|  |
| --- |
| BỘ GIÁO DỤC VÀ ĐÀO ĐẠO  TRƯỜNG ĐẠI HỌC CẦN THƠ  KHOA CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG  LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC  NGHÀNH CÔNG NGHỆ THÔNG TIN  ĐỀ TÀI  XÂY DỰNG WEBSITE THƯƠNG MẠI ĐIỆN TỬ BÁN GIÀY  TRÊN NỀN TẢNG FRAMEWORK EXPRESSJS  VÀ THƯ VIỆN REACTJS  Building an e-commerce website for selling shoes  based on Expressjs Framework and Reactjs library  Sinh viên: Đào Minh Khoa  Mã số: B1809248  Khóa: K44  *Cần Thơ, 05/2022*  BỘ GIÁO DỤC VÀ ĐÀO TẠO  TRƯỜNG ĐẠI HỌC CẦN THƠ  KHOA CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG  LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC  NGHÀNH CÔNG NGHỆ THÔNG TIN  ĐỀ TÀI  XÂY DỰNG WEBSITE THƯƠNG MẠI ĐIỆN TỬ BÁN GIÀY  TRÊN NỀN TẢNG FRAMEWORK EXPRESSJS  VÀ THƯ VIỆN REACTJS  Building an e-commerce website for selling shoes  based on Expressjs Framework and Reactjs library  Giáo viên hướng dẫn Sinh viên thực hiện  Ths. Nguyễn Minh Trung Đào Minh Khoa  B1809248  K44  *Cần Thơ, 05/2022* |

LỜI CẢM ƠN

Trước hết với tình cảm sâu sắc và chân thành nhất, em được bày tỏ lòng biết ơn đến các quý thầy cô giảng viên trường Đại học Cần Thơ đã tận tình giảng dạy và thầy cô trong khoa Công nghệ Thông tin và Truyền thông đã mang lại cho em những kiến thức vô cùng quý giá và bổ ích trong quá trình học tập.

Trong quá trình thực hiện và hoàn thiện luận văn tốt nghiệp, em xin chân thành cảm ơn thầy ThS. Nguyễn Minh Trung, bộ môn Công Nghệ Thông Tin, khoa Công Nghệ Thông Tin và Truyền Thông. Thầy đã luôn tận tình trực tiếp hướng dẫn, giúp đỡ em trong suốt quá trình nghiên cứu, giúp em tiếp cận được đề tài một cách tốt nhất, xây dựng và hoàn thành Luận văn này.

Trong quá trình thực hiện đề tài luận văn tốt nghiệp rất khó khăn nhưng bản thân em đã cố gắng hoàn thành nhiệm vụ đã thể hiện được tinh thần trách nhiệm và tự giác, song song với đó là sự hướng dẫn nhiệt tình của thầy Nguyễn Minh Trung. Vì kiến thức bản thân còn nhiều hạn chế, trong quá trình thực hiện đề tài này em không tránh khỏi những sai sót, kính mong nhận được những ý kiến đóng góp từ quý thầy cô.

Em xin chân thành cảm ơn!

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

*Cần Thơ, ngày tháng năm 2022*

(chữ ký của giảng viên)

NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

*Cần Thơ, ngày tháng năm 2022*

(chữ ký của giảng viên)

TÓM TẮT

Đề tài “Xây dựng website thương mại điện tử bán giày dựa trên nền tảng framework ExpressJS và thư viện ReactJS” được xây dựng đề phù hợp với nhu cầu hiện nay và cũng là quảng bá, mở rộng thị trường cho doanh nghiệp. Đề tài được xây dựng trên nền tảng framework ExpressJS và thư viện ReactJS, được quản lý bởi cơ sở dữ liệu MySQL và công cụ lập trình VS Code. Hệ thống được dùng cho 2 nhóm người dùng: khách hàng và quản lý, website giúp khách hàng mua sản phẩm dễ dàng và tiếp cận được nhiều sản phẩm nhanh chóng thông qua tìm kiếm bằng giọng nói, hỗ trợ khách hàng thông qua flugin messager của facebook. Website còn tích hợp cổng thanh toán trực tuyến VNPAY giúp cho việc thanh toán diễn ra dễ dàng, giảm được nhiều chi phí. Tiếp đến. website hỗ trợ thống kê, quản lý sản phẩm, quản lý hóa đơn, quản lý nhân viên, ... dành cho người quản lý, giúp việc quản lý dễ dàng và linh hoạt.

ABSTRACT

The topic "Building an e-commerce website selling shoes based on the ExpressJS framework and ReactJS library" was created to meet current needs while also promoting and expanding the business market. The theme is based on the ExpressJS framework and ReactJS library, with MySQL as the database and VS Code as the development tool. The system is utilized by two types of users: customers and managers. The website allows customers to buy things fast and easily via voice search, and provides customer care using Facebook's flugin messager. The website also includes the VNPAY online payment gateway, which makes payments simple and affordable. Next. Statistics, product management, invoice management, staff management, etc. are all supported by the website. Making management simple and flexible for managers.

# CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN

## Đặt vấn đề

Ngày nay, Công nghệ thông tin đã và đang được ứng dụng trong rất nhiều lĩnh vực kinh doanh, y tế, giáo dục, … Nổi bật trong đó là lĩnh vực Thương mại điện tử. Thời gian gần đây, thương mại điện tử đang phát triển vượt bậc bởi sự tiện dụng, giảm thiểu nhiều chi phí hơn so với phương thức bán hàng truyền thống. Mô hình này giúp khách hàng dễ dàng tiếp cận với các doanh nghiệp, Khách hàng có thể liên lạc, trao đổi, tìm hiểu thông tin, mua sắm tại nhà mà không cần đến cửa hàng. Đồng thời, doanh nghiệp cũng giảm bớt chi phí cho việc quảng cáo, thuê mặt bằng

Nắm bắt được tình hình đó, đề tài “Xây dựng website thương mại điện tử bán giày dựa trên nền tảng framework ExpressJS và thư viện ReactJS” đã được lên ý tưởng và phát triển thành một website hoàn thiện với các chức năng tiện ích, giúp khách hàng tìm kiếm, xem thông tin sản phẩm dễ dàng, theo dõi đơn hàng, thanh toán nhanh chóng. Giúp người quản lý nắm bắt được nhu cầu mua sắm của khách hàng, quản lý kho hàng, thống kê chi phí doanh thu chính xác.

## 1.2 Mục tiêu đề tài

Website bán giày giúp cho doanh nghiệp hay các chủ cửa hàng giới thiệu và bán sản phẩm một cách nhanh chóng. Giúp người dùng tiếp cận các mặt hàng chỉ với thao tác đơn giản trên máy tính có kết nối internet mà không mất nhiều thời gian hay phải đến trực tiếp cửa hàng.

## 1.3 Đối tượng nghiên cứu

Đối tượng nghiên cứu chính của đề tài là sử dụng ReactJS và ExpressJS để xây dựng website bán quần áo. Ngoài ra, cũng sẽ nghiên cứu thêm về việc xây dựng một kiến trúc hệ thống ứng dụng sử dụng máy chủ dữ liệu (API server) và công cụ quản lý (Web client). Đối tượng nghiên cứu đầy đủ sẽ là một hệ thống gồm 2 thành phần:

- REST API server: xây dựng một server cung cấp các thao tác tạo, thêm, sửa, xóa (CRUD) dữ liệu của hệ thống thông qua HTTP request.

