**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỀ TÀI**

**TRANG BÁN ĐỒ ĐIỆN TỬ SECONDHAND**

|  |  |
| --- | --- |
| **MÔN HỌC :** | **KIẾN TRÚC VÀ THIẾT KẾ PHẦN MỀM** |
| **GVHD :** | **Ts.Võ Văn Hải** |

|  |  |
| --- | --- |
| **Nhóm 5** | |
| **Thành viên** | **MSSV** |
| Võ Thị Hồng Ngọc | 20063231 |
| Lê Thuỵ Thanh Thanh | 20061811 |
| Ngô Thanh Nam | 20056861 |

**MỤC LỤC**

[1. GIỚI THIỆU TỔNG QUAN 2](#_Toc1817)

[2. CÁC CÔNG NGHỆ LIÊN QUAN 4](#_Toc18844)

[2.1. Ngôn ngữ phát triển ứng dụng 4](#_Toc4301)

[2.2. Những framework hổ trợ phát triển ứng dụng 5](#_Toc3066)

[2.2.1. React Native, React.js 5](#_Toc29026)

[2.2.2. Node JS 6](#_Toc8399)

[2.2.3. Cloud Computing 7](#_Toc1609)

[2.2.4. Firebase 8](#_Toc648)

[2.2.5. Redux 8](#_Toc12278)

[2.2.6. Tailwind CSS 8](#_Toc2202)

[2.2.7. Vite  8](#_Toc17810)

[3. RS\_STORE TRANG WEB BÁN ĐỒ ĐIỆN TỬ SECONDHAND 9](#_Toc15599)

[3.1. Giới thiệu 9](#_Toc829)

[3.2. Kiến trúc phần mềm 9](#_Toc17816)

[3.3. Phân tích hệ thống công cụ RS\_Store 10](#_Toc12137)

[3.4. Giao diện đồ họa trang web 12](#_Toc19772)

[4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 15](#_Toc22557)

**RS\_Store: Trang web bán đồ điện tử secondhand**

1. **GIỚI THIỆU TỔNG QUAN**

Trong bối cảnh hiện đại, việc xử lý và tái chế các thiết bị điện tử cũ đang trở thành một vấn đề cấp bách đối với cả môi trường và xã hội. Các thiết bị điện tử, đặc biệt là điện thoại di động,,tv có vòng đời ngắn và thường xuyên được thay thế, dẫn đến lượng rác thải điện tử ngày càng tăng. Quyết định chọn đề tài bán đồ điện tử secondhandxuất phát từ nhiều nguyên nhân chủ quan và khách quan, đòi hỏi những giải pháp hiệu quả và bền vững.

***Nguyên nhân chủ quan:***

* Ý thức bảo vệ môi trường: Với sự gia tăng nhận thức về bảo vệ môi trường và phát triển bền vững, việc tái chế điện tử giúp giảm thiểu lượng rác thải độc hại ra môi trường, bảo vệ hệ sinh thái và sức khỏe con người.
* Giá trị kinh tế: Tái chế điện tử không chỉ giúp tiết kiệm tài nguyên mà còn tạo ra giá trị kinh tế từ việc tái sử dụng linh kiện và vật liệu quý như vàng, bạc, đồng trong các thiết bị cũ.
* Phát triển bền vững: Việc tái chế góp phần quan trọng trong chiến lược phát triển bền vững, giảm thiểu việc khai thác tài nguyên thiên nhiên và năng lượng tiêu thụ cho sản xuất mới.

***Nguyên nhân khách quan:***

* Lượng rác thải điện tử tăng nhanh: Với tốc độ phát triển công nghệ và sự thay thế liên tục các thiết bị điện tử, lượng rác thải điện tử toàn cầu đang tăng nhanh chóng, gây ra áp lực lớn lên hệ thống xử lý rác thải hiện có.
* Quy định và chính sách: Nhiều quốc gia và tổ chức quốc tế đã ban hành các quy định và chính sách khuyến khích hoặc bắt buộc tái chế rác thải điện tử, tạo ra môi trường pháp lý thuận lợi cho các hoạt động tái chế.
* Áp lực từ cộng đồng: Người tiêu dùng ngày càng quan tâm đến các sản phẩm thân thiện với môi trường và các công ty có trách nhiệm xã hội. Áp lực này thúc đẩy các doanh nghiệp tìm kiếm các giải pháp tái chế và quản lý rác thải điện tử hiệu quả.
* Đồ điện tử ra mắt mới hàng năm: Với tốc độ phát triển công nghệ nhanh như hiện nay, sự ra mắt sản phẩm hàng năm của các tập đoàn công nghệ khiến cho nhiều người muốn mua các thiết bị mới nhất, mặc dù mới mua và dùng chưa được lâu.

Với sự phát triển mạnh mẽ của công nghệ và lượng thiết bị điện tử ngày càng tăng, việc xây dựng một hệ thống tái chế điện tử hiệu quả và linh hoạt là điều cần thiết.

***Kiến trúc Micro-service mang lại nhiều lợi ích, bao gồm:***

* Tính linh hoạt: Các dịch vụ nhỏ, độc lập có thể được phát triển, triển khai và mở rộng một cách dễ dàng mà không ảnh hưởng đến toàn bộ hệ thống.
* Tính mô-đun: Hệ thống dễ hiểu và dễ bảo trì, cho phép cải thiện và nâng cấp từng phần một cách hiệu quả.
* Khả năng mở rộng: Hệ thống có thể tự động điều chỉnh tài nguyên để đáp ứng nhu cầu tăng cao đột ngột, giúp cải thiện hiệu suất và độ tin cậy.

***Ứng dụng công nghệ điện toán đám mây:***

* Tăng cường khả năng lưu trữ và xử lý: Điện toán đám mây cung cấp khả năng lưu trữ và xử lý dữ liệu linh hoạt, dễ dàng mở rộng hoặc thu hẹp theo nhu cầu thực tế.
* Giảm chi phí: Sử dụng các dịch vụ đám mây giúp giảm chi phí đầu tư ban đầu và chi phí vận hành do chỉ trả tiền theo nhu cầu sử dụng thực tế.
* Khả năng cân bằng tải: Điện toán đám mây cung cấp các giải pháp cân bằng tải tự động, giúp hệ thống hoạt động ổn định ngay cả khi lượng truy cập tăng cao đột ngột.

***Phát triển web:***

* Tiện lợi cho người dùng: Phát triển web giúp người dùng dễ dàng tương tác với hệ thống, kiểm tra trạng thái của thiết bị tái chế và nhận thông tin cập nhật kịp thời.
* Giảm tải cho máy chủ: Ứng dụng di động và web giúp phân tán tải trọng, giảm áp lực lên máy chủ chính và cải thiện hiệu suất hệ thống.

***Tự động hóa quy trình và báo cáo:***

* Quản lý hiệu quả: Hệ thống tự động hóa các quy trình như nhận, đánh giá, tái chế và báo cáo, giúp quản lý các hoạt động tái chế một cách hiệu quả và minh bạch.
* Báo cáo kịp thời: Tự động tạo các báo cáo hoạt động và tài chính định kỳ giúp doanh nghiệp nắm bắt được tình hình thực tế và đưa ra các quyết định kịp thời.

