

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM



Nguyễn Hoàng Thái Dương	19521409
Nguyễn Âu Duy	19521423
Nguyễn Tiến Đạt	19521344

SE347 Công nghệ Web và Ứng dụng
Cửa hàng thời trang Online

GIẢNG VIÊN GIẢNG DẠY
Trần Anh Dũng

TP. HỒ CHÍ MINH, 2022

LỜI CẢM ƠN

Nhóm xin gửi lời cảm ơn chân thành đến Thầy Trần Anh Dũng đã tận tình hướng dẫn chúng em trong suốt thời gian vừa qua và các bạn học đã góp ý và giúp đỡ nhóm trong quá trình thực hiện đồ án này.

Do kiến thức và thời gian thực hiện hạn chế, đồ án của nhóm vẫn còn nhiều thiếu sót. Nhóm rất mong nhận được góp ý của Thầy và các bạn để đồ án của nhóm được hoàn thiện.

MỤC LỤC

Chương 1. GIỚI THIỆU CHUNG	1
1.1. Lý do chọn đề tài	1
1.2. Tổng quan về chức năng	2
Chương 2. CƠ SỞ LÝ THUYẾT.....	3
2.1. ReactJs.....	3
2.1.1. Giới thiệu	3
2.1.2. Các khái niệm chính trong React (ReactJs).....	3
2.1.3. Cách hoạt động	10
2.1.4. Nhược điểm	10
2.1.5. Ưu điểm và Tại sao nên dùng ReactJs	11
2.1.6. Các framework-package liên quan đến ReactJs.....	12
2.2. Spring boot	21
2.2.1. Khái niệm	21
2.2.2. Nhược điểm của Spring Boot	21
2.2.3. Ưu điểm của Spring Boot	22
2.2.4. Các thư viện được sử dụng trong Spring Boot.....	22
2.2.5. Xây dựng back-end cho đồ án bằng Spring Boot.....	26
2.3. Frameworks và tools khác.....	27
2.3.1. IntelliJ.....	27
2.3.2. Visual Studio Code.....	28
2.3.3. Postman	28

2.3.4. Microsoft SQL Server	29
2.3.5. Github	29
2.3.6. Ngôn ngữ Java	30
2.3.7. Ngôn ngữ thiết kế Web HTML-CSS-Javascript	30
2.3.8. Bootstrap.....	32
Chương 3. XÂY DỰNG HỆ THỐNG.....	33
3.1. Xây dựng kiến trúc hệ thống	33
3.1.1. Sơ đồ hệ thống	33
3.1.2. Sơ đồ use case.....	34
3.1.3. Sơ đồ lớp.....	36
3.1.4. Sơ đồ tuần tự.....	36
3.1.5. Sơ đồ hoạt động	50
3.1.6. Cơ sở dữ liệu.....	59
3.2. Thiết kế giao diện	63
3.2.1. Giao diện ứng dụng	63
3.2.2. Giao diện quản lý.....	68
Chương 4. KẾT LUẬN.....	73
4.1. Kết quả đạt được	73
4.2. Ưu điểm	73
4.3. Nhược điểm	73

DANH MỤC HÌNH

Hình 2-1 Mô hình DOM trong ứng dụng ReactJs	4
Hình 2-2 Ví dụ về JSX	5
Hình 2-3 Component	6
Hình 2-4 Ví dụ props	7
Hình 2-5 Component Lifecycle	8
Hình 2-6 Dữ liệu số người dùng các Framework	12
Hình 2-7 Ví dụ useState	13
Hình 2-8 Ví dụ useEffect	13
Hình 2-9 Thiết lập một bộ chuyển trang bằng React Router	15
Hình 2-10 Ví dụ về cách dùng useNavigate	16
Hình 2-11 Ví dụ cách dùng useLocation	16
Hình 2-12 Ví dụ cách lấy params	17
Hình 2-13 Cách dùng thẻ Link	18
Hình 2-14 Mô hình Redux	19
Hình 2-15 Các thành phần Slice, AsyncThunk trong ReduxToolkit	20
Hình 2-16 Hibernate: các hàm tương ứng sẽ có các anotation query và câu lệnh query tương ứng	23
Hình 2-17 Ví dụ về việc sử dụng anotation	24
Hình 2-18 Cấu trúc Backend đồ án.	26
Hình 2-19 IntelliJ	27
Hình 2-20 Visual Studio Code	28
Hình 2-21 Postman	28
Hình 2-22 SQL Server	29
Hình 2-23 Github	29
Hình 2-24 Java	30
Hình 2-25 HTML-CSS-Javascript	31

Hình 2-26 Bootstrap	32
Hình 3-1 Sơ đồ kiến trúc hệ thống	33
Hình 3-2 Sơ đồ use cases người dùng	34
Hình 3-3 Sơ đồ use case Hệ thống server	35
Hình 3-4 Sơ đồ Database	59
Hình 3-5 Giao diện chính	63
Hình 3-6 Giao diện chính	63
Hình 3-7 Giao diện đăng nhập	64
Hình 3-8 Giao diện đăng ký	64
Hình 3-9 Giao diện sản phẩm	65
Hình 3-10 Giao diện phụ kiện	65
Hình 3-11 Giao diện thông tin liên hệ	66
Hình 3-12 Giao diện giỏ hàng	66
Hình 3-13 Giao diện mua hàng	67
Hình 3-14 Giao diện đặt hàng	67
Hình 3-15 Giao diện Danh sách sản phẩm	68
Hình 3-16 Giao diện dashboard doanh thu	68
Hình 3-17 Giao diện chỉnh sửa/thêm mới sản phẩm	69
Hình 3-18 Giao diện Cập nhập và Quản lý phân loại Sản Phẩm	69
Hình 3-19 Giao diện đơn hàng	70
Hình 3-20 Giao diện Chỉnh sửa Trạng Thái và Chi tiền đơn hàng	70
Hình 3-21 Giao diện Danh Sách khách hàng	71
Hình 3-22 Giao diện danh sách Nhân viên và Quản lý Nhân Viên	71
Hình 3-23 Giao diện tạo mới nhân viên	72

DANH MỤC BẢNG

Bảng 3-1 Users	59
Bảng 3-2 Roles	60
Bảng 3-3 Cart	60
Bảng 3-4 CartInfo	61
Bảng 3-5 Product	61
Bảng 3-6 Category	62

TÓM TẮT ĐỒ ÁN

Việc đề tạo ra một trang Web bán hàng bắt mắt, tiện lợi cho người mua là một trong những vấn đề đang được quan tâm hàng đầu. Có rất nhiều cách để tạo ra một trang Web như thông qua các phần mềm thứ ba để tạo với những người thích thiết kế giao diện nhưng không có hiểu biết sâu rộng về lập trình hay lập trình với các ngôn ngữ, framework hỗ trợ phát triển ứng dụng Web : .NET, HTML-CSS, Javascript, Bootsrap, Vue.js, Angular... Trong đó nhóm đã chọn **ReactJs** cho phần thiết kế giao diện người dùng (front-end) và framework **Java SpringBoot** cho phần xử lý tác vụ logic, kết nối Database, để xây dựng một ứng dụng Web bán quần áo online.

Chương 1. GIỚI THIỆU CHUNG

1.1. Lý do chọn đề tài

Với sự phát triển nhảy vọt của công nghệ thông tin hiện nay, Internet ngày càng giữ vai trò quan trọng trong các lĩnh vực khoa học kỹ thuật và đời sống. Internet là tập hợp kết nối giữa các máy tính, là một mạng máy tính toàn cầu mà bất kì ai cũng có thể truy cập bằng laptop, PC, các thiết bị di động, thiết bị gia dụng thông minh,... Vì thế cùng với sự lớn mạnh của Internet thì hàng loạt các sản phẩm mang tính thương mại ra đời với quy mô từ nhỏ đến lớn ở các ngành nghề, lĩnh vực liên quan và phổ biến, thông dụng ở mọi lĩnh vực khác. Internet đã tạo ra một cuộc cách mạng trao đổi thông tin trong mọi lĩnh vực văn hóa, y tế, xã hội, giáo dục,... nhờ đó góp phần thúc đẩy sự phát triển của thế giới.

Trong đó nổi bật là các sản phẩm, thiết bị thương mại điện tử mà mọi người đều có thể sử dụng dễ dàng và hữu ích. Việc mua bán bây giờ không chỉ thông qua xem hàng – mua hàng trực tiếp mà còn có thể giao dịch qua các trang Web thương mại điện tử một cách tiện lợi nhưng vẫn đảm bảo được sự uy tín, chất lượng từ sản phẩm mà các doanh nghiệp, cửa hàng, siêu thị, cá nhân... cung cấp cho khách hàng cũng như sự hỗ trợ và chính sách bán hàng cần thiết.

Vì thế nhóm đã quyết định chọn đề tài để tạo nên một trang Web bán hàng để có thể đáp ứng nhu cầu người dùng cũng như người bán với các chức năng cần thiết nhất của một cửa hàng.

Thêm vào đó các công nghệ như AI, Blockchain, IoT cũng đang ngày càng phát triển cho hầu hết các lĩnh vực đời sống và tất nhiên việc bán hàng cũng là một thị trường tiềm năng. Bởi với lượng người dùng càng tăng và lượng dữ liệu có được sẽ là một tài nguyên cho việc nghiên cứu, áp dụng các công nghệ trên.

1.2. Tổng quan về chức năng

Chương 2. CƠ SỞ LÝ THUYẾT

2.1. ReactJs

2.1.1. Giới thiệu

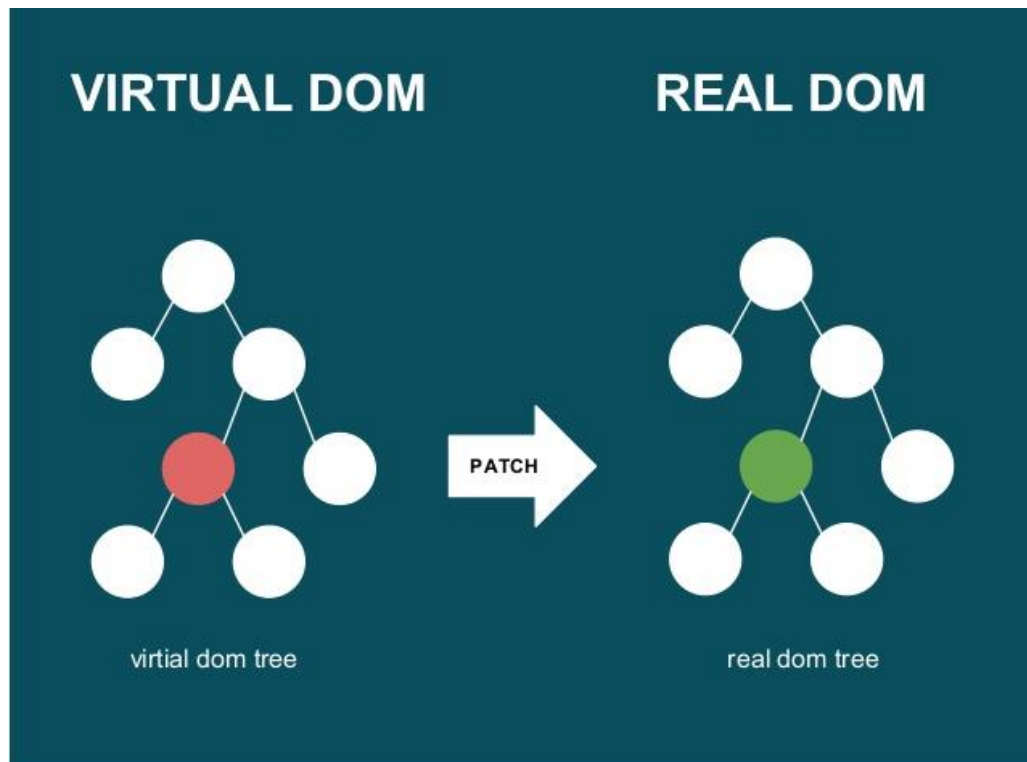
React (hay **ReactJs**, **React.js**) là một thư viện Javascript mã nguồn mở để xây dựng các thành phần giao diện có thể tái sử dụng. Nó được tạo ra bởi **Jordan Walke**, một kỹ sư phần mềm tại **Facebook**. Người bị ảnh hưởng bởi **XHP** (Một nền tảng thành phần HTML cho PHP). ReactJs lần đầu tiên được triển khai cho ứng dụng **Newsfeed** của **Facebook** năm 2011, sau đó được triển khai cho **Instagram.com** năm 2012. Nó được mở mã nguồn (open-sourced) tại **JSConf US** tháng 5 2013

ReactJs đang nổi lên trong những năm gần đây với xu hướng Single Page Application (**SPA**). Trong khi những framework khác cố gắng hướng đến một mô hình MVC hoàn thiện thì React nổi bật với sự đơn giản và dễ dàng phối hợp với những thư viện Javascript khác. Nếu như AngularJS là một Framework cho phép nhúng code Javascript trong code HTML thông qua các attribute như ng-model, ng-repeat...thì với React là một library cho phép nhúng code HTML trong code Javascript nhờ vào JSX, bạn có thể dễ dàng lồng các đoạn HTML vào trong JS. Tích hợp giữa Javascript và HTML vào trong JSX làm cho các component dễ hiểu hơn.

2.1.2. Các khái niệm chính trong React (ReactJs)

2.1.2.1. Virtual DOM

Virtual DOM chỉ là một đại diện ảo của DOM, Virtual DOM ra đời đã tăng hiệu suất các ứng dụng ReactJS một cách đáng kể.



Hình 2-1 Mô hình DOM trong ứng dụng ReactJs

Việc chỉ node gốc mới có trạng thái và khi nó thay đổi sẽ tái cấu trúc lại toàn bộ, đồng nghĩa với việc DOM tree cũng sẽ phải thay đổi một phần, điều này sẽ ảnh hưởng đến tốc độ xử lý. React JS sử dụng Virtual DOM (DOM ảo) để cải thiện vấn đề này. Virtual DOM là một object Javascript, mỗi object chứa đầy đủ thông tin cần thiết để tạo ra một DOM, khi dữ liệu thay đổi nó sẽ tính toán sự thay đổi giữa object và tree thật, điều này sẽ giúp tối ưu hoá việc re-render DOM tree thật.

⇒ Cách hoạt động cũng tựa như commit branch trên Github, Virtual DOM chỉ ghi nhận những thay đổi trên các component xảy ra từ đó chỉ thực việc render lại các component đó trên DOM thật.

2.1.2.2. JSX

JSX là một dạng ngôn ngữ cho phép viết các mã HTML trong Javascript.

Đặc điểm:

- Nhanh hơn: JSX thực hiện tối ưu hóa trong khi biên dịch sang mã Javascript. Các mã này cho thời gian thực hiện nhanh hơn nhiều so với một mã tương đương viết trực tiếp bằng Javascript.
- An toàn hơn. Ngược với Javascript, JSX là kiểu statically typed, nghĩa là nó được biên dịch trước khi chạy, giống như Java, C++. Vì thế các lỗi sẽ được phát hiện ngay trong quá trình biên dịch. Ngoài ra, nó cũng cung cấp tính năng gỡ lỗi khi biên dịch rất tốt.
- Dễ dàng hơn. JSX kế thừa dựa trên Javascript, vì vậy rất dễ dàng để cho các lập trình viên Javascripts có thể sử dụng.

Như hình trên ta có thể thấy việc viết trực tiếp HTML vào function Javascript mà không có lỗi hay cần các dấu “” như xưa.

```
const CatalogNotFound = () => {  
  return (  
    <div>Không tìm thấy kết quả lọc </div>  
  )  
}
```

Hình 2-2 Ví dụ về JSX

2.1.2.3. Component

Component có cú pháp như một **function**.

React được xây dựng xung quanh các component, chứ không dùng template như các framework khác. Trong React, chúng ta xây dựng trang web sử dụng những thành phần (component) nhỏ. Chúng ta có thể tái sử dụng một component ở nhiều nơi, với các trạng thái hoặc các thuộc tính khác nhau, trong một component lại có thể chứa thành phần khác. Mỗi component trong React

có một trạng thái riêng, có thể thay đổi, và React sẽ thực hiện cập nhật component dựa trên những thay đổi của trạng thái. Mọi thứ React đều là component.

```
const Layout = () => {
  return (
    <BrowserRouter>
      <Header />
      <div className='container'>
        <React.Fragment>
          <div className="main">
            <ARoutes />
          </div>
        </React.Fragment>
      </div>
      <Footer />
      <ProductViewModal />
      <LoginModal />
      <RegisterModal />
      <AlertMessage />
    </BrowserRouter>
  )
}

export default Layout
```

Hình 2-3 Component

Ta có thể trong hình thì component cha Layout chính là nơi chứa các component con khác. Và component Layout cũng chính là giao diện chính của Web. Việc thay đổi các các component con chỉ ảnh hưởng đến các component con cần render.

Ví dụ như component LoginModal sẽ là Modal đăng nhập của trang Web và Modal này sẽ được hiện khi người dùng click đăng nhập. Điều này có nghĩa các component khác khi mà LoginModal xuất hiện hay ẩn đi đều không thay đổi vì chỉ có Login Modal là được thay đổi thuộc tính để render lại cho người dùng thấy.

2.1.2.4. Props và State

- **Props:**

Viết tắt của **properties**, nhưng trong React lại là một khái niệm. Props là một đối tượng. Nó sẽ lưu trữ những thuộc tính, giá trị được truyền vào từ component cha hay trong hàm hay bất cứ khi nào Tag của component này được gọi.

```
function Welcome(props) {  
  return <h1>Hello, {props.name}</h1>;  
}  
  
const element = <Welcome name="ReactJS" />;  
ReactDOM.render(  
  element,  
  document.getElementById('root')  
);
```

Hình 2-4 Ví dụ props

Ta có thể thấy element là một biến được gán cho giá trị là một component Welcome và được truyền trong Tag với giá trị name="ReactJS".

Và trong component Welcome thì ta có thể thấy việc truy cập names thông qua props. Ở đây props được là một đối tượng nhận vào giá trị name như sau:

```
props = {  
  name: "ReactJS"  
};
```

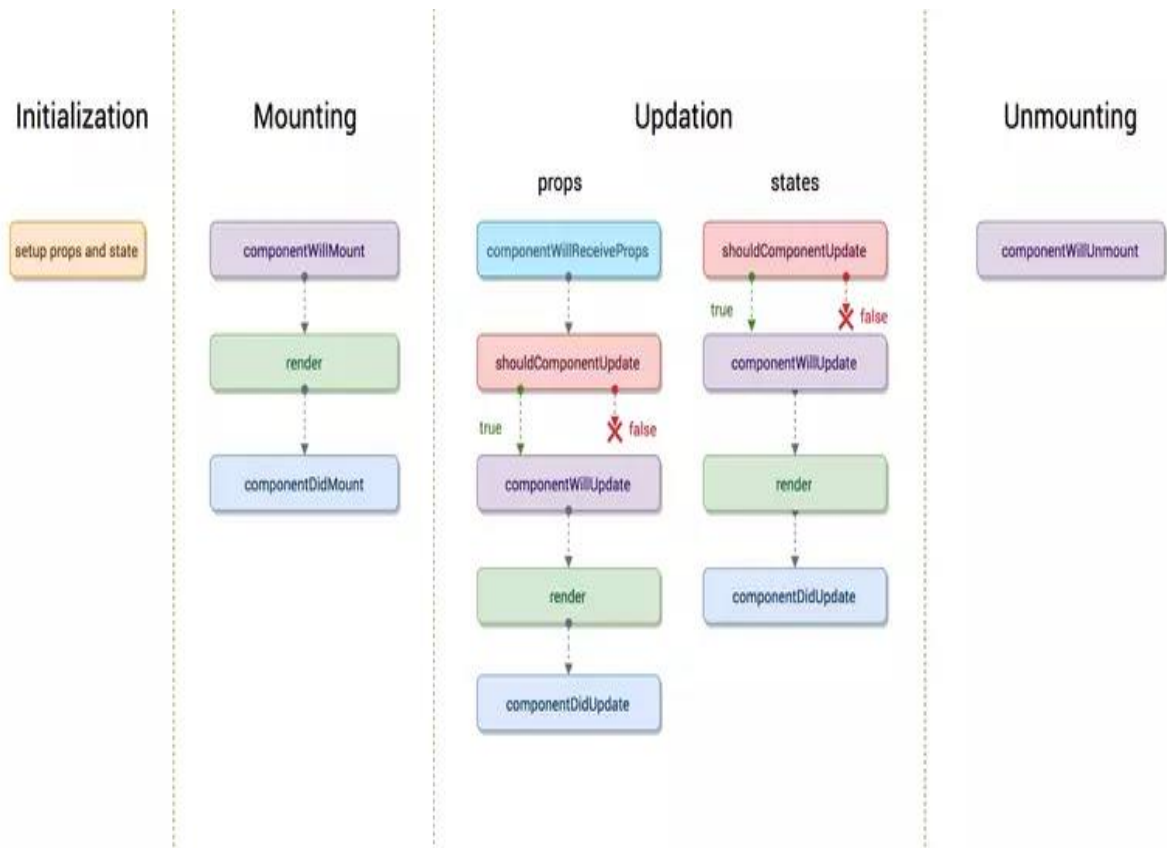
⇒ Lưu ý là chúng ta không nên thay đổi được props trong component con do ReactJS tuân theo quy tắc **pure function**: không làm thay đổi giá trị đầu vào

- **State:**

State cũng có vai trò như Props, nhưng là lưu trữ dữ liệu động của một component.

Do đó State thường được dùng để theo dõi sự thay đổi bên trong component và render lại.

2.1.2.5. Component Lifecycle



Hình 2-5 Component Lifecycle

Trong React Component, components được khởi tạo (hiển thị ra DOM), update, và kết thúc (unmount), đó được gọi là một component life cycle.

React cho phép chúng ta tham gia vào các giai đoạn của mỗi component bằng cách sử dụng các phương thức được xây dựng sẵn trong mỗi giai đoạn đó. Khi một components được khởi chạy nó sẽ phải trải qua 4 giai đoạn chính:

- **Initialization:**

Đây là giai đoạn mà component sẽ bắt đầu khởi tạo state và props.

- **Mounting:**

Giai đoạn này thực hiện sau khi quá trình initialization (khởi tạo) được hoàn thành. Nó sẽ chuyển virtual DOM thành DOM và hiển thị lên người dùng. Component sẽ được render lần đầu tiên.

- **componentWillMount ()**

Được khởi chạy khi một component chuẩn bị được mount (tức là trước khi thực hiện render)

- **componentDidMount()**

Được gọi khi component đã được mount (render thành công).

- **Updating:**

Khi state hoặc props của component thay đổi, các hàm trong nhóm này sẽ được thực thi và quyết định xem có cần phải render lại UI hay không. Trường hợp UI không được re-render khi hàm shouldComponentUpdate () trả về giá trị false.

- **Unmounting:**

Đây là bước cuối cùng trong mỗi component, khi tất cả các tác vụ hoàn thành và bạn tiến hành unmount DOM.

2.1.2.6. React Hook

Trước khi react Hook ra đời thì chúng ta thường khai báo một component là class component và sử dụng các phương thức trong lifecycle một cách nhập nhằng. Trong khi đó functional component thì không thể dùng state và lifecycle dù cách trình bày dễ hiểu hơn class component.

Hooks là một bổ sung mới trong React 16.8.

Hooks là những hàm cho phép bạn “kết nối” React state và lifecycle vào các functional component.

Nhóm hầu như sử dụng React Hook cho đồ án vì nó dễ dùng và dễ hiểu hơn class component xong cũng không phủ nhận class component vẫn sẽ là đối tượng được nhóm nghiên cứu tổng tương lai vì trong một số trường hợp thì class component vẫn có thể mang lại hiệu quả.

2.1.3. Cách hoạt động

ReactJs là một thư viện hỗ trợ cho việc phát triển các ứng dụng Web SPA – single page application. Điều này có nghĩa là ứng dụng chỉ có một trang (thông thường là index.html) cho cả ứng dụng và được chia thành các thành phần – component. Mỗi component sẽ chứa các props- được truyền từ component cha và state- lưu trữ các giá trị có trong component để render lại khi có thay đổi logic hay sự kiện từ phía người dùng.

Chính vì vậy ứng dụng sẽ không cần render lại mà chỉ component thay đổi render lại, từ đó giúp tăng đáng kể hiệu suất người dùng và giúp ích trong việc phát triển do có thể tái sử dụng ở nhiều nơi trong ứng dụng.

2.1.4. Nhược điểm

Reactjs chỉ phục vụ cho tầng View. React chỉ là View Library nó không phải là một MVC framework như những framework khác. Đây chỉ là thư viện của Facebook giúp render ra phần view. Vì thế React sẽ không có phần Model và Controller, mà phải kết hợp với các thư viện khác. React cũng sẽ không có 2-way binding hay là Ajax.

Tích hợp Reactjs vào các framework MVC truyền thống yêu cầu cần phải cấu hình lại.

