**TRƯỜNG ĐẠI HỌC ĐIỆN LỰC**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO CHUYÊN ĐỀ**

**HỌC PHẦN NGÔN NGỮ KỊCH BẢN**

**ĐỀ TÀI: XÂY DỰNG WEBSITE THỜI TRANG AMAZONA**

|  |  |
| --- | --- |
| **Sinh viên thực hiện** | **MAI VIỆT VƯƠNG** |
|  | **NGUYỄN ĐỨC HUY** |
| **Giảng viên hướng dẫn** | **CÙ VIỆT DŨNG** |
| **Ngành** | **CÔNG NGHỆ THÔNG TIN** |
| **Chuyên ngành** | **CÔNG NGHỆ PHẦN MỀM** |
| **Lớp** | **D14CNPM1** |
| **Khóa** | **2019 – 2024** |

***Hà Nội, tháng 12, năm 2022***

**PHIẾU CHẤM ĐIỂM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Họ và tên** | **Chữ kí** | **Điểm** | **Ghi chú** |
| 1 | Mai Việt Vương  MSV: 19810310085 |  |  |  |
| 2 | Nguyễn Đức Hu  MSV: 19810310025 |  |  |  |

|  |  |  |
| --- | --- | --- |
| **Họ và tên giảng viên** | **Chữ kí** | **Ghi chú** |
| Giảng viên 1: |  |  |
| Giảng viên 2: |  |  |

**MỤC LỤC**

[CHƯƠNG 1. KHẢO SÁT HỆ THỐNG 6](#_Toc86603315)

[1.1. Đặt vấn đề 6](#_Toc86603316)

[1.3. Mục tiêu đề tài 7](#_Toc86603317)

[2.1. Tổng quan về *Node.js* 8](#_Toc86603318)

[2.1.1. Giới thiệu về *Node.js* 8](#_Toc86603319)

[2.1.2. Giới thiệu về *MongoDB* 14](#_Toc86603320)

[CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG (UML) 18](#_Toc86603321)

[A. Mô hình nghiệp vụ 18](#_Toc86603322)

[B. Sơ đồ phân rã chức năng Website bán đồ Phượt 18](#_Toc86603323)

[C. DANH MỤC CÁCH KÝ HIỆU 18](#_Toc86603324)

[1. Biểu đồ Use Case 22](#_Toc86603325)

[1.1.Use Case Tổng quát 22](#_Toc86603326)

[1.2. Use Case Đăng nhập 23](#_Toc86603327)

[1.3. Use Case quản lý khách hàng 24](#_Toc86603328)

[1.4. Use Case Quản lý sản phẩm 25](#_Toc86603329)

[1.5. Use Case Mua hàng 26](#_Toc86603330)

[1.6. Biểu đồ hoạt động 27](#_Toc86603331)

[2. Biểu đồ tuần tự 35](#_Toc86603332)

[2.1.Biểu đồ tuần tự của đăng ký 36](#_Toc86603333)

[2.2 biểu đồ tuần tự đăng nhập 37](#_Toc86603334)

[2.3.Biểu đồ tuần tự của đăng xuất 38](#_Toc86603335)

[2.4. Biểu đồ tuần tự chức năng Mua hàng 39](#_Toc86603336)

[2.5. Biểu đồ tuần tự chức năng quản lý khách hàng 40](#_Toc86603337)

[3.Biểu đồ lớp (Class) 40](#_Toc86603338)

[3.1. Biểu đồ lớp quản lý nhân viên 40](#_Toc86603339)

[3.2. Biểu đồ lớp quản lý sản phẩm 41](#_Toc86603340)

[3.3. Biểu đồ lớp cho use case bán hàng 42](#_Toc86603341)

[CHƯƠNG 3: CÀI ĐẶT 43](#_Toc86603342)

[3.1 Giao diện người dùng 43](#_Toc86603343)

[3.2. Giao diện quản lý admin 46](#_Toc86603344)

[TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN 49](#_Toc86603345)

[1. Tổng kết 49](#_Toc86603346)

[2. Kết quả đạt được 49](#_Toc86603347)

[3. Hạn chế 49](#_Toc86603348)

[4. Hướng phát triển 49](#_Toc86603349)

[TÀI LIỆU THAM KHẢO 50](#_Toc86603350)

