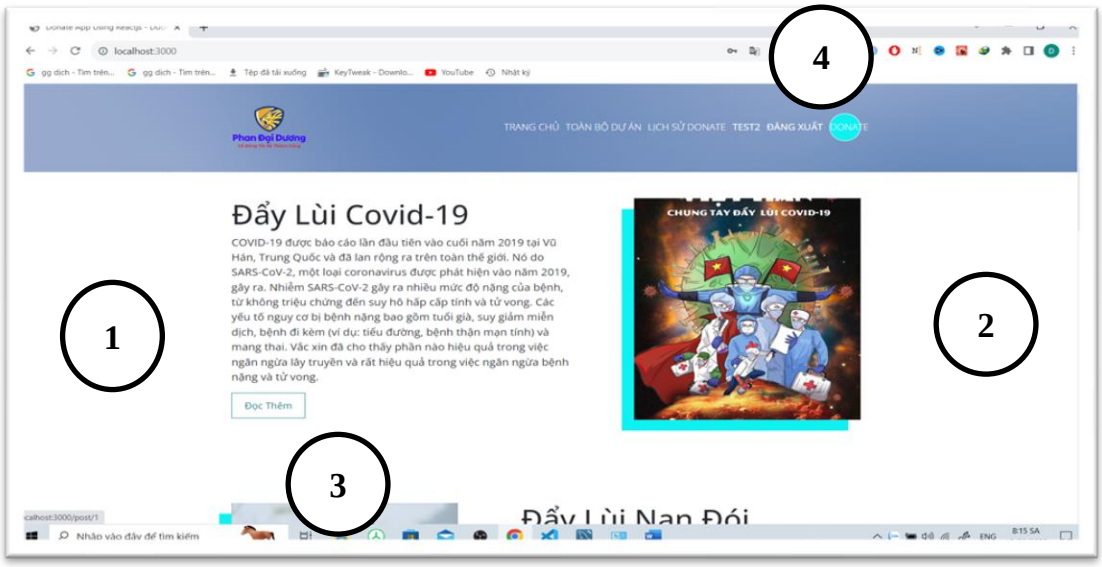


Hình 4.10: Trang lỗi Login

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Form nhập thông tin	Biểu mẫu nhập thông tin đăng nhập
2	Button link	Nút liên kết đến trang website
3	Link	Link liên kết đến trang web
4	Báo Lỗi	Báo các lỗi liên quan đến nhập liệu của người dùng

Bảng 4.5: Bảng mô tả Trang lỗi Login

4.6. **Trang Login Thành Công**

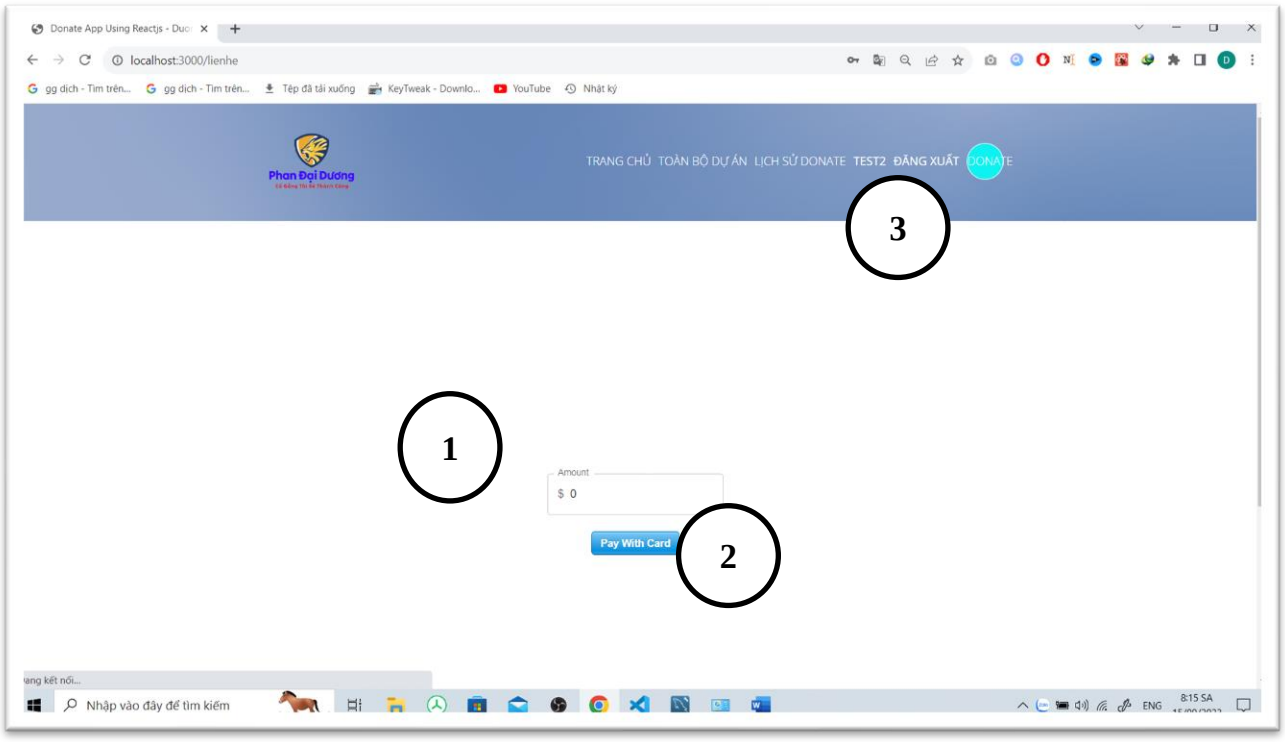


Hình 4.11: Trang Logic Success

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Thông tin về Covid-19	Mô tả ngắn gọn thông tin về Covid-19
2	Ảnh Minh Họa về Covid-19	Mô tả chi tiết về Covid-19 bằng hình ảnh
3	Chức Năng Đọc Thêm	Mở ra một trang bài viết chi tiết Covid -19
4	Đăng Xuất Tài Khoản	Thoát tài khoản đã đăng nhập trên website

Bảng 4.6: Bảng mô tả trang Logic Success

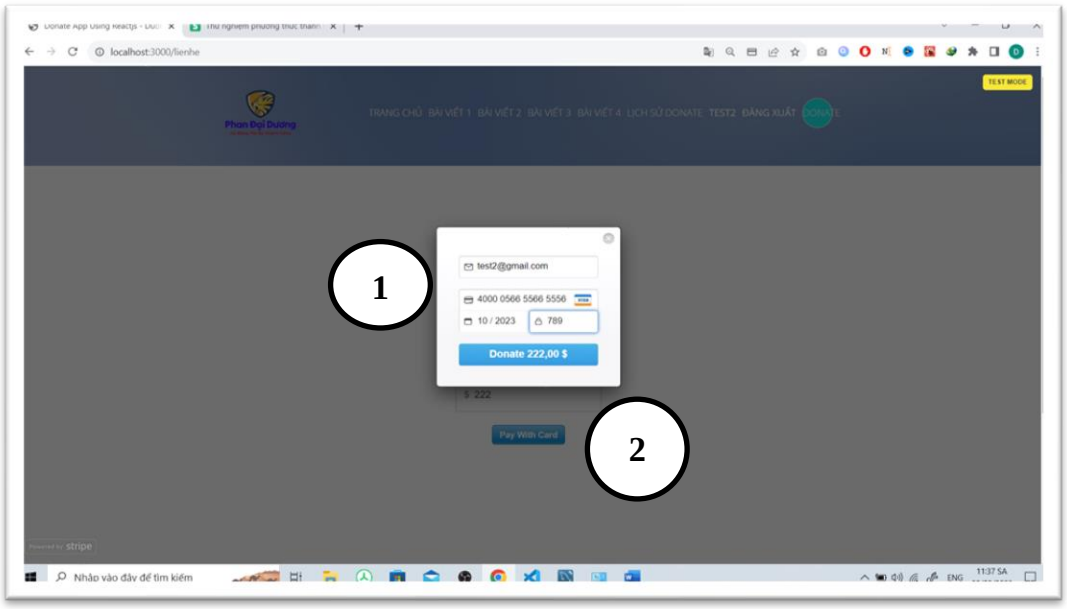
4.7. **Trang Donate**



Hình 4.12: Trang Donate

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Form nhập thông tin	Mô tả số tiền người dùng muốn donate
2	Button Link	Nút liên kết
3	Đăng Xuất Tài Khoản	Thoát tài khoản đã đăng nhập trên website