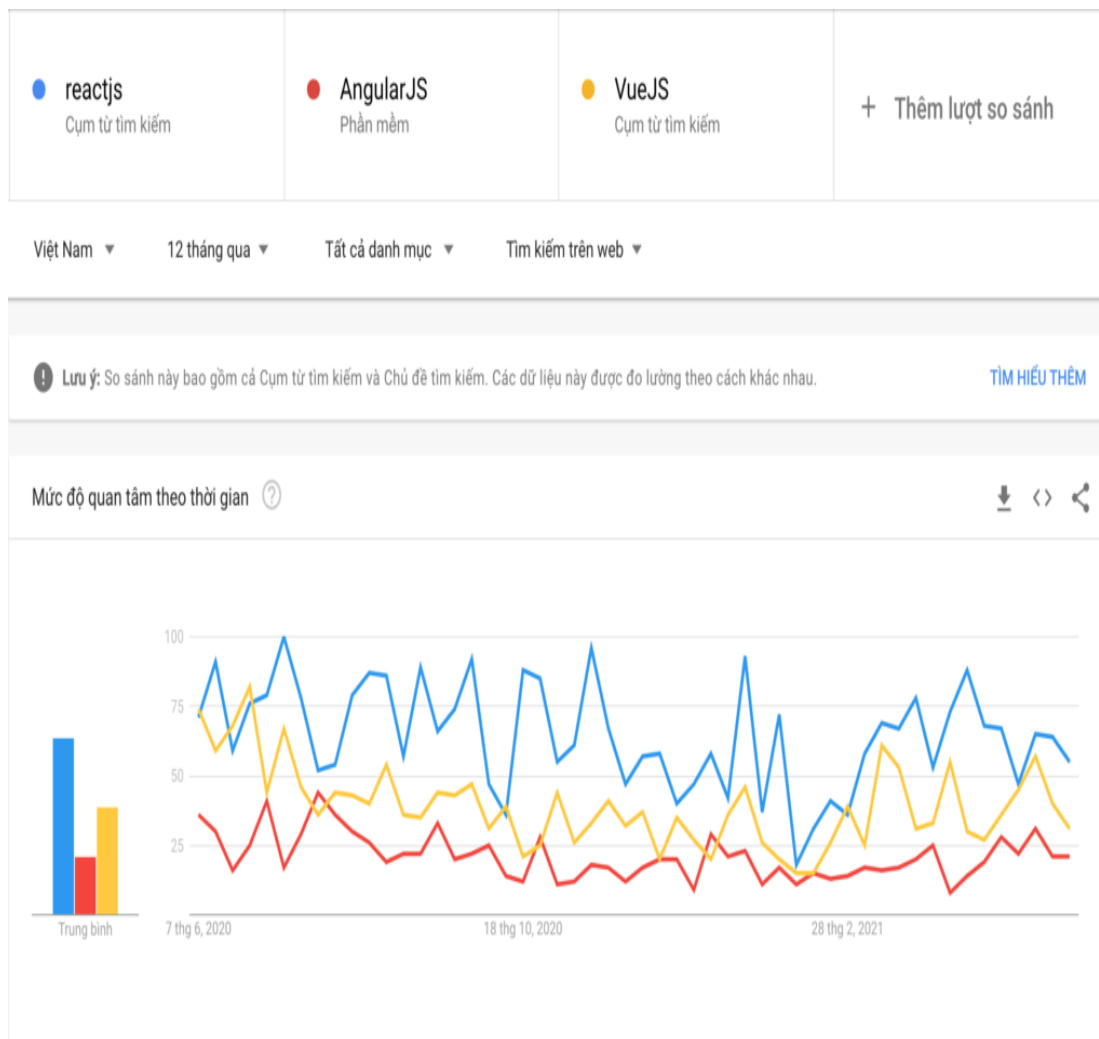
React khá nặng nếu so với các framework khác React có kích thước tương đương với Angular (Khoảng 35kb so với 39kb của Angular). Trong khi đó Angular là một framework hoàn chỉnh.

Khó tiếp cận cho người mới học Web.

2.1.5. Ưu điểm và Tại sao nên dùng ReactJs

Ngoài việc hỗ trợ xây dựng giao diện nhanh, hạn chế lỗi trong quá trình code, cải thiện hiệu suất website thì những tính năng đặc biệt dưới đây có thể là lý do khiến bạn bắt đầu tìm hiểu nó từ bây giờ:

- Phù hợp với đa dạng thể loại website: ReactJS khiến cho việc khởi tạo website dễ dàng hơn bởi vì bạn không cần phải code nhiều như khi tạo trang web thuần chỉ dùng JavaScript, HTML.
- Tái sử dụng các Component.
- Có thể sử dụng cho cả Mobile application: React Native – một framework khác được phát triển cũng chính Facebook tương tự như ReactJs.
- Debug dễ dàng: Facebook đã phát hành một Chrome extension dùng trong việc debug trong quá trình phát triển ứng dụng.
- Cộng đồng người sử dụng lớn, các nguồn như Github đều có thể thấy các source liên quan về ReactJs. Tài liệu tham khảo phong phú và tích hợp với các Framework như Bootstrap cho ra đời React-Bootstrap.
- Hot khi tính tới 2021: Nếu bạn nhìn vào số liệu thống kê từ Google Trend ở Việt Nam ở hình bên dưới, dạo lướt qua các trang tuyển dụng hàng đầu ở Việt Nam như Topdev, Itviec,...bạn sẽ thấy số lượng tuyển dụng cho vị trí React Developer là cực kỳ lớn cùng với mức lương vô cùng hấp dẫn và độ phổ biến hiện tại của ReactJS trên thị trường Việt Nam là như thế nào.



Hình 2-6 Dữ liệu số người dùng các Framework

2.1.6. Các framework-package liên quan đến ReactJs

2.1.6.1. React Hook

Khiến các component trở nên gọn nhẹ hơn.

Giảm đáng kể số lượng code, dễ tiếp cận.

Cho phép chúng ta sử dụng state ngay trong function component

Các Hooks:

a. useState

Hàm này nhận đầu vào là giá trị khởi tạo của 1 state và trả ra 1 mảng gồm có 2 phần tử, phần tử đầu tiên là state hiện tại, phần tử thứ 2 là 1 function dùng để cập nhật state.

```
const [isLoading, setLoading] = useState(false);

onClick() {
  setLoading(true)
}
```

Hình 2-7 Ví dụ useState

b. useEffect

Nó giúp chúng ta xử lý các side effects, useEffect sẽ tương đương với các hàm **componentDidMount**, **componentDidUpdate** và **componentWillUnmount** trong LifeCycle.

```
useEffect(
  () => {
    const subscription = props.source.subscribe();
    return () => {
      subscription.unsubscribe();
    };
  },
  [props.source], // giá trị được subscribe
);
```

Hình 2-8 Ví dụ useEffect

useEffect có thể không cần code cleanup như hàm return.

Có 2 loại useEffect:

c. useEffect with no dependencies

useEffect (**function** ())

useEffect này sẽ luôn thực thi khi hàm render được gọi trong quá trình mà component tồn tại, một số trường hợp sẽ dẫn đến việc loop vô tận.

- useEffect with dependencies

Ở hình trên ta có thể useEffect nhận vào một dependencies là props.source điều này có nghĩa là useEffect sẽ thực thi mỗi khi props.source thay đổi giá trị bao gồm lúc component mount (props.source được khởi tạo). Nếu như ta truyền vào là một state thì useEffect sẽ thực thi mỗi khi setState.

Nếu ta không truyền bất cứ giá trị nào hay state nào thì useEffect sẽ chỉ thực thi mỗi khi component được gọi (mount). Vd:

useEffect(function(),[]) đây là useEffect with dependencies trường hợp không truyền giá trị hay state.

useEffect(function(),[state,props,A,B]) đây là use Effect with dependencies. Trường hợp truyền nhiều giá trị hay state và sẽ căn cứ xem giá trị.state nào thay đổi mà thực thi function() đã được định nghĩa.

d. useRef

Dùng để lưu trữ giá trị của một biến qua các lần render. Sau mỗi lần render, giá trị của một số biến trong component sẽ quay lại giá trị ban đầu, sử dụng useRef để lưu trữ các giá trị này.

Điều này giúp ta có thể quản lý một số tag mà không cần phải gọi document.getelement như trong Javascript thuần.

2.1.6.2. React Router

React-Router là một thư viện định tuyến (routing) tiêu chuẩn trong React. Nó giữ cho giao diện của ứng dụng đồng bộ với URL trên trình duyệt. React-Router cho phép bạn định tuyến "luồng dữ liệu" (data flow) trong ứng dụng của bạn một cách rõ ràng. Nó tương đương với sự khẳng định, nếu bạn có URL này, nó sẽ tương đương với Route này, và giao diện tương ứng.

Phiên bản hiện tại nhóm đang dùng là react-router v6.

a. BrowerRouter, Routes và Rotue

Một bố cục cơ bản của react-router trong việc điều trang trong ReactJs.

Mỗi một project Reactjs chỉ nên có một Browser Router bao bọc toàn bộ chương trình.

Có thể có nhiều Routes để phân chi việc điều hướng trang trong mỗi Component.

Route chỉ có thể được bọc bởi Routes và Routes chỉ chứa các children là Route. Mặc định khi vào chương trình thì ta sẽ được đưa tới Route có path là “/”. Thông thường sẽ page Home/ Trang chủ

Để điều trang ta sẽ dùng thẻ <Link/> trong thư viện react-router-dom

Để điều trang ta sẽ useNavigate trong thư viện react-router-dom

Truy cập pathname bằng useLocation hay useParams để truy cập URL

```
const ARoutes = () => {  
  return (  
    <BrowserRouter>  
      <Routes>  
        <Route path="/" element={<Home />}></Route>  
        <Route path="/catalog/:slug" element={<Product />}></Route>  
        <Route path="/catalog" element={<Catalog />}></Route>  
        <Route path="/cart" element={<Cart />}></Route>  
        <Route path="/accessories" element={<Accessories />}></Route>  
        <Route path="/contact" element={<Contact />}></Route>  
        <Route path="/order" element={<Order />}></Route>  
        <Route path="/customer" element={<CustomerInfo />}></Route>  
      </Routes>  
    </BrowserRouter>  
  )  
}  
  
export default ARoutes
```

Hình 2-9 Thiết lập một bộ chuyển trang bằng React Router

b. useNavigate

```
// v6
import { useNavigate } from 'react-router-dom';

function MyButton() {
  let navigate = useNavigate();
  function handleClick() {
    navigate('/home');
  };
  return <button onClick={handleClick}>Submit</button>;
};
```

Hình 2-10 Ví dụ về cách dùng useNavigate

Khai báo biến khởi tạo bằng useNavigate trong component để có thể điều trang thông qua các pathname đã khai báo trước đó trong route.

Ở các phiên bản react-router trước, useHistory sẽ đảm nhận vai trò này.

c. useLocation

Cách khai báo tương tự như useNavigate.

```
import React from 'react';
import { useLocation } from 'react-router-dom';

const Example = props => {
  const location = useLocation();
  console.log(location);

  // ...
};
```

Hình 2-11 Ví dụ cách dùng useLocation

useLocation sẽ chứa thông tin về path mà bạn đang ở. Nếu bạn đang ở trang admin và trước đó trong route bạn khai báo path của route chứa trang admin là path="/admin" thì location.pathname = "/admin".

d. useParams

```
1  import * as React from 'react';
2  import { Routes, Route, useParams } from 'react-router-dom';
3
4  function ProfilePage() {
5    // Get the userId param from the URL.
6    let { userId } = useParams();
7    // ...
8  }
9
10 function App() {
11   return (
12     <Routes>
13       <Route path="users">
14         <Route path=":userId" element={<ProfilePage />} />
15         <Route path="me" element={...} />
16       </Route>
17     </Routes>
18   );
19 }
```

Hình 2-12 Ví dụ cách lấy params

useParam cho phép bạn lấy các param trên url website.

Các param này được truyền vào bằng cách thêm vào path trong route theo sau các dấu :,?,....

e. Link

Thẻ <Link/> tương tự như thẻ <a/> trong HTML và chức năng như useNavigate nhưng <link/> sẽ được render ra như một component, element UI trên website để người dùng tương tác như các click, ...

```

1  import * as React from "react";
2  import { Link } from "react-router-dom";
3
4  function UsersIndexPage({ users }) {
5    return (
6      <div>
7        <h1>Users</h1>
8        <ul>
9          {users.map((user) => (
10            <li key={user.id}>
11              <Link to={user.id}>{user.name}</Link>
12            </li>
13          ))}
14        </ul>
15      </div>
16    );
17  }

```

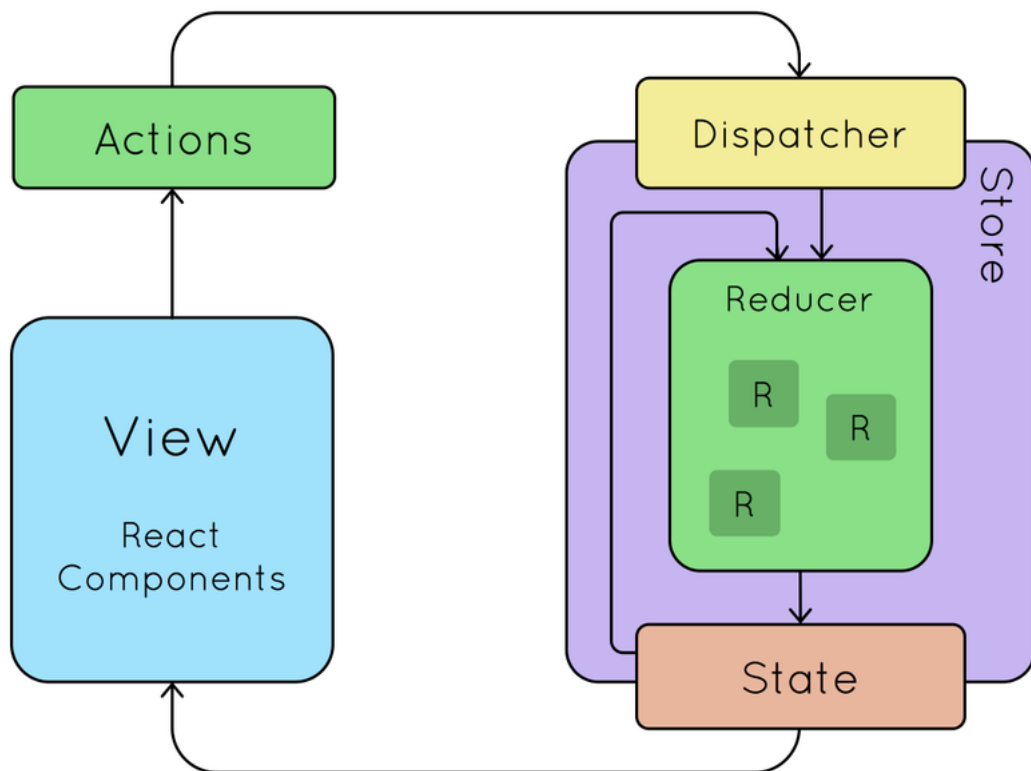
Hình 2-13 Cách dùng thẻ Link

2.1.6.3. Redux Toolkit

a. React Redux - Redux toolkit

- **Redux** là một thư viện Javascript giúp tạo ra thành một lớp quản lý trạng thái của ứng dụng. Tránh trường hợp phải truyền các props từ component cha sang component con và sâu hơn nữa. Vì có những props mà component con này không dùng tới nhưng vẫn phải nhận và truyền lại cho component con sau. Vì thế Redux ra đời để giải quyết vấn đề này.
- **Redux toolkit:** là một package được sinh ra để tiết kiệm thời gian trong việc cấu hình store, các file action hay reducer trong redux khá tách biệt và code lặp lại khá nhiều. Tuy nhiên, đến với redux toolkit, việc config store, action, reducer trở nên dễ dàng, nhanh chóng và ngắn gọn.

Các khái niệm: store, action, reducer và nguyên lý hoạt động



Hình 2-14 Mô hình Redux

- **Store** là nơi lưu trạng thái của ứng dụng và là duy nhất, có thể lưu trữ, truy xuất hoặc cập nhật giá trị state trong store thông qua các action.
- **Action**: là các event, các event này bao gồm type (để reducer biết đây là loại action gì, từ đó thực hiện các hành động cập nhật state thích hợp) và payload (chứa thông tin state mới), các payload này thông thường sẽ được cấu hình từ View hay UI tại nơi người dùng tạo ra các sự kiện như click, ...
- **Reducer**: sau khi action được gọi thông qua dispatch reducer là các pure function lấy state hiện tại, kết hợp với type, payload từ action để cập nhật giá trị state mới.

Các tính năng - tiện ích của redux được thay thế bởi redux toolkit:

```

import { createSlice,createAsyncThunk } from "@reduxjs/toolkit";

import axios from "axios";
import { apiUrl } from "../../utils/constant";

export const getAllProduct = createAsyncThunk(
  'getAllProduct',
  async (data, { rejectWithValue }) => {
    const rs = await axios.get(`${apiUrl}/product`)
    if (rs.status < 200 || rs.status >= 300) {
      return rejectWithValue(rs.data)
    }
    return rs.data
  }
)

export const productSlice = createSlice({
  name: 'productSlice',
  initialState: {
    value: []
  },
  reducers: {

  },
  extraReducers: (builder) => {
    builder.addCase(getAllProduct.fulfilled, (state, action) => {
      state.value = action.payload
    })
  }
})

export default productSlice.reducer

```

Hình 2-15 Các thành phần Slice, AsyncThunk trong ReduxToolkit

- **createSlice:** Với redux, khi định nghĩa action và reducer phải tách ra thành các file khác nhau, trong khi đó đối với slice, reducer và action được kết hợp lại trong cùng một file.
- **createAsyncThunk:** Thông thường các action, reducer không cho phép truyền vào , hay xử lý các hàm bất đồng bộ, vì

createAsyncThunk giúp cho việc này có thể diễn ra trong Redux. Và các xử lý này sẽ được gửi tới phần **extraReducers** trong slice để có thể cập nhật state.

⇒ Vì vậy nhóm đã sử dụng Redux Toolkit thay vì Redux và Redux Thunk mà vẫn đảm bảo với tính thích hợp với thiết đồ án.

2.2. Spring boot

2.2.1. Khái niệm

Spring Boot là một module của Spring Framework, cung cấp tính năng RAD (Rapid Application Development - Phát triển ứng dụng nhanh). Spring Boot ra đời nhằm rút ngắn thời gian cài đặt và cấu hình Spring MVC Project. Giúp các lập trình viên tập trung hơn vào việc phát triển nghiệp vụ thay vì tốn nhiều thời gian cho việc cấu hình dự án. Spring Boot dễ dàng trong việc tích hợp với các hệ sinh thái của Spring như: Spring JDBC, Spring ORM, Spring Security,

2.2.2. Nhược điểm của Spring Boot

Thiếu kiểm soát. Do style cố định, Spring Boot tạo ra nhiều phụ thuộc không được sử dụng dẫn đến kích thước tệp triển khai lớn.

Quá trình chuyển đổi dự án Spring cũ hoặc hiện có thành các ứng dụng Spring Boot nhiều khó khăn và tốn thời gian.

Không thích hợp cho các dự án quy mô lớn. Hoạt động liên tục với các microservices, theo nhiều nhà phát triển, Spring Boot không phù hợp để xây dựng các ứng dụng nguyên khối.

2.2.3. Ưu điểm của Spring Boot

Spring Boot hỗ trợ nhúng trực tiếp các file Server như Tomcat, Jetty hoặc Undertow (Do đó, khi sử dụng Spring Boot không cần phải deploy ra file war). Điều này hỗ trợ cho lập trình viên rất nhiều trong việc phát triển ứng dụng nhanh và đặc biệt là tiết kiệm được rất nhiều thời gian trong việc deploy ứng dụng Spring.

Khác với Spring Framework đơn thuần, Spring Boot hỗ trợ việc tự động cấu hình Spring khi cần thiết. Và đặc biệt Spring Boot không sinh ra code cấu hình và cũng không cần cấu hình bằng XML. Những việc cấu hình thường rất tốn thời gian trong việc phát triển một ứng dụng. Với tiêu chí phát triển một ứng dụng nhanh,

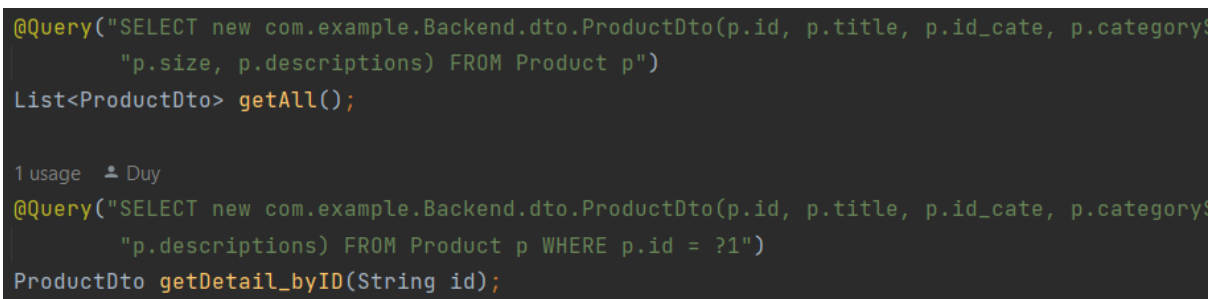
Spring Boot đã giúp cho người dùng giảm thiểu lượng thời gian đáng kể cho việc cấu hình. Spring Boot còn được biết đến là chuẩn cho Microservices (Cloud support, giảm việc setup, config các thư viện hỗ trợ). Do đó, việc triển khai Microservices trên các ứng dụng Spring Boot là rất dễ dàng.

2.2.4. Các thư viện được sử dụng trong Spring Boot

2.2.4.1. Hibernate

Khái niệm: Hibernate là một ORM (object-relationship-mapping: công nghệ cho phép chuyển đổi từ các object trong ngôn ngữ hướng đối tượng sang database quan hệ và ngược lại) framework mã nguồn mở giúp viết các ứng dụng Java có thể tham chiếu các đối tượng (entity) với bảng (table) trong hệ quản trị cơ sở dữ liệu.

Hibernate giúp thực hiện việc giao tiếp giữa tầng ứng dụng và tầng dữ liệu (kết nối, truy suất, lưu trữ...), cải thiện các vấn đề khi sử dụng JDBC như việc không cần phải lặp đi lặp lại các câu lệnh để truy xuất vào database, thay vào đó ta sẽ viết sẵn các kiểu query cần thiết và chỉ cần gọi đến khi cần truy xuất vào database.



```
@Query("SELECT new com.example.Backend.dto.ProductDto(p.id, p.title, p.id_cate, p.category,
    \"p.size, p.descriptions) FROM Product p")
List<ProductDto> getAll();

1 usage  Duy
@Query("SELECT new com.example.Backend.dto.ProductDto(p.id, p.title, p.id_cate, p.category,
    \"p.descriptions) FROM Product p WHERE p.id = ?1")
ProductDto getDetail_byID(String id);
```

Hình 2-16 Hibernate: các hàm tương ứng sẽ có các anotation query và câu lệnh query tương ứng

Sử dụng Hibernate là độc lập với hệ quản trị cơ sở dữ liệu, nghĩa là ta không cần thay đổi câu lệnh HQL khi ta chuyển từ hệ quản trị CSDL MySQL sang Oracle, hay các hệ quản trị CSDL khác... Do đó rất dễ để ta thay đổi CSDL quan hệ, đơn giản bằng cách thay đổi thông tin cấu hình hệ quản trị CSDL trong file cấu hình.

Hibernate còn giúp cho việc mapping giữa Object Java và Table tương ứng trong database tương đối dễ dàng thông qua việc sử dụng các anotation

A screenshot of a code editor showing Java code. The code is for a class named 'Product'. It has two annotations: '@Entity' and '@Table(name = "Product")'. The class has four private String attributes: 'id', 'title', 'id_cate', and 'categorySlug'. Each attribute has a '3 usages' tooltip. The code is as follows:

```
8  
4 usages  Duy +1  
9 @Entity  
10 @Table(name = "Product")  
11 public class Product {  
3 usages  
12 @Id  
13 private String id;  
3 usages  
14 private String title;  
3 usages  
15 private String id_cate;  
3 usages  
16 private String categorySlug;  
3 usages
```

Hình 2-17 Ví dụ về việc sử dụng anotation

2.2.4.2. Spring Data

Spring Data JPA là giảm thiểu việc thực hiện quá nhiều bước để có thể implement được JPA. Spring Data JPA là một phần của Spring Data và nó hỗ trợ Hibernate 5, OpenJPA 2.4 và EclipseLink 2.6.1. Trong đồ án này, chúng em sử dụng JPA như một interface và Hibernate để thực hiện interface này.

-Cách hoạt động: Trong lập trình java, khi phải truy suất dữ liệu từ database, ta cần phải khởi tạo một class DAO (Data access object) ứng với object mà ta cần truy suất để lấy, lưu hoặc xóa dữ liệu. Vì vậy class DAO tốn rất nhiều thời gian để hoàn thiện, Spring data đơn giản hóa việc này bằng cách loại bỏ hoàn toàn việc triển khai class DAO mà chỉ sử dụng nó như một interface. Một interface DAO sẽ cần extends JPA specific Repository interface,

JpaRepository. Điều này cho phép Spring Data tìm thấy interface này và tự động tạo 1 implementation cho nó.

-Bằng cách extends interface, chúng ta nhận được các method CRUD (Create, Read, Update, Delete) phù hợp nhất để truy cập dữ liệu tiêu chuẩn có sẵn trong một DAO tiêu chuẩn.

2.2.4.3. Spring Security

-Khái niệm: Spring security là 1 framework thuộc hệ thống Spring, dành riêng cho việc thiết lập bảo mật của ứng dụng bao gồm authentication và authorization.

- Authentication: quá trình xác minh user, dựa vào thông tin đăng nhập mà user cung cấp. Ví dụ khi login, bạn nhập username và password, nó giúp hệ thống nhận ra bạn là ai.
- Authorization: Quá trình xác định xem user có quyền thực hiện những chức năng nào của hệ thống (đọc/sửa/xóa data), sau khi user đã authenticated thành công.

Hiểu nôm na thì authentication là cái công thứ nhất, xem user có thuộc hệ thống hay không, authorization là cái công thứ hai, xem user được phép làm những gì trong hệ thống đó.

2.2.4.4. Swagger

Open API Specification: là một định dạng mô tả API dành cho REST APIs. Một file OpenAPI cho phép bạn mô tả toàn bộ API bao gồm cả:

- Cho phép những endpoints (/users) và cách thức hoạt động của mỗi endpoint (GET /users, POST /users)
- Các tham số đầu vào & đầu ra của từng hoạt động
- Phương thức xác thực

- Thông tin liên lạc, chứng chỉ, điều khoản sử dụng và những thông tin khác

API specifications có thể được viết bằng YAML hoặc JSON. Định dạng này dễ đọc, dễ hiểu cho cả người dùng lẫn ngôn ngữ máy tính

Swagger là một bộ công cụ mã nguồn mở để xây dựng **OpenAPI specifications** giúp bạn có thể thiết kế, xây dựng tài liệu và sử dụng REST APIs.