- Web client: xây dựng một ứng dụng web giao tiếp với server thông qua REST API, nhằm mục đích quản lý dữ liệu cốt lõi của hệ thống.

## 1.4 Phạm vi nghiên cứu

Phạm vi nghiên cứu của đề tài chủ yếu là làm việc với React JS để tạo ra website bán giày. Ngoài ra, sẽ sử dụng thư viện UI (User Interface) của React JS là Material UI để xây dựng giao diện website, Axios để giao tiếp với API server, phía backend sẽ sử dụng NodeJS với framework ExpressJS để xây dựng nhanh hệ thống RESTful API

## 1.5 Phương pháp nghiên cứu

Nội dung nghiên cứu khá phổ biến, do đó phương pháp nghiên cứu chủ yếu là tìm hiểu thông qua tài liệu trang chủ của các thư viện, thông qua các video trên YouTube và làm các ví dụ trực tiếp. Từ đó, sẽ vận dụng các kiến thức đã tiếp thu được vào thực hiện đề tài. Tương ứng với mỗi thành phần của hệ thống sẽ có những phương pháp nghiên cứu khác nhau: API server: Nghiên cứu cơ bản về NodeJS với framework Express để xây dựng một hệ thống RESTful và hệ quản trị cơ sở dữ liệu MySQL. Web client: Nghiên cứu cơ bản về ứng dụng web sử dụng các thư viện react, redux, axios, tích hợp cổng thanh toán VNPAY và material-ui.

## 1.6 Nội dung nghiên cứu

|  |  |  |
| --- | --- | --- |
| STT | Đối tượng | Nội dung |
| 1 | API và RESTful API | - Tìm hiểu về API và RESTful API |
| 2 | NodeJS – ExpressJS | - Tìm hiểu NodeJS với ExpressJS |
| 3 | MVC | - Tìm hiểu mô hình MVC (model-view-controller) |
| 4 | MySQL | - Tìm hiểu hệ quản trị cơ sở dữ liệu MySQL  Cấu hình MySQL trong NodeJS |
| 5 | ReactJS | - Tìm hiểu ReactJS   * Components * State * Props   - Sử dụng webhook  - Sử dụng routing bằng react-router-dom v6 và chuyển hướng đăng nhập. |
| 6 | Redux | - Tìm hiểu về Redux   * State * Action * Container * Dispatcher   - Sử dụng redux vào website |
| 7 | Axios | - Tìm hiểu về axios.  - Trao đổi với API server thông qua axios. |
| 8 | Thư viện UI Material | - Tìm hiểu về material và các components thư viện cung cấp.  - Xây dựng các components, kết hợp với redux để chia sẻ các state dùng chung. |

*Bảng 1: Nội dung nghiên cứu*

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

## 2.1 Giới thiệu về API và RESTful API

API (Application programming interface) là những quy tắc tương tác để các ứng dụng hoặc thành phần của ứng dụng có thể giao tiếp được với nhau. Mục đích của API là để đơn giản hóa việc lập trình bằng cách trừu tượng hóa đi những cài đặt phức tạp và chỉ phơi bày ra các cách thức để thực hiện hành động mà lập trình viên cần. Trong ngữ cảnh lập trình web, API là cách thức để ứng dụng trao đổi dữ liệu với một dịch vụ trực tuyến. API sẽ cung cấp các chức năng dịch vụ thông qua đường dẫn URL (Uniform Resource Locator) với định nghĩa thông số kỹ thuật như gửi request bằng HTTP (Hypertext Transfer Protocol) và trả về response message theo định dạng XML (Extensible Markup Language) hoặc JSON (JavaScript Object Notation).

REST (Representational State Transfer) là một kiến trúc phần mềm bao gồm các quy tắc để tạo ra dịch vụ web (webservice). Một webservice tuân thủ theo kiến trúc REST thì được gọi là RESTful webservice. Webservice này sử dụng giao thức HTTP để triển khai kiến trúc web. Như vậy, RESTful API chính là kiến trúc thiết kế API tuân thủ theo kiến trúc REST thông qua các phương thức của HTTP (GET, POST, PUT, DELETE,..). Tương ứng với mỗi phương thức HTTP sẽ thực hiện những tác vụ tương ứng:

|  |  |
| --- | --- |
| Phương thức HTTP | Tác vụ |
| GET | Lấy dữ liệu |
| POST | Tạo mới dữ liệu |
| PUT | Cập nhật dữ liệu |
| DELETE | Xóa dữ liệu |

Các tác vụ đọc, tạo, cập nhật, xóa được gọi là CRUD service (Create, Read, Update, Delete). Mỗi tác vụ trên phải được gọi thông qua địa chỉ URI (Uniform Resource Identifier) kèm theo phương thức và payload (có thể có hoặc không, thường là định dạng XML hoặc JSON).



*Nguồn:* [*https://www.rlogical.com/blog/what-is-rest-api-vs-web-api-vs-soap-api*](https://www.rlogical.com/blog/what-is-rest-api-vs-web-api-vs-soap-api)

RESTful API sử dụng giao thức stateless (là một giao thức truyền thông không sử dụng session) và theo tiêu chuẩn nên hệ thống sẽ nhanh, đáng tin cậy và có thể mở rộng dễ dàng. Thông thường, RESTful API sẽ xác thực người dùng khi gửi yêu cầu đối với những tác vụ nguy hiểm như cập nhật hoặc xóa dữ liệu hoặc chỉ cho phép đối với người quản trị.

## 2.2 ReactJS



Nguồn: https://200lab.io/blog/reactjs-la-gi

ReactJS (hay React, React.js) là một thư viện mã nguồn mở (MIT License) dùng để xây dựng giao diện người dùng được viết bằng JavaScript. React được phát hành lần đầu tiên vào năm 2013 bởi Facebook. React hiện nay được duy trì bởi Facebook và cộng đồng lập trình viên. React được sử dụng bởi những doanh nghiệp công nghệ hàng đầu như Facebook, Twitter, Instagram.

Sức mạnh của React nằm ở việc tự động cập nhật lại UI (user interfaces) khi dữ liệu thay đổi với chi phí ít nhất mà không cần phải tải lại toàn bộ trang web. Điều này làm cho ứng dụng phản hồi lại nhanh hơn, tiết kiệm băng thông, tăng tốc ứng dụng và tăng trải nghiệm người dùng. Trong React sẽ có một số khái niệm cơ bản về JSX, components, props, state và hook.

### 2.2.1 JavaScript XML

JSX (JavaScript XML) là một cú pháp mở rộng của JavaScript cho phép định nghĩa các thành phần HTML trong React. Nói một cách đơn giản, cú pháp này cho phép ta viết các HTML tag trực tiếp trong JavaScript.

Trên thực tế, các trình duyệt ngay cả các trình duyệt mới nhất cũng không hỗ trợ cú pháp của JSX. Do đó mã nguồn sử dụng JSX cần được chuyển về JavaScript thông qua một thư viện có tên là Babel (một JavaScript compiler).

### 2.2.2 Components

Components là những thành phần UI được chia nhỏ ra, độc lập và có thể tái sử dụng. Component có thể là những function (stateless) hoặc class (stateful) trong JS.

Functional component: là một hàm Javascript (hoặc ES6) trả về một React element. Theo tài liệu chính thức của React, hàm dưới đây là một component hợp lệ.



Function này là một component React hợp lệ vì nó nhận một "props" làm tham số và trả về 1 React element.