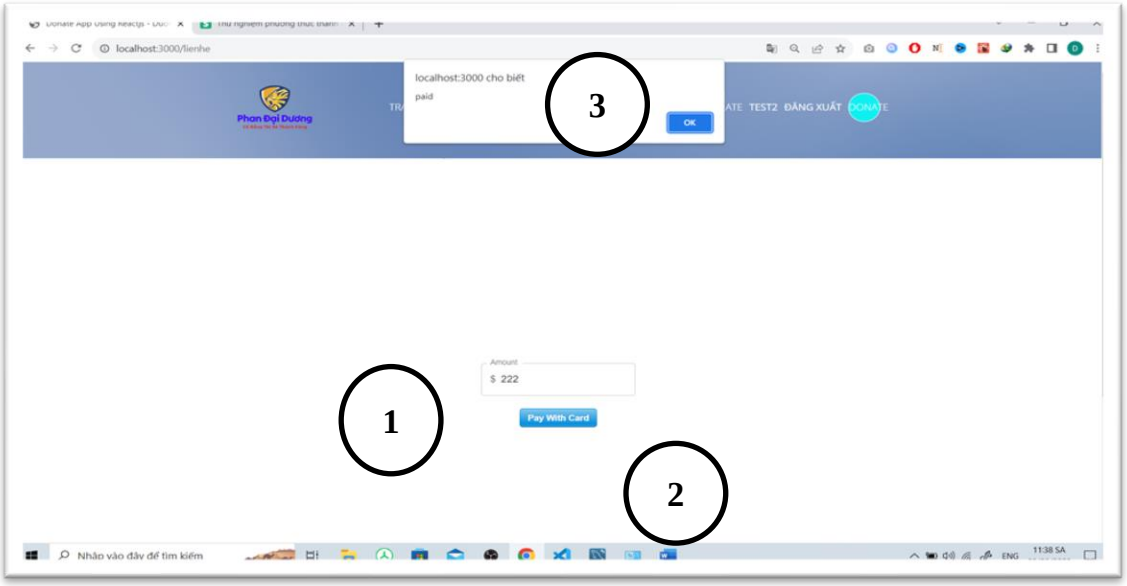
Bảng 4.7: Bảng mô tả Trang Donate



Hình 4.13: Form Donate

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Form nhập thông tin	Mô tả số tiền người dùng muốn donate
2	Button Link	Nút liên kết

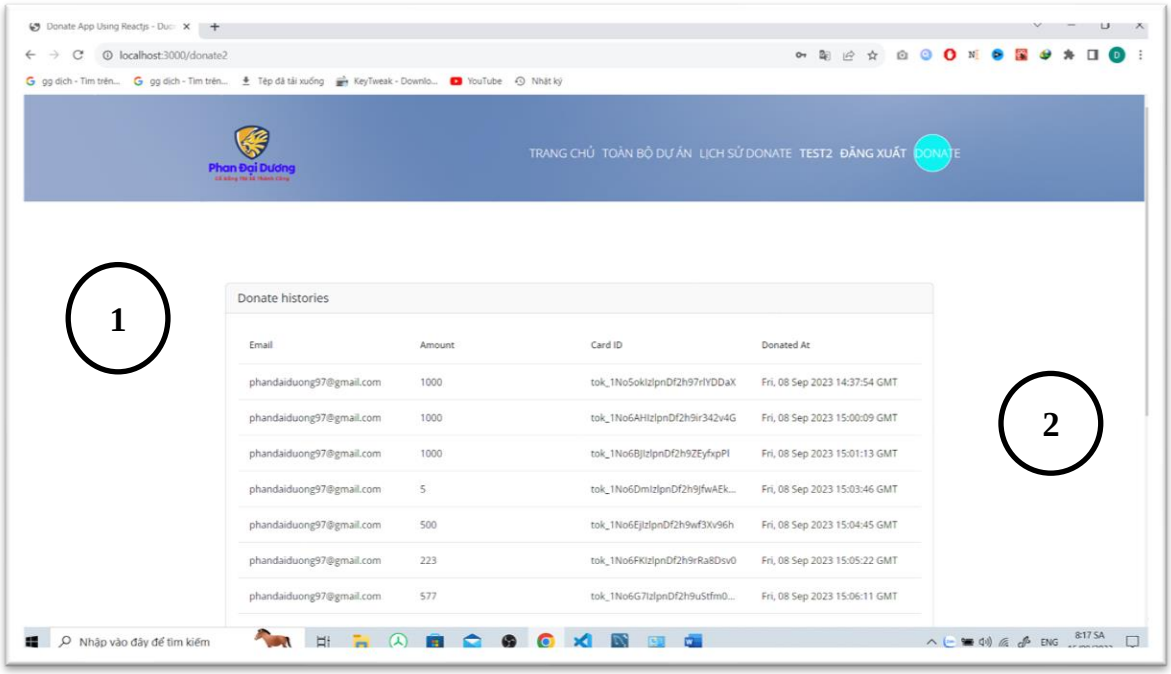
Bảng 4.8: Bảng mô tả Form Donate



Hình 4.14: Trang Thanh Toán Thành Công

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Form nhập thông tin	Mô tả số tiền người dùng muốn donate
2	Button Link	Nút liên kết
3	Alert	Mô tả thông báo trạng thái khi donate thành công

Bảng 4.9: Bảng mô tả Trang Thanh toán thành công

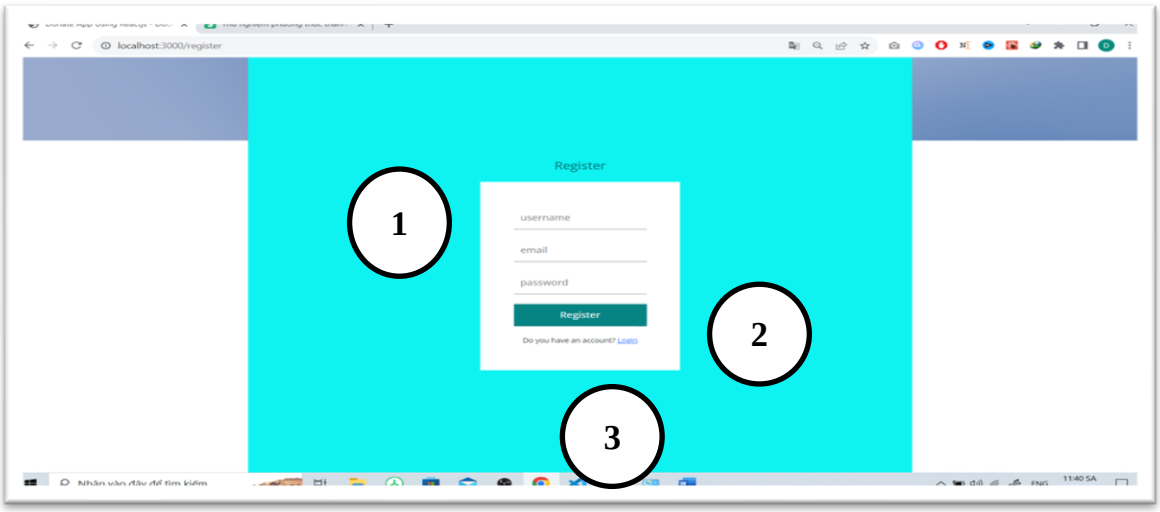


Hình 4.15: Trang lịch sử Donate sau khi thanh toán

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Table dữ liệu	Chứa thông tin lịch sử người dùng donate
2	Dữ liệu đươc thêm vào	Mô tả dữ liệu được thêm vào sau khi donate