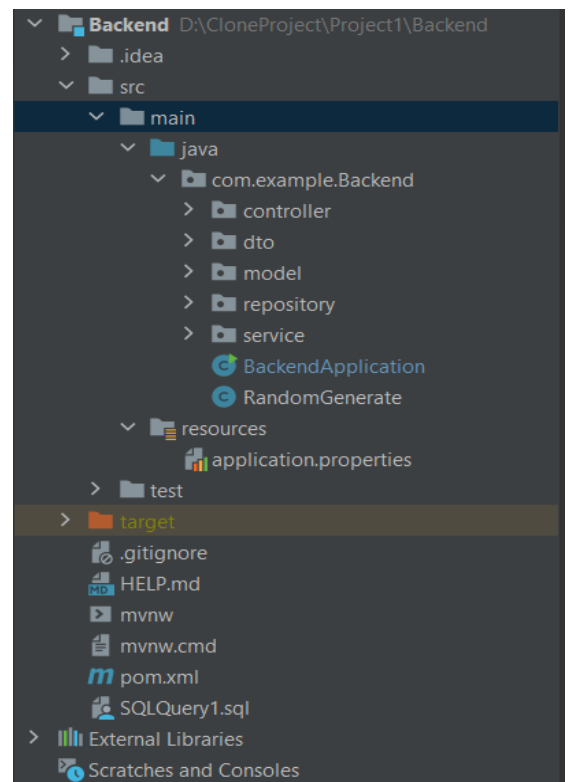
2.2.5. Xây dựng back-end cho đề án bằng Spring Boot

Ở đề án này, chúng em đã tìm hiểu và xây dựng một codebase để triển khai ứng dụng Spring Boot làm phần backend cho website của đề án, codebase này tương đối hoàn chỉnh và đáp ứng được các nhu cầu như dễ đọc, dễ triển khai, bảo trì, nâng cấp, sửa chữa và hoàn toàn có thể được tái sử dụng cho các đề án khác có liên quan.

Các file và tệp quan trọng của đề án:

Tệp model chứa các model là các class object cần quản lí. Các object này đóng vai trò là entity có khả năng mapping đến các table bên dưới database

Tệp dto chứa các class dto (data transfer object) được sử dụng làm chỗ chứa tạm thời cho dữ liệu trong quá trình đọc, viết,



Hình 2-18 Cấu trúc Backend đề án.

ghi, xóa dữ liệu từ database lên trên hoặc ngược lại.

Tập repository chứa các repository của đồ án, các repository này kế thừa từ JPA repository

Tập service chứa 2 phần:

- Các interface của service tương ứng với từng model mà service đó hướng đến.
- Các implement của các service interface.

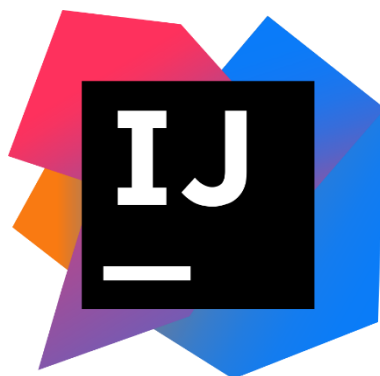
Tập controller chứa các class controller (các RestController trong Spring Boot) các controller này giúp điều hướng và mapping các request được gửi đến từ server

File pom.xml chứa các thông tin cần thiết để định nghĩa project và các plugin cũng như các dependency được khai báo và sử dụng trong project Spring Boot

File BackendApplication là file java chính để chạy project Spring Boot, file này còn có các hàm cấu hình security của project.

2.3. Frameworks và tools khác

2.3.1. IntelliJ

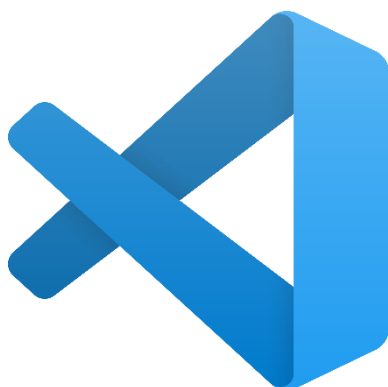


Hình 2-19 IntelliJ

Được phát hành lần đầu tiên vào năm 2001, IntelliJ IDEA là một môi trường phát triển tích hợp cho Windows, macOS và Linux. Nó chủ yếu được sử dụng để phát triển phần mềm bằng ngôn ngữ lập trình Java, nhưng cũng hỗ trợ các ngôn ngữ khác hoặc tự nhiên hoặc sử dụng một plugin. Nó cũng là cơ sở cho Android Studio của Google, môi trường

phát triển chính thức để tạo các ứng dụng Android. Nhóm sử dụng cho việc code Back-end.

2.3.2. Visual Studio Code



Hình 2-20 Visual Studio Code

Visual Studio Code là một trình soạn thảo mã nguồn được phát triển bởi Microsoft dành cho Windows, Linux và macOS. Nó hỗ trợ chức năng debug, đi kèm với Git, có chức năng nổi bật cú pháp (syntax highlighting), smart code, snippets, và cải tiến mã nguồn. Nó cũng cho phép tùy chỉnh, do đó, người dùng có thể thay đổi theme, phím tắt, và các tùy chọn khác. Nó miễn phí và là phần mềm mã nguồn mở theo giấy phép MIT. Nhóm sử dụng cho việc code Front-end.

2.3.3. Postman



Hình 2-21 Postman

Postman là một công cụ cho phép chúng ta thao tác với API, phổ biến nhất là REST.

Postman hiện là một trong những công cụ phổ biến nhất được sử dụng trong thử nghiệm các API. Với Postman, ta có thể gọi Rest API mà không cần viết dòng code nào.

Postman hỗ trợ tất cả các phương thức HTTP (GET, POST, PUT, PATCH, DELETE, ...). Bên cạnh đó, nó còn cho phép lưu lại lịch sử các lần request, rất tiện cho việc sử dụng lại khi cần.

Nhóm sử dụng Postman cho việc test API.

2.3.4. Microsoft SQL Server



Hình 2-22 SQL Server

SQL Server chính là một hệ quản trị dữ liệu quan hệ sử dụng câu lệnh SQL để trao đổi dữ liệu giữa máy cài SQL Server và máy Client. Một Relational Database Management System – RDBMS gồm có: databases, datase engine và các chương trình ứng dụng dùng để quản lý các bộ phận trong RDBMS và những dữ liệu khác.

Các thành cơ bản trong SQL Server gồm có: Reporting Services, Database Engine, Integration Services, Notification Services, Full Text Search Service, ... Tất cả kết hợp với nhau tạo thành một giải pháp hoàn chỉnh giúp cho việc phân tích và lưu trữ dữ liệu trở nên dễ dàng hơn.

Nhóm sử dụng CSDL SQL Server làm Database cho ứng dụng.

2.3.5. Github



Hình 2-23 Github

GitHub là một hệ thống quản lý dự án và phiên bản code, hoạt động giống như một mạng xã hội cho lập trình viên. Các lập trình viên có thể clone lại mã nguồn từ một repository và Github chính là một dịch vụ máy chủ repository công cộng, mỗi người có thể tạo tài khoản trên đó để tạo ra các kho chứa của riêng mình để có thể làm việc.

GitHub được coi là một mạng xã hội dành cho lập trình viên lớn nhất và dễ dùng nhất với các tính năng cốt lõi như:

Wiki, issue, thống kê, đổi tên project, project được đặt vào namespace là user.

Watch project: theo dõi hoạt động của project của người khác. Xem quá trình người ta phát triển phần mềm thế nào, project phát triển ra sao

Follow user: theo dõi hoạt động của người khác.

Github giúp ta quản lý source code dễ dàng, tracking sự thay đổi version.

2.3.6. Ngôn ngữ Java



Hình 2-24 Java

Java là ngôn ngữ lập trình hướng đối tượng được sử dụng các cú pháp C và C++, thường được dùng trong phát triển phần mềm, trang web, game hay ứng dụng

Java được phát triển bởi James Gosling và đồng nghiệp ở Sun MicroSystem năm 1991 và được phát hành 1994 đến năm 2010 được Oracle mua lại.

Các phiên bản của Java:

- Java SE: Là nền tảng cơ bản phát triển giao diện ứng dụng Winform.
- Java EE: Dựa trên SE nhưng dùng để phát triển web.
- Java ME: Phát triển dành cho mobile.

2.3.7. Ngôn ngữ thiết kế Web HTML-CSS-Javascript

Đây là bộ ba ngôn ngữ nền tảng cơ bản cho việc thiết kế Web từ lâu đời, ReactJs chỉ là thư viện được mở rộng và đóng gói các dòng lệnh của các ngôn ngữ trên. Trong đó:



Hình 2-25 HTML-CSS-Javascript

- **HTML:** (*HyperText Markup Language* – *Ngôn ngữ đánh dấu siêu văn bản*) là một ngôn ngữ đánh dấu được thiết kế để tạo nên các trang web với các mẫu thông tin được trình bày trên WWW. Cùng với CSS, Javascript, HTML tạo ra bộ ba nền tảng kỹ thuật cho các Website.

HTML là cốt lõi của mọi trang web. Bất kể sự phức tạp của một trang web hoặc số lượng công nghệ liên quan. Đó là một kỹ năng thiết yếu cho bất kỳ chuyên gia web. Đó là điểm khởi đầu cho bất cứ ai học cách tạo nội dung cho web. Và thật may mắn cho những bạn mới bắt đầu là HTML rất dễ học.

- **CSS:** (*Cascading Style Sheets*). Ngôn ngữ lập trình này chỉ ra cách các yếu tố HTML của trang web thực sự sẽ xuất hiện trên giao diện của trang. Nếu HTML cung cấp các công cụ thô cần thiết để cấu trúc nội dung trên một trang web thì CSS sẽ giúp định hình kiểu nội dung này để trang web xuất hiện trước người dùng theo một cách đẹp hơn.

- **Javascript:** JavaScript là ngôn ngữ lập trình dựa trên logic. Nó có thể được sử dụng để sửa đổi nội dung trang web. JavaScript là ngôn ngữ lập trình cho phép các nhà phát triển web thiết kế các trang web tương tác.

Nhóm sử dụng để code cho Front-end bởi vì chủ yếu ReactJs sẽ dùng các cú pháp ES6 (thuộc Javascript) và các dòng HTML trong render component. Và SCSS (là tập CSS đã được biên dịch).

2.3.8. Bootstrap



Hình 2-26 Bootstrap

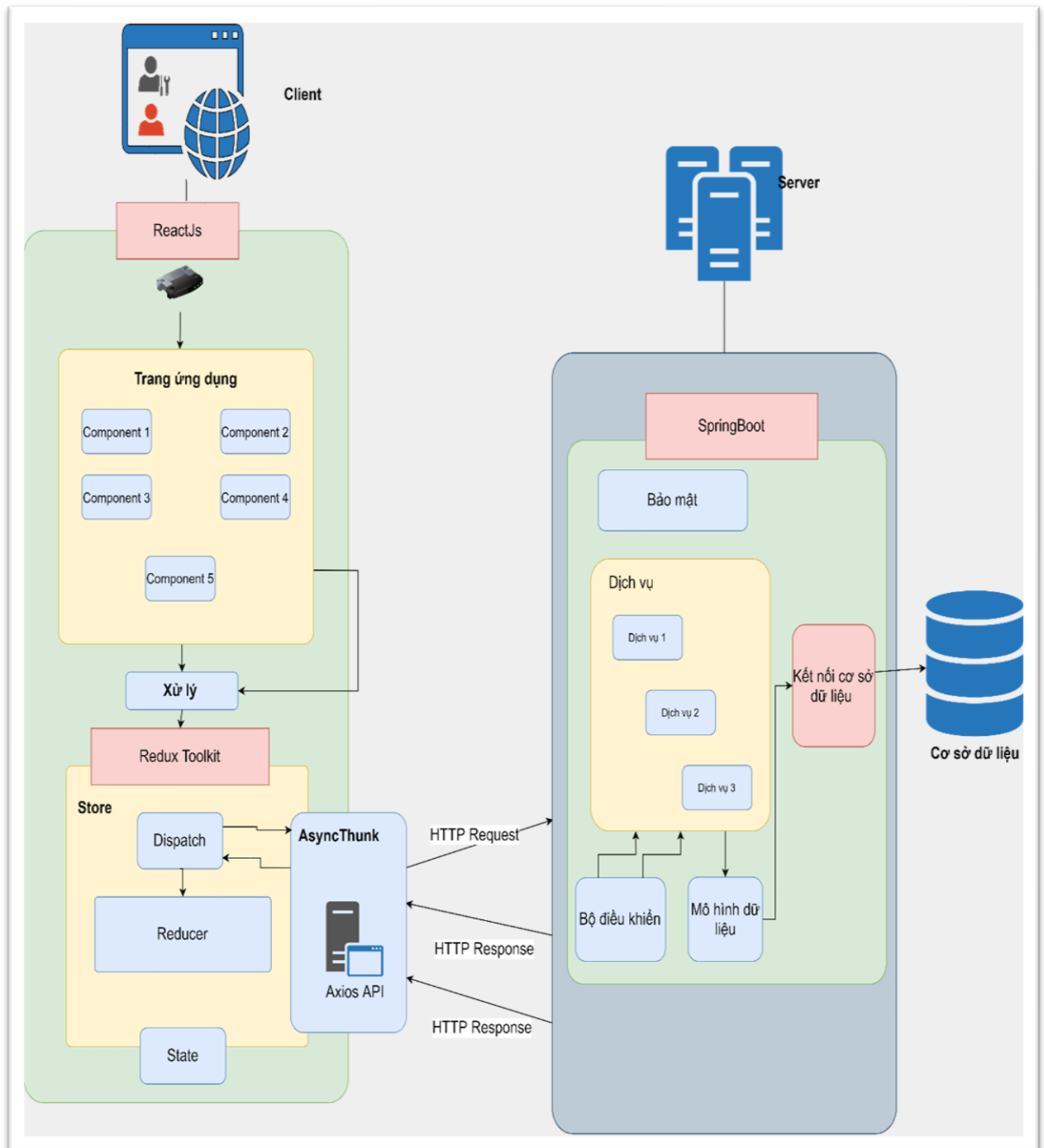
Bootstrap là một framework bao gồm các HTML, CSS và JavaScript template dùng để phát triển website chuẩn **Responsive-tương thích với mọi phiên bản nền tảng (mobile desktop hay tablet, ...)**.

Nhóm sử dụng Bootstrap 5 là một phiên bản mới của Bootstrap giúp thiết kế ứng dụng trông đẹp mắt hơn.