Hoặc theo ES6 arrow function:



Functional component cũng được biết tới với cái tên là stateless components. Bởi vì chúng không thể làm nhiều thứ phức tạp như quản lý React State (data) hoặc xử lý vấn đề liên quan tới life-cycle trong functional components. Tuy nhiên, từ phiên bản React 16.8, nhà phát hành giới thiệu tính năng React Hooks. Với Hooks, chúng ta có thể sử dụng state và những features khác trong functional components.

Class components: Các Class components là những class ES6. Chúng phức tạp hơn functional components.

Class components còn có:

- Phương thức khởi tạo, có hàm về vòng đời component, hàm render().

- State (dữ liệu ứng dụng).



- Tóm lại, một class components là một class kế thừa từ React.Component, có thể nhận props (trong hàm khởi tạo) nếu cần.

### 2.2.3 Props và State

Props là những thuộc tính được truyền vào một component và chỉ có thể đọc. Ví dụ như thẻ Post có thuộc tính được truyền vào là title. Truy xuất bằng cú pháp props.title sẽ cho giá trị là “My Post Title”.



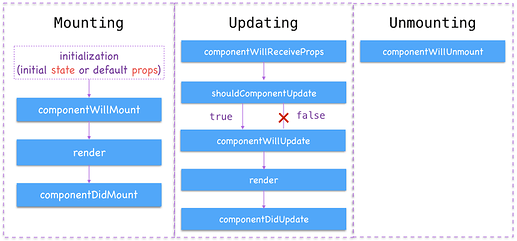
State là trạng thái thuộc về chính component đó, được quản lý bởi chính nó và không được truy xuất từ bên ngoài. Chỉ có thể sử dụng state khi dùng stateful component.

### 2.2.4 LifeCycle

LifeCycle là một vòng đời của một React component từ lúc được render lần đầu tiên và mỗi lần render lại (mounting) và khi gỡ bỏ component (unmounting).

Có 2 phương thức được tự động gọi khi sự kiện mounting (component được render lần đầu tiên hoặc được render lại) và unmounting (component bị gỡ bỏ) xảy ra lần lượt là componentDidMount và componentWillUnmount.

Có thể ghi đè 2 phương thức này khi sử dụng stateful component (class).



Nguồn:<https://stories.magestore.com/t/tim-hi-u-v-lifecycle-c-a-class-trong-react-js/1427>

### 2.2.5 Hook

Hook là tính năng được thêm vào React ở phiên bản 16.8. Cho phép sử dụng một số tính năng chỉ có ở stateful component (class) khi dùng stateless component (function) như state (useState), life cycle (useEffect), ...

## 2.3 Thư viện Redux



Nguồn: <https://redux.js.org/>

Redux là một thư viện JavaScript mã nguồn mở (MIT License) dùng để quản lý state của ứng dụng, được sử dụng phổ biến với React hoặc Angular để xây dựng giao diện người dùng. Redux được tạo bởi Dan Abramov và Andrew Clark vào năm 2015.

State trong redux sẽ có các tính chất sau:

− Từ một nguồn duy nhất: state của ứng dụng được là một state toàn cục được lưu trữ thành một object tree.

− State chỉ có thể đọc: state không thể thay đổi trực tiếp mà chỉ có thể phát động bằng một action (object) mô tả những gì đã xảy ra.

− State được thay đổi bởi những reducer (JS function): khi muốn thay đổi state sẽ phát động ra một action, reducer sẽ tiếp nhận và xử lý thay đổi state tương 19 ứng. Vì reducer chỉ là những hàm js bình thường, có thể truyền vào dữ liệu, gọi theo thứ tự hoặc tái sử dụng các reducer.

Redux được thiết kế cho việc quản lý tập trung, chia sẻ state giữa các component trở nên đơn giản, nhất quán. Để sử dụng redux với react cần phải nắm một số khái niệm cơ bản sau:

− Action: là những thông tin payload được dùng để gửi dữ liệu đến store. Hiểu đơn giản, action chính là một event được gửi lên store kèm theo dữ liệu và loại action sẽ được thực hiện.

− Reducer: là những hàm thuần túy thực hiện việc nhận action được gửi lên và lấy state hiện tại của ứng dụng, xử lý và trả về một state mới. Global state chính là một object tree lưu trữ toàn bộ state của ứng dụng, action gửi lên sẽ mô tả những gì xảy ra. Khi đó, reducer sẽ tiếp nhận action và lấy một nhánh state nhỏ tương ứng từ global state, sau đó tạo ra một nhánh state mới dựa trên action và cập nhật vào global state.

− Store: là một object kết nối action và reducer lại với nhau, chịu trách nhiệm lưu giữ state của ứng dụng, cho phép truy xuất và cập nhật state thông qua dispatch (là một dịch vụ của store cho phép gửi action thông qua nó).

## 2.4 Thư viện axios

Axios là một thư viện HTTP client dựa trên Promise dùng để gửi các request HTTP bất đồng bộ đến REST endpoint để sử dụng các dịch vụ CRUD.

Promise là một cải tiến để loại bỏ try catch, callback rườm rà trong xử lý bất đồng bộ, thay vào đó là then – catch. Hiểu đơn giản là khi thực hiện một tác vụ bất đồng bộ, sau khi thực hiện thành công thì “then” sẽ được gọi, ngược lại khi có lỗi phát sinh thì “catch” sẽ được gọi. Tương tự như promise, axios cũng có thể sử dụng theo kiểu gửi request lồng nhau.

Ngoài ra, axios còn hỗ trợ interceptor dùng để thực hiện một số công việc khác trước khi gửi request hoặc ngay khi nhận được response. Ví dụ như tạo mới access token khi nhận được response thông báo lỗi “401 – Unauthorized”.

## 2.5 Thư viện React UI – Material UI

Material UI là một thư viện các React Component đã được tích hợp thêm cả Google's Material Design. Các component trong Material UI là những React component, hỗ trợ hầu hết các thành phần cốt yếu để tạo dựng nên một trang web hoàn chỉnh. Khi sử dụng Material UI thì hầu như không cần phải cài thêm một thư viện UI bổ sung nào nữa thì cũng đủ để đáp ứng nhu cầu của dự án đưa ra.

Dưới đây là danh sách các component mà Material UI hỗ trợ:

Layout: Box, Container, Gird, Grid List, Hidden.

Input: Button, Button Group, Checkbox, Floating Action Button, Date/Time, Radio, Select, Slider, Switch, TextField, Transfer List.

Navigation: Bottom Navigation, Breadcrum, Drawer, Link, Menu, Stepper, Tabs.

Surfaces: App bar, Paper, Card, Accordion.

Feedback: Progress, Dialog, Snackbar, Backdrop.

Data Display: Avatar, Badge, Chip, Divider, Icons, Material Icons, List, Table, Tooltip, Typography.

Utils: Click Away Listener, CSS Baseline, Modal, No SSR, Popover, Popper, Portal, Textarea Autosize, Transition, useMediaQuery.

Lab: Alert, Autocomplete, Data Grid, Pagination, Rating, Skeleton, Speed Dial, Timeline, Toggle Button, Tree View.

## 2.6 Cơ sở dữ liệu MySQL



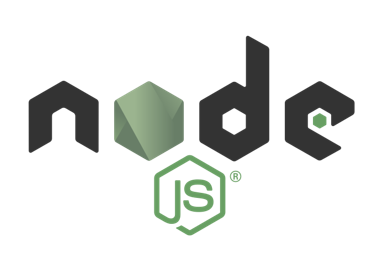
Nguồn: <https://vi.cleanpng.com/png-gwfs4v/>

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở miễn phí, được những doanh nghiệp lớn nhất thế giới như Facebook, Google, Adobe sử dụng để tiết kiệm thời gian và chi phí cho việc quản lý dữ liệu hiệu quả. MySQL được sáng lập bởi Michael "Monty" Widenius và David Axmark năm 1995. Hiện nay được phát triển bởi tập đoàn Oracle.