Bảng 4.10: Bảng mô tả Trang lịch sử Donate sau khi thanh toán

4.8. **Đăng ký**

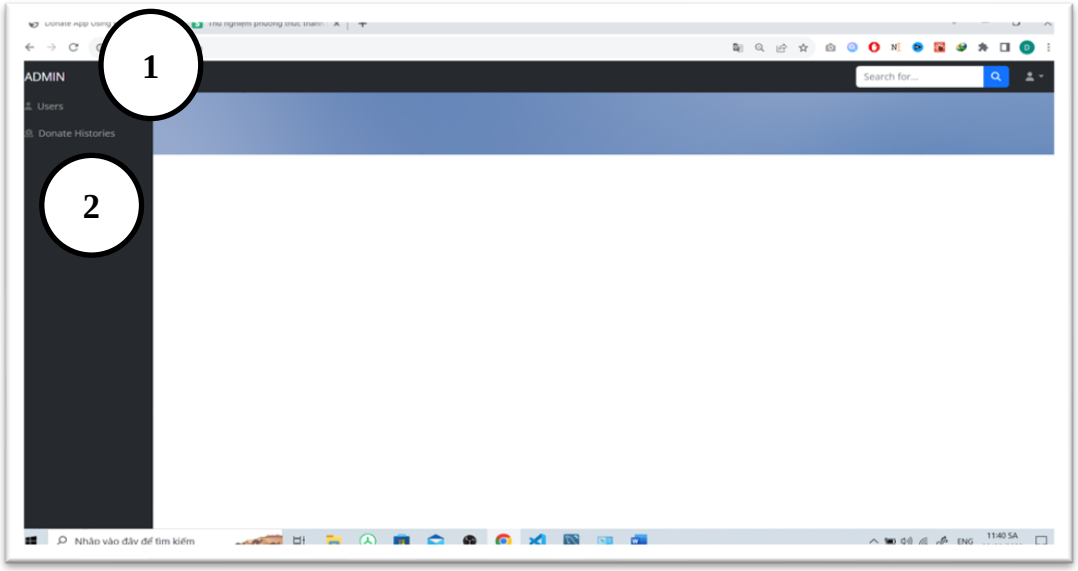


Hình 4.16: Trang đăng ký

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Form nhập thông tin	Biểu mẫu nhập thông tin đăng ký
2	Button link	Nút liên kết đến trang website
3	Link	Link liên kết đến trang web

Bảng 4.11: Bảng mô tả Trang đăng ký

4.9. **Trang Admin**

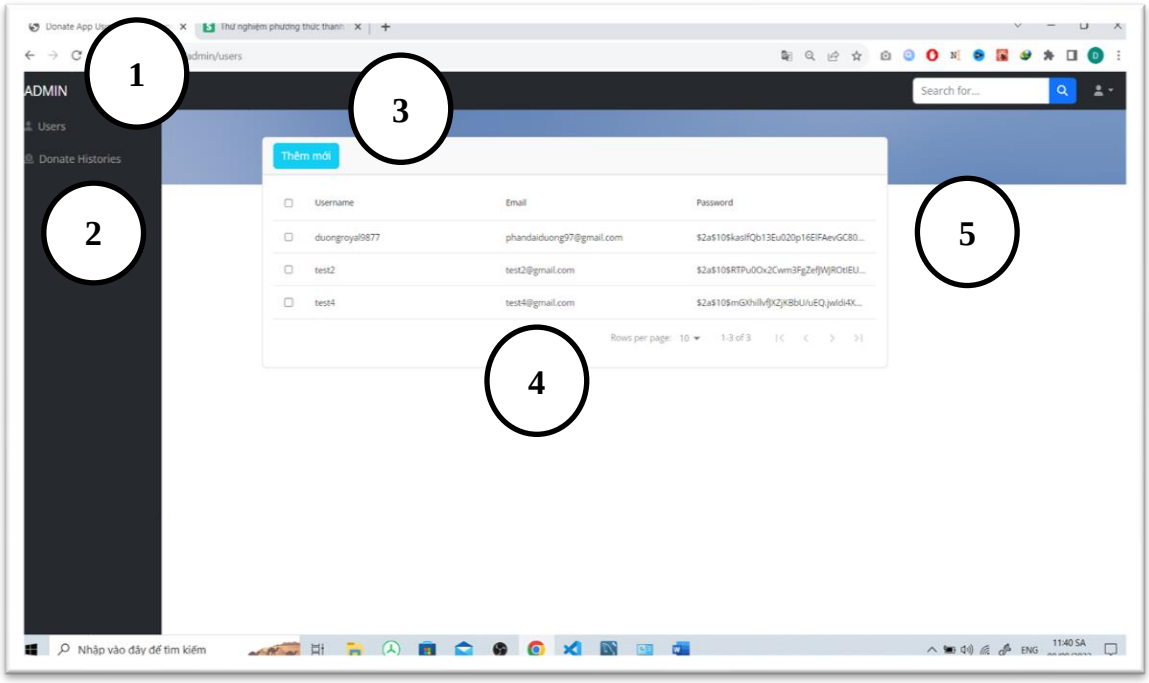


Hình 4.17: Trang chủ trang Admin

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Trang Admin Quản Lý User	Mô tả trang Admin Quản Lý User
2	Trang lịch sử donate	Mô tả trang lịch sử donate

Bảng 4.12: Bảng mô tả Trang chủ trang Admin

4.10. **Trang Admin Quản Lý User**

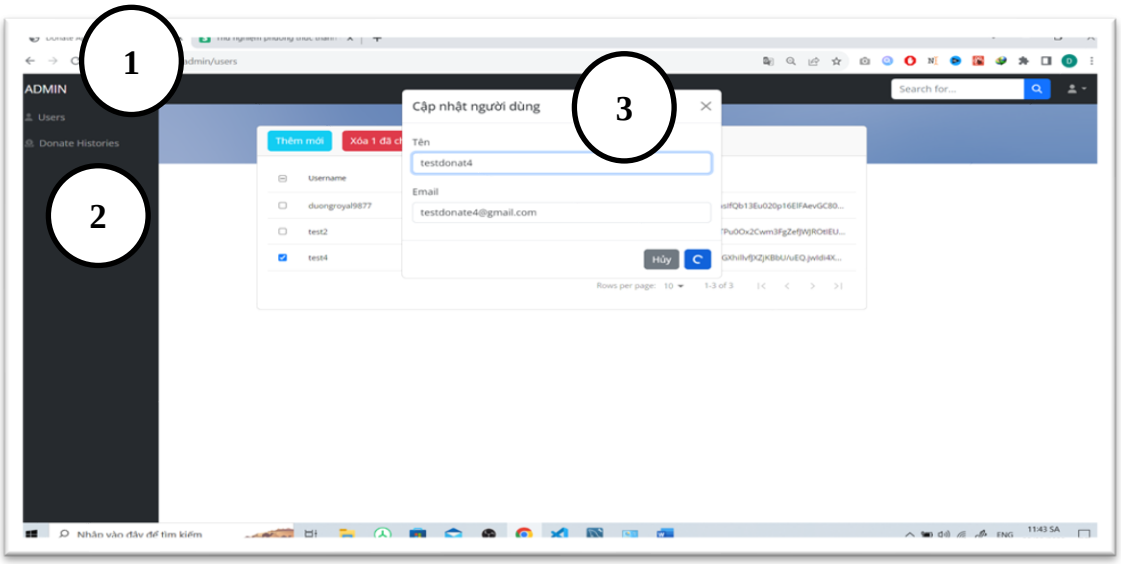


Hình 4.18: Trang Admin Quản Lý User

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Trang Admin Quản Lý User	Mô tả trang Admin Quản Lý User
2	Trang lịch sử donate	Mô tả trang lịch sử donate
3	Button Link	Nút liên kết
4	Row per page	Hàng trên mỗi trang, dùng thu nhỏ hoặc phóng to các hàng dữ liệu
5	Table dữ liệu	Dữ liệu donate của người dùng