**LỜI NÓI ĐẦU**

Ngày nay *Công Nghệ Thông Tin* là một ngành rất phổ biến trong xã hội. Nó được ứng dụng rộng rãi trong rất nhiều ngành nghề, các lĩnh vực khác nhau và đạt được hiệu quả cao trong công việc. Đặc biệt là trong công tác quản lý, tin học giúp giảm tải sức người, tiết kiệm chi phí lẫn thời gian, gọn nhẹ hơn rất nhiều so với phương thức quản lý bằng giấy tờ như trước kia. Ứng dụng tin học vào công tác quản lý còn giúp thu hẹp không gian lưu trữ dữ liệu, tránh thất lạc dữ liệu, giảm thiểu các sai số, thiếu sót. Hơn thế nó còn giúp việc tra cứu, tìm kiếm thông tin một cách nhanh chóng, chính xác. Để giúp công việc quản lý đạt được những yêu cầu trên, *Công Nghệ Thông Tin* đã được đưa vào sử dụng và dần cho thấy sự hữu ích của nó. Với những thành tựu trong ngành, ta không còn phải đối mặt với những tập hồ sơ dày cộp, những ngăn tủ đựng hồ sơ chiếm nhiều diện tích mà ta có thể phải mất nhiều thời gian để tìm kiếm thông tin, dữ liệu cần thiết. Giờ đây tất cả các bất tiện đó đều được giải quyết bằng phần mềm quản lý – một sản phẩm của ngành *Công Nghệ Thông Tin*.

Chính vì sự hữu ích của ngành *Công Nghệ Thông Tin* mang lại, cùng với mong muốn xây dựng một website quản lý tiện lợi và giúp người dùng dễ dàng tiếp cận. Chúng em đã thực hiện đề tài ***“Xây dựng website thời trang Amazona”***. Chúng em tin rằng đây sẽ là một đề tài rất hay và có thể phát triển lớn hơn trong tương lai.

Là những sinh viên mới tiếp cận các phương pháp phân tích thiết kế hệ thống nên trong quá trình thực hiện không tránh khỏi những thiếu sót. Chúng em rất mong nhận được sự góp ý từ thầy cô để đề tài có thể đạt kết quả tốt hơn, sát với nhu cầu thực tiễn và có thể áp dụng trong tương lai.

Chúng em xin chân thành cảm ơn!

# CHƯƠNG 1. KHẢO SÁT HỆ THỐNG

## 1.1. Đặt vấn đề

Hiện nay trong khi đời sống con người nâng cao, nhu cầu du lịch ngày càng phát triển thì nhu cầu tìm những khách sạn có phòng nghỉ chất lượng và dịch vụ tốt để nghỉ ngơi thư giãn ngày càng phổ biến. Bằng cách truy cập vào các website tìm kiếm thông tin phòng tại khách sạn giúp khách hàng chọn được phòng theo ý thích của mình.

Trước kia khi chưa ứng dụng tin học vào công tác quản lý thì các công việc như cập nhật thông tin khách hàng, xem các thông tin về dịch vụ của khách sạn, thông tin về số phòng, và nhân viên, hoá đơn thanh toán, tất cả đều làm thủ công nên các cán bộ quản lý phải mất rất nhiều thời gian mới tổng hợp được những thông tin cần thiết. Mặt khác hiện nay trong các khách sạn, các cán bộ làm công tác quản lý thông tin của khách sạn phải quản lý một lượng lớn thông tin về cán bộ công nhân viên, tình trạng phòng... Việc này mất rất nhiều thời gian và công sức của con người nên việc ứng dụng tin học vào công tác quản lý khách sạn theo yêu cầu của người dùng ngày càng trở nên hết sức cần thiết, quan trọng.

Nhận thức được thực trạng đó chúng em mong muốn xây dựng một webste bán đồ Phượt, giờ đây khách hàng vẫn có thể dễ dàng mua sắm trực tuyến tại gia lựa chọn dịch vụ đi kèm, còn đối với các đại lý cửa hàng giờ đây đã có thể quản lý các mặt hàng sản phẩm, khách hàng nhân viên của mình một cách dễ dàng hơn, tất cả đã được gói gọn vào website và dịch vụ đi kèm.

Với đề tài này chúng em xin được trình bày một cách thức quản lý website cho phép người quản trị dễ dàng trong việc quản lý của hàng và các sản phẩm hiệu quả tới khách hàng và các đơn vị hợp tác.

**1.2. Mục đích nghiên cứu**

* Tìm hiểu về hệ thống phần mềm *Node.js.*
* Tìm hiểu các công cụ hỗ trợ như: *HTML, CSS, JS, Web Service.*
* Tìm hiểu về *MongoDB* và tương tác kiến thức *csdl.*
* Cách điều khiển dữ liệu *(Controller).*

## 1.3. Mục tiêu đề tài

Website với mục đích cung cấp cho khách hàng các thông tin chính xác về các phòng tại các khách sạn đang có trên cả nước và cách thức đặt phòng qua mạng. Các thông tin được cập nhật thường xuyên và nhanh chóng. Vì vậy, rút ngắn được khoảng cách giữa khách hàng và các khách sạn cung cấp dịch vụ, đưa thông tin về các phòng tại các khách sạn nhanh chóng đến cho khách hàng.