MySQL hỗ trợ trên nhiều hệ điều hành: Windows, Linux, MacOS,...

MySQL cũng được công nhận bởi DB-Engines là hệ quản trị cơ sở dữ liệu của năm 2019.

## 2.7 NodeJS



Nguồn: <https://nodejs.org/en/about/resources/>

NodeJS là một nền tảng được xây dựng trên V8 JavaScript Engine – trình thông dịch thực thi mã JavaScript, giúp xây dựng các ứng dụng web một cách đơn giản và dễ dàng mở rộng.

NodeJS được phát triển bởi Ryan Dahl vào năm 2009 và có thể chạy trên nhiều hệ điều hành khác nhau: OS X, Microsoft Windows, Linux.

### 2.7.1 Blocking I/O và Nonblocking I/O

I/O là quá trình giao tiếp (lấy dữ liệu vào, trả dữ liệu ra) giữa một hệ thống thông tin và môi trường bên ngoài. Với CPU, thậm chí mọi giao tiếp dữ liệu với bên ngoài cấu trúc chip như việc nhập/ xuất dữ liệu với memory (RAM) cũng là tác vụ I/O.

Blocking I/O: Yêu cầu thực thi một IO operation, sau khi hoàn thành thì trả kết quả lại. Process/Theard gọi bị block cho đến khi có kết quả trả về hoặc xảy ra ngoại lệ.

Nonblocking I/O: Yêu cầu thực thi IO operation và trả về ngay lập tức (timeout = 0). Nếu operation chưa sẵn sàng để thực hiện thì thử lại sau. Tương đương với kiểm tra IO operation có sẵn sàng ngay hay không, nếu có thì thực hiện và trả về, nếu không thì thông báo thử lại sau.

### 2.7.2 Synchronous và Asynchronous

Synchronous: Các sự kiện diễn ra theo thứ tự. Một sự kiện chỉ được bắt đầu khi sự kiện trước kết thúc.

Asynchronous: Không theo thứ tự, các hành động có thể xảy ra đồng thời hoặc chí ít, mặc dù các hành động bắt đầu theo thứ tự nhưng kết thúc thì không. Một hành động có thể bắt đầu (và thậm chí kết thúc) trước khi hành động trước đó hoàn thành.

Libuv – một thư viện multi-platform hỗ trợ asynchronous I/O giúp Nodejs mang lại cơ chế Asynchronous Event-Driven Non-Blocking I/O.

### 2.7.3 Callback

Callback là một hàm sẽ được thực hiện sau khi một hàm khác đã thực hiện xong.

Trong JavaScript, hàm là đối tượng. Do đó, các hàm có thể lấy các hàm làm đối số và có thể được trả về bởi các hàm khác. Các hàm thực hiện điều này được gọi là higher - order function (Hàm bậc cao hơn). Bất kỳ hàm nào được truyền dưới dạng đối số được gọi là hàm callback.

Javascript là một ngôn ngữ lập trình hướng sự kiện và bất đồng bộ nên callback function đóng vai trò rất quan trọng.

ES6 định nghĩa 3 state cho một lời gọi hàm không đồng bộ:

* Pending: hàm đang được thực hiện và chưa trả về kết quả. Trong lúc này, nếu cố tình console.log biến kết quả sẽ nhận được output < pending > .
* Fulfilled: hàm đã thực hiện xong – thành công và trả về kết quả.
* Rejected: hàm đã thực hiện xong – không thành công. Thường thì sẽ bắt exception tại bước này.

Máy chủ NodeJS có thể nhận rất nhiều các yêu cầu (request) từ rất nhiều người dùng. Vì vậy để nâng cao khả năng phục vụ, tất cả các API của NodeJS được thiết kế hỗ trợ Callback.

### 2.7.4 Event Loop

NodeJS là một ứng dụng đơn luồng (Single Thread), nó hoạt động phía trên một nền tảng được viết bởi C++, nền tảng này sử dụng đa luồng (Multi-Thread) để thực hiện đồng thời các nhiệm vụ.

Mỗi yêu cầu (request) từ phía người dùng được NodeJS coi là một sự kiện (event), chúng được đặt vào một Event Queue (Hàng đợi sự kiện). NodeJS sử dụng quy tắc FIFO (First In First Out), điều này có nghĩa là những yêu cầu đến trước sẽ được xử lý trước.

Event Loop là một vòng lặp vô tận, nó sẽ chuyển các yêu cầu sang Thread Pool (Bể chứa các luồng), đồng thời mỗi yêu cầu sẽ được đăng ký một hàm Callback. Khi một yêu cầu được xử lý xong, hàm Callback tương ứng sẽ được gọi thực thi.

Thread Pool là một chương trình viết bằng ngôn ngữ C++, nó hỗ trợ đa luồng (Multi Threads), chính vì vậy tại đây các yêu cầu sẽ được xử lý trên các luồng khác nhau. NodeJS cũng hỗ trợ đa tiến trình (Multi Processes), điều này có nghĩa là chúng có thể được thực thi trên các lõi (Core) khác nhau.

Khi một yêu cầu được xử lý xong, NodeJS sẽ gọi hàm Callback (Đã được đăng ký cho yêu cầu này) để thực thi nó.

Nếu mỗi kết nối tới Server đều mở ra một luồng (Thread) sẽ rất tốn bộ nhớ. Điều này đã được chứng thực khi so sánh Apache và Nginx (Hai Web Server triển khai các ứng dụng PHP). Apache đã tiêu tốn bộ nhớ hơn rất nhiều so với Nginx.

NodeJS giống với Nginx là chúng chỉ sử dụng một luồng đơn (Single thread) để đón tiếp các kết nối từ phía người dùng, và coi mỗi yêu cầu của người dùng là một sự kiện.

Các hoạt động I/O rất tốn tài nguyên của hệ thống, vì vậy NodeJS quản lý chặt chẽ việc sử dụng các hoạt động I/O. Vì vậy chỉ cần sử dụng Callback khi bạn thực thi các nhiệm vụ liên quan tới I/O.

### 2.7.5 Module

Node.js sử dụng kiến trúc Module để đơn giản hóa việc tạo ra các ứng dụng phức tạp. Module là giống như các thư viện trong C, C#, Java, … Mỗi module chứa một tập các hàm chức năng có liên quan đến một "đối tượng" của Module.

Các module được giữ tách biệt riêng với nhau, tách riêng với code base, khi nào cần sử dụng những cái nào thì gọi chúng ra và kết hợp lại với nhau tùy logic xử lý của bạn.

### 2.7.6 NPM - Node Package Manager

NPM viết tắt của Node Package Manager là một công cụ tạo và quản lý các thư viện lập trình Javascript cho NodeJS. Trong cộng đồng Javascript, các lập trình viên chia sẻ hàng trăm nghìn các thư viện với các đoạn code đã thực hiện sẵn một chức năng nào đó. Nó giúp cho các dự án mới tránh phải viết lại các thành phần cơ bản, các thư viện lập trình hay thậm chí cả các framework.

Các loại package

Dựa theo chức năng mà ta chia package ra làm 2 loại, đó là Simple dependencies và Development dependencies.