Bảng 4.13: Bảng mô tả Trang Admin quản lý User

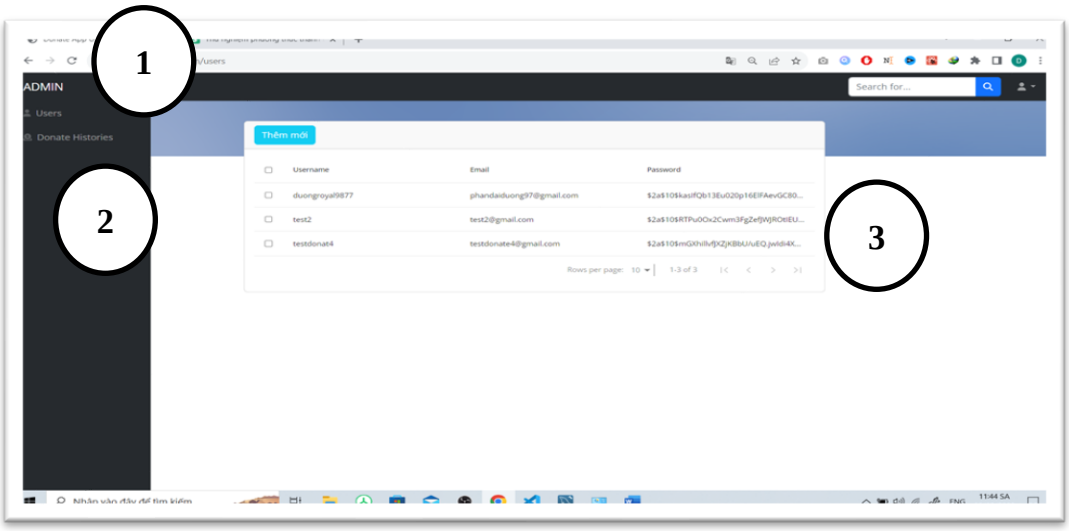
4.11. **Trang Admin sửa User**



Hình 4.19: Trang Admin Sửa Tài Khoản User

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Trang Admin Quản Lý User	Mô tả trang Admin Quản Lý User
2	Trang lịch sử donate	Mô tả trang lịch sử donate
3	Form cập nhật người dùng	Biểu mẫu cập nhật tài khoản người dùng

Bảng 4.14: Bảng mô tả Trang Admin Sửa tài khoản User

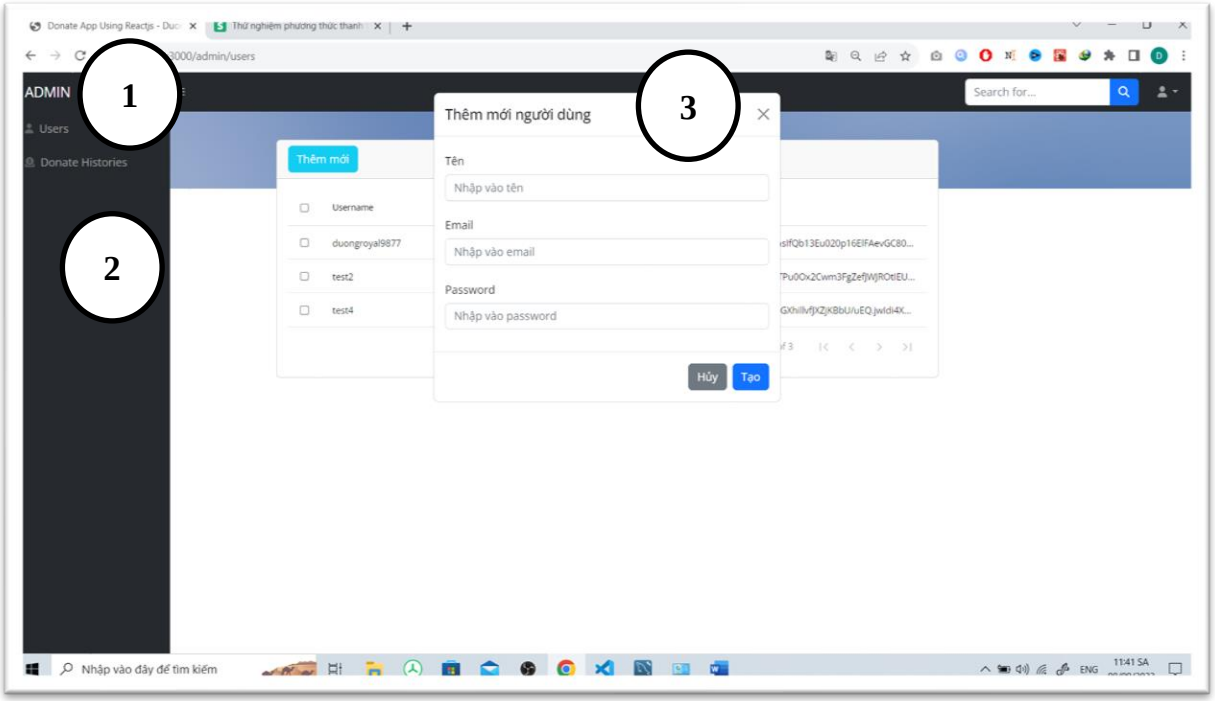


Hình 4.20: Trang sau khi Admin Sửa Tài Khoản User

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Trang Admin Quản Lý User	Mô tả trang Admin Quản Lý User
2	Trang lịch sử donate	Mô tả trang lịch sử donate
3	Table dữ liệu	Biểu mẫu sau khi cập nhật tài khoản người dùng

Bảng 4.15: Bảng mô tả Trang sau khi Admin sửa tài khoản User

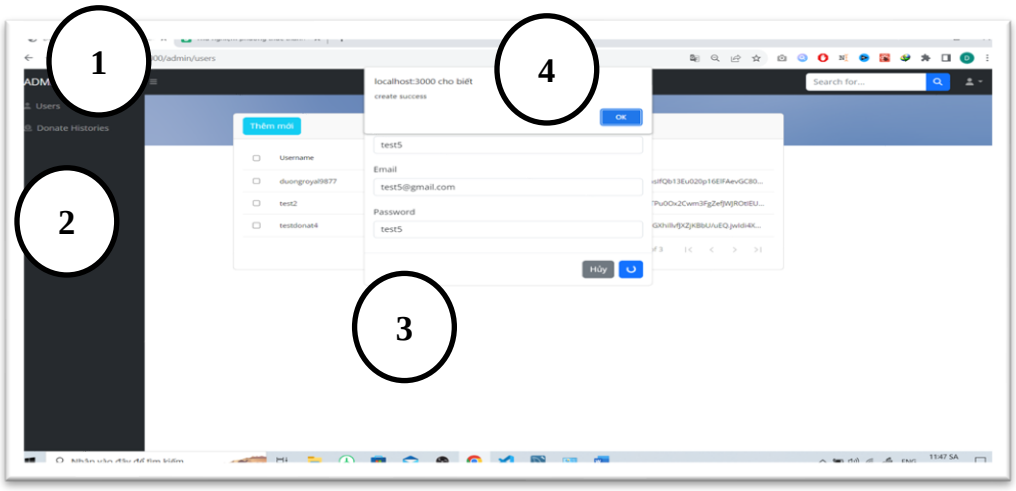
4.12. **Trang thêm User**



Hình 4.21: Trang Admin Thêm Tài Khoản User

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Trang Admin Quản Lý User	Mô tả trang Admin Quản Lý User
2	Trang lịch sử donate	Mô tả trang lịch sử donate
3	Form thêm mới người dùng	Biểu mẫu thêm mới tài khoản người dùng