Chương 3. XÂY DỰNG HỆ THỐNG

3.1. Xây dựng kiến trúc hệ thống

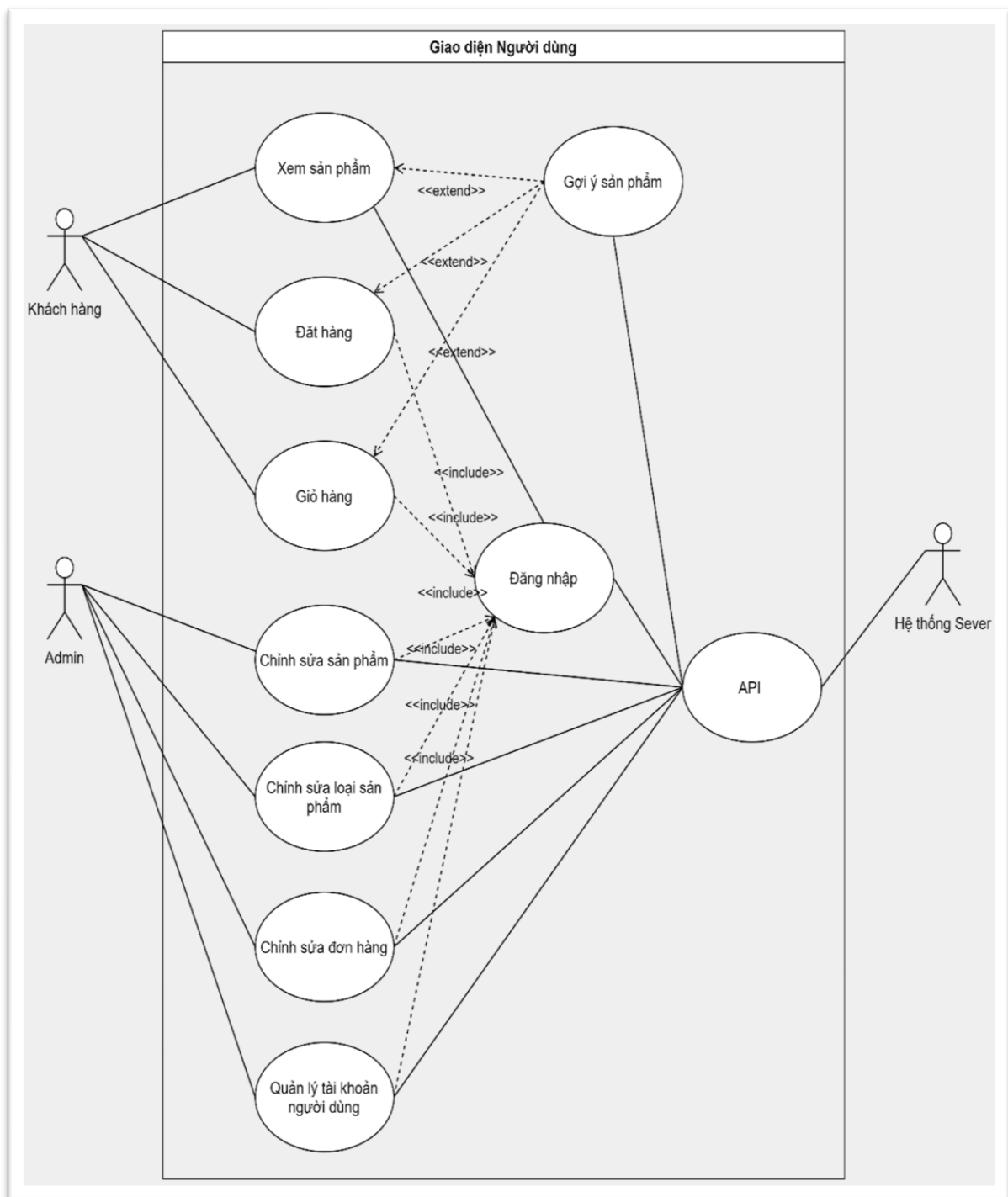
3.1.1. Sơ đồ hệ thống



Hình 3-1 Sơ đồ kiến trúc hệ thống

3.1.2. Sơ đồ use case

3.1.2.1. Người dùng/ Khách hàng

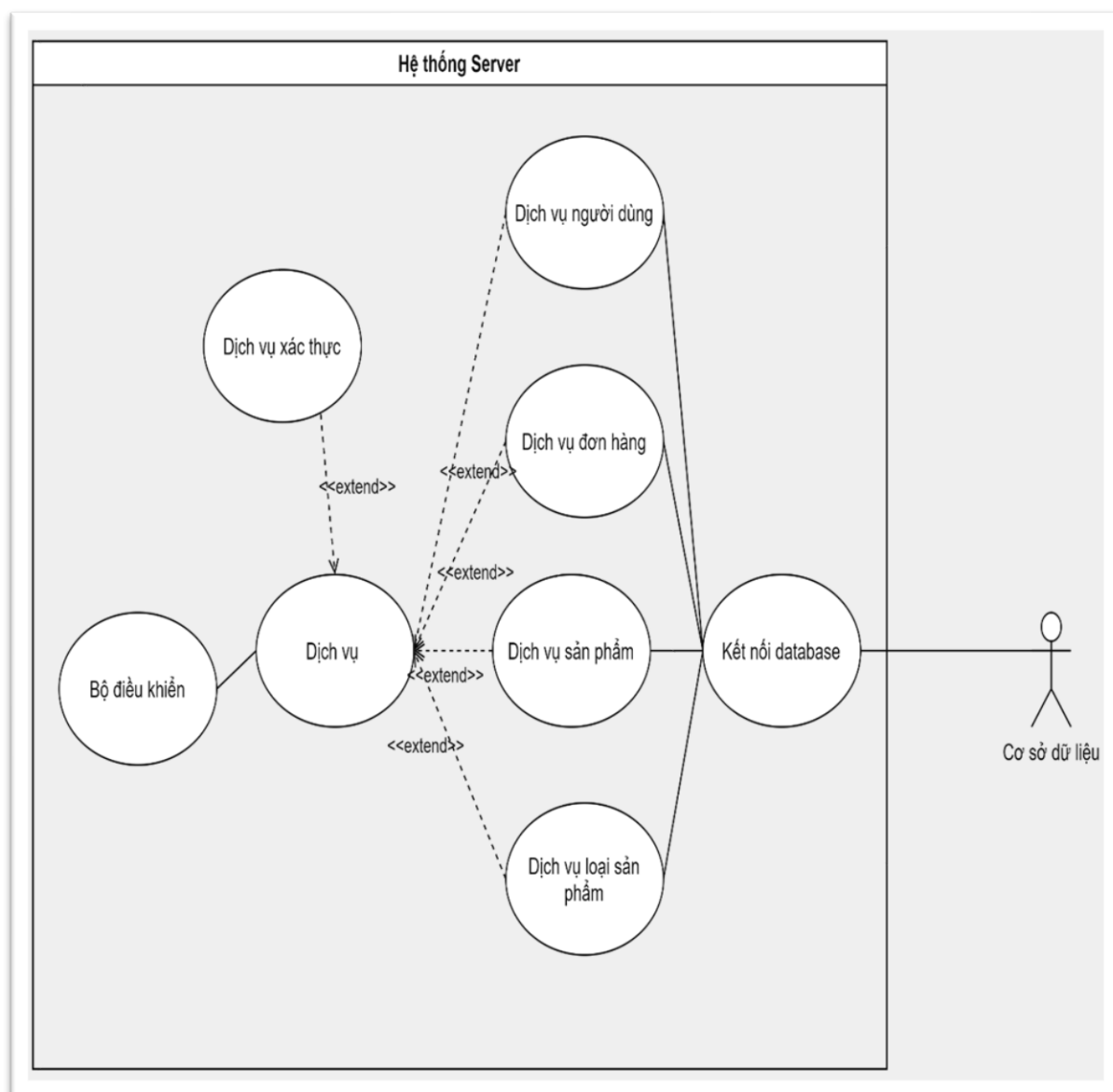


Hình 3-2 Sơ đồ use cases người dùng

a. Tác nhân

b. Use cases

3.1.2.2. Hệ thống



Hình 3-3 Sơ đồ use case Hệ thống server

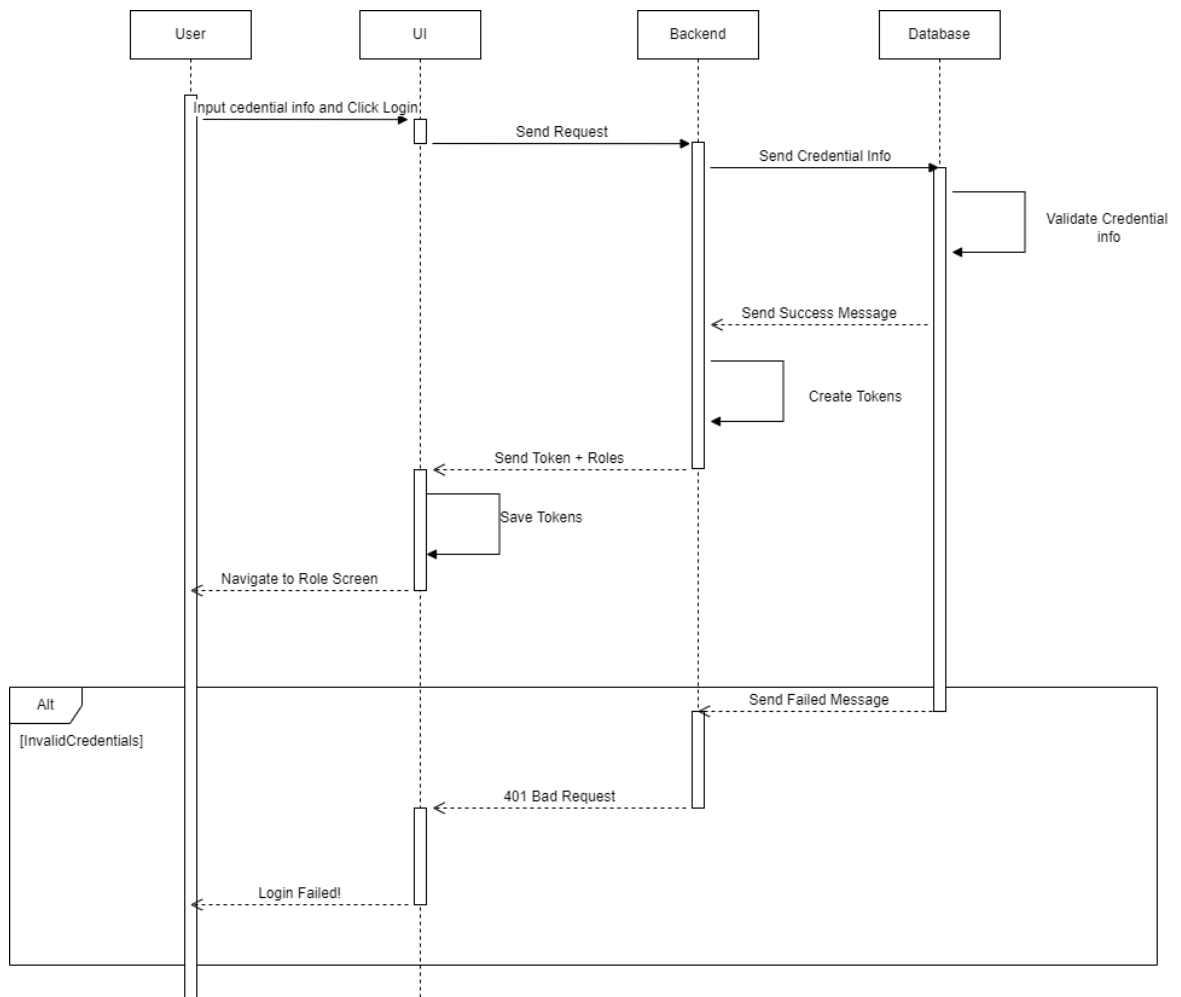
a. Tác nhân

b. Use cases

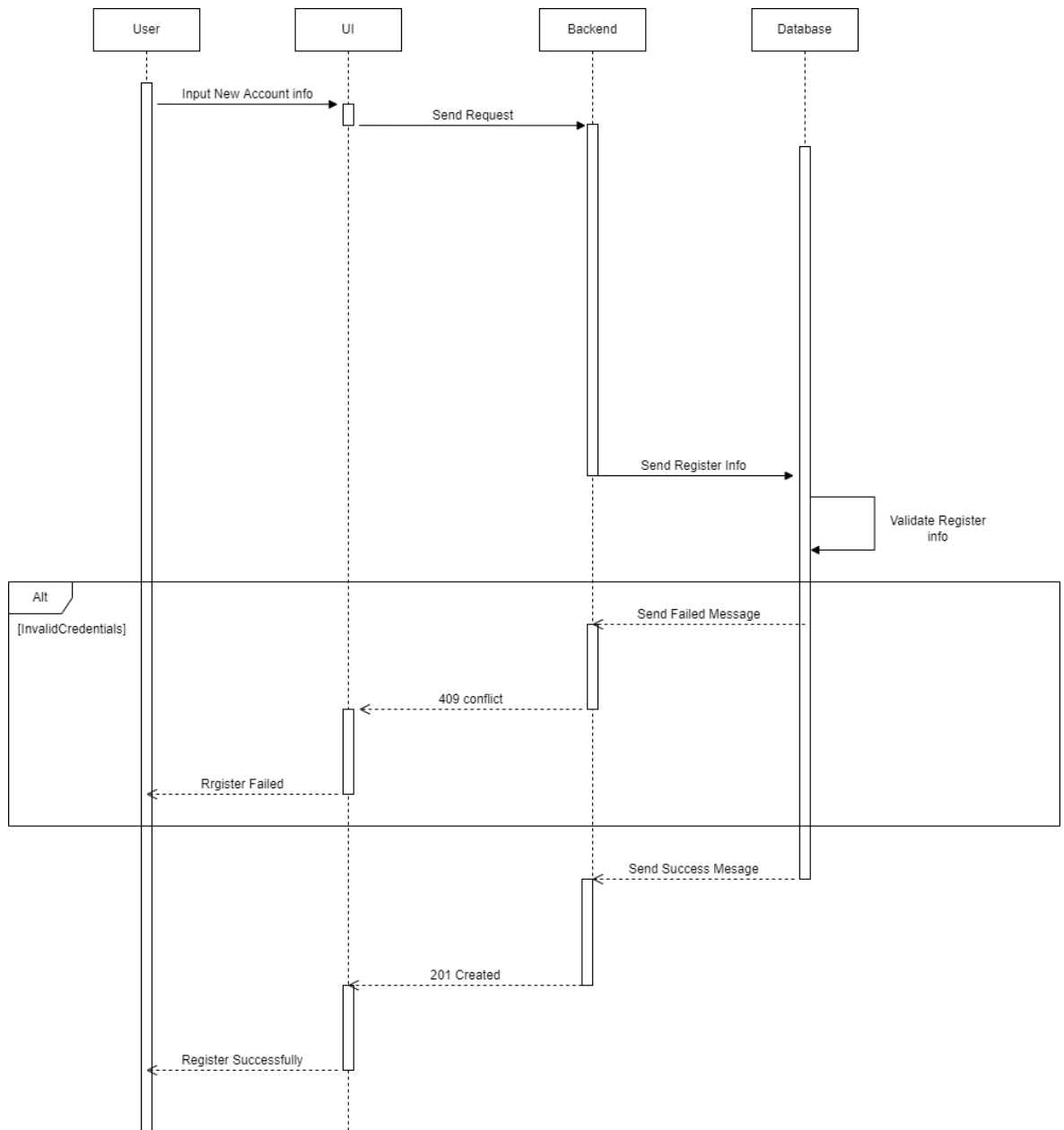
3.1.3. Sơ đồ lớp

3.1.4. Sơ đồ tuần tự

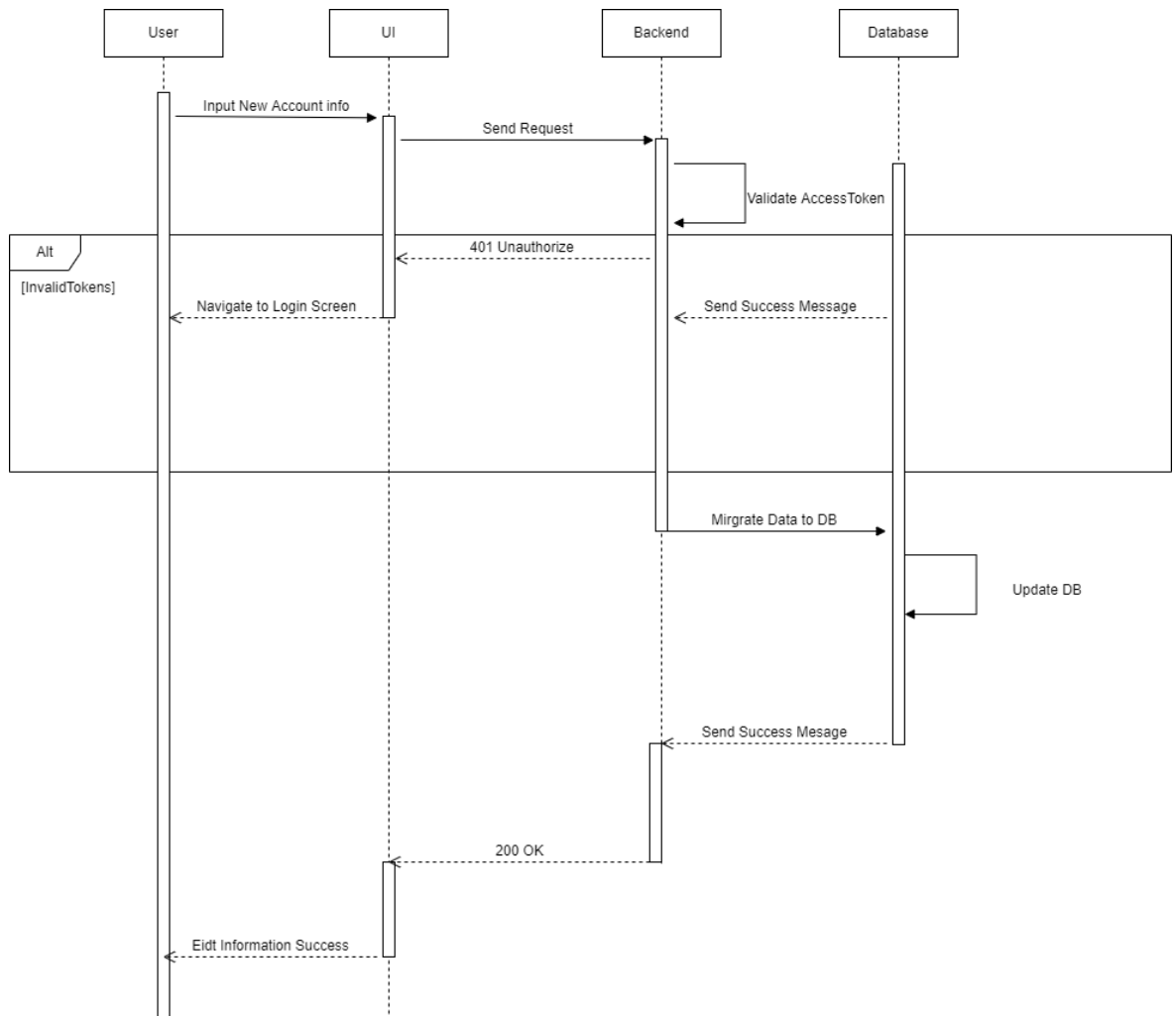
3.1.4.1 Login



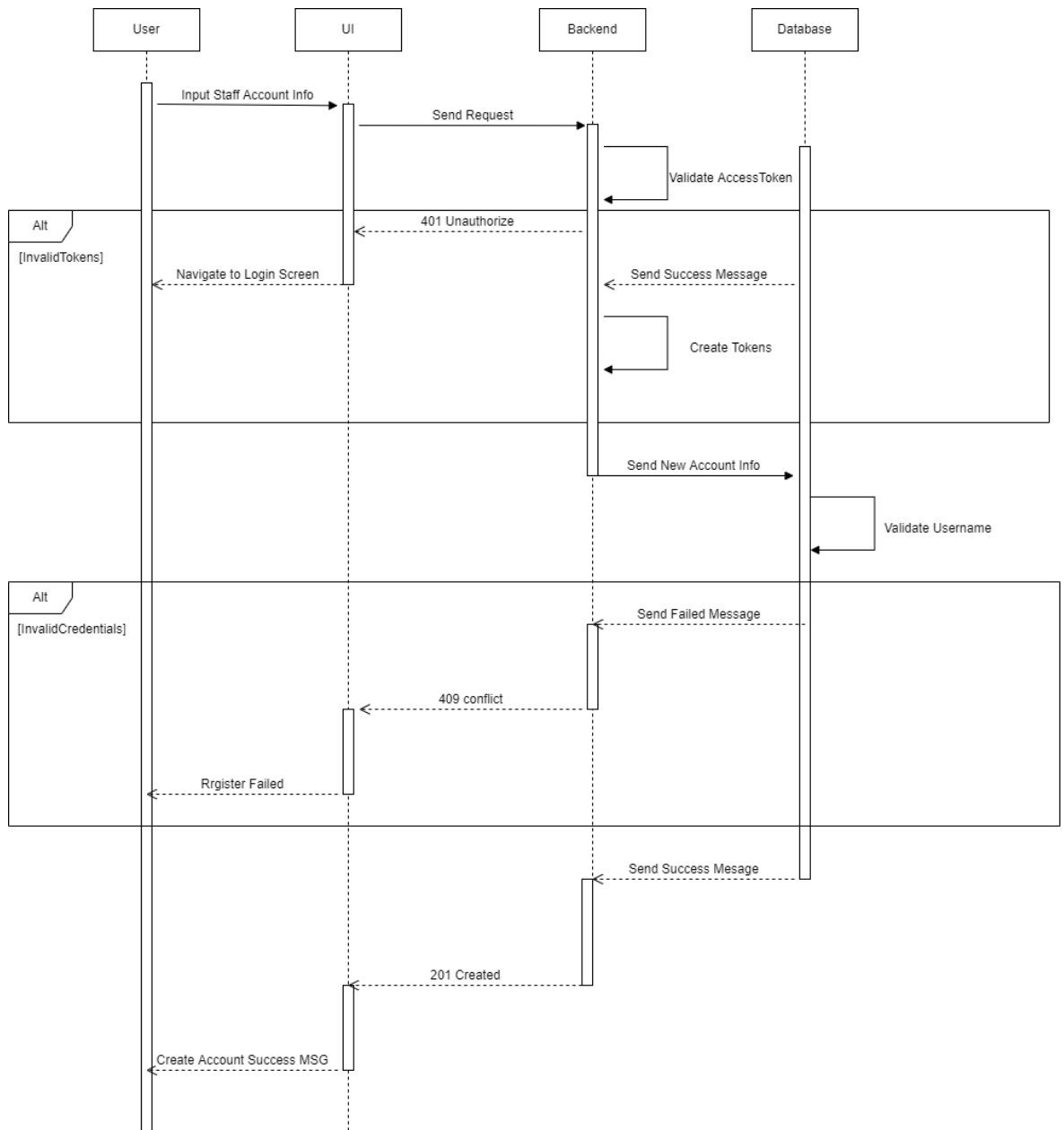
3.1.4.2 Register



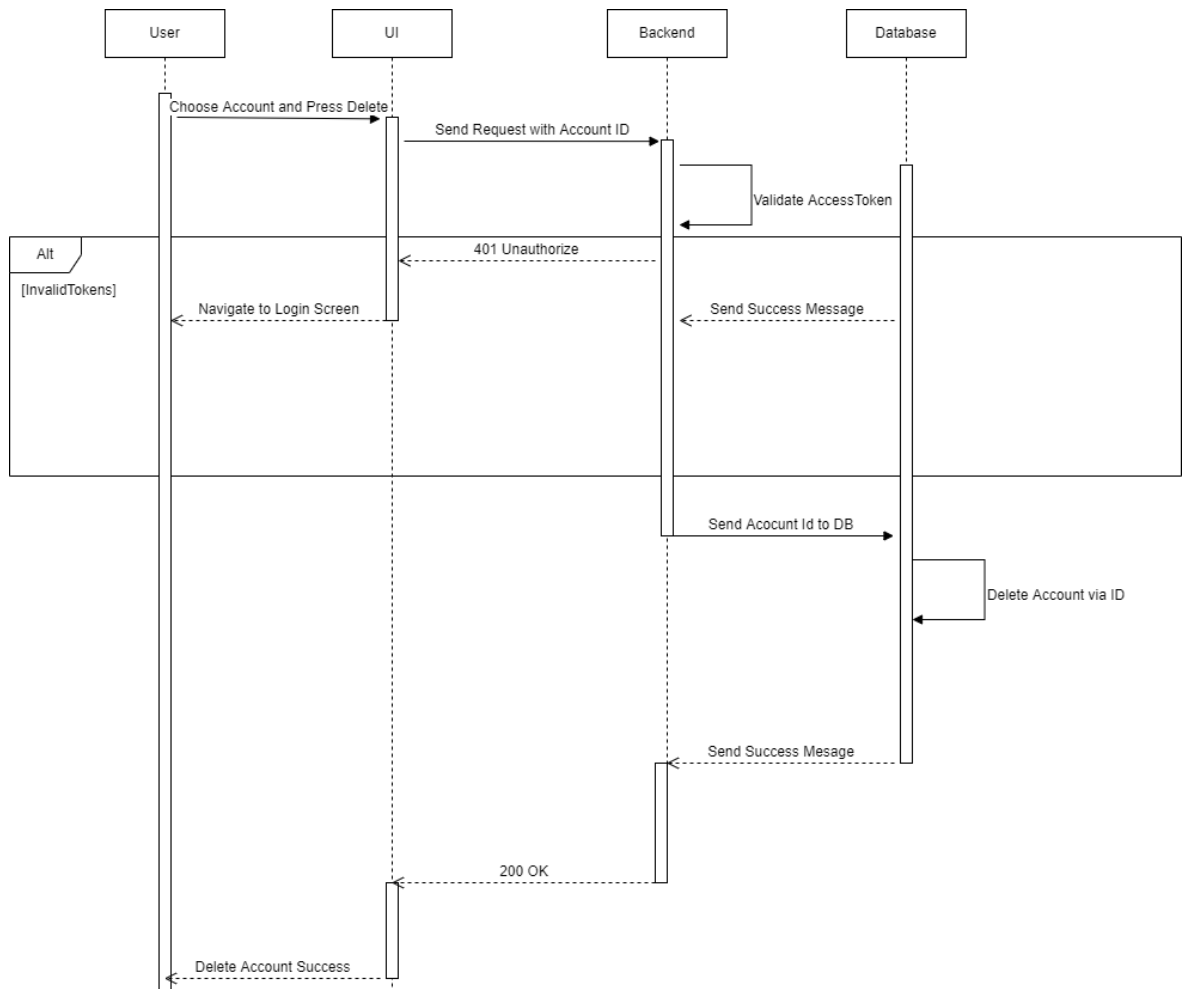
3.1.4.3 Edit Profile



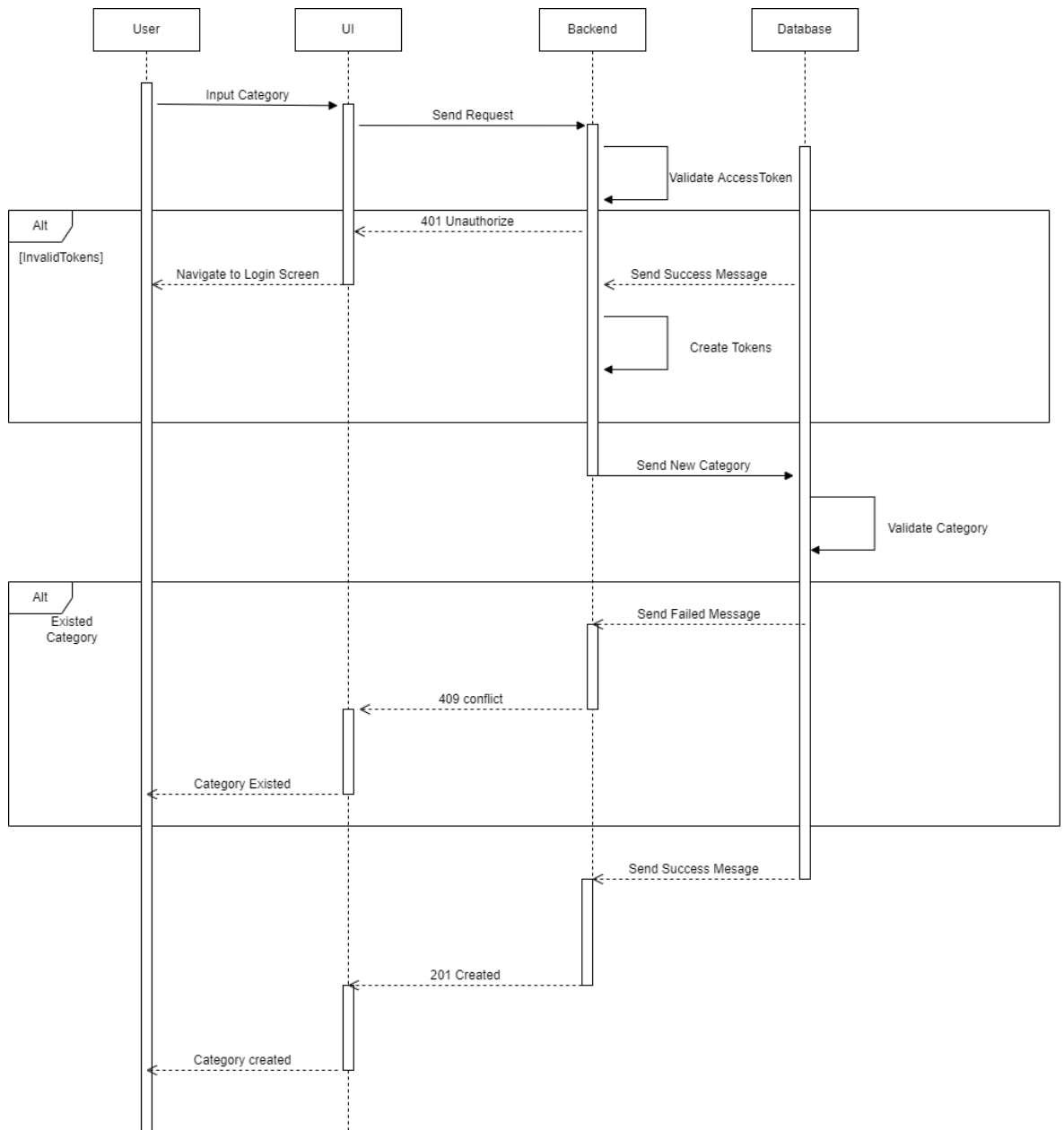
3.1.4.4 Create Staff Account



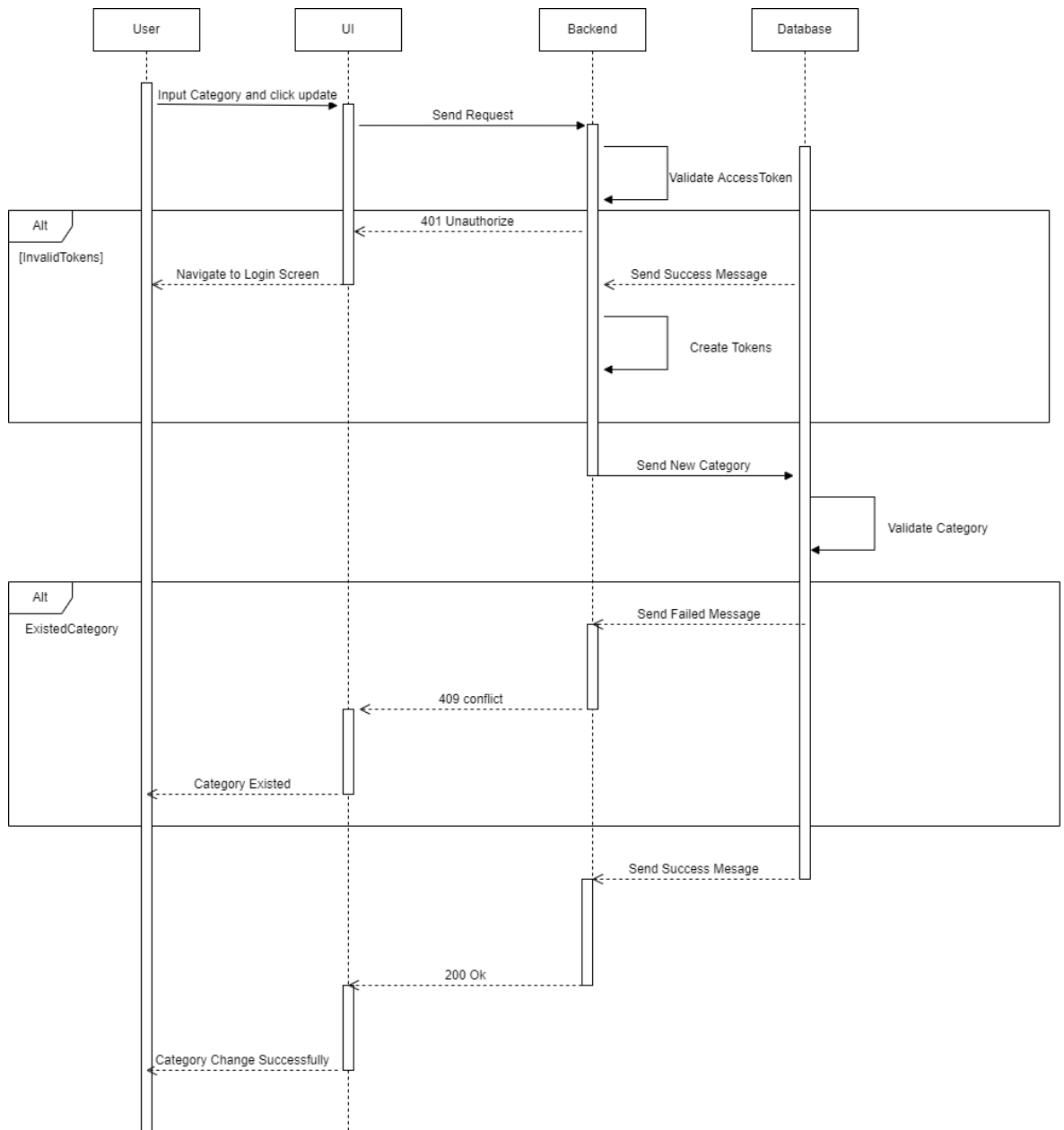
3.1.4.5 Delete Account



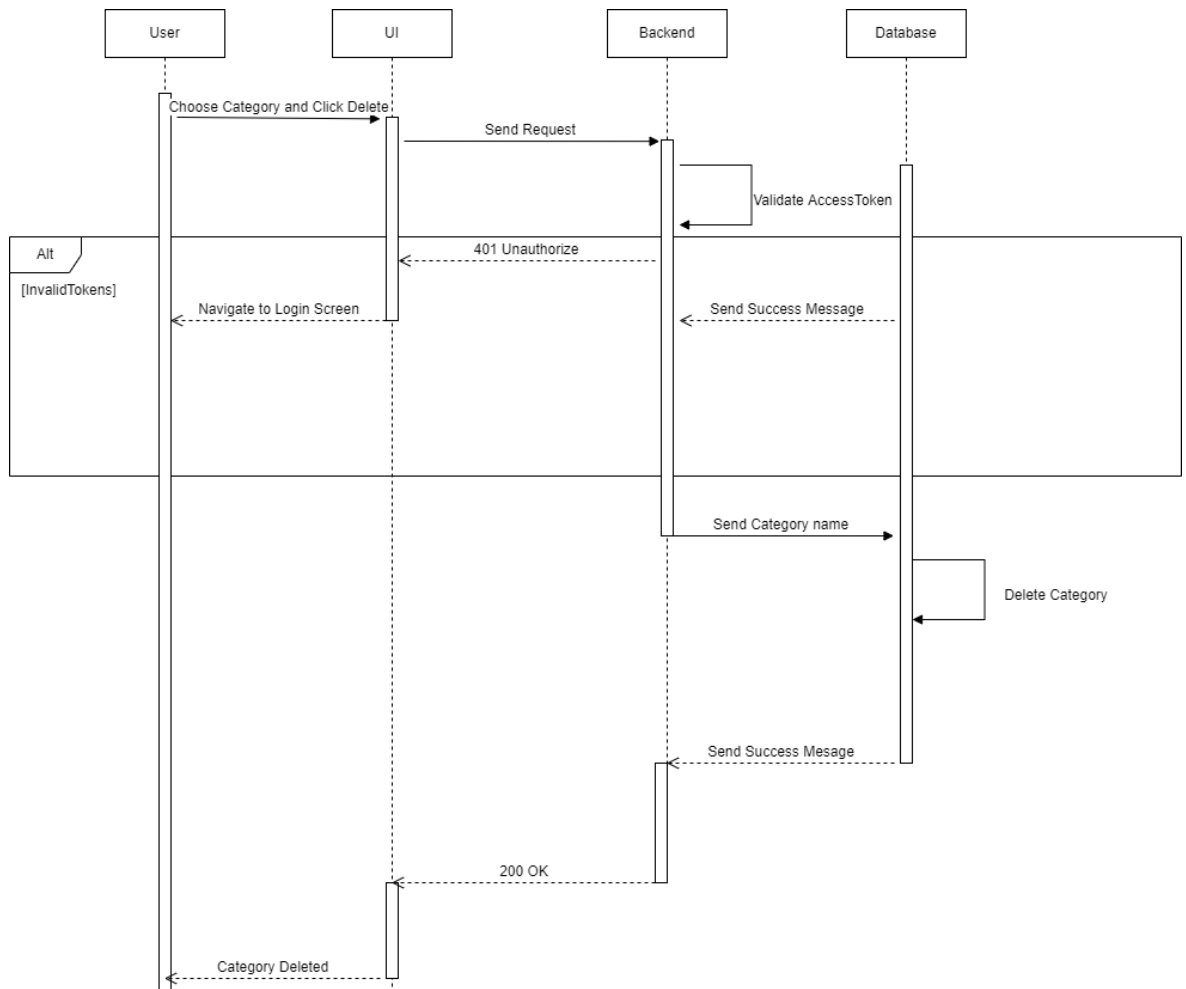
3.1.4.6 Create new Category



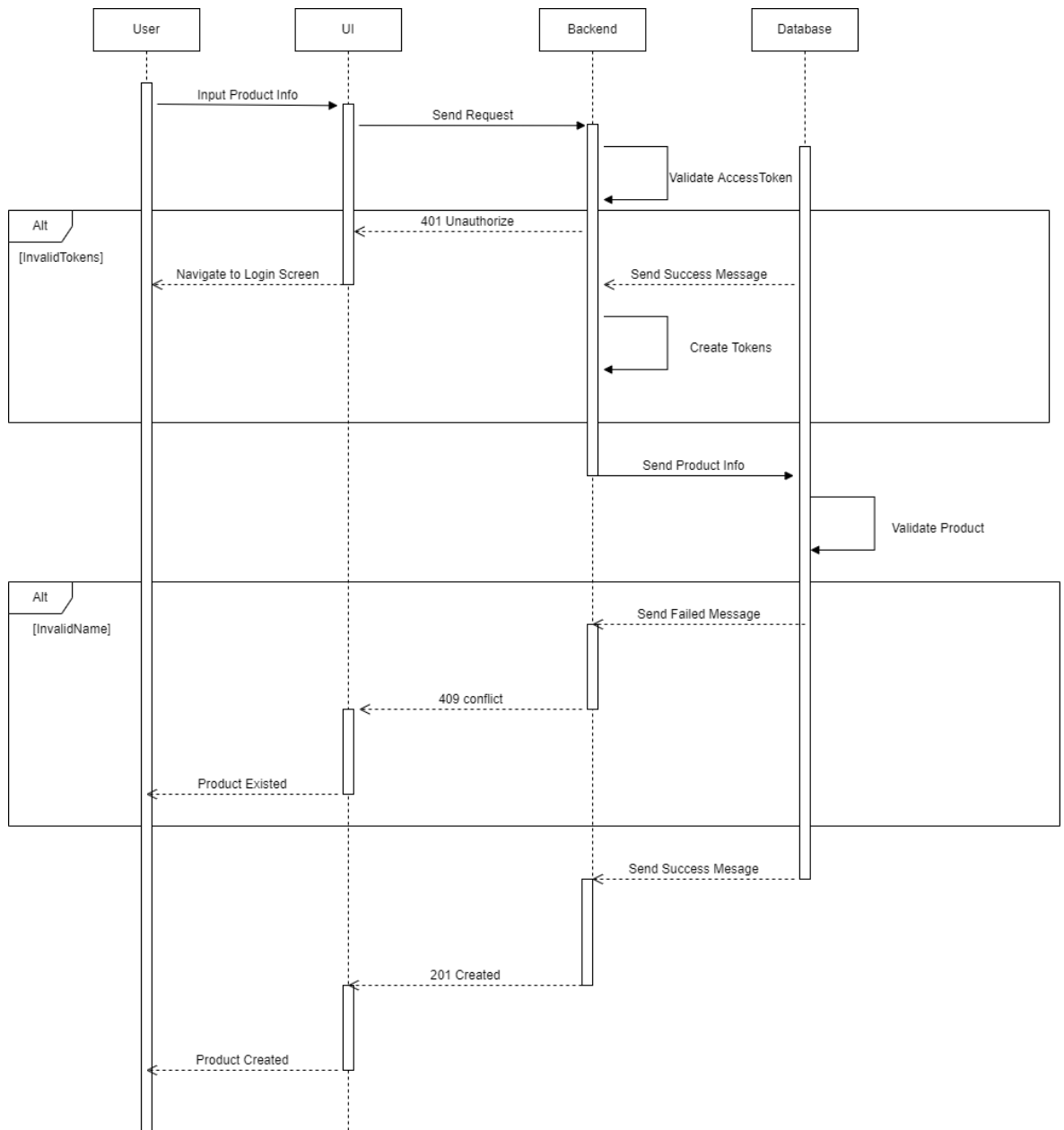
3.1.4.7 Edit Category



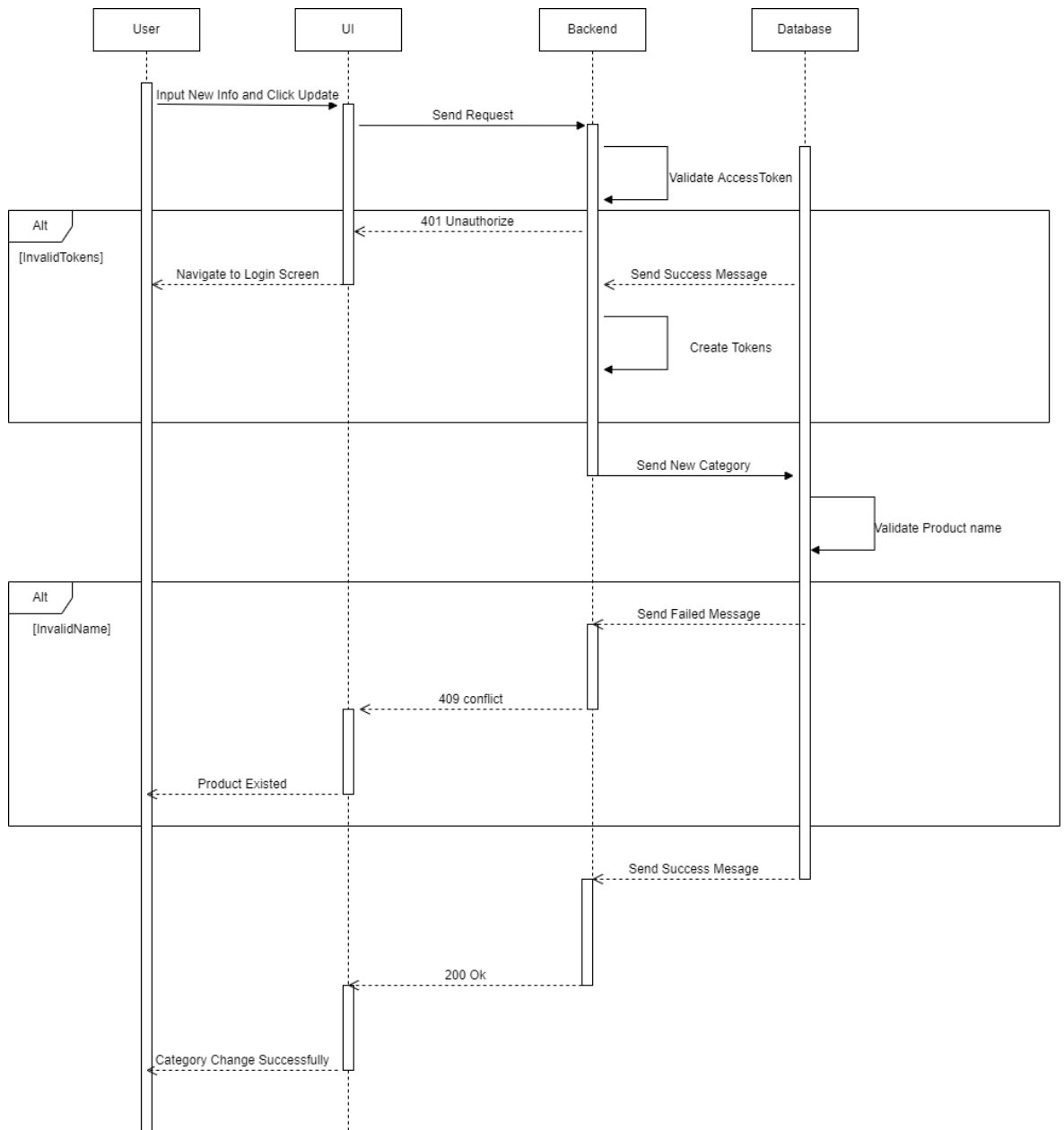
3.1.4.8 Delete Category



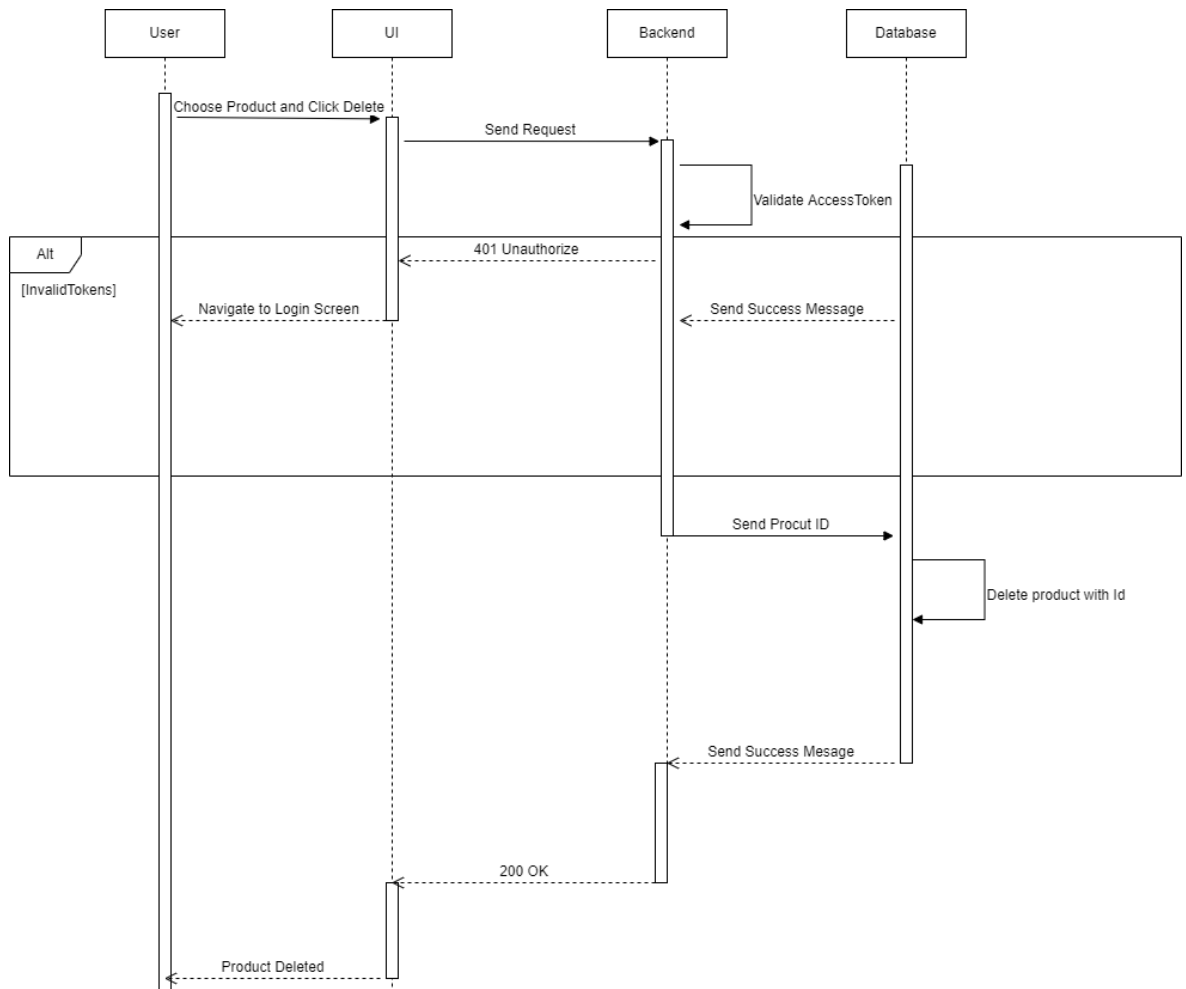
3.1.4.9 Create Product



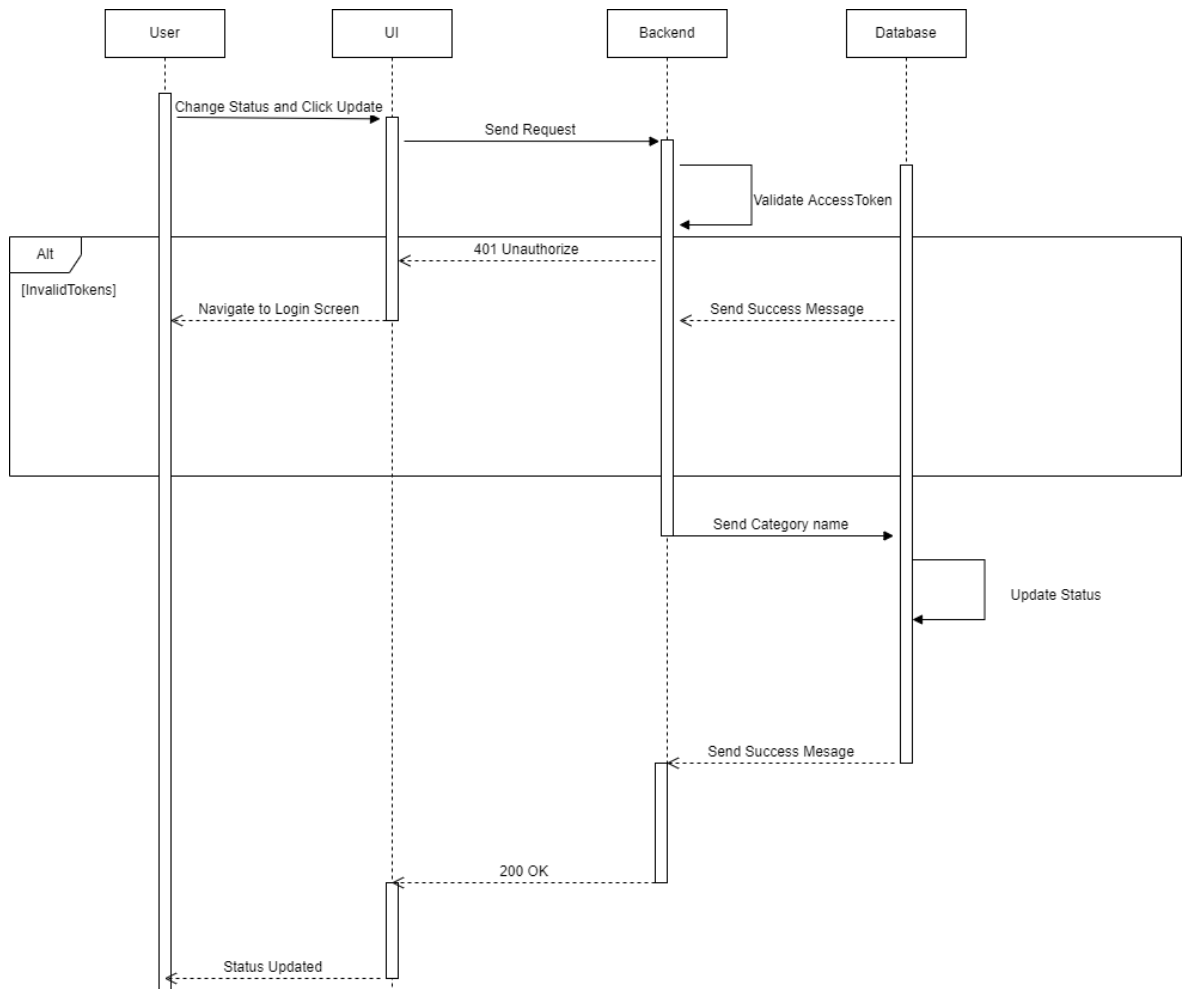
3.1.4.10 Edit Product



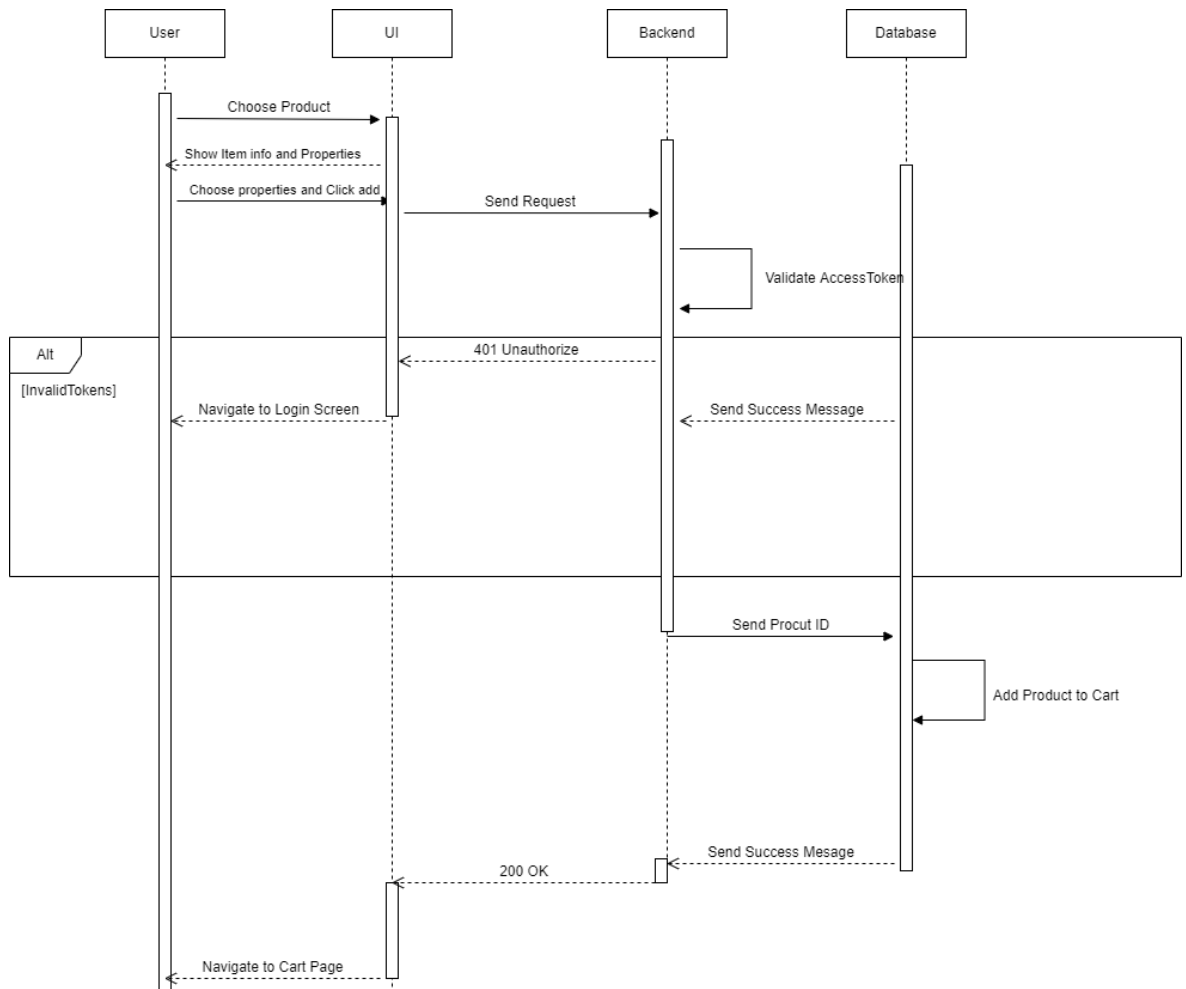
3.1.4.11 Delete Product



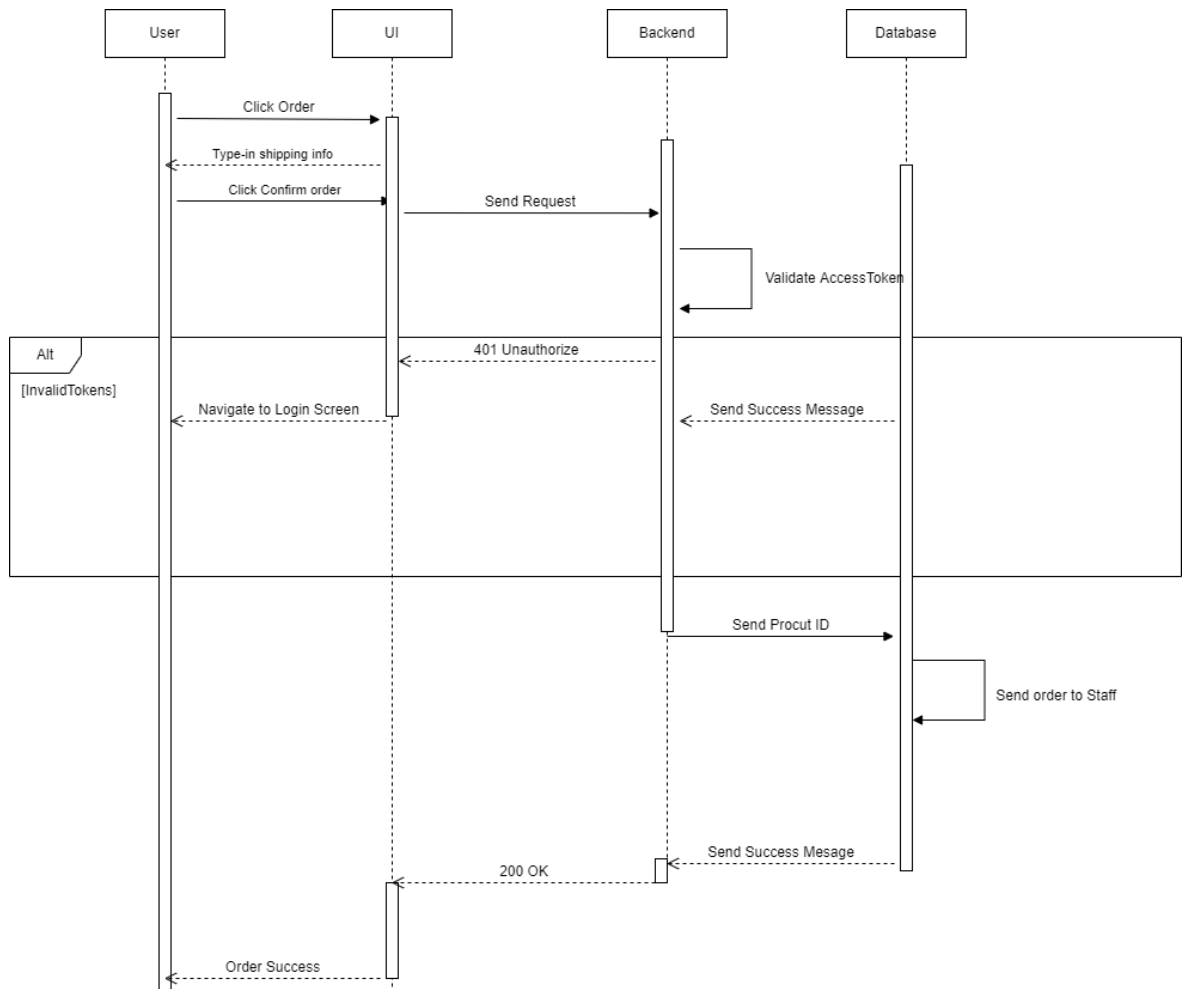
3.1.4.12 Update Cart Status



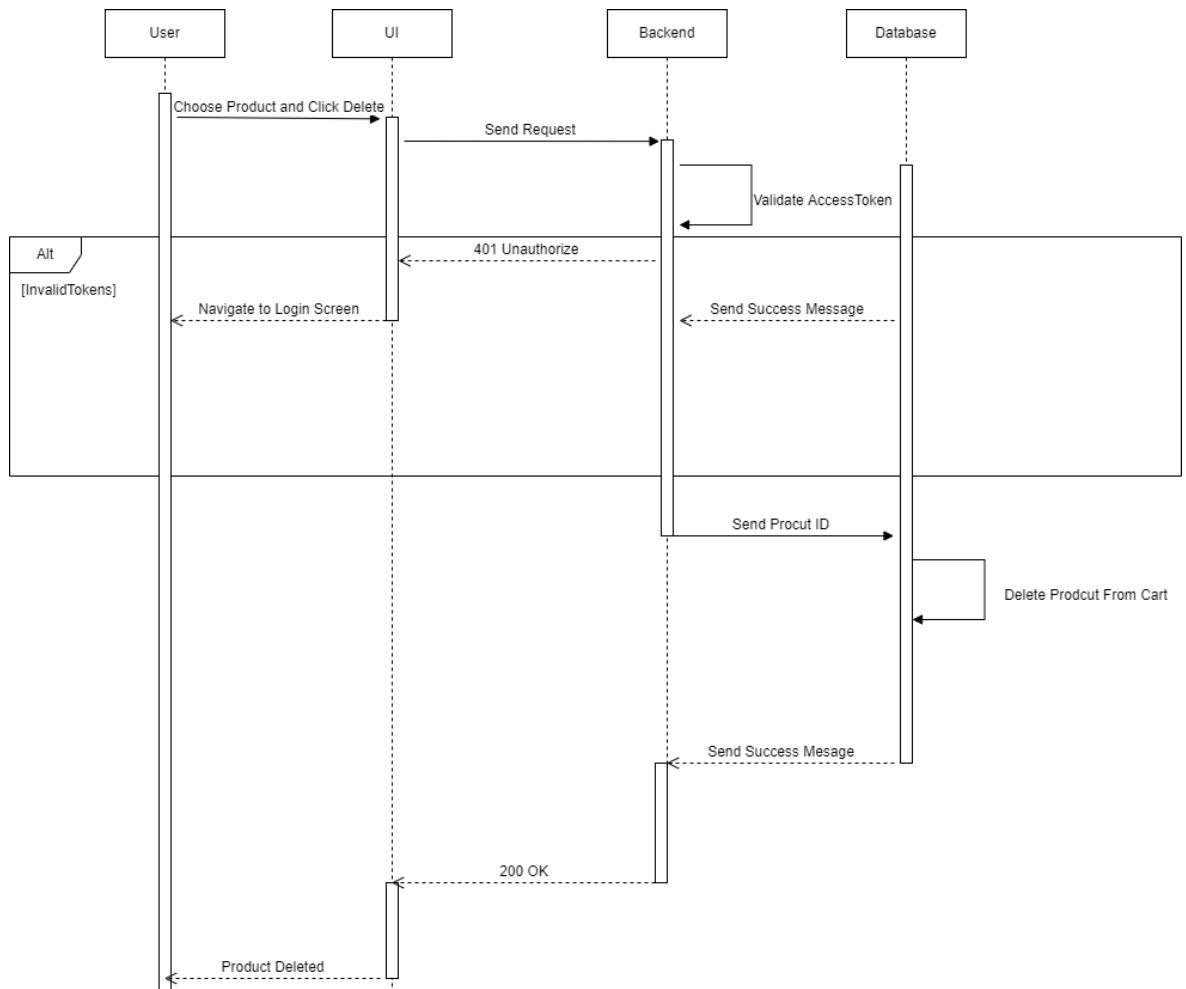