Simple dependencies là những package bắt buộc phải có trong quá trình chạy sản phẩm. Khi cài đặt Simple dependencies, Npm sẽ tự động cài đặt tất cả các dependencies cần thiết.

Development dependencies là những package bắt buộc khi phát triển cũng như phát hành sản phẩm. Khi cài đặt Development dependencies, Npm sẽ chỉ cài đặt các dependencies cần thiết.

## 2.8 ExpressJS Framework



Nguồn: <https://itnavi.com.vn/blog/expressjs-la-gi/?amp>

Expressjs là một framework được xây dựng trên nền tảng của Nodejs. Nó cung cấp các tính năng mạnh mẽ để phát triển web hoặc mobile. Expressjs hỗ trợ các method HTTP và middleware tạo ra API vô cùng mạnh mẽ và dễ sử dụng.

Định tuyến (Routing)

Routing trong Node.js là một khái niệm nói đến việc xác định ứng dụng sẽ đáp ứng như thế nào khi người dùng tạo một request đến một endpoint (Điểm cuối) cụ thể nào đó. Điểm cuối đó thường là một URI hoặc một đường dẫn (Path) với một Request method (POST, PUT, GET, ... ) cụ thể.

Cấu trúc định tuyến cơ bản:

app.METHOD (Path, Handler...)

Trong đó:

* app : là một instance của express
* METHOD: là một HTTP Method

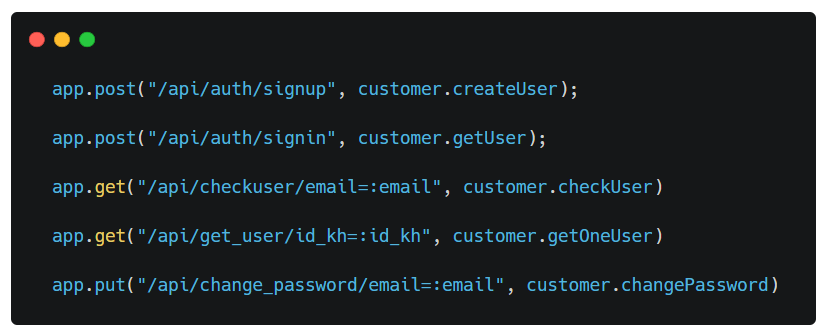
Express hỗ trợ rất nhiều loại HTTP methods khác nhau, bao gồm: get, post, put, head, delete, options, trace, copy, lock, mkcol, move, purge, unlock, report, mkactivity, checkout, merge, m-search, notify, subscribe, unsubscribe, patch.

* Path: là một đường dẫn trên máy chủ

Route path có thể là một chuỗi thông thường (String) hoặc là một chuỗi có ký hiệu biểu thức chính quy (string patterns) hoặc là một biểu thức chính quy (regular expressions)

* Handler : là một function sẽ thực thi khi một route được trùng khớp

Đơn giản là một hoặc nhiều function sẽ được gọi khi một route trùng khớp để đáp ứng một yêu cầu nào đó. Lưu ý các handler sẽ được gọi đúng theo thứ tự truyền vào.



Route parameters

Route parameters là những vị trí trên URL được đánh dấu bằng cách đặt tên, mục đích là để lấy ra các giá trị tương ứng. Tất cả cá giá trị đối số sẽ được đặt vào đối tượng req trong thuộc tính params. Với tên thuộc tính trùng khớp với từ khóa được xác định trên URL.

Ví dụ, chúng ta định nghĩa một path là /api/get\_user/id\_kh=:id\_kh. Thì ở đây :id\_kh chính là một route param. Khi đó nếu người dùng truy cập đường dẫn như là /api/get\_user/id\_kh=KH01 ta lấy ra được :id\_kh = KH01 và giá trị này sẽ nằm ở req.params.id\_kh.

Response methods

Sau việc tiếp nhận và xử lý, thì việc tiếp theo đó là đáp ứng (Response). Trong express định nghĩa sẵn một số phương thức hỗ trợ hay dùng nhất là:

|  |  |
| --- | --- |
| Tên phương thức | Ý nghĩa |
| res.json() | Trả về một dữ liệu dạng JSON |
| res.redirect() | Chuyển hướng đến một đường dẫn nào đó |
| res.render() | Trả về một view template |
| res.send() | Gửi dữ liệu dạng text |

# CHƯƠNG 3: NỘI DUNG

## 3.1 Đặc tả

Website có nhiệm vụ là giới thiệu sản phẩm, chức năng đặt hàng và theo dõi đơn hàng thông minh với giao diện thân thiện, dễ sử dụng.

Người quản trị sẽ thêm các sản phẩm lên website với đầy đủ các thông tin về sản phẩm như: tên sản phẩm, thông tin sản phẩm, kích thước, màu sắc, thương hiệu, loại sản phẩm, giá bán, hình ảnh sản phẩm. Khi khách hàng truy cập vào website của cửa hàng, khách hàng có thể lựa chọn hoặc tìm kiếm sản phẩm mình cần mua. Khách hàng muốn mua ngay hoặc thêm sản phẩm vào giỏ hàng để thanh toán sau, khách hàng cần phải đăng nhập để thêm sản phẩm vào giỏ hàng và thanh toán sản phẩm, khách hàng có thể đăng ký tài khoản nếu chưa có tài khoản.

Sau khi khách hàng thêm các sản phẩm cần mua vào giỏ hàng, khách hàng có thể vào lại giỏ hàng để xem lại các mặt hàng, nếu có sản phẩm nào không mua thì khách hàng có thể xóa sản phẩm đó khỏi giỏ hàng.

Khi khách hàng đã lựa chọn xong, khách hàng sẽ tiến hành đặt hàng và thanh toán qua hai hình thức là thanh toán khi nhận hàng và thanh toán trực tuyến qua VNPAY. Khách hàng có thể hủy đơn hàng nếu quản lý chưa xác nhận đơn hàng. Khi khách hàng gửi yêu cầu đặt hàng hệ thống sẽ kiểm tra thông tin khách hàng có hợp lệ hay không, nếu không hợp lệ người quản trị sẽ liên hệ lại khách hàng hoặc hủy đơn hàng đó. Ngược lại, thông tin đơn hàng sẽ được cập nhật trong danh sách đơn hàng của khách hàng, khách hàng cũng có thể theo dõi đơn hàng thông qua thanh trang thái của đơn hàng đó. Nếu khách hàng cần thêm thông tin thì có thể nhắn tin qua flugin messager.

Quản trị viên sẽ tương tác trên giao diện dành cho quản trị viên. Hệ thống cho phép quản trị viên quản lý được số lượng sản phẩm được nhập vào, thêm sản phẩm, sửa và xóa sản phẩm. Nhận thông tin khách hàng mua sản phẩm, quản lý danh sách khách hàng, thông tin hóa đơn của khách hàng, thống kê sản số hóa đơn, doanh thu và sản phẩm bán chạy nhất.

## 3.2 Các chức năng chính của hệ thống

1. Khách hàng

* Khách hàng chưa đăng ký tài khoản: có thể truy cập website để đăng ký tài khoản, tìm kiếm sản phẩm bằng giọng nói, lọc sản phẩm, xem thông tin của sản phẩm và có thể nhắn tin với cửa hàng để được hỗ trợ, … Nhưng không thể thêm sản phẩm vào giỏ hàng và mua sản phẩm.
* Khách hàng đã có tài khoản: có thể truy cập website và đăng nhập, nếu có tài khoản khách hàng sẽ có tất cả các chức năng của người chưa có tài khoản, thêm sản phẩm vào giỏ hàng và thanh toán được sản phẩm. việc thanh toán sản phẩm sẽ được thực hiện thông qua 2 hình thức: thanh toán khi nhận sản phẩm và thanh toán trực tuyến qua VNPAY.