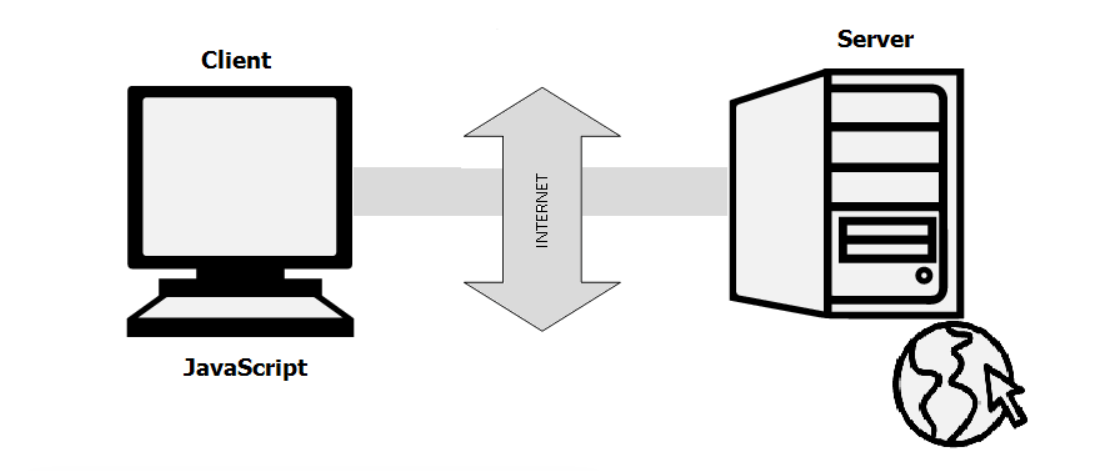
Trang web bán đồ điện tử cũ chúng tôi đề xuất sẽ bao gồm các chức năng chính như: nhận thiết bị điện tử từ khách hàng, đánh giá và định giá thiết bị. Hệ thống sẽ được chia thành hai phần chính: Front-end và Back-end. Phần Front-end sẽ bao gồm website để người dùng và quản trị viên tương tác. Phần Back-end sẽ bao gồm các dịch vụ xử lý dữ liệu và giao tiếp với cơ sở dữ liệu.

Với những giải pháp trên, hệ thống tái chế điện tử không chỉ giúp giảm thiểu lượng rác thải điện tử, bảo vệ môi trường mà còn mang lại hiệu quả kinh tế và nâng cao chất lượng dịch vụ cho người dùng. Chúng tôi tin rằng việc áp dụng kiến trúc Micro-service cho hệ thống tái chế điện tử sẽ mang lại hiệu quả cao trong việc quản lý và xử lý rác thải điện tử, đồng thời đáp ứng nhu cầu ngày càng tăng về bảo vệ môi trường và sử dụng tài nguyên một cách bền vững.

Bài báo của chúng tôi chia làm 4 phần: *Phần 1 Giới thiệu tổng quan về hệ thống*; *Phần 2 Các công nghệ liên quan*; *Phần 3 Tái chế điện tử*; *Phần 4 Kết luận và hướng phát triển*

1. **CÁC CÔNG NGHỆ LIÊN QUAN**
   1. **Ngôn ngữ phát triển ứng dụng**
      1. **Ngôn ngữ lập trình Javascrip**

JavaScript rất mạnh mẽ ngôn ngữ kịch bản phía máy khách. JavaScript được sử dụng chủ yếu để tăng cường sự tương tác của người dùng với trang web. Nói cách khác, bạn có thể làm cho trang web của mình sinh động và tương tác hơn với sự trợ giúp của JavaScript. JavaScript cũng đang được sử dụng rộng rãi trong phát triển trò chơi và di động phát triển ứng dụng.



Là một ngôn ngữ kịch bản, JavaScript không thể tự chạy. Trên thực tế, trình duyệt chịu trách nhiệm chạy mã JavaScript. Khi người dùng yêu cầu một trang HTML có JavaScript trong đó, tập lệnh sẽ được gửi đến trình duyệt và trình duyệt có quyền thực thi nó. Ưu điểm chính của JavaScript là tất cả các trình duyệt web hiện đại đều hỗ trợ JavaScript. Vì vậy, bạn không phải lo lắng về việc liệu khách truy cập trang web của bạn có sử dụng Internet Explorer hay không, Google Chrome, Firefox hoặc bất kỳ trình duyệt nào khác. JavaScript sẽ được hỗ trợ. Ngoài ra, JavaScript chạy trên bất kỳ opera hệ thống ting kể cả Windows, Linux hoặc Mac. Do đó, JavaScript khắc phục được nhược điểm chính của vbscript (Hiện không được dùng nữa) chỉ giới hạn ở IE và Windows.

* 1. **Những framework hổ trợ phát triển ứng dụng**
     1. **React Native, React.js**

React Native là một framework phát triển ứng dụng di động sử dụng ngôn ngữ JavaScript. Nó được tạo ra bởi Facebook để xây dựng các ứng dụng di động đa nền tảng với giao diện người dùng tương tự như ứng dụng native.

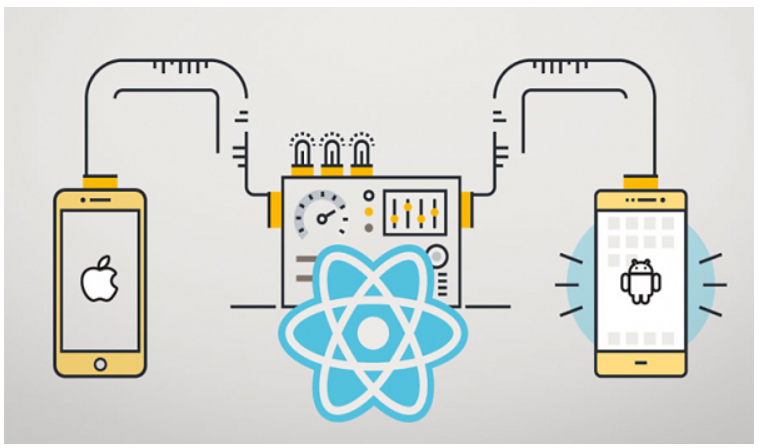
React Native cho phép lập trình viên sử dụng cùng một codebase để phát triển ứng dụng cho cả hệ điều hành iOS và Android, giúp tiết kiệm thời gian và công sức. Nó sử dụng các thành phần UI được xây dựng lại từ React, một thư viện JavaScript phổ biến để xây dựng giao diện người dùng web.

React Native hoạt động chủ yếu nhờ vào sự tích hợp của 2 thread là Main Thread và JS Thread cho các ứng dụng mobile. Trong đó:

- Nhiệm vụ của Main Thread được thể hiện trong việc cập nhật các giao diện người dùng (UI), rồi sau đó xử lý các tương tác của người dùng.

- JS Thread đảm nhận việc thực thi và tiến hành xử lý code Javascript.

Đây là 2 luồng hoạt động hoàn toàn đối lập nhau trong React. Chính vì vậy, để hai Thread có thể tương tác được với nhau thì phải sử dụng tới một cầu nối Bridge. Chúng sẽ cho phép 2 Thread có thể giao tiếp mà không quá phụ thuộc vào nhau cũng như thực hiện các chuyển đổi từ thread này sang thread khác. Các dữ liệu từ hai Thread này đều sẽ được vận hành ngay khi tiếp nối các dữ liệu cho nhau.