## 2.1. Tổng quan về *Node.js*

### 2.1.1. Giới thiệu về *Node.js*

***Node.js là gì?***

*Node.js* là một hệ thống phần mềm được thiết kế để viết các ứng dụng *internet* có khả năng mở rộng, đặc biệt là *máy chủ web.* Chương trình được viết bằng *JavaScript,* sử dụng kỹ thuật điều khiển theo sự kiện, nhập/xuất không đồng bộ để tối thiểu tổng chi phí và tối đa khả năng mở rộng. *Node.js* bao gồm có *V8 JavaScript engine* của Google, *libUV,* và vài thư viện khác.

*Node.js* được tạo bởi Ryan Dahl từ năm 2009, và phát triển dưới sự bảo trợ của Joyent.

Mục tiêu ban đầu của Dahl là làm cho trang web có *khả năng push* như trong một số ứng dụng web như *Gmail.* Sau khi thử với vài ngôn ngữ Dahl chọn *JavaScript* vì một *API Nhập/Xuất không đầy đủ.* Điều này cho phép anh có thể định nghĩa một quy ước *Nhập/Xuất* điều khiển theo sự kiện *non-blocking.*

Vài môi trường tương tự được viết trong các ngôn ngữ khác bao gồm *Twisted* cho *Python, Perl Object Environment* cho *Perl, libevent* cho *C* và *EventMachine* cho *Ruby.* Khác với hầu hết các chương trình *JavaScript, Node.js* không chạy trên một trình duyệt mà chạy trên *Server. Node.js* sử dụng nhiều chi tiết kỹ thuật của *CommonJS.* Nó cung cấp một môi trường *REPL* cho kiểm thử tương tác.

*Node.js* được *InfoWorld* bình chọn là *“Công Nghệ của năm”* vào 2012.

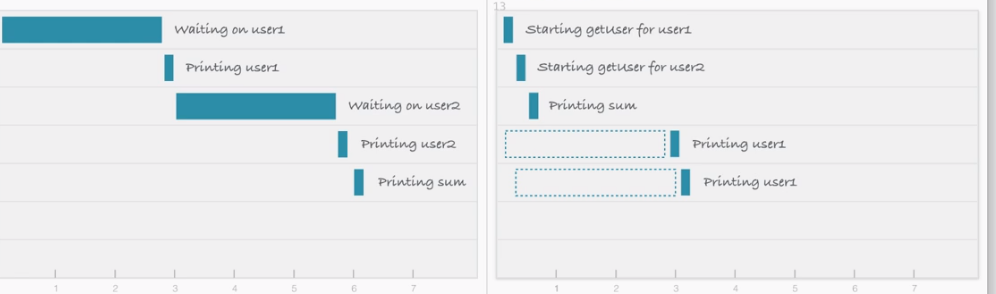
***Tại sao lại là Node.js?***

*Chúng ta sẽ đi tìm hiểu tại sao Node.js lại phổ biến đến vậy nhé!*

*I/O* là *Input/Output.* Nó có thể là bất cứ thứ gì từ đọc/viết các *file* nội bộ cho đến tạo *HTTP* đến *API.*

*I/O* tốn thời gian và do đó nó sẽ chặn các chức năng khác.

Xét một tình huống là chúng ta cần dữ liệu *backend* chi tiết cho *user1* và *user2* sau đó hiển thị chúng lên màn hình hoặc bảng điều khiển. Để phản hồi cho lệnh này sẽ mất thời gian, nhưng cả 2 dữ liệu *user* có thể thực hiện độc lập và cùng lúc.



*Blocking I/O (trái) và Non-Blocking I/O (phải).*

***Blocking I/O***

Trong phương thức *blocking,* dữ liệu của *user2* sẽ không bắt đầu cho đến khi dữ liệu của *user1* xuất trên màn hình.

Nếu đây là một *web server,* chúng ta phải bắt đầu một *thread* mới cho từng *user* mới. Nhưng *JavaScript* là đơn luồng *(tính chất “xong cái này mới đến cái khác”).* Và điều này làm *JavaScript* không phù hợp cho các tác vụ đa luồng.

Đó là lý do *non-blocking* xuất hiện.