1. Quản lý

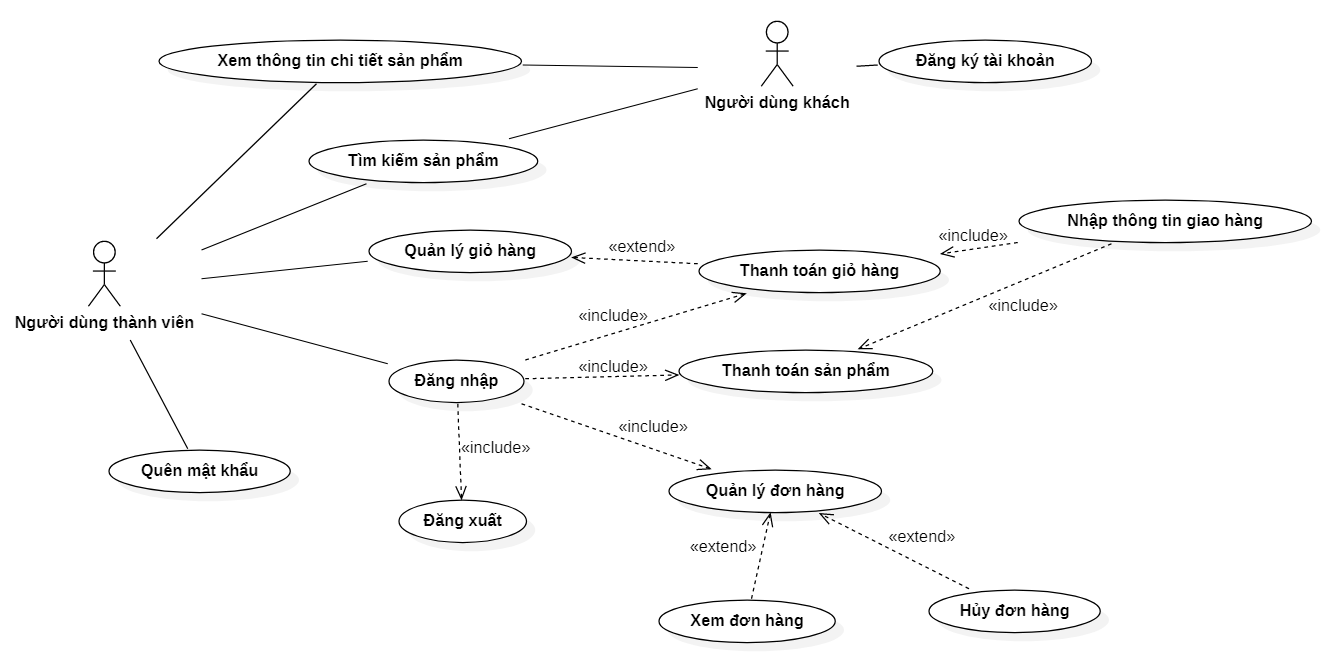
* Người quản lý sẽ nắm tất cả các quyền trong hệ thống như xem thống kê doanh thu, thêm sửa xóa và khuyễn mãi sản phẩm, nhập hóa đơn, thêm sửa và xóa nhân viên, phân quyền cho nhân viên, thêm các danh mục, xác nhận và hủy hóa đơn. Ngoài ra, quản lý có thể phân quyền cho nhân viên phụ trách với các chức vụ khác nhau:
  + Nhân viên: xem đơn hàng.
  + Nhân viên kho: thêm sản phẩm mới và nhập hóa đơn.
  + Giao hàng: chấp nhận đơn hàng hoặc không nhận đơn hàng đó.

## 3.3 Môi trường vận hành

* Website hoạt động tốt trên trình duyệt Chrome, Edge, …
* Chương trình tạo máy chủ Web (Web server) XAMPP.
* Yêu cầu kết nối Internet.

## 3.4 Usecase diagram

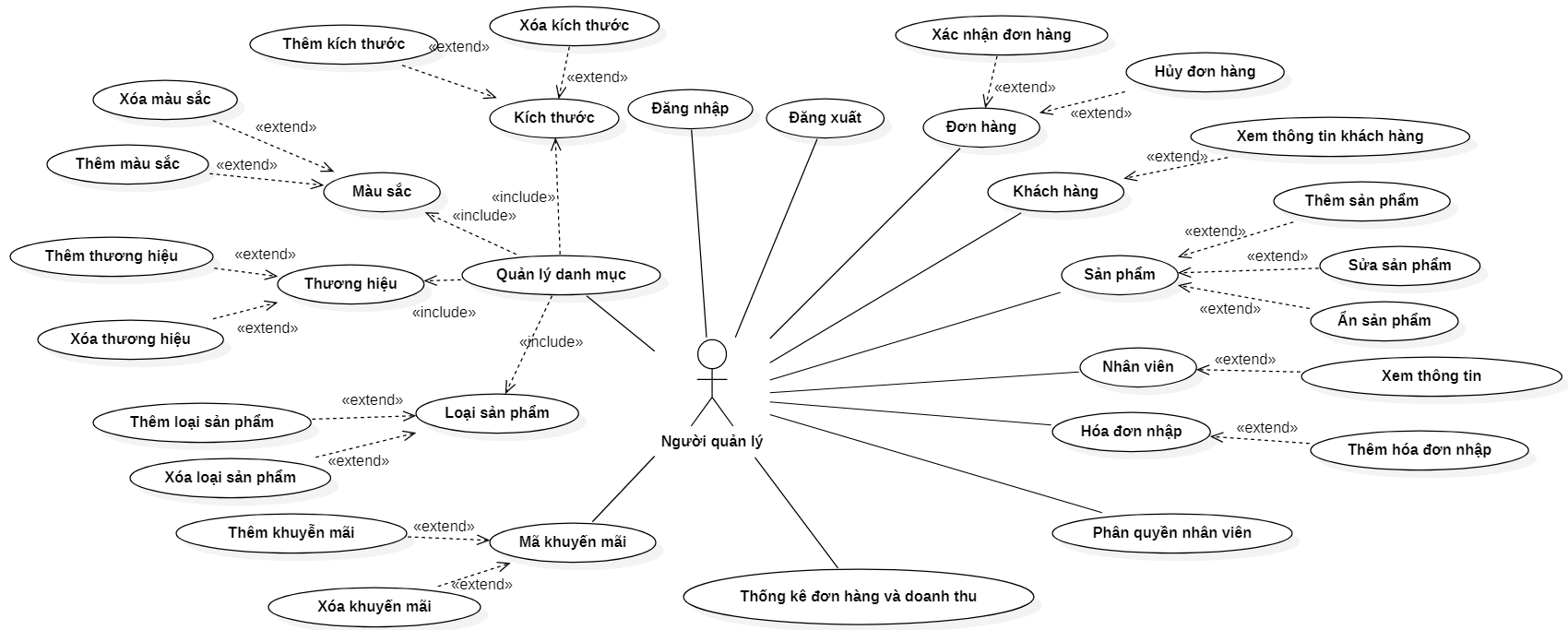
### 3.4.1 Usecase diagram người dùng thành viên và người dùng khách



Người dùng khách (khách hàng không có tài khoản): khách hàng có thể thực hiện các chức năng sau: Đăng ký tài khoản, xem thông tin chi tiết sản phẩm, tìm kiếm sản phẩm.