* + 1. **Node JS**

NodeJS là một môi trường runtime chạy JavaScript đa nền tảng và có mã nguồn mở, được sử dụng để chạy các ứng dụng web bên ngoài trình duyệt của client. Nền tảng này được phát triển bởi Ryan Dahl vào năm 2009, được xem là một giải pháp hoàn hảo cho các ứng dụng sử dụng nhiều dữ liệu nhờ vào mô hình hướng sự kiện (event-driven) không đồng bộ.

Node.JS là một trong những nền tảng phổ biến nhất hiện nay cho mục đích phát triển ứng dụng mạng phía server. Những đặc điểm khiến Node.JS là lựa chọn hàng đầu cho các developer hiện nay:

- Tốc độ cực nhanh: Được xây dựng dựa trên engine JavaScript V8 của Google Chrome, do đó các thư viện của nó có khả năng thực thi code chỉ rất nhanh.

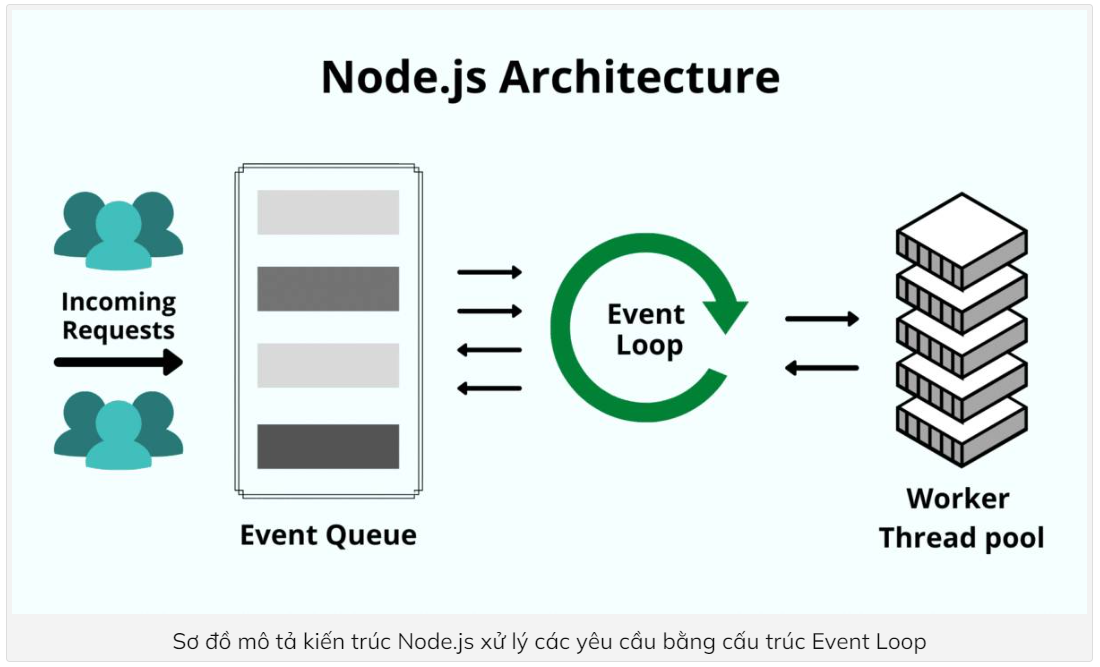
- NPM: Với hơn 50,000 package khác nhau, các developer có thể dễ dàng lựa chọn bất kỳ tính năng nào để xây dựng cho ứng dụng của mình.

- Lập trình không đồng bộ: Mọi API của Node.JS đều có tính không đồng bộ (non-blocking), do đó một server dựa trên Node.JS không cần phải đợi API trả về dữ liệu.

- Không có buffering: Node.JS giúp tiết kiệm thời gian xử lý file khi cần upload âm thanh hoặc video vì các ứng dụng này không bao giờ buffer dữ liệu mà chỉ xuất dữ liệu theo từng phần (chunk).

- Đơn luồng: Node.JS sử dụng mô hình đơn luồng với vòng lặp sự kiện. Do đó các ứng dụng có thể xử lý số lượng request lớn hơn rất nhiều so với các server truyền thống như Apache HTTP Server.

Node.js sử dụng cấu ​​trúc “Single Thread Event Loop” để xử lý nhiều máy khách cùng một lúc.



* + 1. **Cloud Computing**

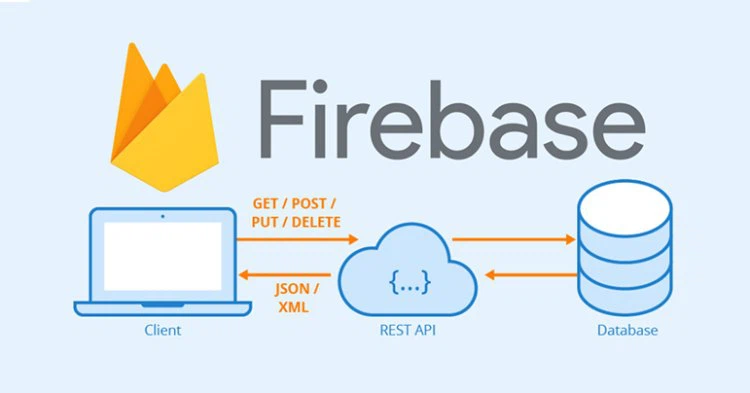
Cloud Computing, hay còn được gọi là “điện toán đám mây”, là một phương pháp phân phối tài nguyên máy tính như máy chủ, lưu trữ, cơ sở dữ liệu, mạng, phần mềm và phân tích qua Internet. Thay vì sở hữu và duy trì cơ sở hạ tầng vật lý, các tổ chức có thể truy cập các tài nguyên này theo yêu cầu, chỉ trả tiền cho những gì họ sử dụng và mở rộng quy mô hoạt động khi cần. Cloud Computing cung cấp giải pháp linh hoạt và tiết kiệm chi phí cho các doanh nghiệp thuộc mọi quy mô.

Cloud Computing hoạt động thông qua một mạng lưới các máy chủ từ xa được lưu trữ trên Internet. Các máy chủ này chịu trách nhiệm lưu trữ, quản lý và xử lý dữ liệu, cũng như chạy các ứng dụng và cung cấp dịch vụ.

Người dùng có thể truy cập các tài nguyên này từ bất cứ nơi nào có kết nối Internet, sử dụng nhiều loại thiết bị như máy tính, máy tính xách tay, điện thoại thông minh hoặc máy tính bảng. Cơ sở hạ tầng đám mây cho phép việc cộng tác liền mạch, chia sẻ dữ liệu và truy cập từ xa vào các ứng dụng và tệp nhất định nằm trong quyền truy cập.

* + 1. **Firebase**

Firebase là tập hợp các dịch vụ [điện toán đám mây](https://en.wikipedia.org/wiki/Cloud_computing" \o "Cloud computing)[phụ trợ](https://en.wikipedia.org/wiki/Backend_as_a_service" \o "Phần cuối như một dịch vụ) và nền tảng phát triển ứng dụng do [Google](https://en.wikipedia.org/wiki/Google" \o "Google) cung cấp. Nó lưu trữ cơ sở dữ liệu, dịch vụ, xác thực và tích hợp cho nhiều ứng dụng, bao gồm Android, iOS, JavaScript, Node.js, Java, Unity, PHP và C++.