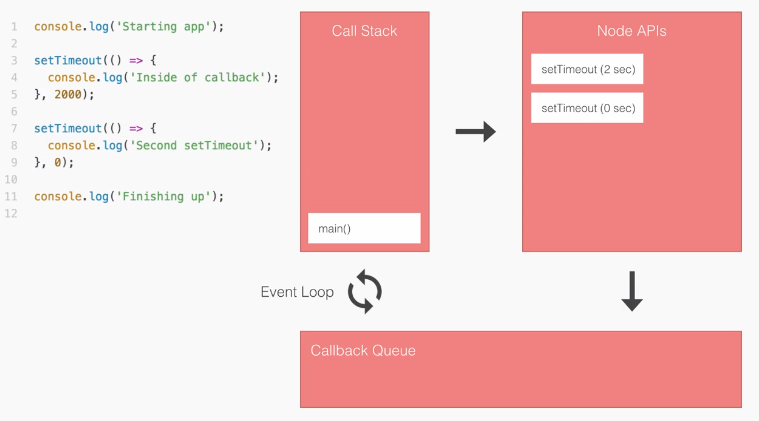
***Non-blocking I/O***

Mặt khác, sử dụng *request non-blocking,* bạn có thể *request* dữ liệu cho *user2* mà không phải đợi phản hồi *request* từ *user1.* Bạn có thể bắt đầu cả 2 *request* song song.

*Non-blocking I/O* loại bỏ nhu cầu đa luồng vì *server* có thể xử lý nhiều *request* cùng một lúc.

***JavaScript Event Loop***

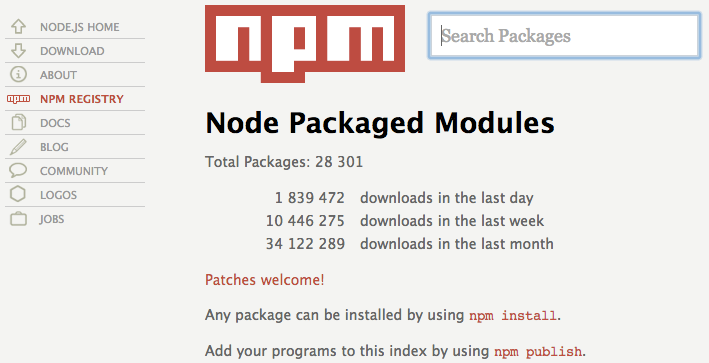
*Event Loop* là cơ chế giúp *JavaScript* có thể thực hiện nhiều thao tác cùng một lúc *(concurrent model),* trước giờ vẫn nghe nói *Node.js* có thể xử lý cả hàng ngàn *request* cùng một lúc mặc dù nó chỉ dùng một *thread* duy nhất *(Single Threaded).* Nếu như ở *PHP* hay *Java* thì với mỗi một *request* sẽ sinh ra một *thread* để xử lý *request* đó, các *thread* hoạt động độc lập, được cấp bộ nhớ, giao tiếp ngoại vi và trả về kết quả.



*Cách thức JavaScript Event Loop hoạt động*

***npm***

*npm –* viết tắt của *Node package manager* là một công cụ tạo và quản lý các thư viện lập trình *JavaScript* cho *Node.js.* 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 là cả *framework.*



*npm* có sẵn khi bạn tải *Node.js* về. Để kiểm tra xem trên hệ thống của bạn đã được cài *npm* chưa chúng ta sử dụng lệnh *npm -v,* nếu một phiên bản hiện ra thì hệ thống của bạn đã được cài đặt *npm.*

Vì *npm* là một phần mềm cài đặt trên máy tính của bạn nên bạn có thể sử dụng nó để cài đặt các thư viện *JavaScript* từ trên *Internet.* Để cài đặt một thư viện nào đó, chỉ cần mở cửa sổ *Terminal (hoặc CMD)* và thực thi lệnh giống dưới đây:

*npm install package-name*

*Ví dụ: ta thử tải Vuejs về sử dụng sẽ dùng lệnh:*

*npm install vue*

*Khi ai đó muốn sử dụng Vue.js chúng ta chỉ cần sử dụng lệnh require():*

*var Vue = require(‘vue’);*

***Cài đặt global và local***

Có 2 cách để cài đặt một gói bằng *npm:*

* ***Local:*** sẽ tạo ra thư mục *node\_modules* nếu chưa có trong *project* hoặc nếu có rồi sẽ lấy *code* của gói cần cài đặt đưa vào đây, tức chỉ hiện diện trong thư mục của *project* hiện tại. Khi cần sử dụng bạn có thể sử dụng lệnh *require().*
* ***Global:*** sẽ lưu trữ *code* của gói trong các *file* hệ thống cố định trong máy, chỉ có thể dùng các *package* này thông qua các hàm *CLI (Command Line Interface)* ví dụ như *glub.* Không thể dùng *package* thông qua *require().*

Mặc định thì các *package* khi cài đặt đều sẽ là cài trên *project* của bạn.