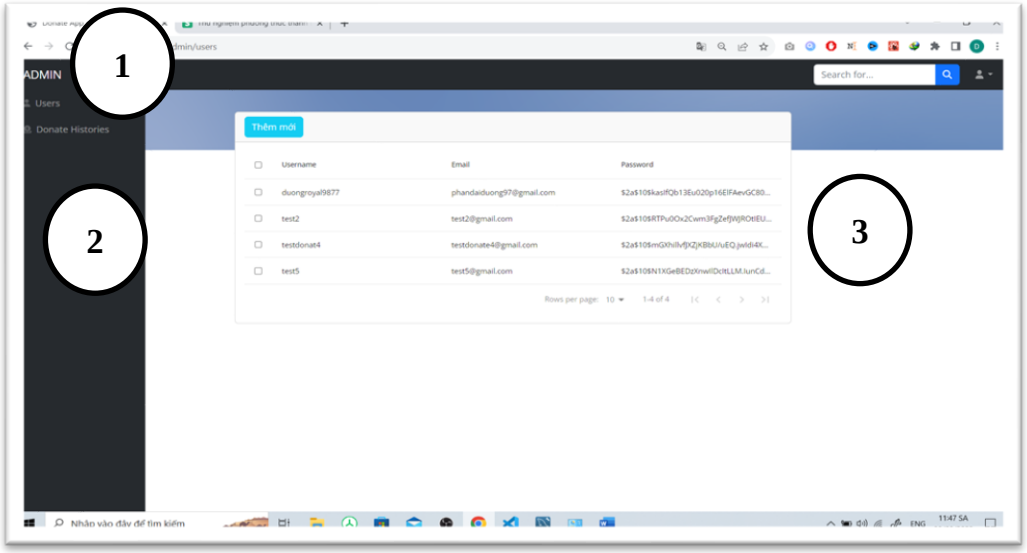
Bảng 4.16: Bảng mô tả trang Admin thêm tài khoản User



Hình 4.22: Trang nhập thông tin khi Admin Thêm Tài Khoản User

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Trang Admin Quản Lý User	Mô tả trang Admin Quản Lý User
2	Trang lịch sử donate	Mô tả trang lịch sử donate
3	Form thêm mới người dùng	Biểu mẫu thêm mới tài khoản người dùng
4	Alert	Mô tả thông báo trạng thái khi thêm mới tài khoản người dùng thành công

Bảng 4.17: Bảng mô tả Trang nhập thông tin khi Admin thêm tài khoản User

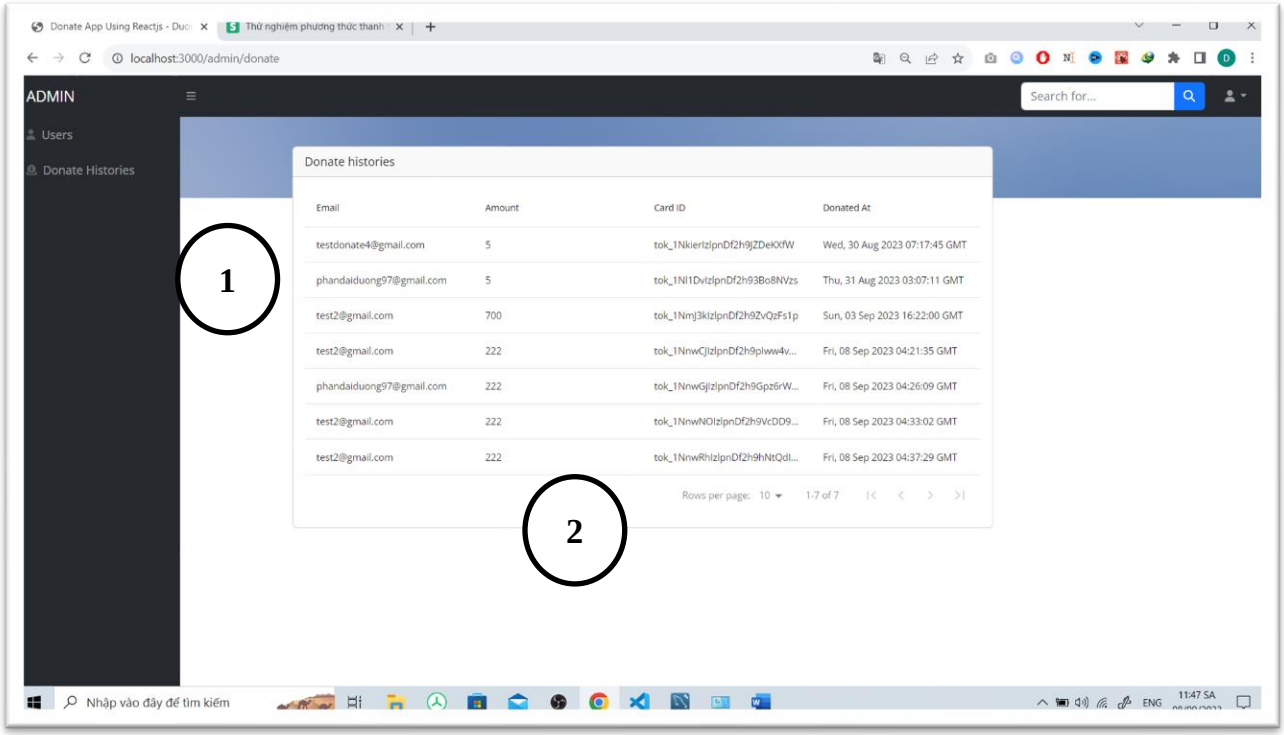


Hình 4.23: Trang nhập thông tin khi Admin Thêm Tài Khoản User

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Trang Admin Quản Lý User	Mô tả trang Admin Quản Lý User
2	Trang lịch sử donate	Mô tả trang lịch sử donate
3	Table dữ liệu	Biểu mẫu sau khi thêm tài khoản người dùng