* + 1. **Redux**

Redux là một [thư viện JavaScript mã](https://en.wikipedia.org/wiki/JavaScript_library" \o "Thư viện JavaScript)[nguồn mở](https://en.wikipedia.org/wiki/Open-source_software" \o "Phần mềm mã nguồn mở) để quản lý và tập trung [trạng thái](https://en.wikipedia.org/wiki/State_(computer_science)" \o "Nhà nước (khoa học máy tính)) ứng dụng . Nó được sử dụng phổ biến nhất với các thư viện như [React](https://en.wikipedia.org/wiki/React_(web_framework)" \o "Phản ứng (khung web)) hoặc [Angular](https://en.wikipedia.org/wiki/Angular_(web_framework)" \o "Góc (khung web)) để xây dựng [giao diện người dùng](https://en.wikipedia.org/wiki/User_interface" \o "Giao diện người dùng) . Tương tự như (và lấy cảm hứng từ) [kiến ​​trúc Flux](https://en.wikipedia.org/wiki/React_(JavaScript_library)" \l "Unidirectional_data_flow" \o "Phản ứng (thư viện JavaScript)) của Facebook , nó được tạo ra bởi Dan Abramov và Andrew Clark. Kể từ giữa năm 2016, người bảo trì chính là Mark Erikson và Tim Dorr.

Redux là một thư viện nhỏ với API đơn giản, hạn chế được thiết kế để trở thành nơi chứa có thể dự đoán được cho trạng thái ứng dụng. Nó hoạt động theo kiểu tương tự như [hàm rút gọn](https://en.wikipedia.org/wiki/Fold_(higher-order_function)" \o "Gấp (chức năng bậc cao hơn)) , một khái niệm [lập trình hàm](https://en.wikipedia.org/wiki/Functional_programming" \o "Lập trình chức năng) .

***Nguyên lý hoạt động của Redux dựa trên một số khái niệm và quy trình cơ bản sau:***

*Single source of truth (Nguồn dữ liệu đơn duy nhất):* Tất cả trạng thái của ứng dụng được lưu trữ trong một đối tượng gọi là "store". Store là một cây trạng thái (state tree) không thể thay đổi trực tiếp.

*State is read-only (Trạng thái chỉ đọc):* Trạng thái không được thay đổi trực tiếp. Để thay đổi trạng thái, ta cần tạo ra một hành động (action) mô tả sự thay đổi.

*Changes are made with pure functions (Thay đổi được thực hiện bằng các hàm thuần túy):* Reducer là các hàm thuần túy nhận vào trạng thái hiện tại và hành động, và trả về một trạng thái mới. Việc thay đổi trạng thái xảy ra thông qua việc gọi các reducer.

*Unidirectional data flow (Luồng dữ liệu một chiều):* Khi một hành động được gửi đến store, nó sẽ được chuyển đến reducer để xử lý và tạo ra trạng thái mới. Sau đó, trạng thái mới được cập nhật trong store và thông báo cho các thành phần ứng dụng khác về sự thay đổi.

*Time-travel debugging (Gỡ lỗi theo thời gian):* Redux hỗ trợ tính năng gỡ lỗi theo thời gian, cho phép bạn xem lại các hành động đã xảy ra và trạng thái tại mỗi thời điểm trong quá trình phát triển.

Quy trình cơ bản khi sử dụng Redux bao gồm:

*Định nghĩa trạng thái ban đầu:* Xác định trạng thái ban đầu của ứng dụng và khởi tạo store.

*Định nghĩa các hành động (actions):* Xác định các hành động mà ứng dụng có thể thực hiện, mô tả các sự kiện và dữ liệu đi kèm.

* + 1. **TailwindCSS**

CSS Tailwind là một [khung CSS](https://en.wikipedia.org/wiki/CSS_framework" \o "CSS framework)[nguồn mở](https://en.wikipedia.org/wiki/Open-source" \o "Mã nguồn mở) . Tính năng chính của thư viện này là, không giống như các khung CSS khác như [Bootstrap](https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)" \o "Bootstrap (front-end framework)) , nó không cung cấp một loạt các lớp được xác định trước cho [các thành phần](https://en.wikipedia.org/wiki/HTML_element" \o "HTML element) như nút hoặc bảng. Thay vào đó, nó tạo ra một danh sách các lớp CSS "tiện ích" có thể được sử dụng để tạo kiểu cho từng thành phần bằng cách trộn và kết hợp.

* + 1. **Vite**

Vite là một máy chủ phát triển cục bộ được viết bởi Evan You, [[1]](https://en.wikipedia.org/wiki/Vite_(software)" \l "cite_note-1) người tạo ra [Vue.js](https://en.wikipedia.org/wiki/Vue.js" \o "Vue.js) và được sử dụng theo mặc định bởi [Vue](https://en.wikipedia.org/wiki/Vue.js" \o "Vue.js) và cho các mẫu dự án [React](https://en.wikipedia.org/wiki/React_(software)" \o "Phản ứng (phần mềm)) . Nó có hỗ trợ cho [TypeScript](https://en.wikipedia.org/wiki/TypeScript" \o "Bản đánh máy) và [JSX](https://en.wikipedia.org/wiki/JSX_(JavaScript)" \o "JSX (JavaScript)) . Nó sử dụng Rollup và [esbuild](https://en.wikipedia.org/wiki/Esbuild" \o "Esbuild) nội bộ để đóng gói. [[2]](https://en.wikipedia.org/wiki/Vite_(software)" \l "cite_note-2)

Nó giám sát các tệp khi chúng đang được chỉnh sửa và khi lưu tệp, trình duyệt web sẽ tải lại mã đang được chỉnh sửa thông qua quy trình có tên là Thay thế mô-đun nóng (HMR) [[3],](https://en.wikipedia.org/wiki/Vite_(software)" \l "cite_note-3) quy trình này hoạt động bằng cách chỉ tải lại tệp cụ thể đang được thay đổi bằng mô-đun ES6 (ESM) thay vì biên dịch lại toàn bộ ứng dụng.

Vite cung cấp hỗ trợ tích hợp cho kết xuất phía máy chủ (SSR). Theo mặc định, nó lắng nghe trên cổng TCP 5173. Có thể định cấu hình Vite để phân phát nội dung qua [HTTPS](https://en.wikipedia.org/wiki/HTTPS" \o "HTTPS) và các yêu cầu proxy (bao gồm [WebSocket](https://en.wikipedia.org/wiki/WebSocket" \o "WebSocket) ) tới máy chủ web phụ trợ (chẳng hạn như [Apache HTTP Server](https://en.wikipedia.org/wiki/Apache_HTTP_Server" \o "Máy chủ HTTP Apache) hoặc [lighttpd](https://en.wikipedia.org/wiki/Lighttpd" \o "Lighttpd) ).

Đặc điểm và chức năng :

ViteJS ra đời sau nên các chức năng cũng na ná Vue-CLI. Cũng scaffold, dev server with hot reload,v.v..

Ngoài ra, ViteJS hỗ trợ các ngôn ngữ, biến thể JS, CSS rất tốt như:

+ Hỗ trợ Typescript, JSX và TSX

+ Hỗ trợ các CSS preprocessor như .scss, .less

+ Static assets như images và JSON files

Tuy nhiên, Vite cũng có một vài khác biệt như:

+ Vite không based trên Webpack. Thay vào đó là sử dụng Rollup.