*npm update package\_name*

Các câu lệnh này có thể sử dụng *flag -g* để thực hiện cập nhật cho các gói được cài đặt *global.*

***Kiểm tra các gói cài đặt***

Để kiểm tra các gói đã được thêm thông qua *npm* sử dụng câu lệnh *npm ls,* nếu kiểm tra các cài đặt *global* thêm tham số *-g.*

*npm ls*

*npm ls -g*

***Package.json***

Để quản lý các gói cài đặt cục bộ bằng *npm* thì cách tốt nhất là thông qua *file package.json,* chính là *file* nằm trong thư mục gốc của *project. File JSON* này chứa các nội dung:

* Các gói thư viện lập trình mà *project* sử dụng.
* Cho phép xác định phiên bản chính xác của các gói thư viện lập trình được sử dụng.
* Các gói bạn xây dựng có thể chia sẻ dễ dàng với các lập trình viên khác trên toàn cầu thông qua *npm.*

Lệnh *npm init -yes* sẽ tạo ra *file package.json* mẫu.

Có một số các thuộc tính trong *package.json,* chúng ta cùng điểm qua:

* *name: tên gói thư viện*
* *version: phiên bản gói*
* *description: phần mô tả về gói thư viện*
* *homepage: trang chủ của gói*
* *author: tác giả*
* *contributors: tên người đóng góp cho package*
* *dependencies: danh sách các gói phụ thuộc, tự động được cài theo*
* *repository: loại repository và url của package, thông thường là git*
* *main: index.js*
* *keywords: các từ khóa*

***Require***

*Require* làm 3 thứ:

* Tải *module* đi kèm với *Node.js* như hệ thống *file* và *HTTP* từ *Node.js API.*
* Tải thư viện thứ 3 như *Express* và *Mongoose* mà bạn cần cài đặt từ *npm.*
* Giúp *require file* của bạn và *mô-đun hóa project.*

*Require* là một chức năng, và nó nhận tham số *path* tinh chỉnh và trả về *module.export.*

***Node Modules***

*Module Node* là 1 *block code* có thể tái sử dụng mà sự tồn tại của nó không vô tình ảnh hưởng đến các *code* khác.

Bạn có thể tạo *Module* của riêng mình và sử dụng nó với nhiều ứng dụng khác nhau.

*Node.js* có 1 bộ *built-in module* bạn có thể sử dụng mà không cần cài đặt thêm gì nữa.

***V8 turbo-charges JavaScript bằng cách tận dụng C++***

*V8* là một công cụ *runtime* nguồn mở được viết bằng *C++.*

*JavaScript ­­-> V8 (C++) -> Machine Code.*

*V8* thực hiện một tệp lệnh gọi là *ECMAScript* như được chỉ định trong *ECMA-262.*

*ECMAScript* được tạo ra bởi *ECMA International* để chuẩn hóa *JavaScript.*

Về cơ bản viêcj này cho phép bạn thêm các tính năng vào *JavaScript* ngoài *V8* của *Chrome* như *SpiderMonkey* của *Mozilla, Chakra* của *Microsoft, v.v…*

***Events***

Có 2 loại *event* trong *Node:*

* *Event System: C++* có lõi từ một thư viện gọi là *libuv. (Vd: đọc xong một file)*
* *Event Custom:* Lõi *JavaScript.*

### 2.1.2. Giới thiệu về *MongoDB*

***MongoDB là gì?***

*MongoDB* là một *database* hướng tài liệu *(document),* một dạng *NoSQL database.* Vì thế, *MongoDB* sẽ tránh cấu trúc *table-based* của *relational database* để thích ứng với các tài liệu như *JSON* có một *schema* rất linh hoạt gọi là *BSON. MongoDB* sử dụng lưu trữ dữ liệu dưới dạng *Document JSON* nên mỗi một *collection* sẽ có các kích cỡ và các *document* khác nhau. Các dữ liệu được lưu trữ trong *document* kiểu *JSON* nên truy vấn sẽ rất nhanh.

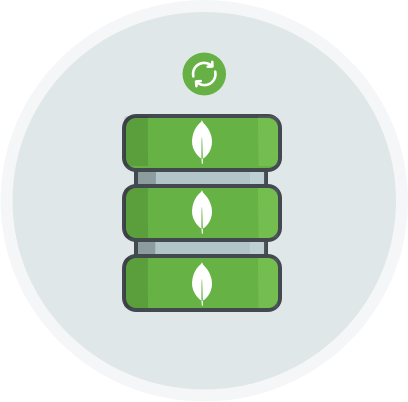
***Định nghĩa thêm về MongoDB***

*MongoDB* lần đầu ra đời bởi *MongoDB Inc.,* tại thời điểm đó là thế hệ 10, vào tháng Mười năm 2007 nó là một phần của sản phẩm *PaaS (Platform as a Service)* tương tự như *Windows Azure* và *Google App Engine.* Sau đó nó đã được chuyển thành nguồn mở từ năm 2009.