Bảng 4.18: Bảng mô tả Trang nhập thông tin khi Admin thêm tài khoản User

4.13. **Trang Lịch Sử Admin**

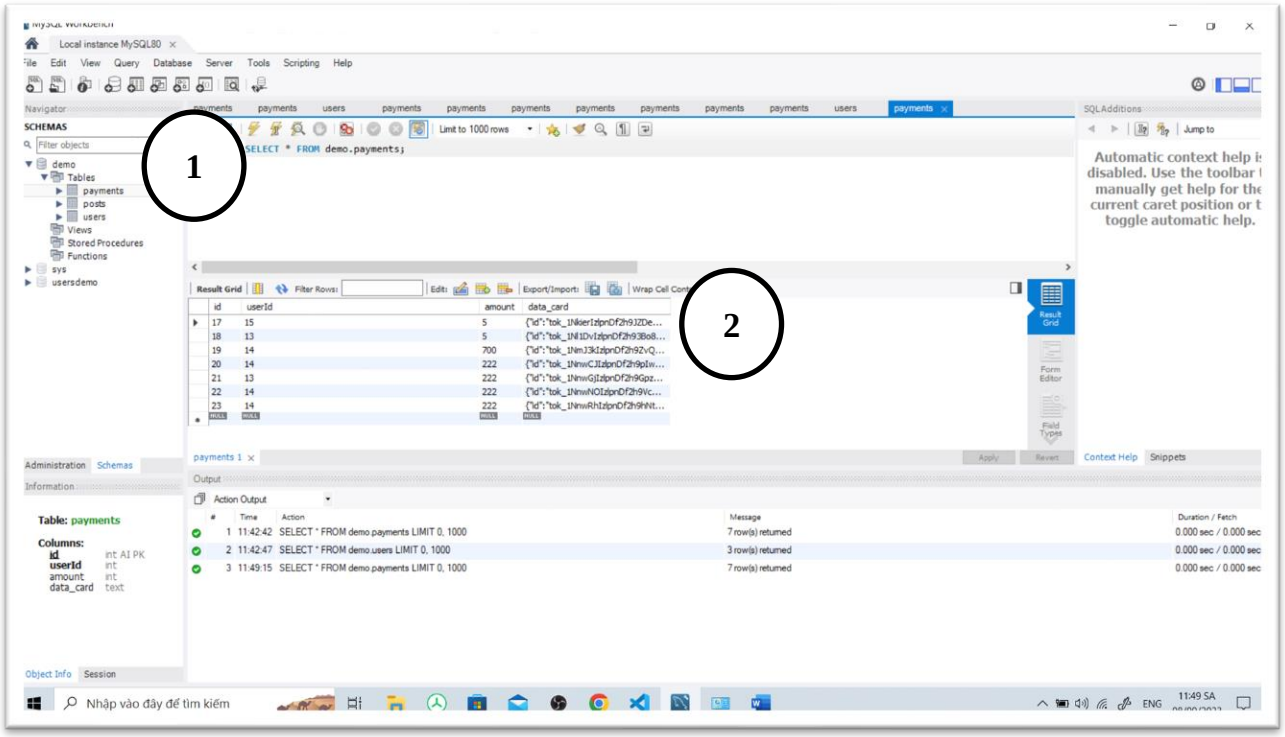


Hình 4.24: Trang Lịch Sử Admin

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Table dữ liệu	Chứa thông tin lịch sử người dùng donate
2	Row per page	Hàng trên mỗi trang, dùng thu nhỏ hoặc phóng to các hàng dữ liệu

Bảng 4.19: Bảng mô tả Trang lịch sử Admin

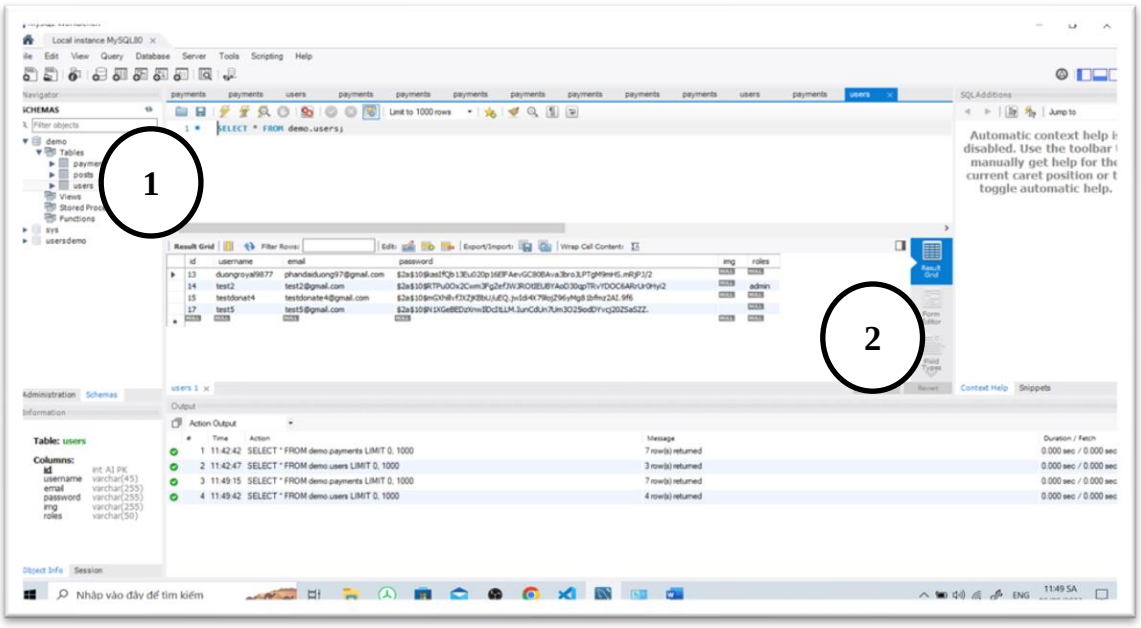
4.14. **Trang Dữ Liệu MySQL**



Hình 4.25: Trang Dữ Liệu MySQL

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Date base Mysql	Chứa thông tin dữ liệu data base
2	Table dữ liệu donate	Table dữ liệu lịch sử người dùng donate

Bảng 4.20: Bảng mô tả Trang dữ liệu MySQL

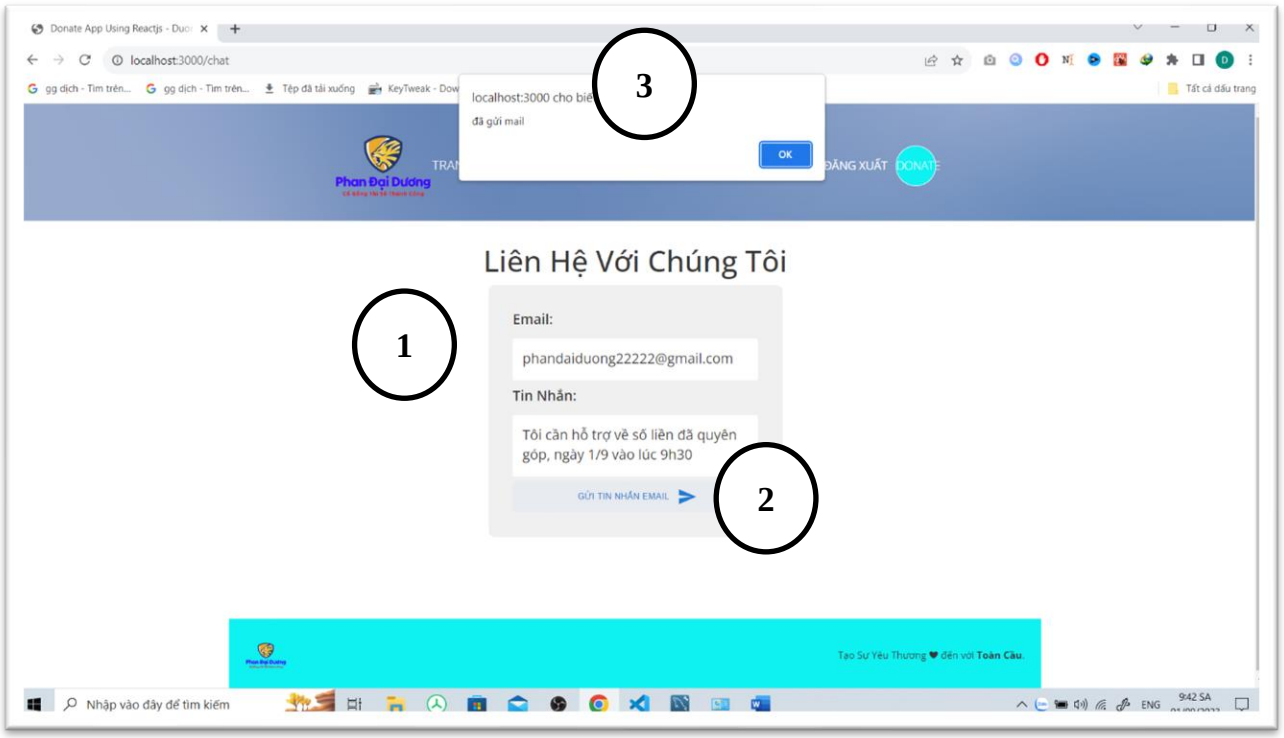


Hình 4.26: Trang Dữ Liệu MySQL

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Date base Mysql	Chứa thông tin dữ liệu data base
2	Table dữ liệu donate	Table dữ liệu tài khoản người dùng và người quản lý