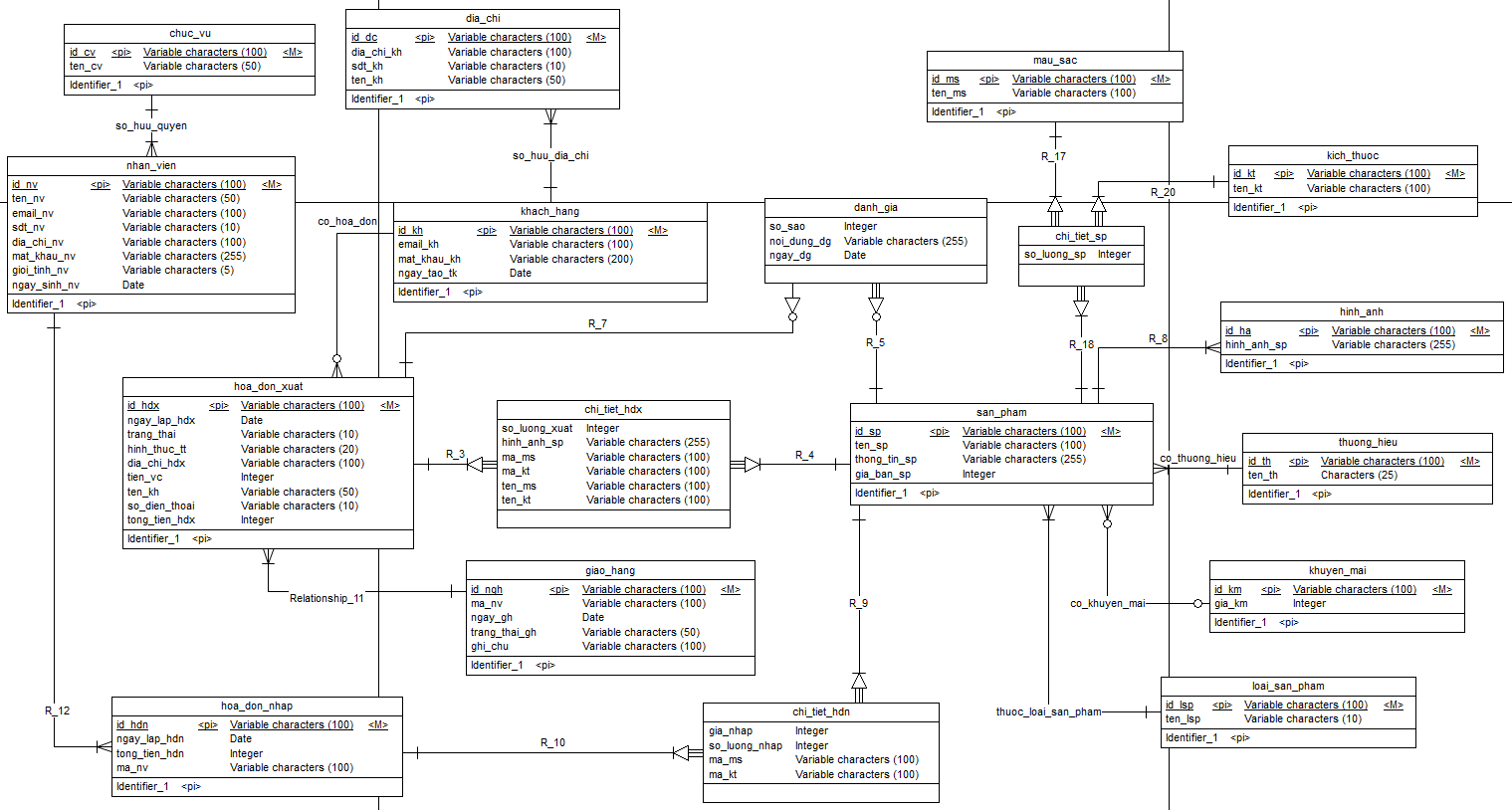
Người dùng thành viên (khách hàng đã có tài khoản): khách hàng có thể thực hiện các chức năng sau: Đăng nhập tài khoản, xem thông tin chi tiết sản phẩm, tìm kiếm sản phẩm, thêm sản phẩm vào giỏ hàng, thanh toán sản phẩm, thanh toán giỏ hàng, thêm thông tin giao hàng, xem lịch sử đơn hàng và hủy đơn hàng khi quản lý chưa xác nhận đơn hàng.