+ DevServer sử dụng native ES modules trên trình duyệt

1. **RS\_STORE TRANG WEB BÁN ĐỒ ĐIỆN TỬ SECONDHAND** 
   1. **Giới thiệu**

Để triển khai các giải pháp mà chúng tôi nêu ra trong phần giới thiệu, chúng tôi đề xuất xây dựng trang web RS\_Store. Trang web này chúng tôi xây dựng dựa vào các công nghệ như ReactJS, React Navtive, Node JS, Redux, Tailwind CSS, Vite. Đồng thời, để triển khai trang web RS\_Store đến thiết bị người dùng, chúng tôi cũng dùng các dịch vụ như Firebase. Trong đó, ReactJS tạo trang web trên các trình duyệt như Google, phục vụ cho các chức năng của người dùng.

Firebase dùng lưu trữ dữ liệu của toàn hệ thống như dữ liệu thiết bị điện tử, dữ liệu tài khoản, thông tin người dùng…Các dịch vụ điện toán đám mây sử dụng trong Firebase gồm: Authentication, Cloud Firestore. Cụ thể các dịch vụ này đảm nhiệm các công việc như sau: Authentication xác nhận thông tin đăng nhập, đăng ký như email và mật khẩu ; Cloud Firestore nơi lưu trữ các thông tin thiết bị điện tử .

* 1. **Kiến trúc phần mềm**

Microservices là một kỹ thuật phát triển phần mềm, một biến thể thuộc kiến trúc hướng dịch vụ (SOA), cấu trúc một ứng dụng như một tập hợp các dịch vụ được ghép lỏng lẻo. Microservices là một thiết kế kiến trúc dùng để xây dựng một ứng dụng phân tán thông qua các container. Mỗi chức năng của ứng dụng hoạt động như một dịch vụ độc lập, do đó kiến trúc này được đặt tên mà micro-service. Thông qua kiến trúc này, mỗi dịch vụ có thể dễ dàng mở rộng quy mô và cập nhật mà không làm gián đoạn hay ảnh hưởng bất kỳ dịch vụ nào khác trong ứng dụng.

Một framework microservices xây dựng một hệ thống phân tán, có khả năng mở rộng và quy mô lớn, giúp giảm thiểu hiện tượng tắc nghẽn cho cơ sở dữ liệu trung tâm. Đồng thời kiến trúc này còn cải thiện các khả năng kinh doanh cho doanh nghiệp, chẳng hạn như cho phép các ứng dụng phân phối và triển khai liên tục, tiếp cận các hệ thống công nghệ hiện đại.

- Ưu điểm:

*+ Tính mô đun cao:* Ứng dụng được chia thành các dịch vụ nhỏ (như dịch vụ quản lý người dùng, dịch vụ giỏ hàng, dịch vụ hóa đơn, dịch vụ quản lý sản phẩm) giúp dễ dàng bảo trì và nâng cấp từng phần riêng lẻ mà không ảnh hưởng đến toàn bộ hệ thống.

+ *Khả năng mở rộng:*Dễ dàng mở rộng các dịch vụ độc lập dựa trên nhu cầu sử dụng. Ví dụ, nếu có nhiều người dùng truy cập để mua hàng, chỉ cần mở rộng dịch vụ giỏ hàng mà không cần mở rộng toàn bộ hệ thống.

+ *Độ tin cậy*: Nếu một dịch vụ gặp sự cố (ví dụ, dịch vụ hóa đơn), các dịch vụ khác vẫn hoạt động bình thường. Điều này giúp giảm thiểu thời gian ngừng hoạt động và nâng cao trải nghiệm người dùng.

*+ Dễ dàng triển khai*: Các dịch vụ có thể được triển khai độc lập và nhanh chóng, cho phép cập nhật và triển khai các tính năng mới một cách nhanh chóng và an toàn.

+ *Công nghệ đa dạng:*Có thể sử dụng công nghệ phù hợp nhất cho từng dịch vụ. Ví dụ, dùng Node.js cho dịch vụ giỏ hàng, Python cho dịch vụ quản lý sản phẩm, tăng cường hiệu suất và tối ưu hóa từng dịch vụ.

- Nhược điểm:

+ *Quản lý phức tạp*: Quản lý, giám sát và điều phối giữa các dịch vụ trở nên phức tạp hơn. Cần thiết lập hệ thống quản lý dịch vụ, giám sát và log tập trung để theo dõi và bảo trì hệ thống.

*+ Giao tiếp giữa các dịch vụ:*Việc giao tiếp giữa các dịch vụ qua mạng có thể làm tăng độ trễ và chi phí liên lạc. Cần tối ưu hóa các giao tiếp này để đảm bảo hiệu suất của hệ thống.

+ *Tính nhất quán dữ liệu:* Đảm bảo tính nhất quán dữ liệu giữa các dịch vụ (ví dụ, giữa dịch vụ giỏ hàng và dịch vụ hóa đơn) là một thách thức. Cần thiết lập các chiến lược đồng bộ hóa và quản lý giao dịch phân tán.

+ *Độ phức tạp trong kiểm thử:* Kiểm thử các ứng dụng Microservices phức tạp hơn do có nhiều dịch vụ tương tác với nhau. Cần có các môi trường kiểm thử phức tạp và công cụ kiểm thử tự động để đảm bảo chất lượng dịch vụ.

+*Yêu cầu về kỹ năng và công nghệ:* Đòi hỏi đội ngũ kỹ thuật có kỹ năng cao và hiểu biết về nhiều công nghệ khác nhau. Việc đào tạo và duy trì đội ngũ này cũng tốn kém và phức tạp.

* 1. **Phân tích hệ thống công cụ RS\_Store**

Đối tượng người dùng cho hệ thống RS\_Store gồm có người mua và quản trị viên của trang web. Các chức năng của từng đối tượng được mô tả trong Bảng 3.3.1. Trong đó, chúng ta tập trung vào các chức năng chính của trang web mà hiện nay người mua đang gặp phải vấn đề khó khăn như mua các đồ điện tử đã qua sử dung đã được kiểm tra.

Chức năng chính của ứng dụng được phân chia theo đối tượng người dùng:

- Chức năng cho Người dùng

+ Xem thông tin cá nhân

*Mô tả:* Người dùng có thể xem thông tin cá nhân của mình sau khi đã đăng nhập vào trang web.

*Mục tiêu:* Giúp người dùng quản lý và cập nhật thông tin cá nhân một cách thuận tiện và bảo mật.

+ Đăng nhập tài khoản

*Mô tả:* Người dùng có thể đăng nhập vào trang web bằng tài khoản đã được tạo trước đó.

*Mục tiêu:* Đảm bảo an ninh và tính riêng tư cho người dùng khi truy cập vào hệ thống.

+ Quản lý giỏ hàng

*Mô tả:* Người dùng có thể xem danh sách các đồ điện tử đã thêm vào giỏ hàng, xóa đồ điện tử khỏi giỏ hàng và xem chi tiết mô tả của từng sản phẩm trong giỏ.

*Mục tiêu:* Tạo sự thuận tiện và linh hoạt trong việc mua sắm, giúp người dùng dễ dàng quản lý và theo dõi các sản phẩm muốn mua.

+ Tra cứu hóa đơn

*Mô tả:* Người dùng có thể xem thông tin hóa đơn, bao gồm giá sản phẩm và giá giao hàng.