Bảng 4.21: Bảng mô tả Trang Dữ Liệu MySQL

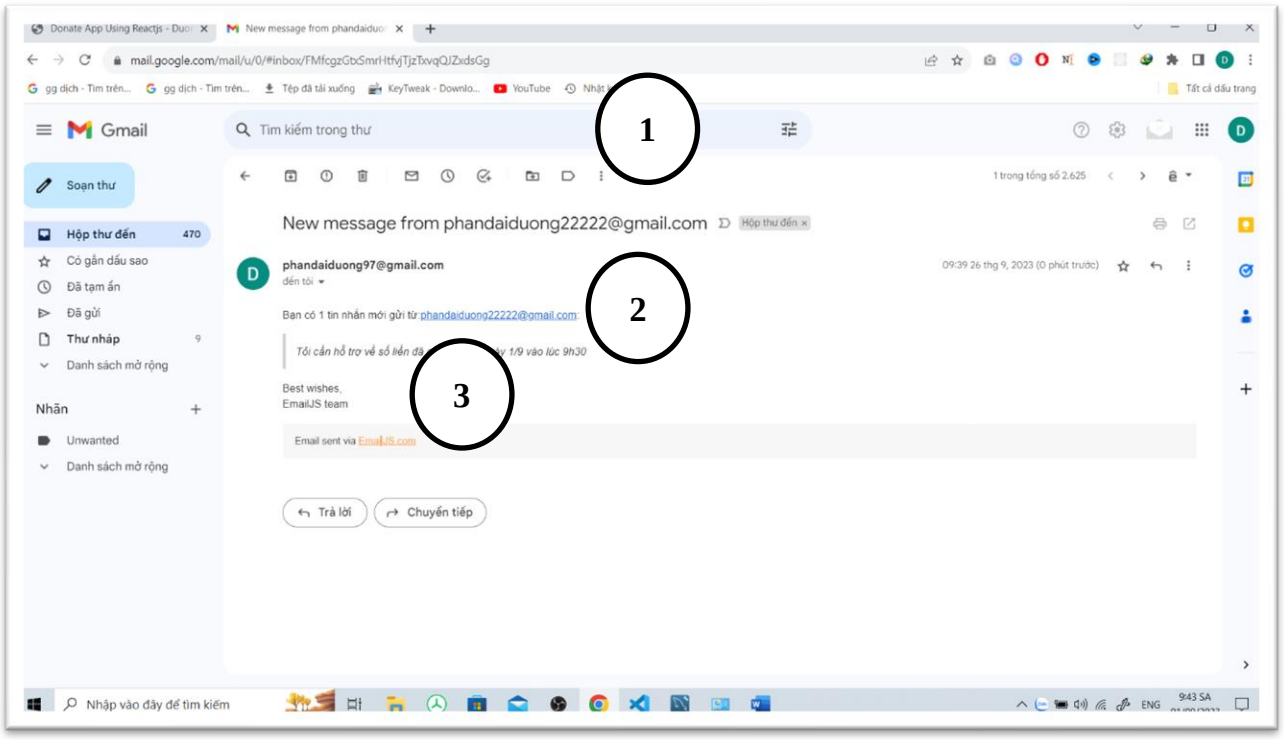
4.15. **Trang Gửi Mail hỗ trợ**



Hình 4.27: Trang gửi mail hỗ trợ

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Form nhập thông tin	Thông tin dữ liệu người dùng nhập
2	Button link	Gửi thông tin, tin nhắn của người dùng tới Email
3	Thông báo	Thông báo trạng thái

Bảng 4.22: Bảng mô tả Trang Dữ Liệu MySQL

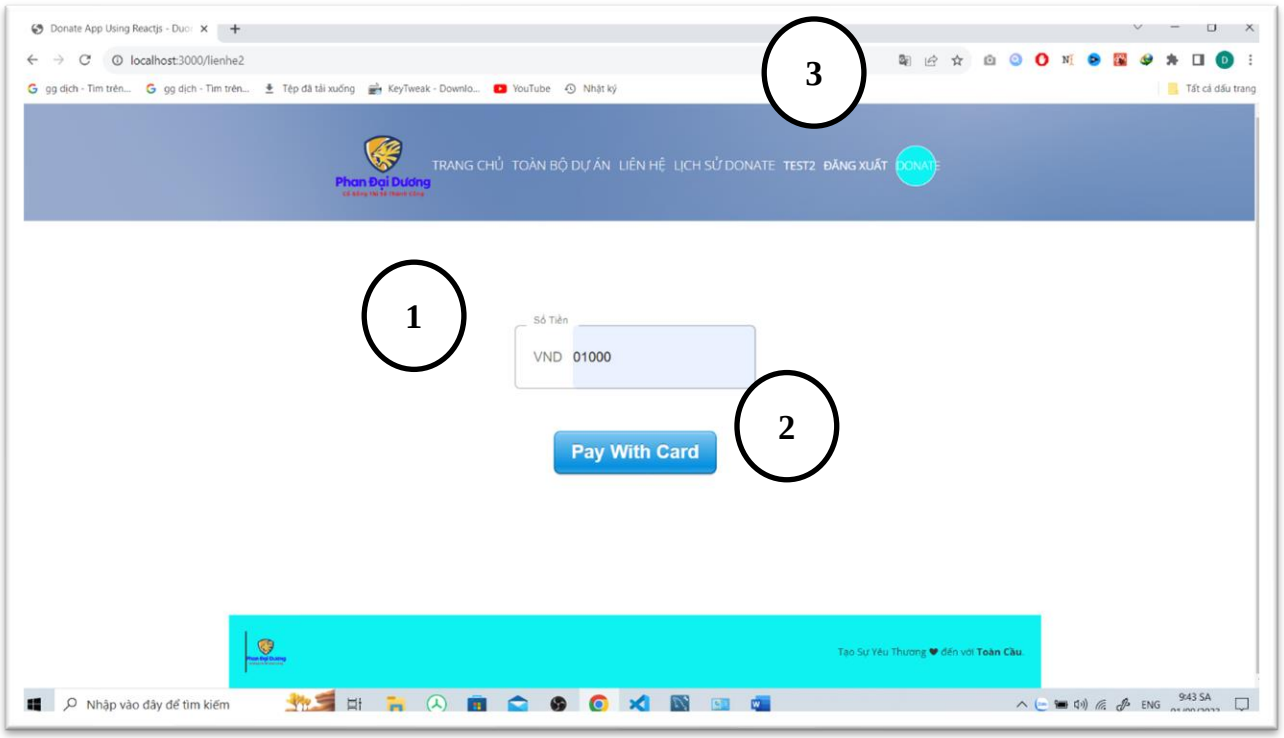


Hình 4.28: Trang Dữ Liệu MySQL

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Email người dùng	Thông tin dữ liệu người dùng
2	Thông tin người dùng	Gửi thông tin, tin nhắn của người dùng tới Email
3	Thư viện Reactjs Mailjs	Thông báo gửi từ Mailjs