3.1.4.13 Add Item to Cart



3.1.4.14 Order Cart Item

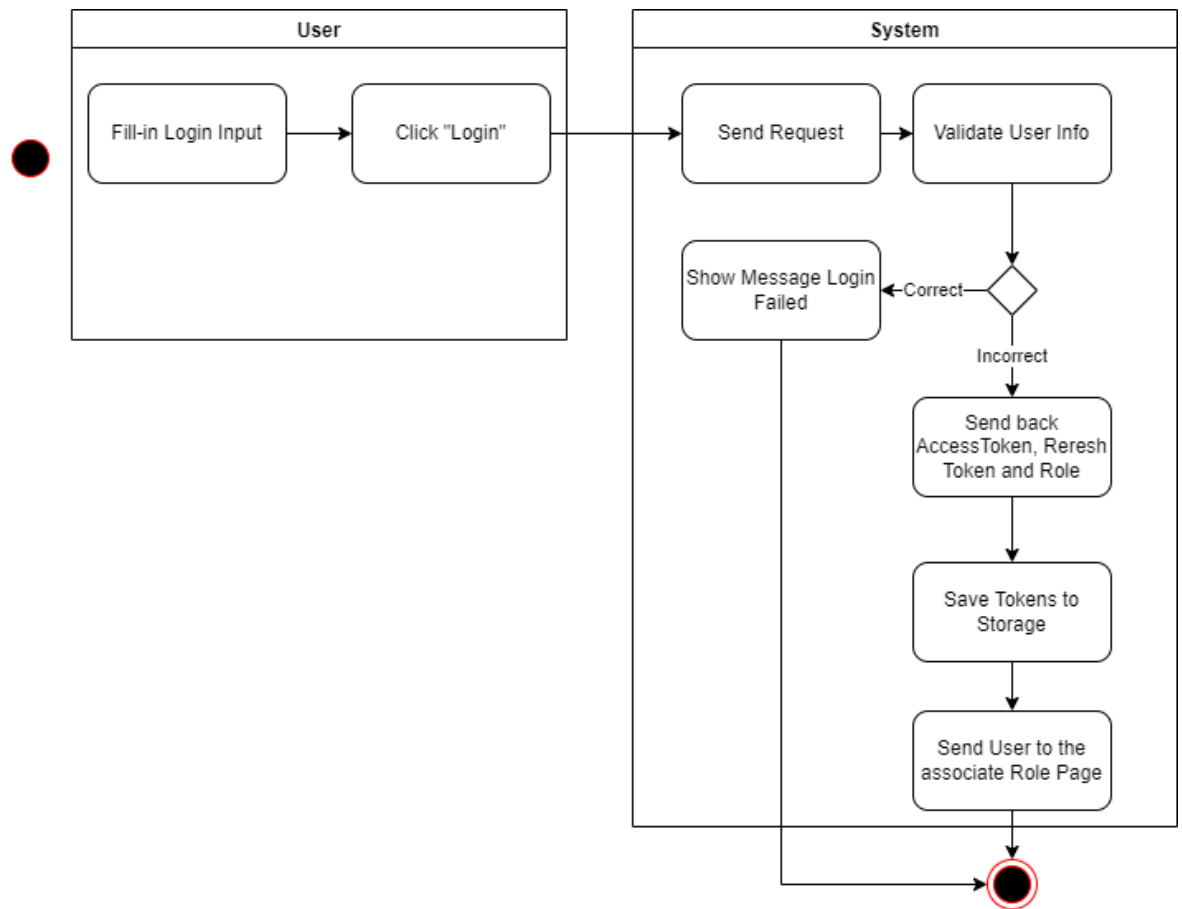


3.1.4.15 Remove Cart Item

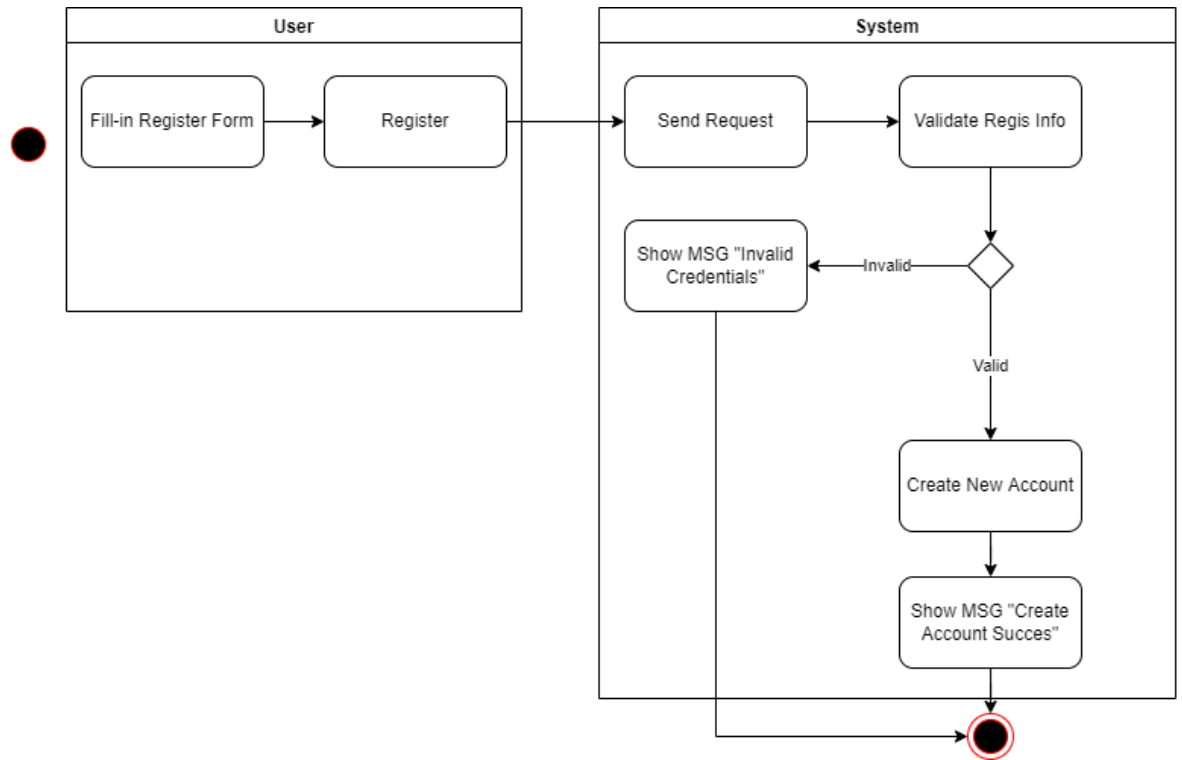


3.1.5. Sơ đồ hoạt động

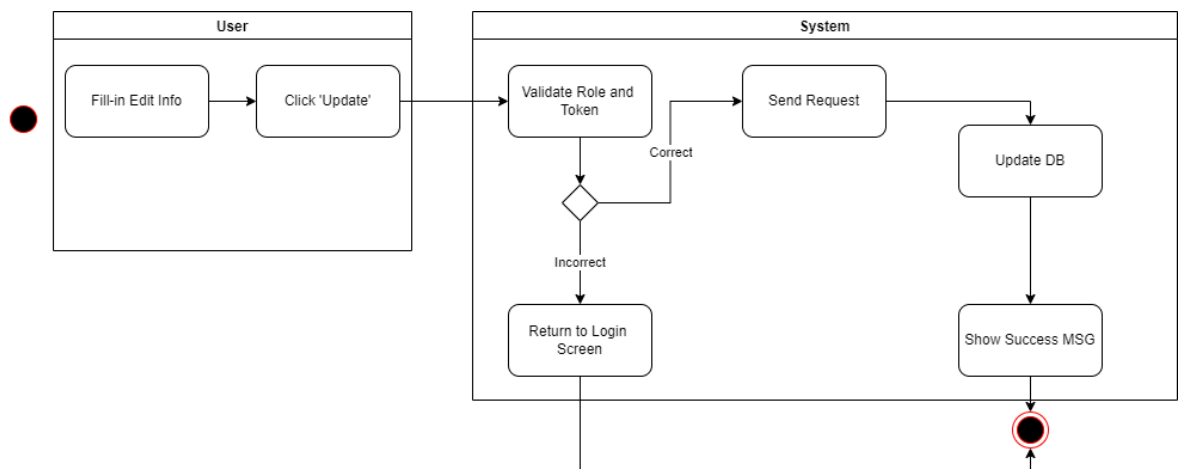
3.1.5.1 Login



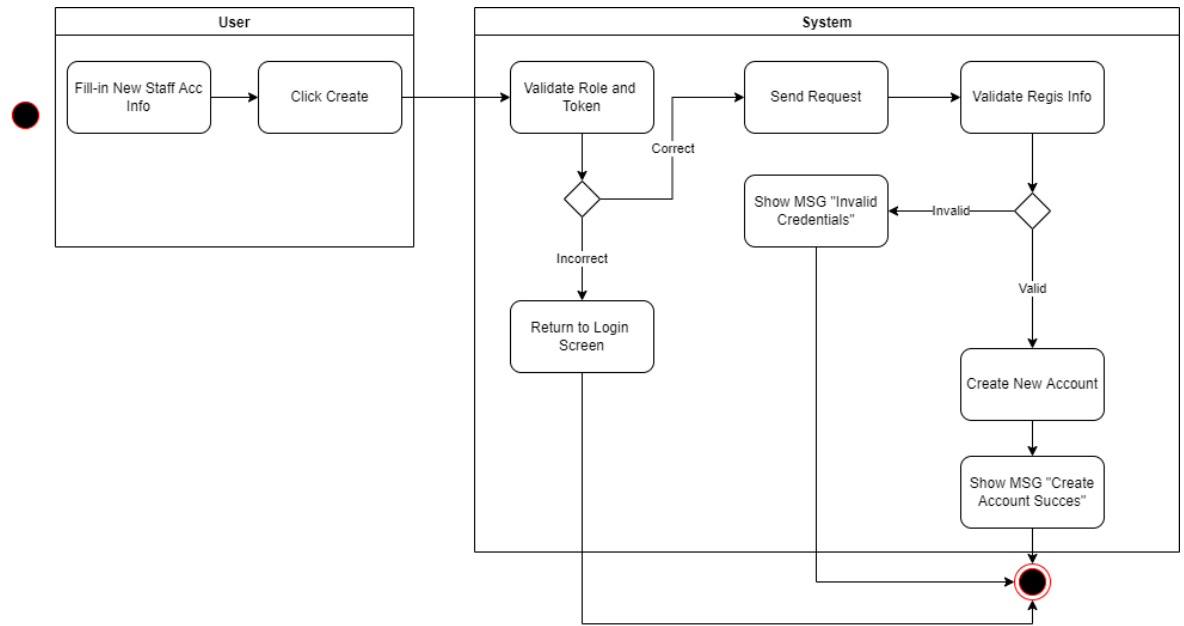
3.1.5.2 Register



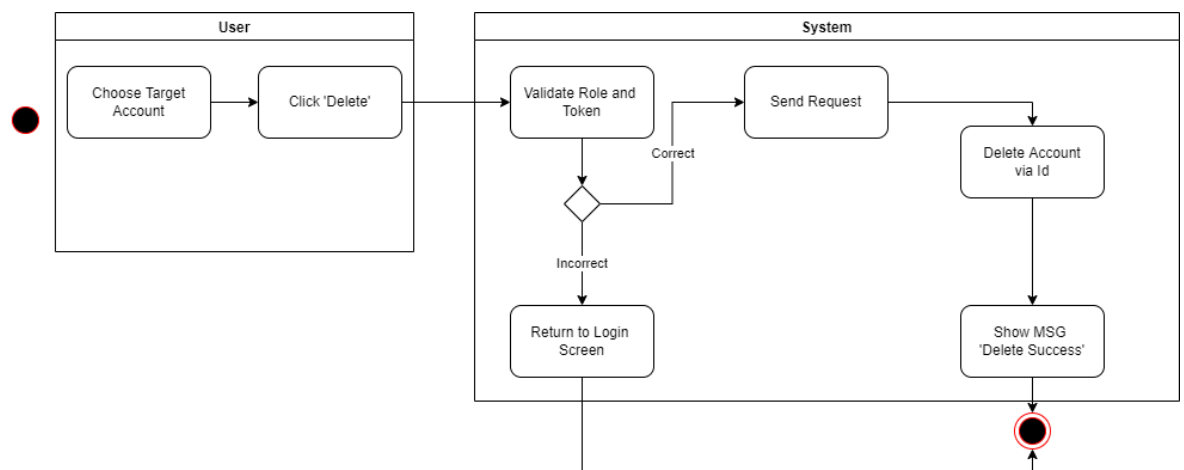
3.1.5.3 Edit Profile



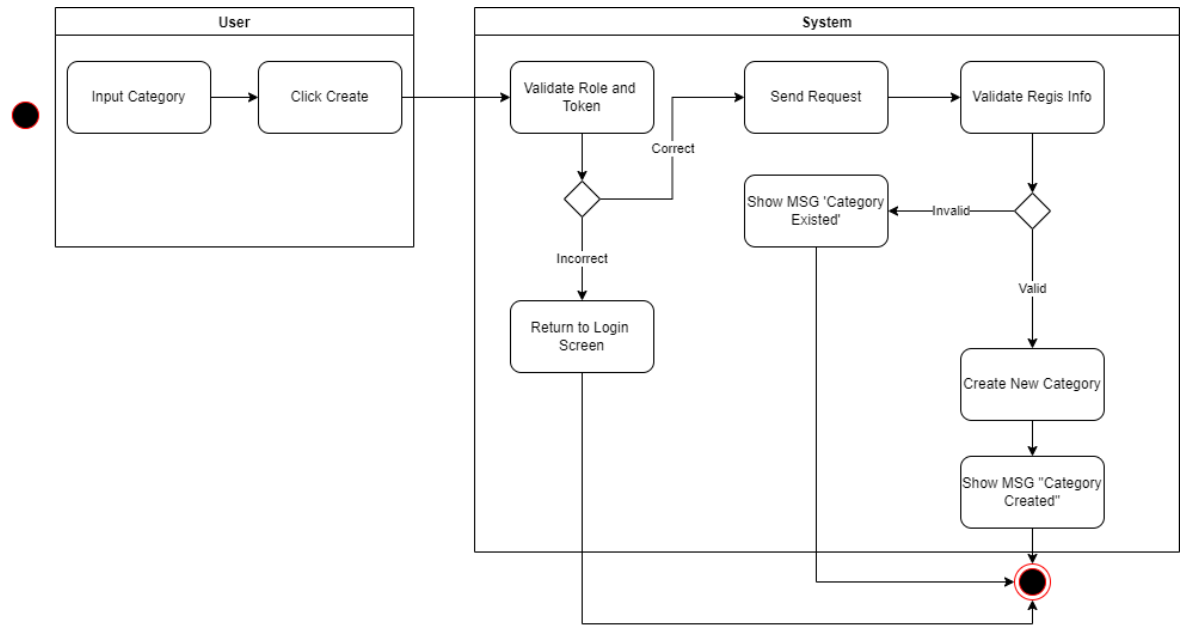
3.1.5.4 Create Staff Account



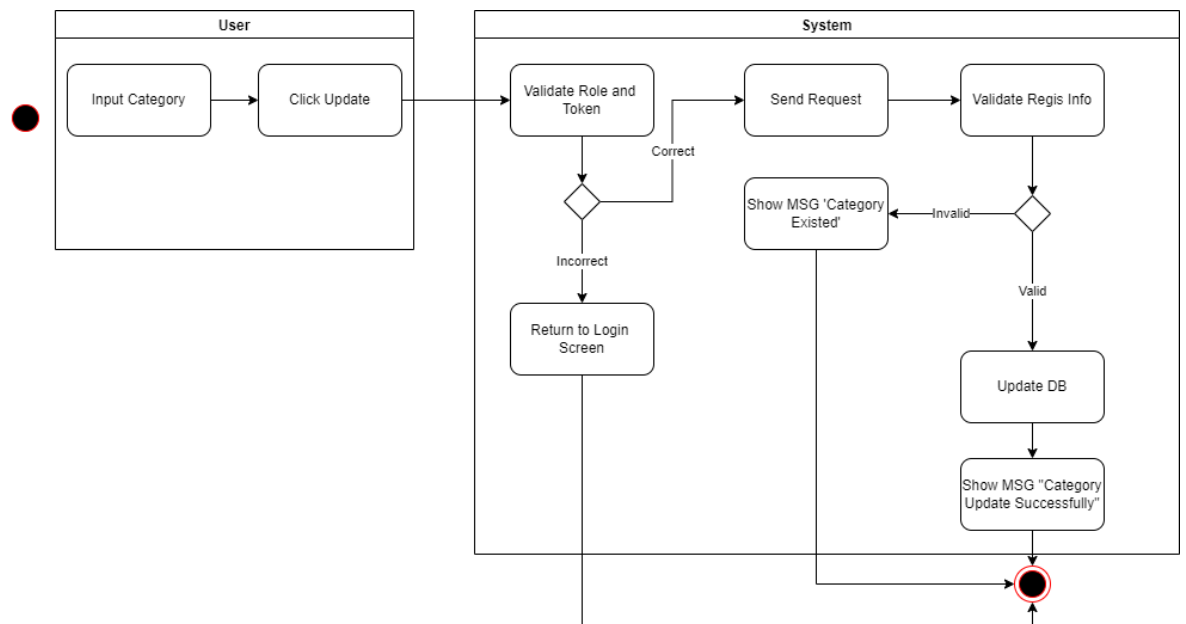
3.1.5.5 Delete Account



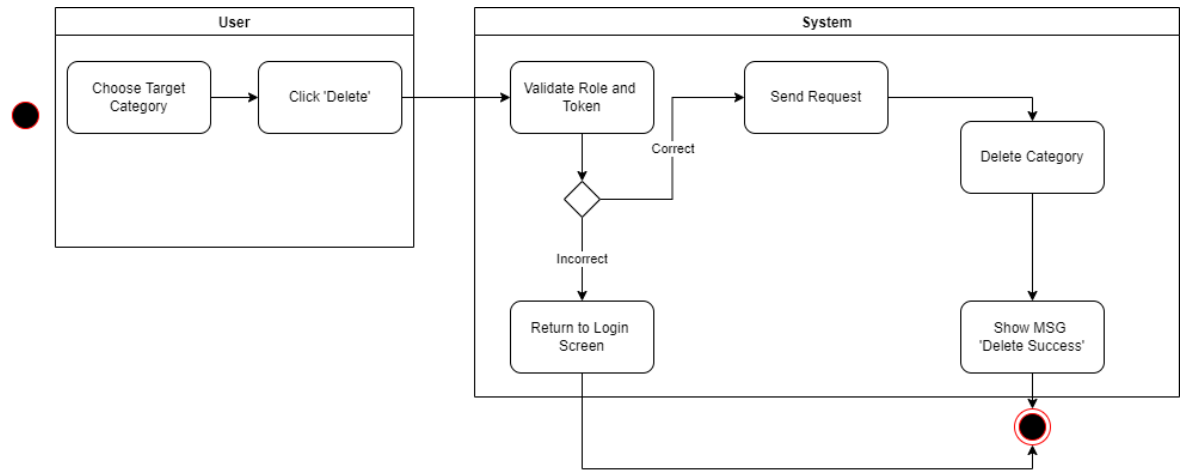
3.1.5.6 Create new Category



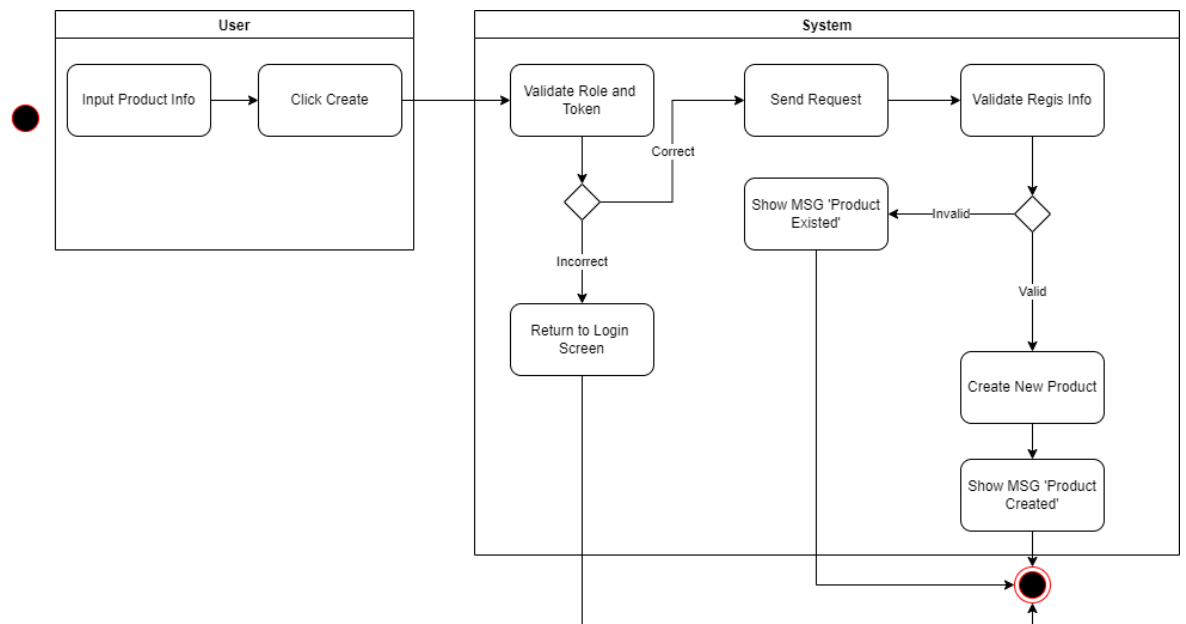
3.1.5.7 Edit Category



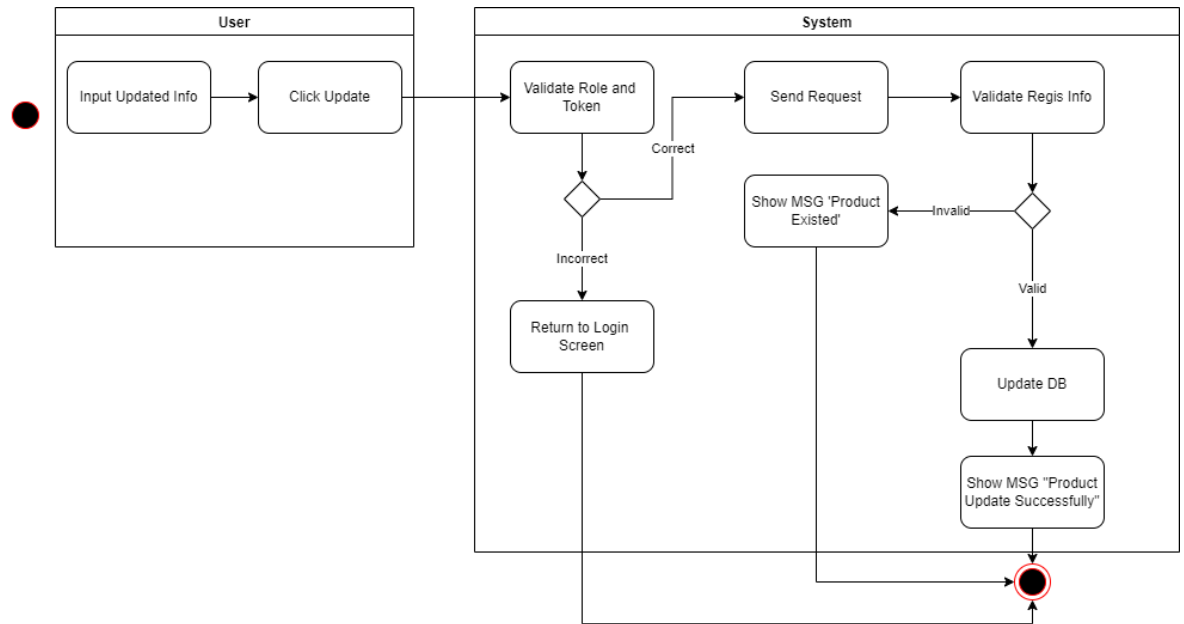
3.1.5.8 Delete Category



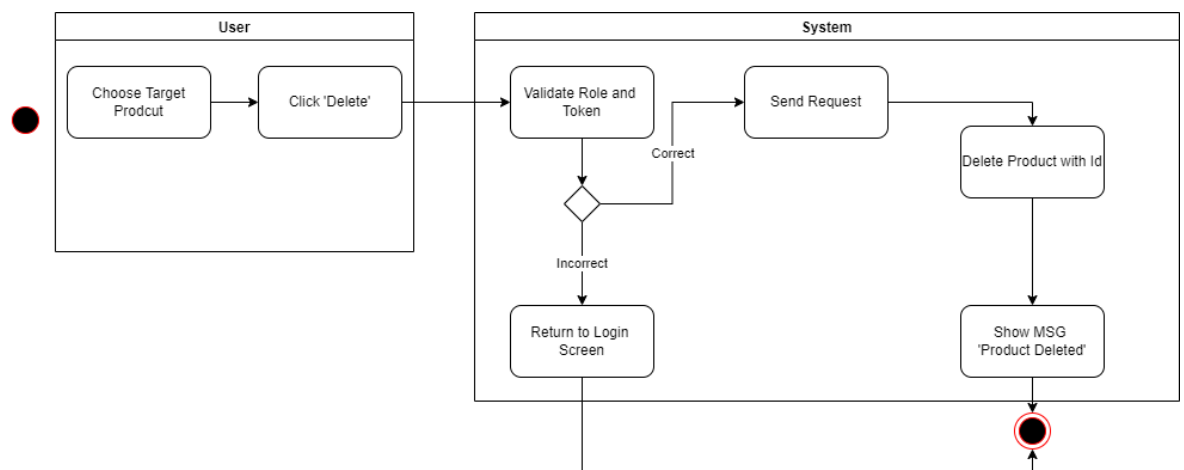
3.1.5.9 Create Product



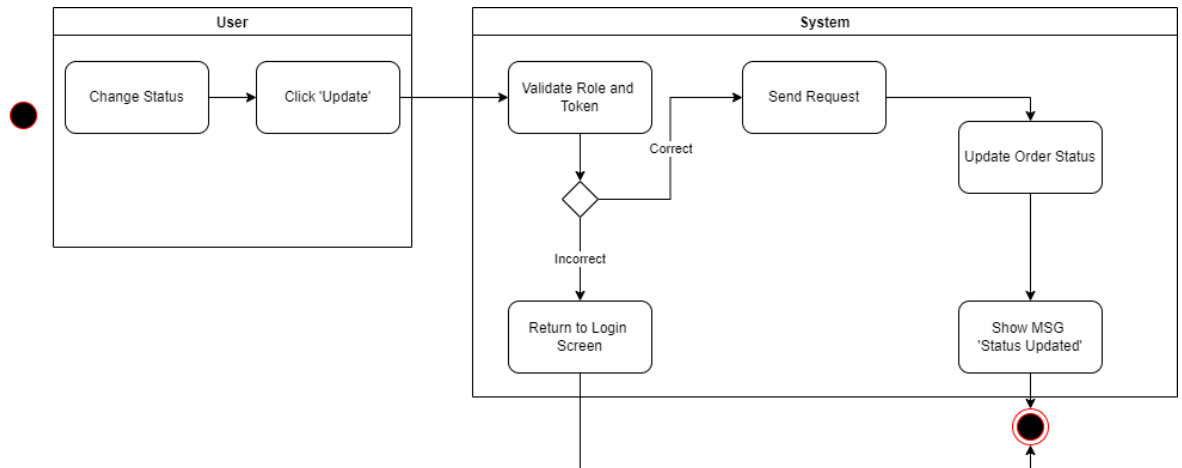
3.1.5.10 Edit Product



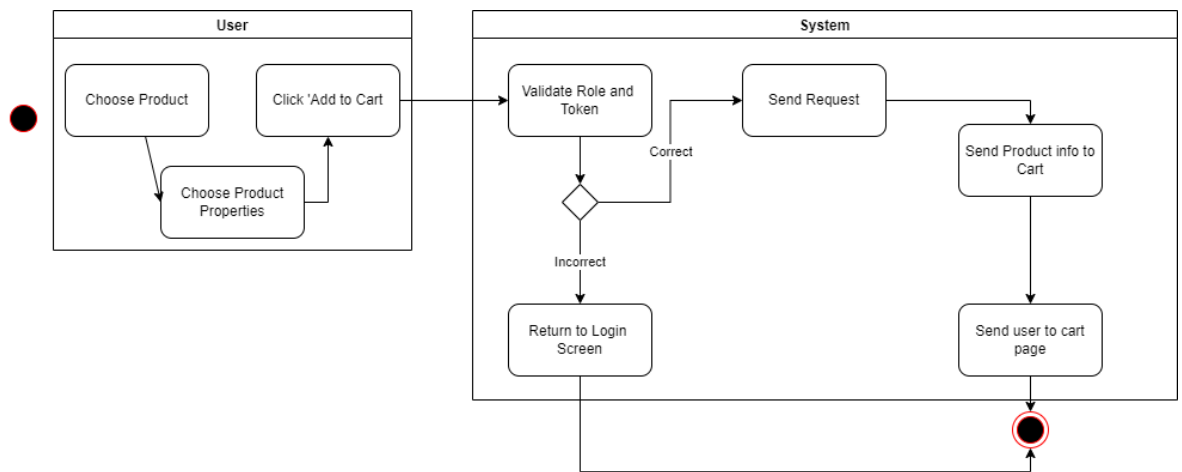
3.1.5.11 Delete Product



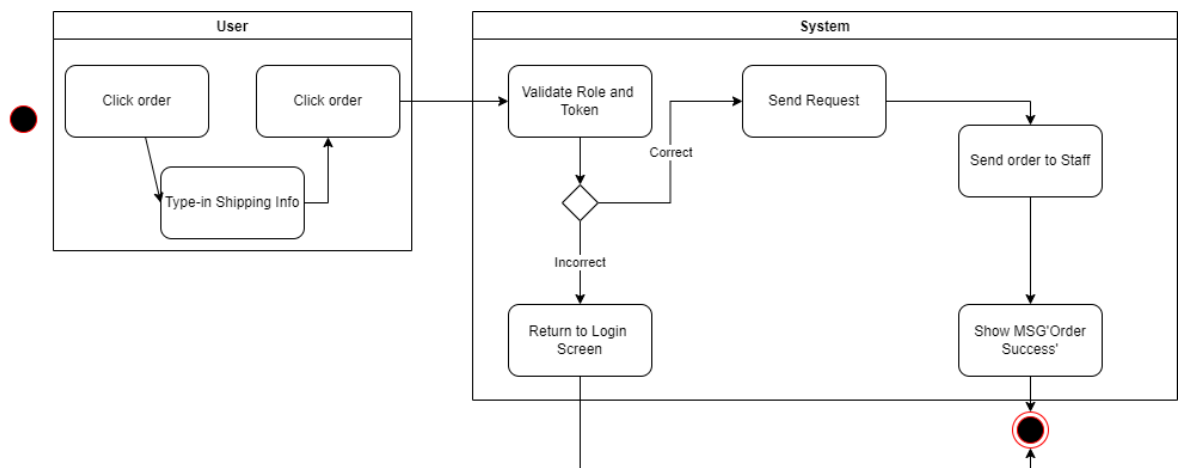
3.1.5.12 Update Cart Status



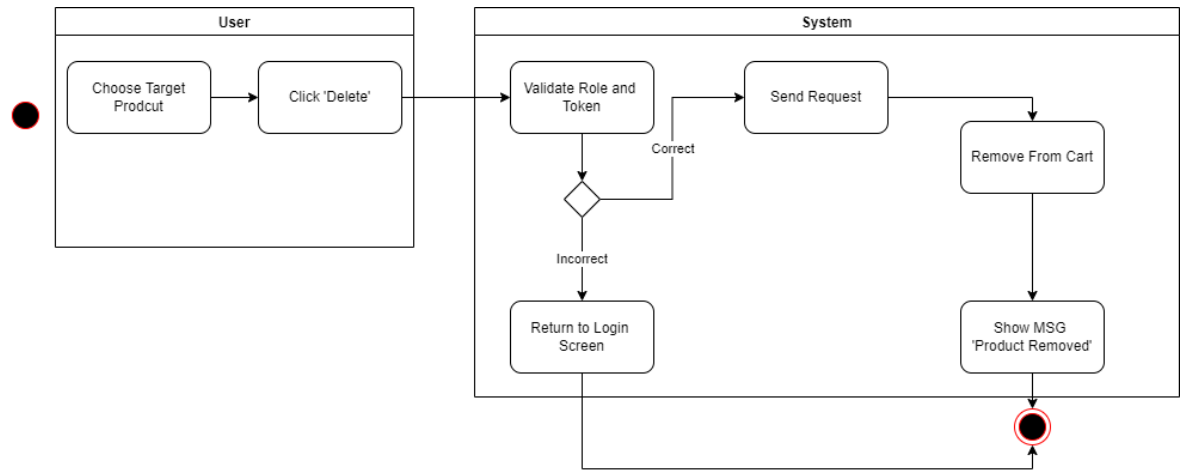
3.1.5.13 Add Item to Cart



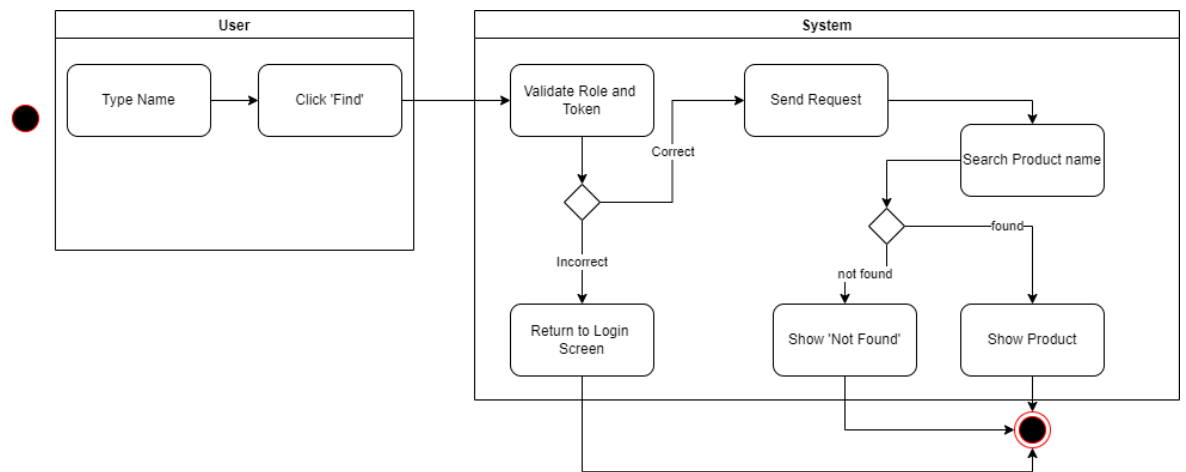
3.1.5.14 Order Cart Item



3.1.5.15 Remove Cart Item

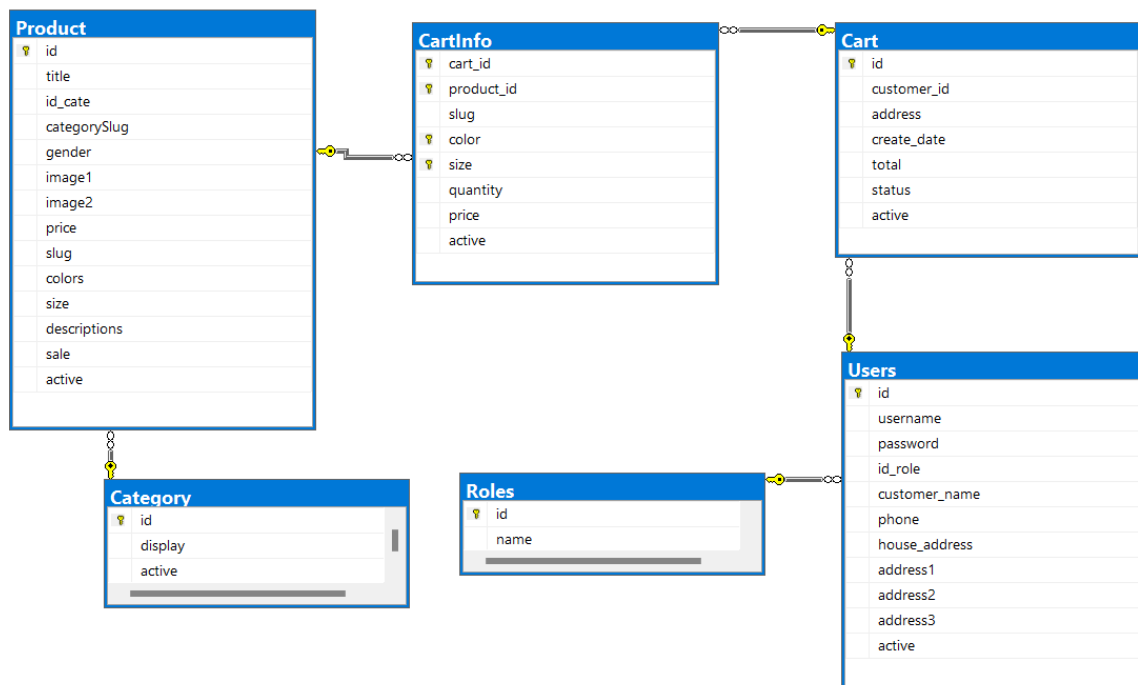


3.1.5.16 Find Product



3.1.6. Cơ sở dữ liệu

3.1.6.1. Sơ đồ Database



Hình 3-4 Sơ đồ Database

3.1.6.2. Cấu trúc bảng

a. Users

Bảng 3-1 Users

Tên cột	Kiểu dữ liệu	Kích cỡ	Mô tả
<u>id</u>	varchar	5	Id của user