*MongoDB* đã trở thành một trong nhưng *NoSQL database* nổi trội nhất bấy giờ, được dùng làm *backend* cho rất nhiều *website* như *eBay, SourceForge* và *The New York Times.*

***Các feature của MongoDB gồm có:***

* ***Các ad hoc query:*** hỗ trợ *search* bằng *field,* các phép *search* thông thường, *regular expression searches* và *range queries.*
* ***Indexing:*** bất kì *field* nào trong *BSON document* cũng có thể được *index.*
* ***Replication:*** có ý nghĩa là *“nhân bản”,* là có một phiên bản giống hệt phiên bản đang tồn tại, đang sử dụng. Với cơ sở dữ liệu, nhu cầu lưu trữ lớn, đòi hỏi *cơ sở dữ liệu* toàn vẹn, không bị mất mát trước những sự cố ngoài dự đoán là rất cao. Vì vậy, người ta nghĩ ra khái niệm *“nhân bản”,* tạo một phiên bản *cơ sở dữ liệu* giống hệt *cơ sở dữ liệu* đang tồn tại, và lưu trữ ở một nơi khác, đề phòng có sự cố.
* ***Aggregation:*** các *Aggregation operation* xử lý các bản ghi dữ liệu và trả về kết quả đã được tính toán. Các phép toán tập hợp các nhóm giá trị từ nhiều *Document* lại với nhau, và có thể thực hiện nhiều phép toán đa dạng trên dữ liệu đã được nhóm đó để trả về một kết quả duy nhất. Trong đó *SQL, count (\*)* và *GROUP BY* là tương đương với *Aggregation* trong *MongoDB.*
* ***Lưu trữ file:*** *MongoDB* được dùng như một hệ thống *file* tận dụng những *function* trên và hoạt động như một cách phân phối qua *sharding.*



***Khi nào sử dụng MongoDB?***

* ***Quản lý và truyền tải content:*** Quản lý đa dạng nhiều *product* của *content* chỉ trong một kho lưu trữ *data* cho phép thay đổi và phản hồi nhanh chóng mà không chịu thêm phức tạp từ hệ thống *content.*
* ***Cấu truc Mobile và Social:*** *MongoDB* cung cấp một *platform* có sẵn, phản xạ nhanh, và dễ mở rộng cho phép rất nhiều khả năng đột phá, phân tích *real-time,* và hỗ trợ toàn cầu.
* ***Quản lý data khách hàng:*** Tận dụng khả năng *query* nhanh chóng cho phân tích *real-time* trên *cơ sở dữ liệu* người dùng cực lớn với các mô hình *data* phức tạp bằng các *schema* linh hoạt và tự động *sharding* cho mở rộng chiều ngang.

***Ưu điểm của MongoDB***

* Dữ liệu lưu trữ phi cấu trúc, không có tính ràng buộc, toàn vẹn nên tính sẵn sàng cao, hiệu suất lớn và dễ dàng mở rộng lưu trữ.
* Dữ liệu được *caching (ghi đệm)* lên *RAM,* hạn chế truy cập vào *ổ cứng* nên tốc độ đọc và ghi cao.

***Nhược điểm của MongoDB***

* Không ứng dụng được cho các mô hình giao dịch nào yêu cầu dộ chính xác cao do không ràng buộc.
* Không có cơ chế *transaction (giao dịch)* để phục vụ các ứng dụng ngân hàng.
* Dữ liệu lấy *RAM* làm trọng tâm hoạt động vì vậy khi hoạt động yêu cầu một bộ nhớ *RAM* lớn*.*
* Mọi thay đổi về dữ liệu mặc định đều chưa được ghi xuống *ổ cứng* ngay lập tức vì vậy khả năng bị mất dữ liệu từ nguyên nhân mất điện đột xuất là rất cao.

# CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG (UML)

## 2.1. Phân tích yêu cầu của hệ thống

Sau khi khảo sát hiện trạng ở trên, một yêu cầu cấp thiết cần đặt ra là xây dựng và phát triển một Website đáp ứng được nhu cầu tìm kiếm phòng, đặt phòng, các dịch vụ kèm theo phòng... Việc xây dựng website phải đáp ứng được những yêu cầu sau:

**Quản lý khách hàng:**

Mỗi khách hàng của khách sạn đều được khách sạn quản lý những thông tin sau: Họ tên, giới tính, địa chỉ, điện thoại, số CMND hoặc Passport (hoặc các văn bằng khác có hình).