*Mục tiêu:* Giúp người dùng nắm bắt thông tin chi tiết về giao dịch và chi phí, đảm bảo tính minh bạch và rõ ràng.

- Chức năng cho Quản trị viên

+ Quản lý người dùng

*Mô tả:* Quản trị viên có thể xem thông tin người dùng và danh sách mua của họ.

*Mục tiêu:* Hỗ trợ quản trị viên trong việc theo dõi và quản lý hoạt động của người dùng, đảm bảo an ninh và hiệu quả cho hệ thống.

+ Quản lý đồ điện tử

*Mô tả:* Quản trị viên có thể thêm mới, xóa hoặc chỉnh sửa thông tin của các sản phẩm điện tử.

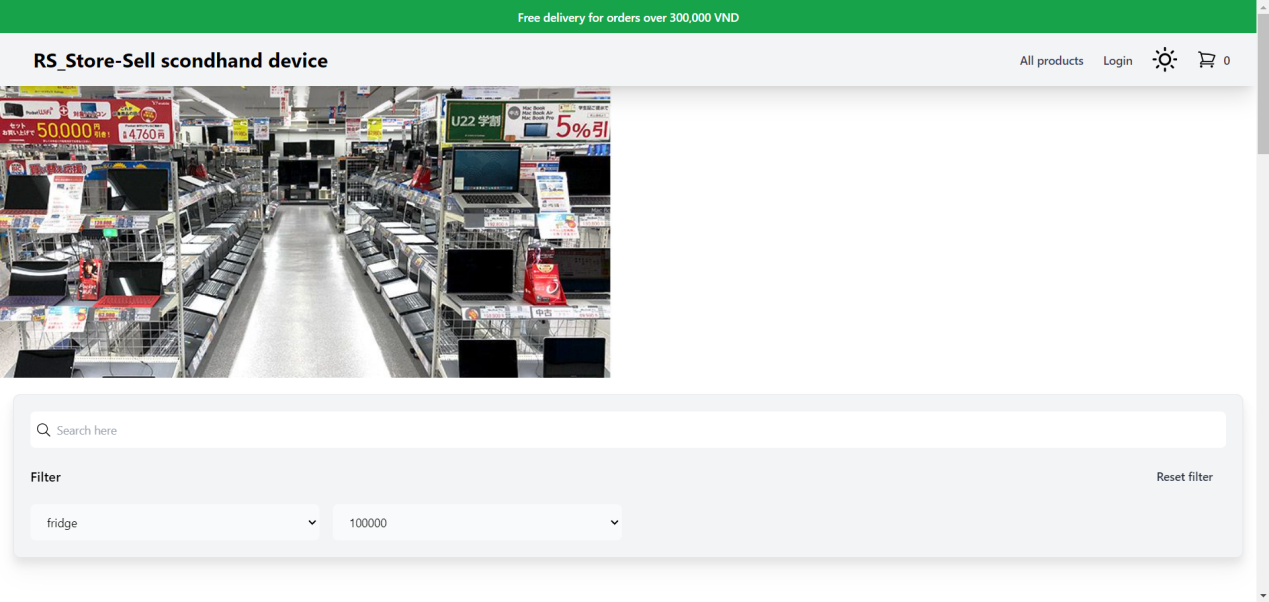
*Mục tiêu:* Cho phép quản trị viên dễ dàng quản lý danh mục sản phẩm, cập nhật thông tin một cách nhanh chóng và chính xác.

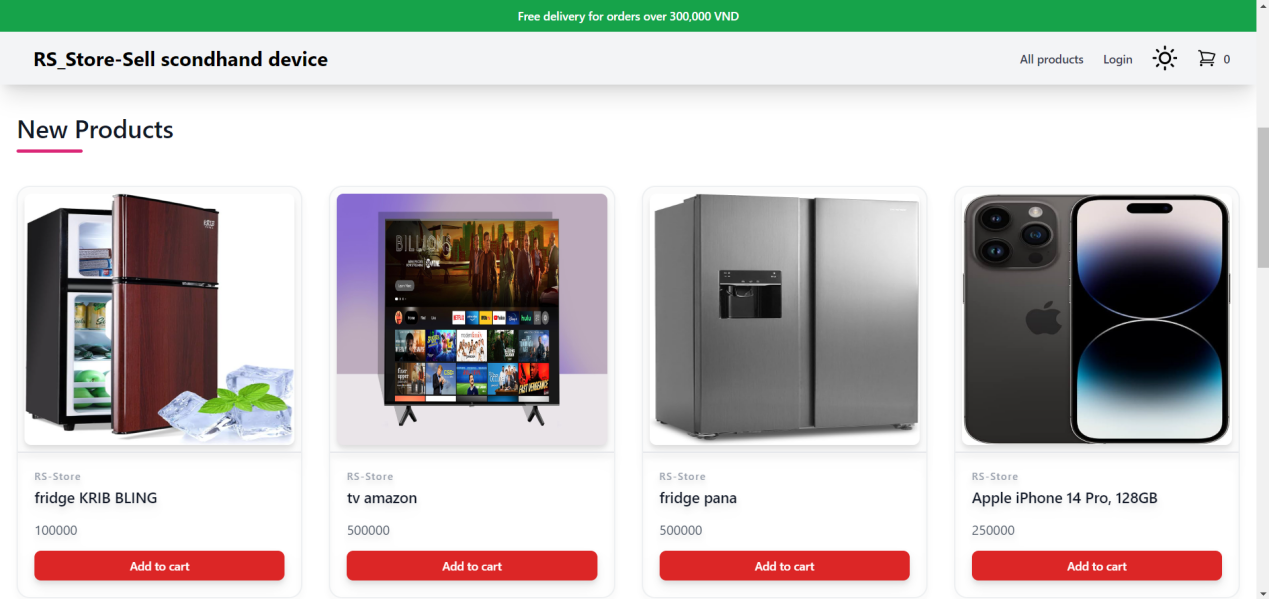
*Bảng 3.3.1 Danh các chức năng trong trang web RS\_Store*