### 3.4.2 Usecase diagram quản lý

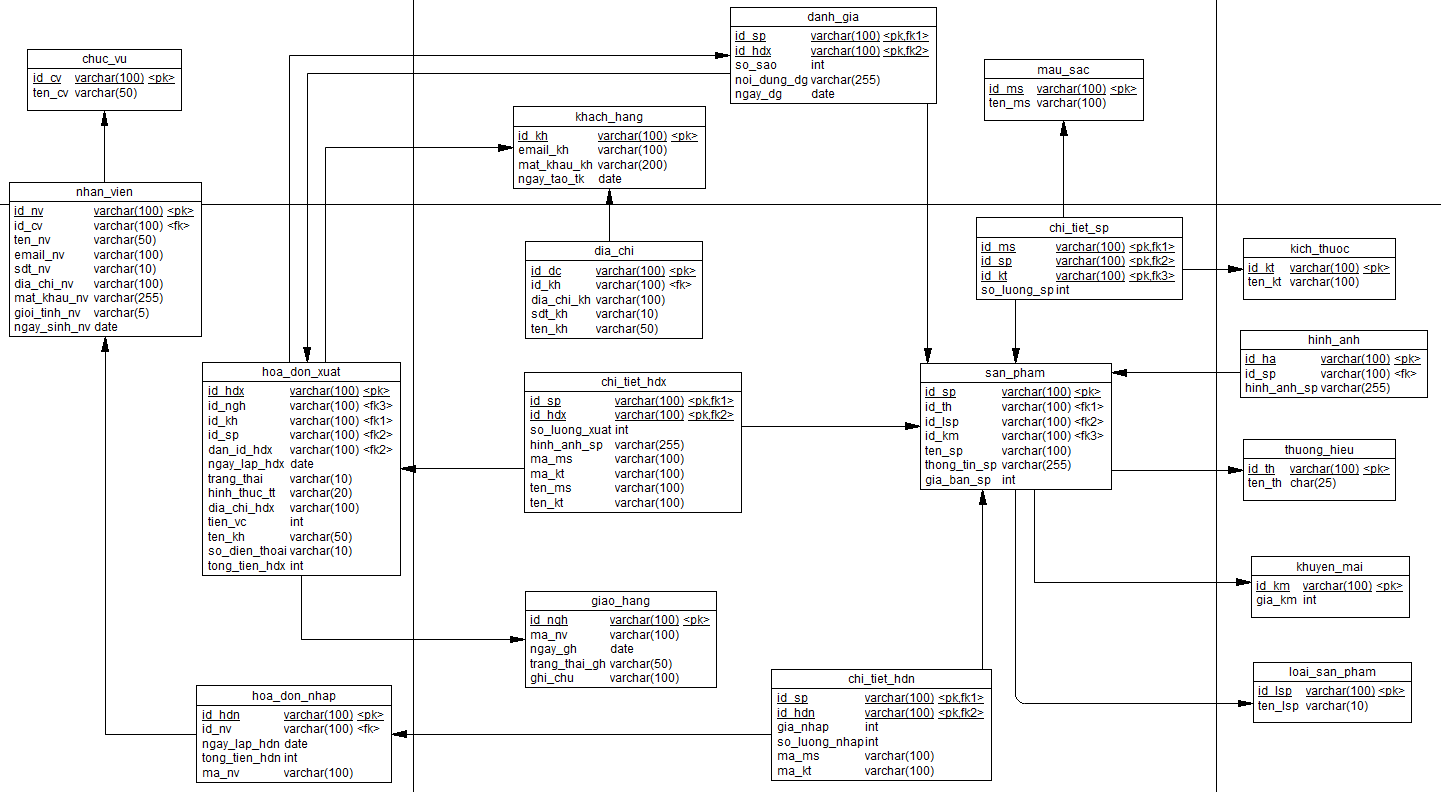


Sơ đồ usecase của quản lý. Trong đó, quản lý có toàn quyền sử dụng các chức năng của hệ thống. quản lý có thể đăng nhập, đăng xuất vào hệ thống. quản lý các danh mục màu sắc, kích thước, loại sản phẩm, mã khuyễn mãi, thông tin đơn hàng, hóa đơn nhập, thông tin khách hàng, thông tin sản phẩm, nhân viên và phân quyền nhân viên. Ngoài ra, quản lý còn xem được thống kê doanh thu của cửa hàng, lợi nhuận, số lượng sản phẩm tồn kho và đã bán.

3.5 Mô hình dữ liệu mức quan niệm (CDM)



3.6 Mô hình dữ liệu mức vật lý (PDM)

z

3.7 Các bảng thực thể

Bảng khách hàng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| khach\_hang | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_kh | varchar(100) | Khóa chính | ID khách hàng |
| 2 | email\_kh | varchar(100) |  | Email khách hàng |
| 3 | mat\_khau\_kh | varchar(255) |  | Mật khẩu khách hàng |
| 4 | ngay\_tao\_tk | date |  | Ngày tạo tài khoản |

Bảng địa chỉ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| dia\_chi | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_dc | varchar(100) | Khóa chính | ID địa chỉ |
| 2 | dia\_chi\_kh | varchar(100) |  | Địa chỉ khách hàng |
| 3 | sdt\_kh | varchar(10) |  | Số điện thoại khách hàng |
| 4 | ten\_kh | varchar(20) |  | Tên khách hàng |

Bảng sản phẩm

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| san\_pham | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_sp | varchar(100) | Khóa chính | ID sản phẩm |
| 2 | ten\_sp | varchar(100) |  | Tên sản phẩm |
| 3 | thong\_tin\_sp | varchar(255) |  | Thông tin sản phẩm |
| 4 | gia\_ban\_sp | integer |  | Giá bán sản phẩm |

Bảng chi tiết sản phẩm

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| chi\_tiet\_sp | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | so\_luong | integer |  | Số lượng sản phẩm |

Bảng màu sắc

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| mau\_sac | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_ms | varchar(100) | Khóa chính | ID màu sắc |
| 2 | ten\_ms | varchar(100) |  | Tên màu sắc |

Bảng kích thước

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| kich\_thuoc | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_kt | varchar(100) | Khóa chính | ID kích thước |
| 2 | ten\_kt | varchar(5) |  | Tên kích thước |

Bảng hình ảnh

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| hinh\_anh | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_ha | varchar(100) | Khóa chính | ID hình ảnh |
| 2 | hinh\_anh\_sp | varchar(255) |  | hình ảnh sản phẩm |

Bảng loại sản phẩm

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| loai\_san\_pham | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_lsp | varchar(100) | Khóa chính | ID loại sản phẩm |
| 2 | ten\_lsp | varchar(10) |  | Tên loại sản phẩm |

Bảng thương hiệu

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| thuong\_hieu | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_th | varchar(100) | Khóa chính | ID thương hiệu |
| 2 | ten\_th | varchar(25) |  | Tên thương hiệu |

Bảng khuyến mãi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| khuyen\_mai | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_km | varchar(100) | Khóa chính | ID khuyễn mãi |
| 2 | gia\_km | integer |  | Giá khuyễn mãi |

Bảng đánh giá

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| danh\_gia | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | so\_sao | integer | Khóa chính | Số sao |
| 2 | noi\_dung\_dg | varchar(255) |  | Nội dung đánh giá |
| 3 | ngay\_dg | date |  | Ngày đánh giá |

Bảng hóa đơn xuất

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| hoa\_don\_xuat | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_hdx | varchar(100) | Khóa chính | ID hóa đơn xuất |
| 2 | ten\_kh | varchar(50) |  | Tên khách hàng |
| 3 | so\_dien\_thoai | varchar(10) |  | Số điện thoại |
| 4 | tong\_tien\_hdx | integer |  | Tổng tiền hóa đơn xuất |
| 5 | ten\_vc | integer |  | Tiền vận chuyển |
| 6 | ngay\_lap\_hdx | date |  | Ngày lập hóa đơn xuất |
| 7 | trang\_thai | varchar(10) |  | Trạng thái |
| 8 | hinh\_thuc\_tt | varchar(20) |  | Hình thức thanh toán |
| 9 | dia\_chi\_hdx | varchar(100) |  | Địa chỉ hóa đơn xuất |

Bảng chi tiết hóa đơn xuất

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| chi\_tiet\_hdx | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | so\_luong\_xuat | integer | Khóa chính | Số lượng xuất |
| 2 | hinh\_anh\_sp | varchar(255) |  | Hình ảnh sản phẩm |
| 3 | ma\_ms | varchar(100) |  | Mã màu sắc |
| 4 | ma\_kt | varchar(100) |  | Mã kích thước |
| 5 | ten\_ms | varchar(100) |  | Tên màu sắc |
| 6 | ten\_kt | varchar(100) |  | Tên kích thước |

Bảng giao hàng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| giao\_hang | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_ngh | varchar(100) | Khóa chính | ID người giao hàng |
| 2 | ma\_nv | varchar(100) |  | Mã nhân viên |
| 3 | ngay\_gh | date |  | Ngày giao hàng |
| 4 | trang\_thai\_gh | varchar(50) |  | Trạng thái giao hàng |
| 5 | ghi\_chu | varchar(100) |  | Ghi chú |

Bảng hóa đơn nhập

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| hoa\_don\_nhap | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_hdn | varchar(100) | Khóa chính | ID hóa đơn nhập |
| 2 | ngay\_lap\_hdn | date |  | Ngày lập hóa đơn nhập |
| 3 | tong\_tien\_hdn | integer |  | Tổng tiền hóa đơn nhập |
| 4 | ma\_nv | varchar(100) |  | Mã nhân viên |

Bảng nhân viên

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| nhan\_vien | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_nv | varchar(100) | Khóa chính | ID nhân viên |
| 2 | ten\_nv | varchar(50) |  | Tên nhân viên |
| 3 | email\_nv | varchar(100) |  | Email nhân viên |
| 4 | sdt\_nv | varchar(10) |  | Số điện thoại nhân viên |
| 5 | dia\_chi\_nv | varchar(100) |  | Địa chỉ nhân viên |
| 6 | mat\_khau\_nv | varchar(255) |  | Mật khẩu nhân viên |
| 7 | gioi\_tinh\_nv | varchar(5) |  | Giới tính nhân viên |
| 8 | ngay\_sinh\_nv | date |  | Ngày sinh nhân viên |

Bảng chức vụ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| chuc\_vu | | | | |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | id\_cv | varchar(100) | Khóa chính | ID chức vụ |
| 2 | ten\_cv | varchar(50) |  | Tên chức vụ |