**Quản lý phòng:**

Các phòng của khách sạn được quản lý dựa vào phòng số, loại phòng, trạng thái, giá cơ bản. Khách sạn gồm có các loại phòng tương ứng với giá (trong đó có 10% VAT).

**Quản lý đăng ký - thuê phòng:**

Tất cả các thông tin đăng ký thuê phòng đều được quản lý dựa trên người đặt phòng, họ tên khách đăng ký, thời gian đặt phòng, thời gian trả phòng, số tiền thanh toán và trạng thái thanh toán.

**Quản lý thông tin nhận phòng:**

Khách sạn quản lý những thông tin sau: Họ tên người nhận phòng, ngày nhận, giờ nhận. Tất cả thông tin nhận phòng đều dựa vào thông tin đăng ký thuê phòng đã khai báo trước đó. Vì khách hàng có đăng ký mới được nhận phòng theo đúng thông tin mà khách đã đăng ký.

**Quản lý việc trả phòng:**

Dựa trên thông tin đăng ký thuê phòng và thông tin nhận phòng, việc trả phòng được khách sạn quản lý các thông tin sau: Số trả phòng, ngày trả, giờ trả, họ tên người trả (có thể trả phòng trước thời hạn đăng ký). Nếu khách hàng muốn gia hạn thêm thời gian ở tại khách sạn thì phải tiến hành làm thủ tục đăng ký lại. Bên cạnh việc trả phòng của khách thì khách sạn sẽ kiểm tra lại tình trạng phòng.

**Quản lý thông tin huỷ đăng ký:**

Cũng như việc quản lý các công việc trên thì việc huỷ đăng ký được quản lý các thông tin sau: Số huỷ đăng ký, ngày huỷ đăng ký, giờ huỷ đăng ký, họ tên người huỷ đăng ký.

**Quản lý tài khoản nhân viên:**

Tất cả những nhân viên làm việc tại khách sạn đều được quản lý các thông tin sau: Họ nhân viên, Tên nhân viên, Chức vụ của nhân viên.

**Quản lý dịch vụ:**

Khách sạn còn quản lý thông tin về dịch vụ: tên dịch vụ, giá tiền, đơn vị. Tại khách sạn có nhiều loại dịch vụ như: điện thoại, ăn uống, giặt ủi, đưa đón khách, chuyên chở đồ cho khách (khi khách yêu cầu)...

**Quy trình quản lý:**

Khi khách hàng có nhu cầu ở **khách sạn** thì khách hàng có thể liên hệ với khách sạn để tiến hành làm thủ tục **đăng ký phòng** hoặc khách hàng có thể làm thủ tục thuê và nhận phòng để ở ( trường hợp phòng còn trống ). Có hai hình thức liên hệ: **đến trực tiếp, qua điện thoại.**

Khi khách hàng làm thủ tục đăng ký thì khách hàng phải khai báo đầy **đủ thông tin** về mình cũng như thông tin về cơ quan (nếu có) khi **Bộ phận Lễ tân** (BPTL) yêu cầu. Hầu hết khách hàng đều phải **đặt cọc** **một số tiền nhất** định dựa vào **loại phòng** mà mình đã đăng ký.

Trường hợp các tổ chức thuê phòng thì tổ chức sẽ **cử đại diện** đến khách sạn làm **thủ tục đăng ký**.

Trong thời gian trước ngày nhận phòng tối thiểu là 1 tuần các khách hàng đăng ký theo đoàn phải gửi cho khách sạn danh sách các thành viên để khách sạn phân bố phòng cho hợp lý.

Khi khách hàng đến nhận phòng thì ngoài số tiền đặt cọc, khách hàng phải gửi cho BPLT CMND\_Passport ( hoặc các văn bằng có hình còn thời hạn sử dụng ). BPLT căn cứ vào đó để kiểm tra lại thông tin khi khách đăng ký phòng. Sau khi kiểm tra xong BPLT sẽ chỉ trả lại các văn bằng này khi khách hàng làm xong thủ tục trả phòng. Sau đó BPLT sẽ kiểm tra lại tình trạng phòng để tiến hành giao phòng cho khách.

Trong thời gian ở khách sạn, khách có thể sử dụng các loại dịch vụ, yêu cầu phục vụ. Mọi yêu cầu sẽ do BPLT đảm nhiệm và đáp ứng trong phạm vi hoạt động của khách sạn. Khách hàng sử dụng dịch vụ chưa thanh toán thì Bộ phận dịch vụ sẽ ghi hoá đơn nợ và chuyển đến BPLT tính vào hoá đơn tổng.

Sau khi đặt phòng khách có thể huỷ đăng ký và phải bồi thường cho khách sạn theo quy định đã thoả thuận. Vì thế tiền đặt cọc là phải có trước khi nhận phòng. Để tránh rủi ro thì số tiền đặt cọc phải lớn hơn hay bằng số tiền phải bồi thường. Tuy nhiên trước 7 ngày nhận phòng khách huỷ đăng ký không phải bồi thường.