Bảng 4.23: Bảng mô tả Trang Dữ Liệu MySQL

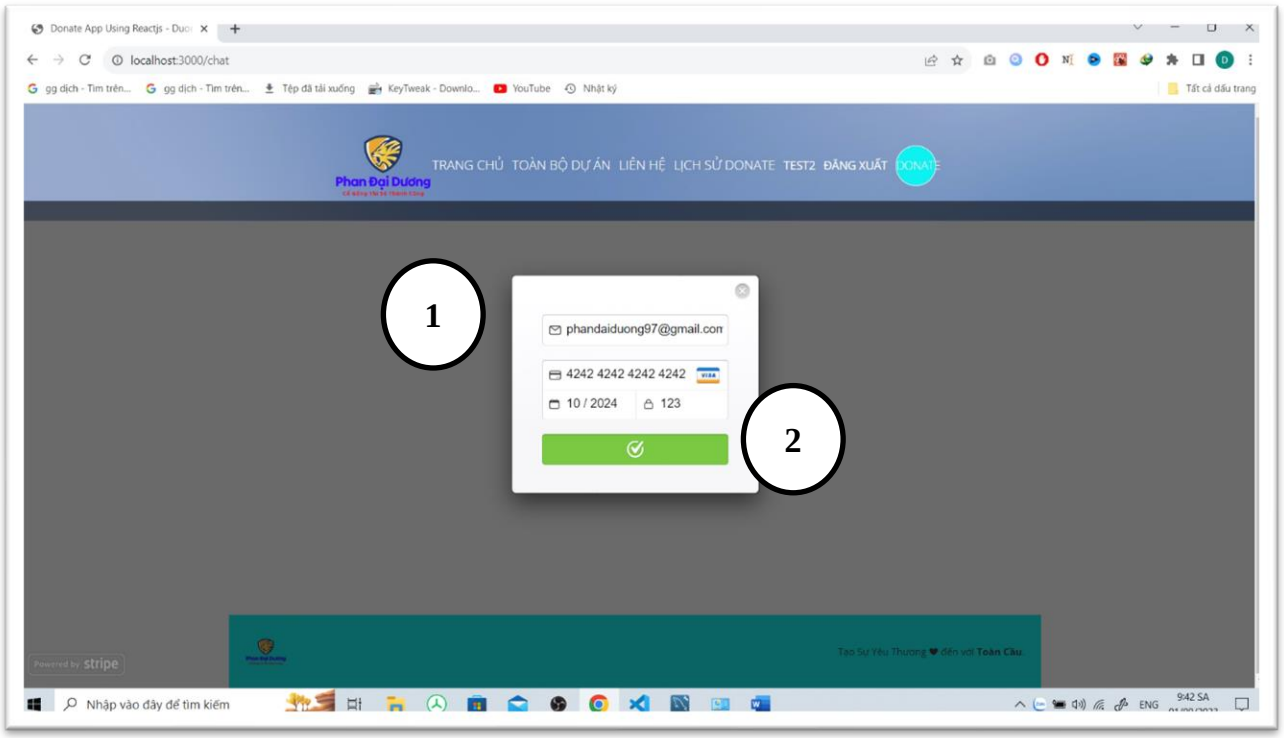
4.16. Donate bằng Tiếng Việt



Hình 4.29: Trang Donate

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Form nhập thông tin	Mô tả số tiền người dùng muốn donate
2	Button Link	Nút liên kết
3	Đăng Xuất Tài Khoản	Thoát tài khoản đã đăng nhập trên website

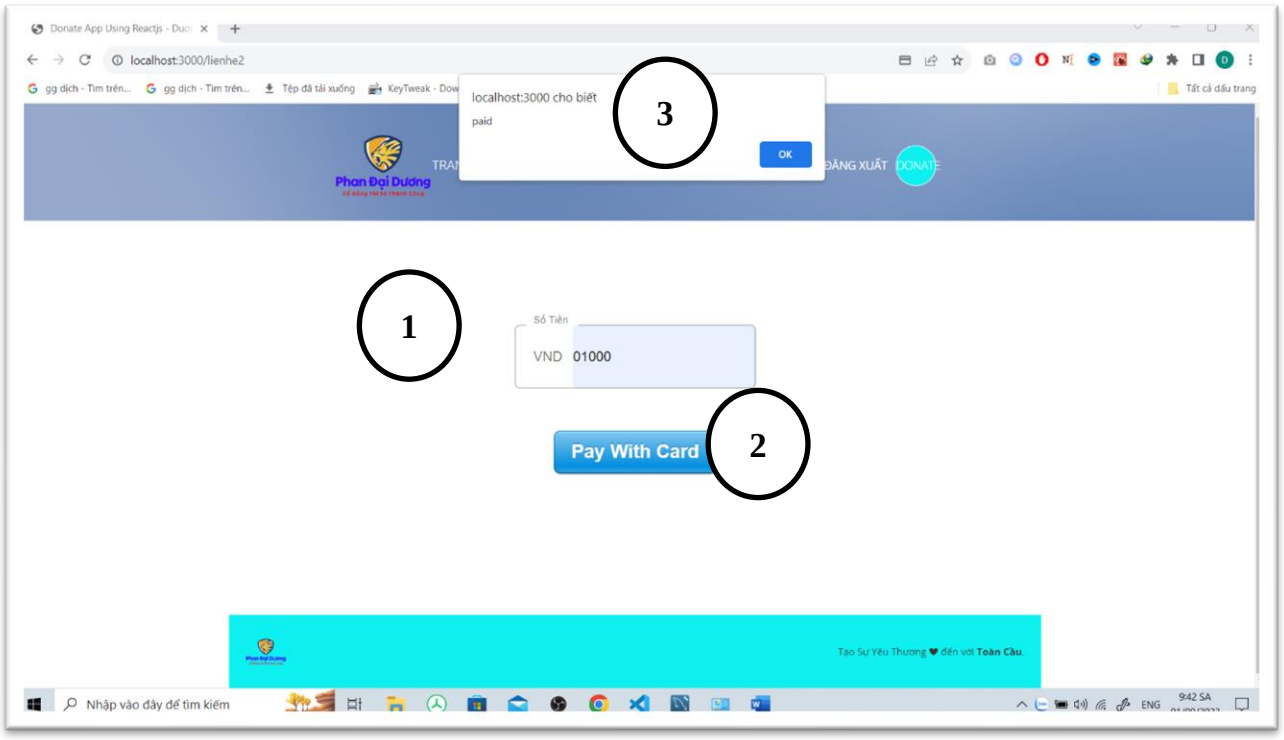
Bảng 4.24: Bảng mô tả Trang Donate



Hình 4.30: Trang Donate

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Form nhập thông tin	Mô tả số tiền người dùng muốn donate
2	Button Link	Nút liên kết

Bảng 4.25: Bảng mô tả Trang Donate



Hình 4.31: Trang Thanh Toán Thành Công

Ký hiệu	Chức năng/Thông tin	Ý nghĩa sử dụng
1	Form nhập thông tin	Mô tả số tiền người dùng muốn donate (VND)
2	Button Link	Nút liên kết
3	Alert	Mô tả thông báo trạng thái khi donate thành công

Bảng 4.26: Bảng mô tả Trang Thanh toán thành công

KẾT LUẬN

Kết quả đạt được

Các chức năng của Website gồm có:

- Đăng xuất, Đăng nhập hệ thống
- Gửi mail hỗ trợ
- Đăng ký tài khoản
- Lịch sử donate (Lịch Sử Quyên Góp)
- Donate
- Pagination
- Row per page
- Thêm, xóa, sửa, cập nhật tài khoản người dùng của trang admin

5.3 Hướng phát triển của đề tài

- Thêm các lựa chọn cho donate
- Quản lý bài viết donate
- Quên mật khẩu
- Một số chức năng chưa tối ưu
- Phần phân tích thiết kế còn lủng củng chưa rõ ràng
- Giao diện chưa được thẩm mỹ
- Vẫn trên ý tưởng này nhưng em sẽ tối ưu tốt hơn để phù hợp với nhiều máy dùng hơn..
- Chú trọng vào tối ưu tốc độ xử lý của hệ thống để đem lại một trải nghiệm tốt nhất cho người dùng.
- Nâng cao bảo mật, thiết kế cơ sở dữ liệu sử dụng mạng để liên kết được nhiều người sử dụng

TÀI LIỆU THAM KHẢO

- [1] Jordan Walke https://react.dev/, ngày tham khảo: 29/05/2013
- [2] https://www.mysql.com/products/workbench/, ngày tham khảo: 23/07/2023
- [3] https://nodejs.org/, ngày tham khảo: 20/08/2023
- [4] https://stripe.com/, ngày tham khảo: 10/08/2023
- [5] https://www.youtube.com/watch?v=DpYE5zPDRVQ, ngày tham khảo: 23/07/2023

PHỤ LỤC

Link drive phần source code:

https://drive.google.com/drive/folders/1pWnMlzis4teawidMTYe-W1LnMav6Qxmb?usp=sharing