username	varchar	50	Email được đặt là tài khoản
password	varchar	max	Mật khẩu
id_role	varchar	100	Id của role
customer_name	varchar	50	Tên user
phone	varchar	20	Số điện thoại user

house_address	varchar	100	Địa chỉ nhà
address1	varchar	100	Địa chỉ phường
address2	varchar	100	Địa chỉ quận
address3	varchar	100	Địa chỉ thành phố
active	bit		Cờ hiệu dành cho xóa

b. Roles

Bảng 3-2 Roles

Tên cột	Kiểu dữ liệu	Kích cỡ	Mô tả
<u>id</u>	varchar	5	Id của role
name	varchar	50	Tên của role

c. Cart

Bảng 3-3 Cart

Tên cột	Kiểu dữ liệu	Kích cỡ	Mô tả
<u>id</u>	varchar	9	Id của cart dùng để tham chiếu đến các cart info
Customer_id	varchar	5	Id của customer
Adress	nvarchar	Max	Địa chỉ giao hàng của cart
Create_date	Smalldatetime		Ngày tạo đơn
Total	Int		Tổng tiền đơn
Status	nvarchar	50	Trạng thái đơn
Active	Bit		Cờ hiệu dành cho xóa

d. CartInfo

Bảng 3-4 CartInfo

Tên cột	Kiểu dữ liệu	Kích cỡ	Mô tả
<u>cart_id</u>	varchar	9	Id của cart
product_id	varchar	10	Id của product
slug	varchar	50	Phân loại của product
color	varchar	50	Màu sắc
size	varchar	50	Kích cỡ
quantity	int		Số lượng product
price	int		Giá
active	bit		Cờ hiệu dành cho xóa

e. Product

Bảng 3-5 Product

Tên cột	Kiểu dữ liệu	Kích cỡ	Mô tả
<u>id</u>	varchar	10	Id của product
title	nvarchar	100	Tên của product
<u>id_cate</u>	varchar	10	Id của category
categoryslug	varchar	max	Phân loại của category