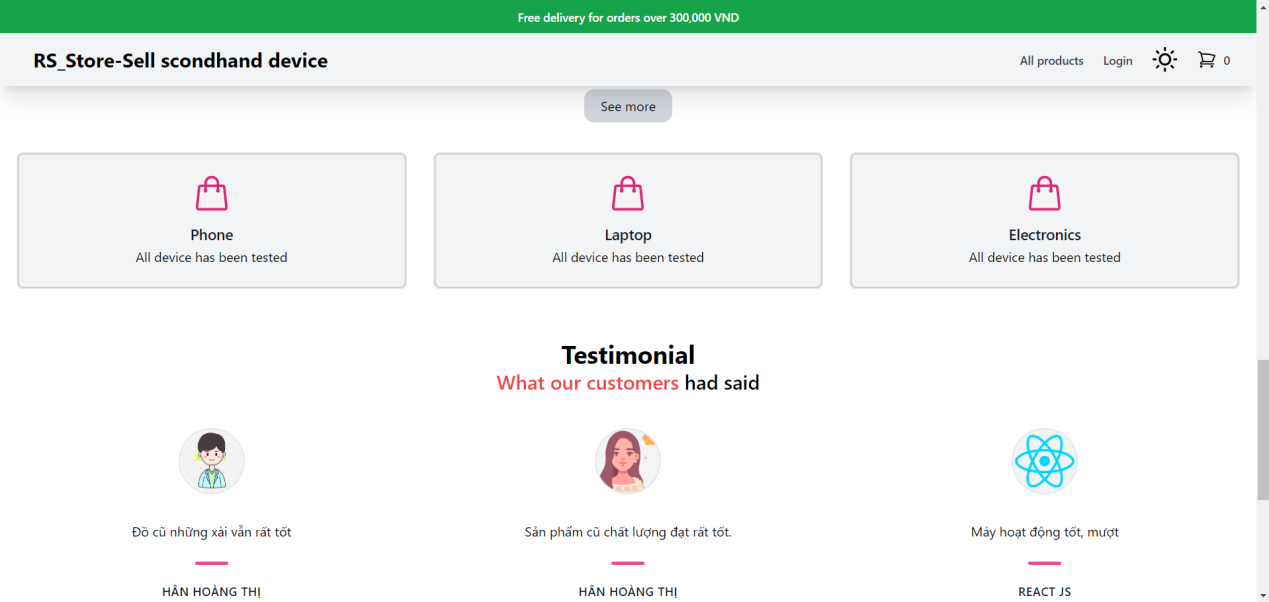
|  |  |  |
| --- | --- | --- |
| Đối tượng | Tên chức năng | Mô tả |
| Người dùng | Xem thông tin cá nhân | Xem thông tin cá nhân của người dùng đã đăng nhập vào trang web |
| Đăng nhập tài khoản | Đăng nhập vào trang web với tài khoản được tạo |
| Quản lý giỏ hàng | Gồm các chức năng như xem danh sách các đồ điện tử đã cho vào giở hàng, xóa đồ điện tử trong giỏ hàng, xem chi tiết mô tả đồ điện tử |
| Tra cứu hóa đơn | Xem thông tin hóa đơn (giá đồ điện tử, giá giao hàng) |
| Quản trị viên | Quản lý người dùng | cho phép xem thông tin người dùng, danh sách mua |
| Quản lý đồ điện tử | Cho phép thêm, xóa, sửa thông tin của đồ điện tử |

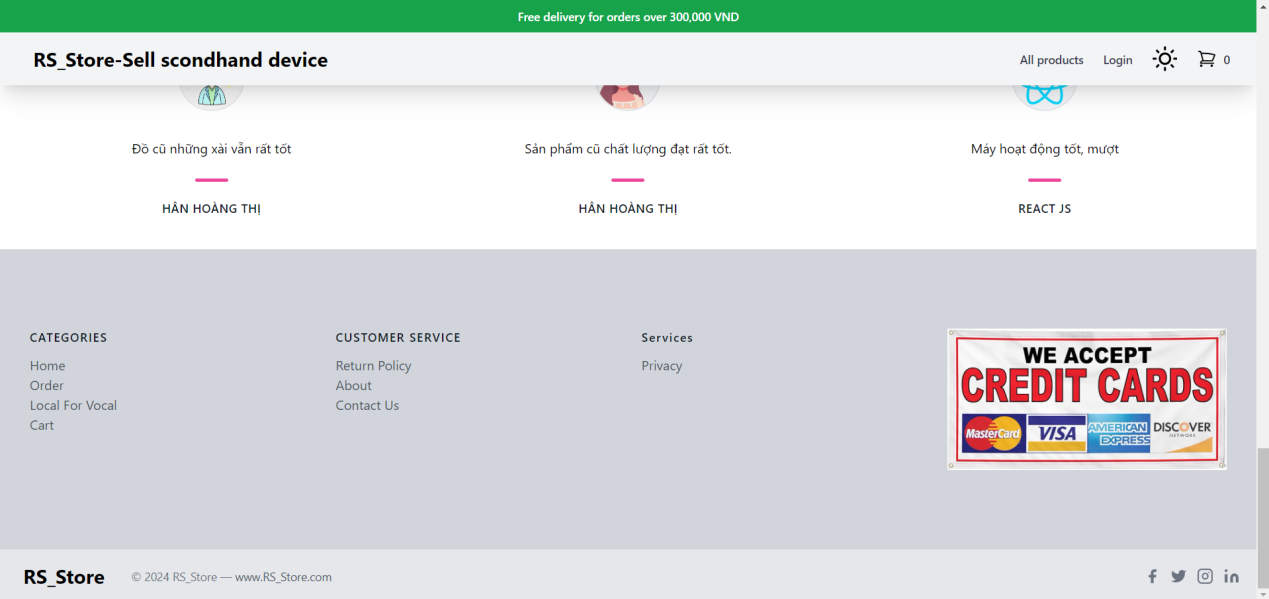
Từ những chức năng trên, chúng tôi tập trung vào các chức năng chính như: thêm đồ vào giỏ hàng, thanh toán. Sau đó chúng tôi mở rộng các chức năng tương ứng như trong *Hình 3.3.1.*

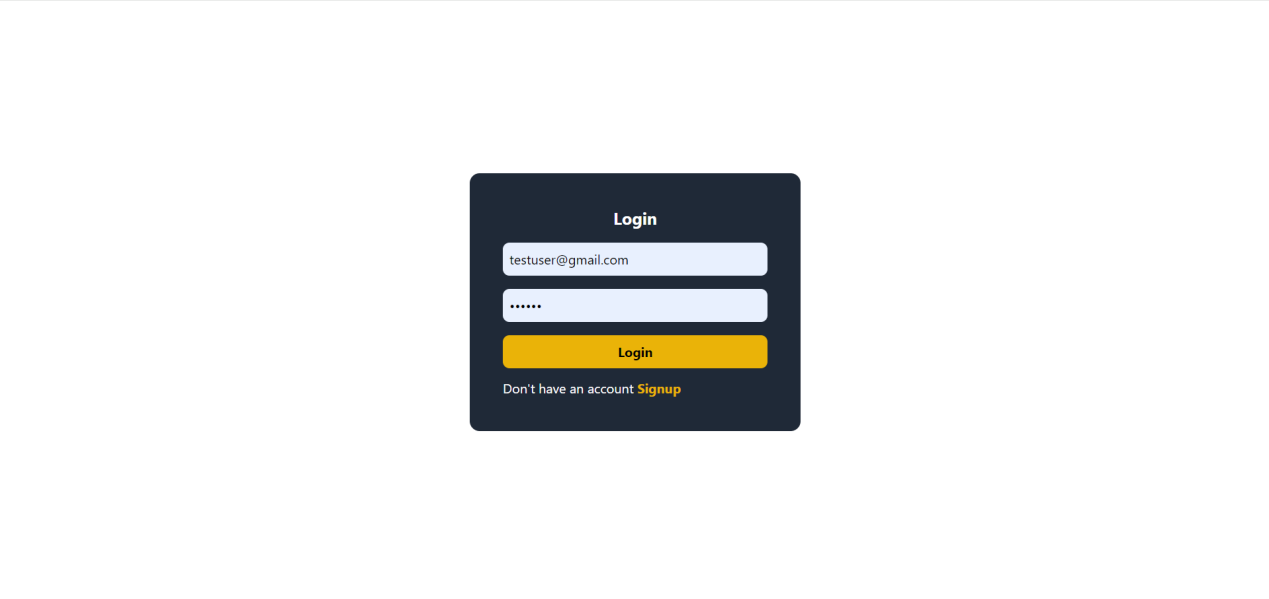
* 1. **Giao diện đồ họa trang web**



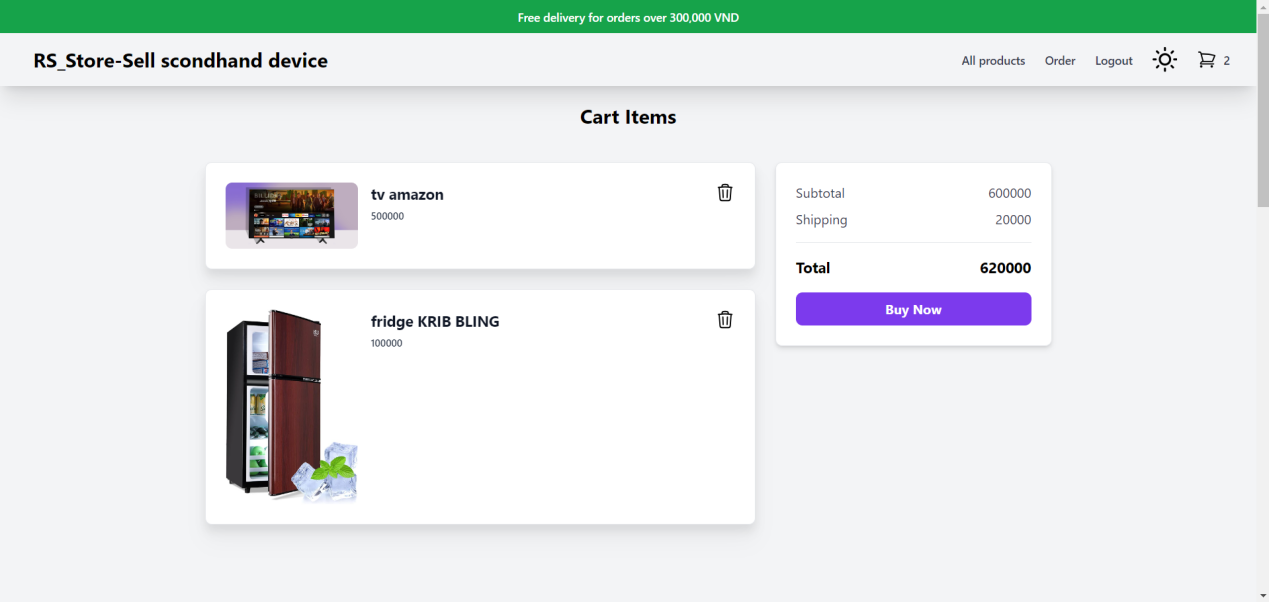


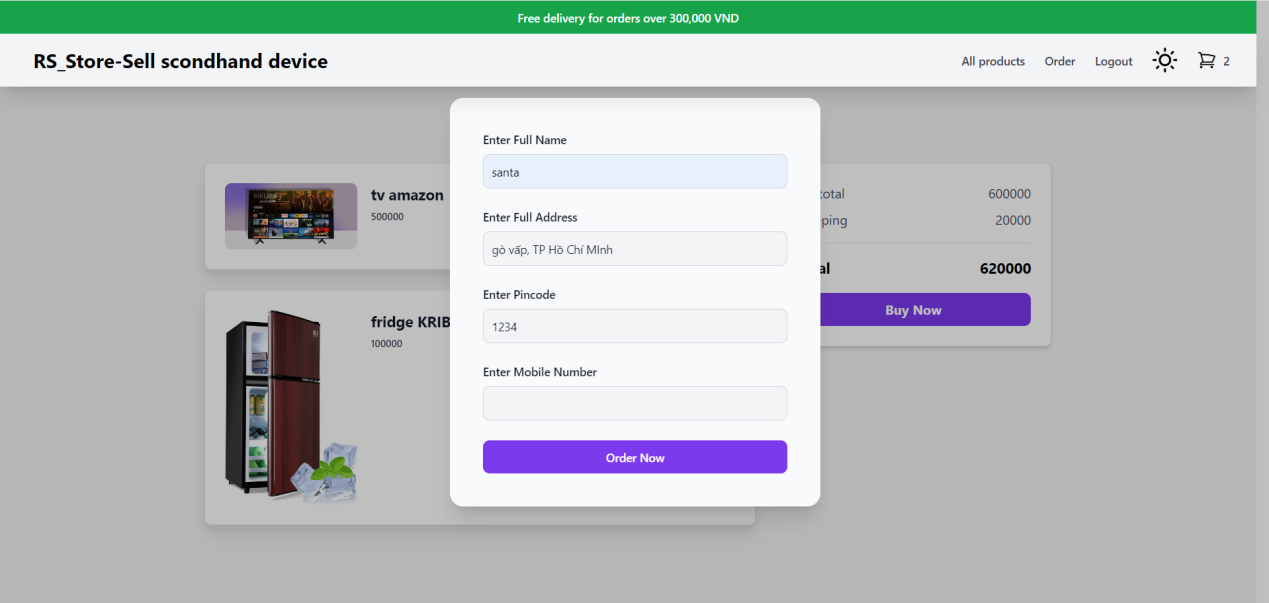












1. **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

Bài báo cáo này chúng tôi đã đưa ra giải pháp cho bài toán xây dựng trang web RS\_Store nhằm hỗ trợ cho việc mua đồ cũ đã qua sử dụng của mọi người. Giải pháp của chúng tôi là triển khai hệ thống website trên nền tảng điện toán đám mây. Đồng thời cũng kiểm chứng được tính ổn định của hệ thống khi triển khai trên nền tảng này. Trong cả hai giải pháp trên chúng tôi đều áp dụng công nghệ phù hợp nhằm tối ưu tài nguyên cũng như đảm bảo được tính hiệu quả của hệ thống. Cụ thể là phần website chúng tôi triển khai trên công nghệ như React Native, ReactJS, Vite tạo website cho người dùng, quản trị viên. Phần triển khai trên điện toán đám mây chúng tôi dùng Authentication , Cloud Firestore. Tuy nhiên, công cụ vẫn còn một số hạn chế là quá trình chúng tôi thử nghiệm chỉ với số lượng truy xuất vừa và nhỏ. Ngoài ra, các chức năng trên hệ thống web dành cho nhà quản trị vẫn còn hạn chế, chưa đáp ứng hết các chức năng hỗ trợ. Trong tương tai, chúng tôi sẽ dành tài nguyên và nhân lực để giải quyết các hạn chế trên.

**TÀI LIỆU THAM KHẢO**

**<https://sunteco.vn/wp-content/uploads/2023/06/Dac-diem-va-cach-thiet-ke-theo-Microservices-Architecture.png>**

**<https://vsi-international.com/microservice-api-ung-dung-trong-thuc-te/>**

**<https://viblo.asia/p/gioi-thieu-ve-kien-truc-microservices-4P8566O35Y3>**

**<https://topdev.vn/blog/phat-trien-phan-mem-theo-kien-truc-microservice/>**

**<https://stringee.com/vi/blog/post/redux-la-gi>**

**<https://viblo.asia/p/redux-trong-react-la-gi-1Je5EDX0lnL>**

**<https://www.marketenterprise.vn/blog/framework-tailwind-css.html>**

**<https://viblo.asia/p/tim-hieu-ve-tailwind-css-924lJp6WKPM>**

**<https://vntalking.com/vitejs-la-gi-mot-frontend-build-tool-toc-do.html>**