gender	nvarchar	10	Giới tính sản phẩm
image1	varchar	max	Link ảnh 1
image2	varchar	max	Link ảnh 2
price	int		Giá product
slug	varchar	100	Phân loại product
colors	varchar	100	Các màu sắc product
size	varchar	50	Các kích cỡ sản phẩm
descriptions	nvarchar	max	Mô tả sản phẩm
sale	int		Phần trăm sale sản phẩm
active	bit		Cờ hiệu dành cho xóa

f. Category

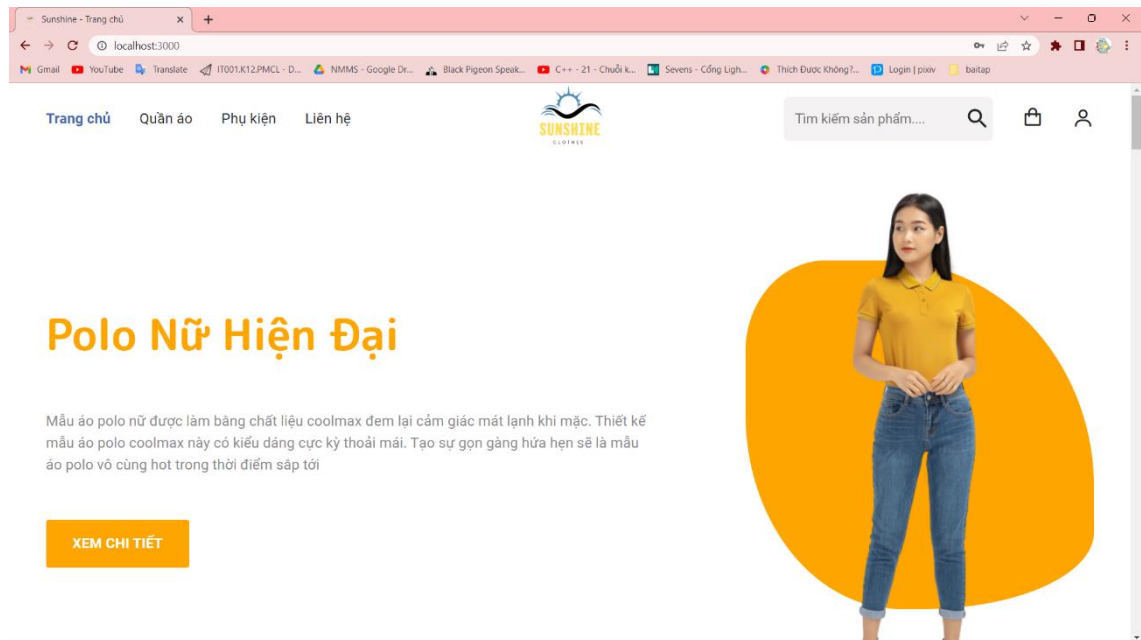
Bảng 3-6 Category

Tên cột	Kiểu dữ liệu	Kích cỡ	Mô tả
<u>id</u>	varchar	10	Id của category
active	bit		Cờ hiệu dành cho xóa
display	varchar	50	

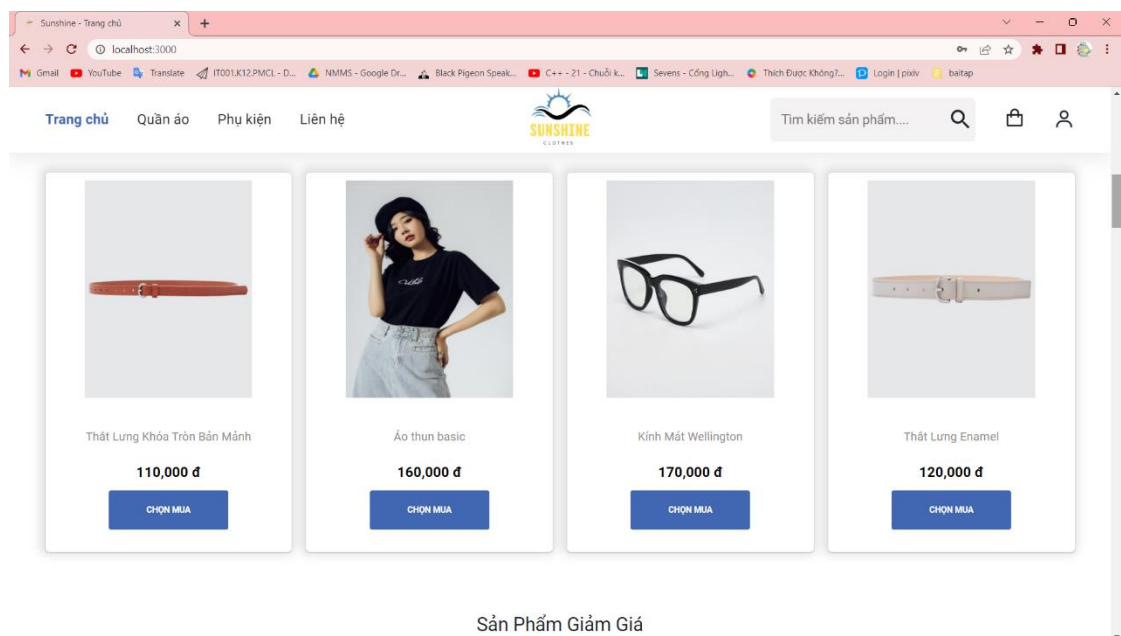
3.2. Thiết kế giao diện

3.2.1. Giao diện ứng dụng

3.2.1.1. Giao diện chính của trang web:



Hình 3-6 Giao diện chính



Hình 3-5 Giao diện chính

3.2.1.2. Giao diện đăng nhập:

Đăng nhập

Đăng nhập

Welcome back!
Enter your email address and password to access admin panel.

Email

Mật khẩu

ĐĂNG NHẬP

This beautiful theme yours!

"Best investment i made for a long time. Can only recommend it for other users."

- Admin User

Don't have an account? [Register](#)

Hình 3-7 Giao diện đăng nhập

3.2.1.3. Giao diện đăng ký:

Register Form

Đăng ký

Biểu mẫu đăng ký.
If You Really Want To Know, Look In The Register.

Họ và tên

Email

Mật khẩu

Lặp lại Mật khẩu

XÁC NHẬN ĐĂNG KÝ

This beautiful theme yours!

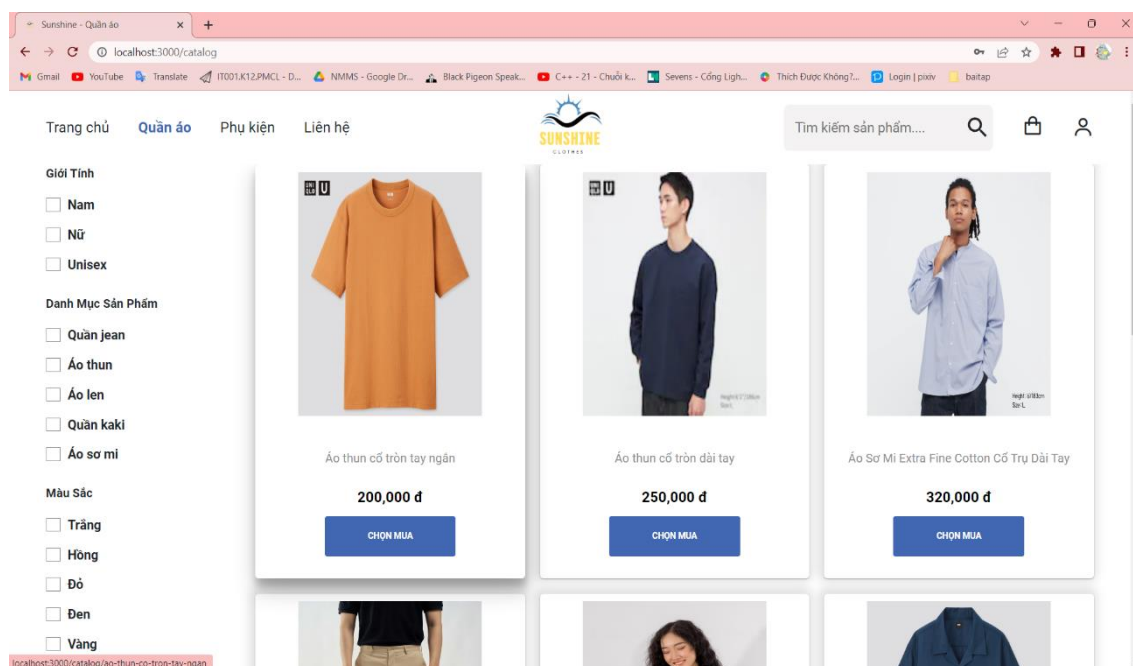
"Chúc quý khách hàng sẽ có những trải nghiệm tuyệt vời khi mua sắm ở đây."

- Admin User

Đã có tài khoản? [Đăng nhập](#)

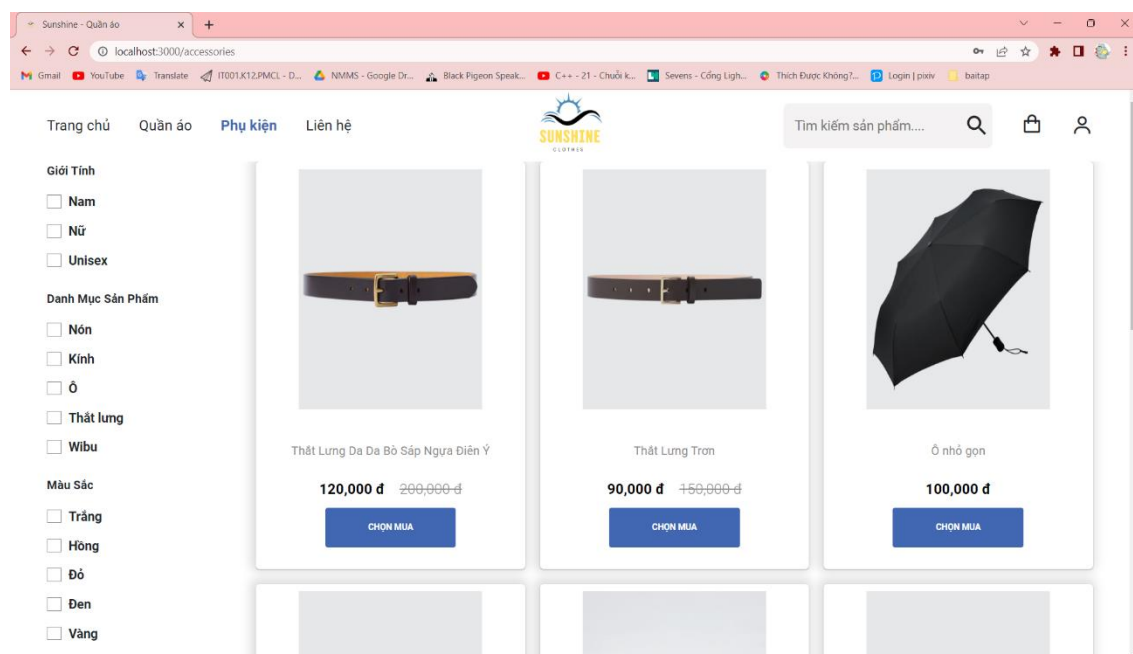
Hình 3-8 Giao diện đăng ký

3.2.1.4. Giao diện sản phẩm:



Hình 3-9 Giao diện sản phẩm

3.2.1.5. Giao diện Phụ Kiện:



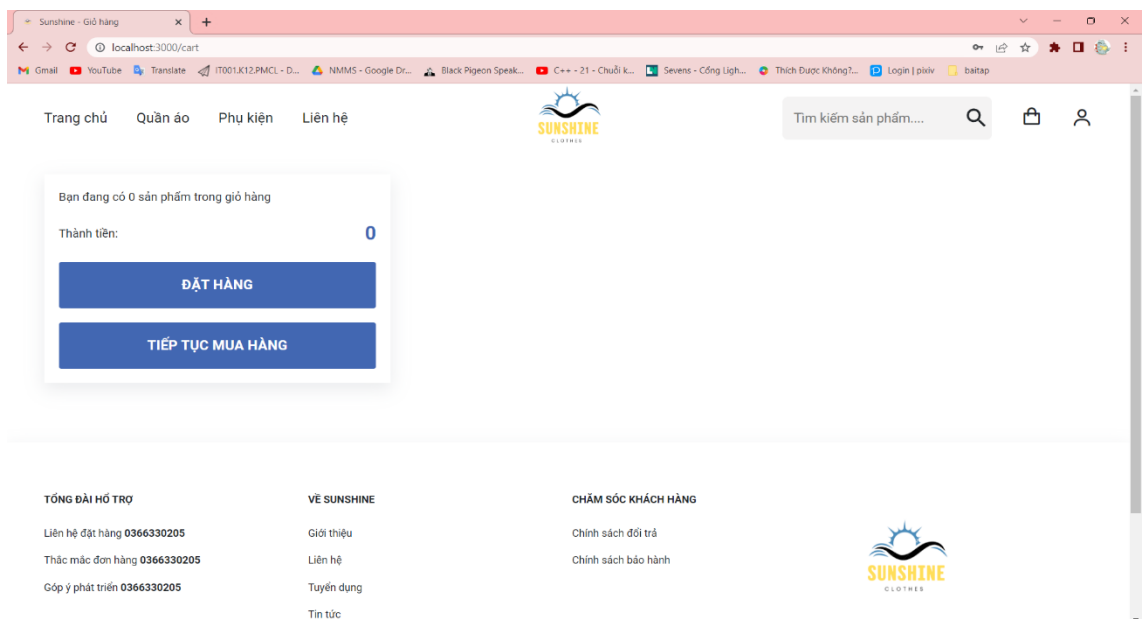
Hình 3-10 Giao diện phụ kiện

3.2.1.6. Giao diện thông tin liên hệ:



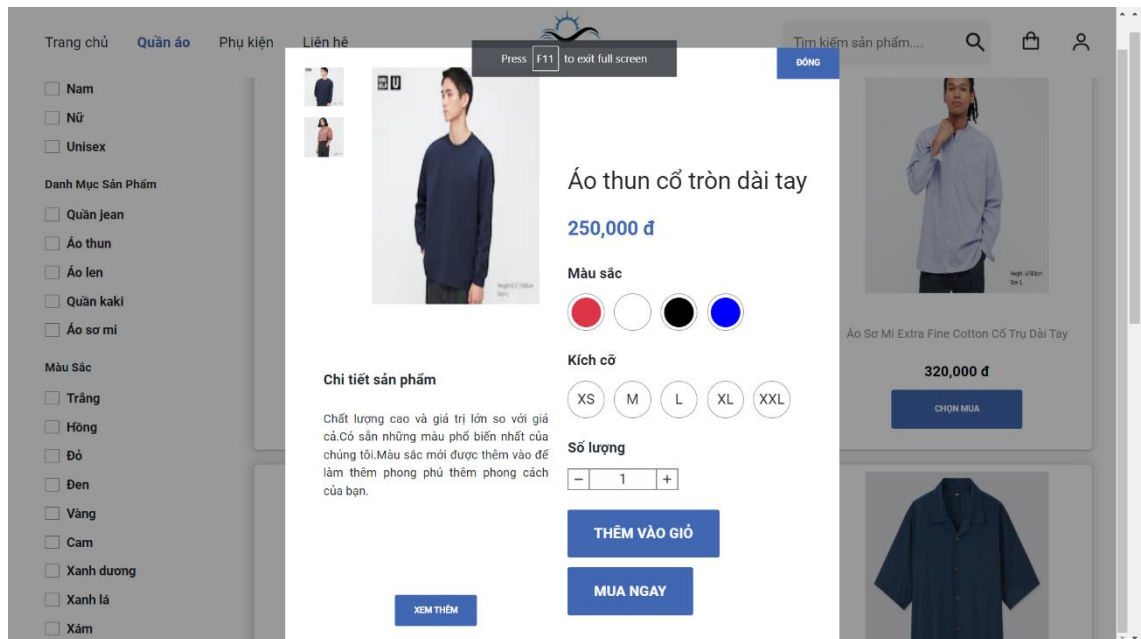
Hình 3-11 Giao diện thông tin liên hệ

3.2.1.7. Giao diện giỏ hàng:



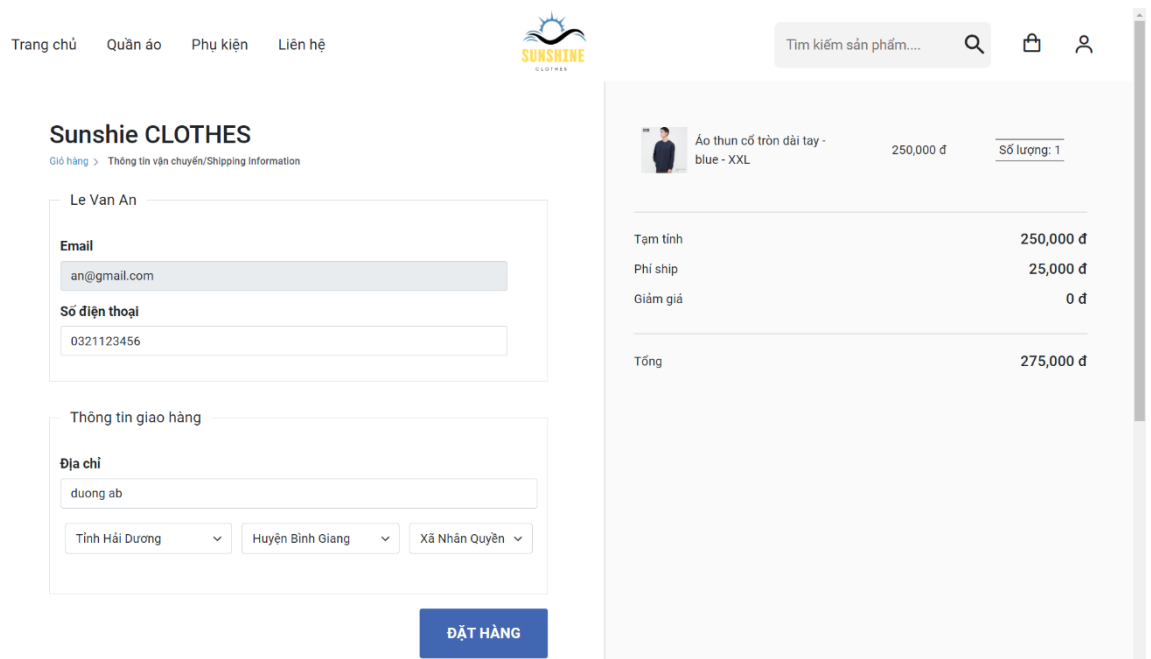
Hình 3-12 Giao diện giỏ hàng

3.2.1.8. Giao diện Mua hàng:



Hình 3-13 Giao diện mua hàng

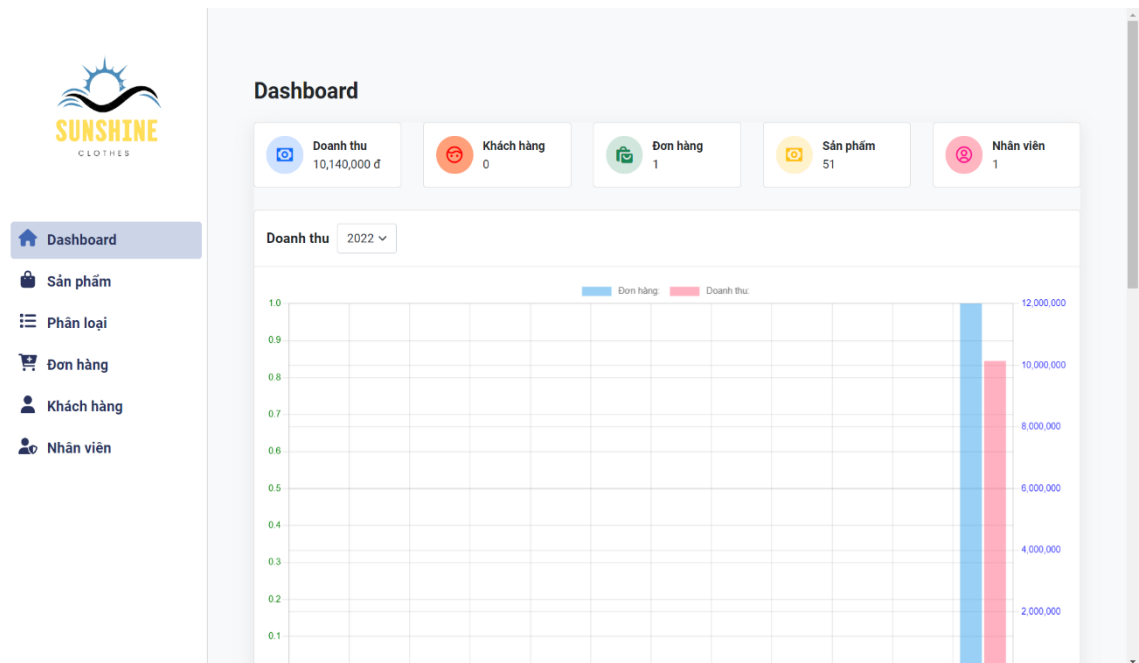
3.2.1.9. Giao diện đặt hàng:



Hình 3-14 Giao diện đặt hàng

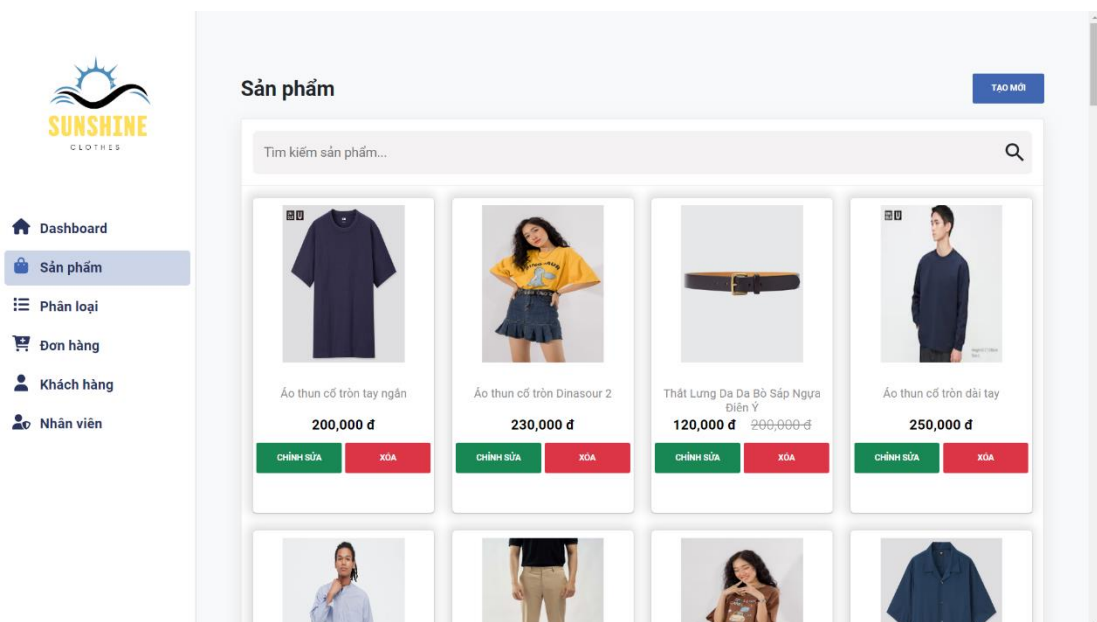
3.2.2. Giao diện quản lý

3.2.2.1. Giao diện Dashboard doanh thu:



Hình 3-16 Giao diện dashboard doanh thu

3.2.2.2. Giao diện Danh sách sản phẩm:



Hình 3-15 Giao diện Danh sách sản phẩm

3.2.2.3. Giao diện Chỉnh sửa/thêm mới sản phẩm:

SẢN PHẨM

Chỉnh sửa thông tin sản phẩm

Tên sản phẩm
Áo thun cổ tròn tay ngắn

Giá
200000

Giới tính
☐ Nam ☐ Nữ ☒ Unisex

Loại sản phẩm
Áo thun

Hình ảnh 1
Choose File No file chosen

Hình ảnh 2
Choose File No file chosen

Màu sắc
[Color selection circles]

Kích cỡ
[Size selection buttons: XS, S, M, L, XL, XXL]

Hình 3-17 Giao diện chỉnh sửa/thêm mới sản phẩm

3.2.2.4. Giao diện Cập nhập và Quản lý phân loại Sản Phẩm:

PHÂN LOẠI

Tên	Tùy chọn
Quần jean	[icon]
Áo thun	[icon]
Áo len	[icon]
Quần kaki	[icon]
Áo sơ mi	[icon]

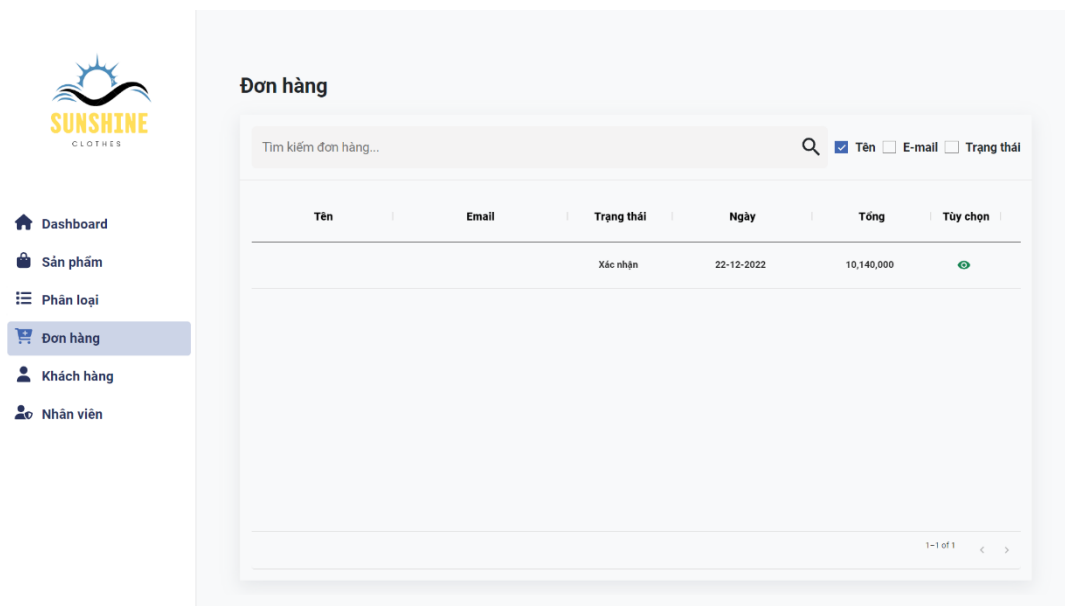
Loại sản phẩm
[input field]

CẬP NHẬT

1-5 of 10

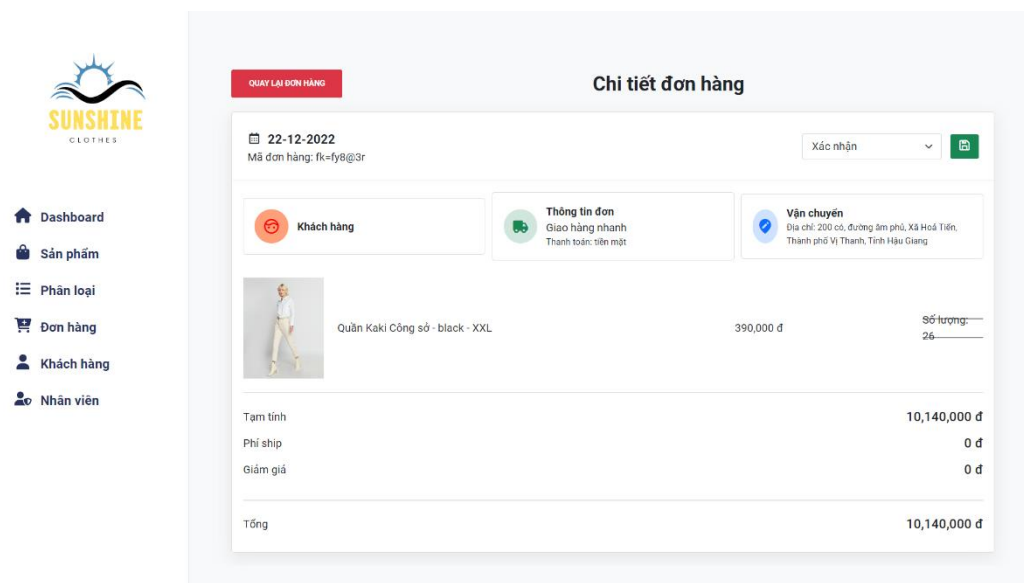
Hình 3-18 Giao diện Cập nhập và Quản lý phân loại Sản Phẩm

3.2.2.5. Giao diện đơn hàng:



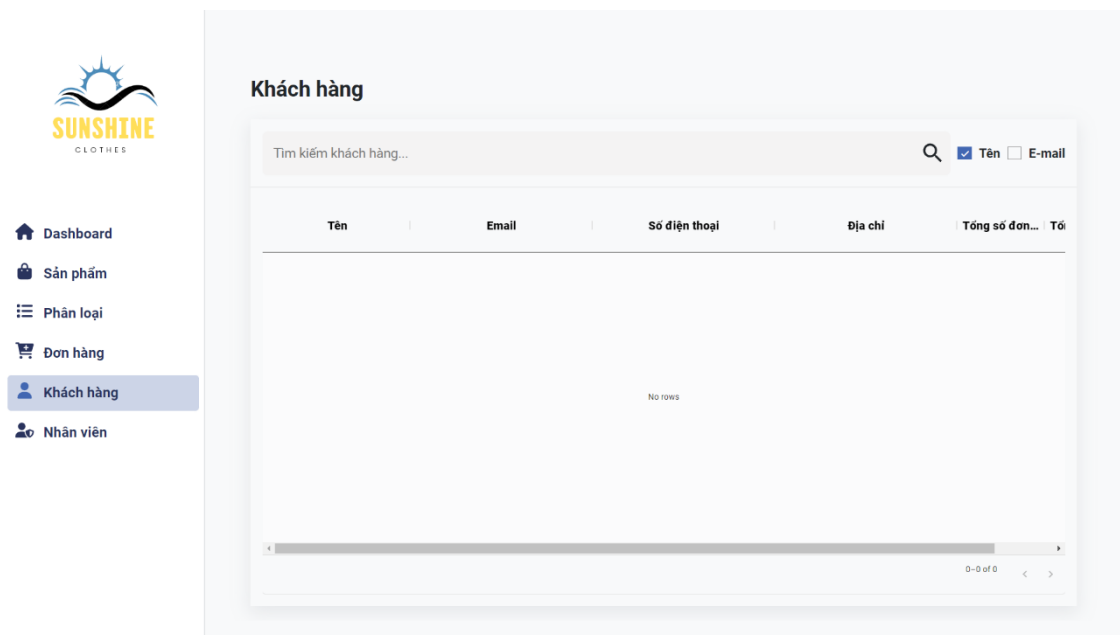
Hình 3-19 Giao diện đơn hàng

3.2.2.6. Giao diện Chỉnh sửa Trạng Thái và Chi tiền đơn hàng:



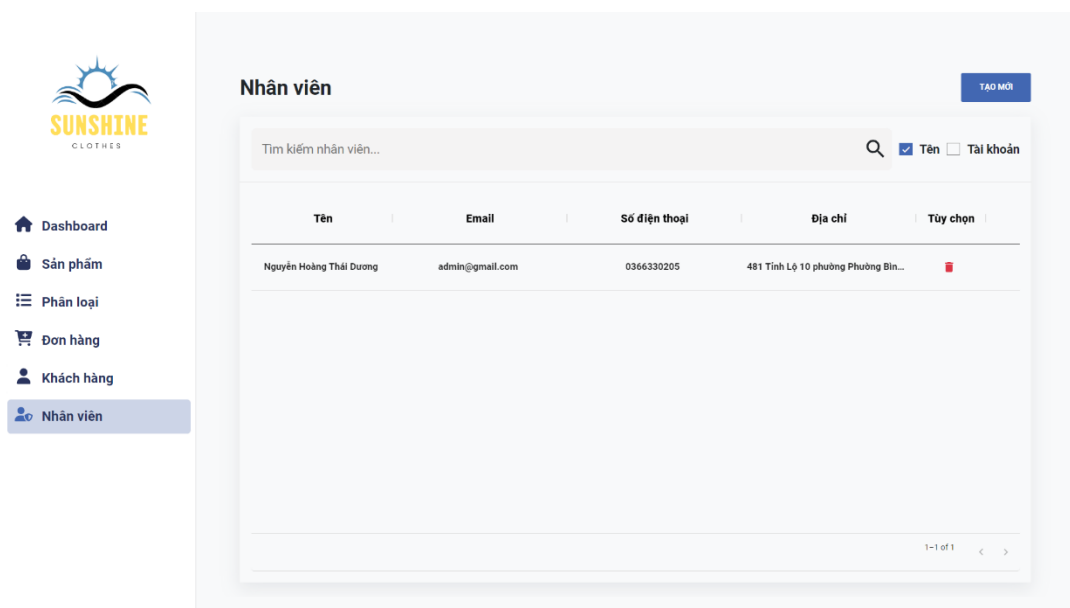
Hình 3-20 Giao diện Chỉnh sửa Trạng Thái và Chi tiền đơn hàng

3.2.2.7. Giao diện Danh Sách khách hàng:



Hình 3-21 Giao diện Danh Sách khách hàng

3.2.2.8. Giao diện danh sách Nhân viên và Quản lý Nhân Viên:



Hình 3-22 Giao diện danh sách Nhân viên và Quản lý Nhân Viên

3.2.2.9. Giao diện tạo mới nhân viên:

The screenshot displays the 'Sunshine Clothes' management dashboard. On the left is a sidebar with navigation links: Dashboard, Sản phẩm, Phân loại, Đơn hàng, Khách hàng, and Nhân viên (highlighted). The main content area is titled 'Nhân viên' and includes a search bar with the placeholder 'Tìm kiếm nhân viên...'. A modal window titled 'Đăng ký tài khoản' is open, featuring input fields for 'Tài khoản/Email', 'Tên', 'Số điện thoại', and 'Địa chỉ', followed by a blue 'LƯU' (Save) button. In the background, a table with a 'Tùy chọn' header and a 'Bin...' button is partially visible. The bottom right corner of the interface shows a pagination indicator '1-1 of 1'.

Hình 3-23 Giao diện tạo mới nhân viên

Chương 4. KẾT LUẬN

4.1. Kết quả đạt được

Qua bài báo cáo trên, nhóm đã thực hiện được việc áp dụng các kiến thức mà mình tìm hiểu được về ReactJs – SpringBoot để thiết kế một website bán hàng cơ bản mà vẫn đáp ứng được các chức năng thiết yếu.

Quá trình thiết kế cũng như tìm hiểu đã được ghi nhận lại cho các lập trình viên hay các bạn sinh viên muốn sử dụng tài liệu tham khảo cho các chủ đề liên quan.

4.2. Ưu điểm

Tài liệu tham khảo về các Frameworks như ReactJs, Springboot và Flask phong phú giúp nhóm dễ dàng tiếp cận hơn cho việc xây dựng Đồ Án.

Website bán hàng là một trong những web có nhiều người sử dụng cũng như thiết kế nên dễ tham khảo.

Website có giao diện đơn giản thân thiện với người dùng cũng như Responsive trên đa thiết bị.

Sử dụng mô hình kiến trúc Client-Sever thông qua API nên phần thiết kế cũng như công việc các thành viên Front – Back được tách biệt và dễ dàng xây dựng.

4.3. Nhược điểm

Tính năng cơ bản đã được đáp ứng đầy đủ nhưng còn thiếu các tính năng nâng cao khác như đánh giá, bình luận, thanh toán thẻ,...

Giao diện màu sắc tuy thân thiện nhưng vẫn còn đơn giản.

Mặt hàng còn chưa được phân loại nhỏ hơn và nhiều loại sản phẩm.

TÀI LIỆU THAM KHẢO

[1]: Tổng quan về HTML-CSS-Javascript: [Lập trình HTML, CSS và JavaScript trong việc xây dựng website \(codegym.vn\)](#).

[2]: Redux: https://www.youtube.com/watch?v=g_K1w8e0ILo&list=WL&index=3&t=2197s

[3]: Lập trình ReactJs, HTML-CSS-Javascript: <https://fullstack.edu.vn/>

[4]: Tài liệu ReactJs:

Giới thiệu ReactJs <https://viblo.asia/p/gioi-thieu-ve-reactjs-phan-i-cac-khai-niem-co-ban-V3m5WzjblO7>

Props và State: <https://viblo.asia/p/su-khac-nhau-giua-props-va-state-trong-reactjs-OeVKBvrJKkW>

ReactJs Docs: <https://reactjs.org/docs/getting-started.html>

Component Lifecycle: <https://viblo.asia/p/lifecycle-component-trong-reactjs-gGJ59jzxKX2>

ReactHook: <https://viblo.asia/p/cung-tim-hieu-ve-cac-hook-trong-react-hooks-Ljy5VYgjlra>

[5]: React-Router: <https://reactrouter.com/docs/en/v6>

[6] Tài liệu Spring Boot:

Trang tài liệu Spring : <https://spring.io/projects/spring-boot>

Video về Spring:

https://www.youtube.com/watch?v=O_XL9oQ1_To&t=947s

Hibernate: <https://topdev.vn/blog/hibernate-la-gi-sao-phai-dung-no-thay-jdbc/>

Spring Datal: [Giới thiệu về Spring Data JPA - Tại sao cần sử dụng nó? - Học Spring Boot \(hocspringboot.net\)](#)