Khi hết thời hạn thuê phòng như đã đăng ký thì khách phải trả phòng cho khách sạn. Khi khách làm thủ tục trả phòng, BPLT sẽ kiểm tra lại tình trạng phòng, tổng hợp lại các hoá đơn sử dụng dịch vụ chưa thanh toán trong thời gian khách ở khách sạn. sau khi tổng hợp in ra hoá đơn tổng để khách thanh toán và trả lại CMND\_Passport cho khách. Hình thức thanh toán ở khách sạn chủ yếu bằng tiền mặt (tiền Việt nam\_VNĐ).

**2.2. Phân tích thiết kế với UML**

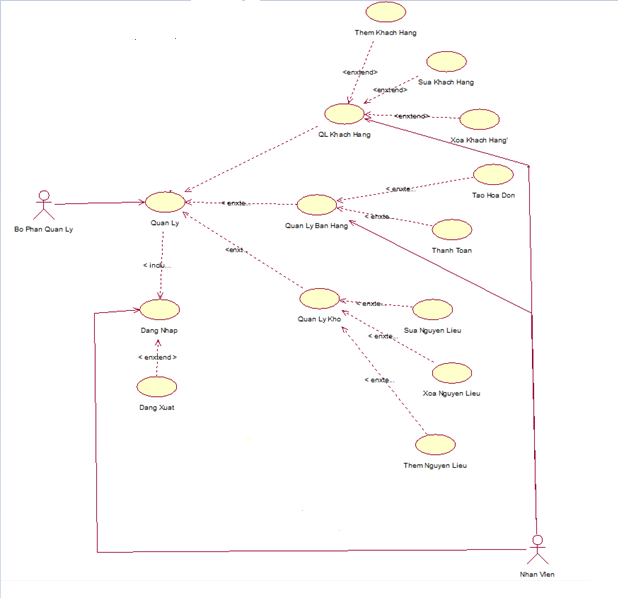
**2.2.1.** **Xác định các tác nhân của hệ thống**

* ***Admin (Người quản trị):***
* Quản trị viên có thể đăng nhập website của họ.
* Quản trị viên có thể xem, thay đổi trạng thái phòng, thêm, sửa, xoá, tìm kiếm các sản phẩm.
* Quản trị viên có thể xem, thêm, sửa, xoá, tìm kiếm các dịch vụ.
* Quản trị viên có thể xem, thêm, sửa, xoá, tìm kiếm các thông tin của khách hàng.
* ***Visitors (Khách hàng):***
* Khách hàng có thể tìm kiếm thông tin theo tên, giá cả ...
* Khách hàng có thể xem tất cả các sản phẩm có mặt ở website.
* Khách hàng có thể đặt mua sản phẩm ở website.

Bảng 2. . Các tác nhân của hệ thống

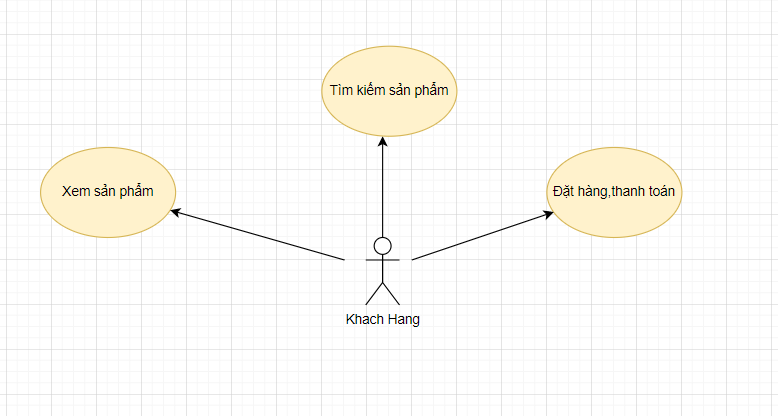
|  |  |
| --- | --- |
| **Tác nhân** | **Chức năng** |
| **Khách hàng** | Xem thông tin |
| Tìm kiếm phòng |
| Đặt mua sản phẩm |
| Trao đổi ý kiến, phản hồi |
| **Người quản trị** | Quản lý sản phẩm |
| Quản lý danh mục |
| Đăng nhập hệ thống |
| Quản lý dịch vụ |
|
|

* + 1. **Biểu đồ Use case**
       1. **Biểu đồ UseCase tổng quát**

**

Hình 2.1: *Biểu đồ Usecase tổng quát*

* + - 1. **Biểu đồ UseCase Khách Hàng**

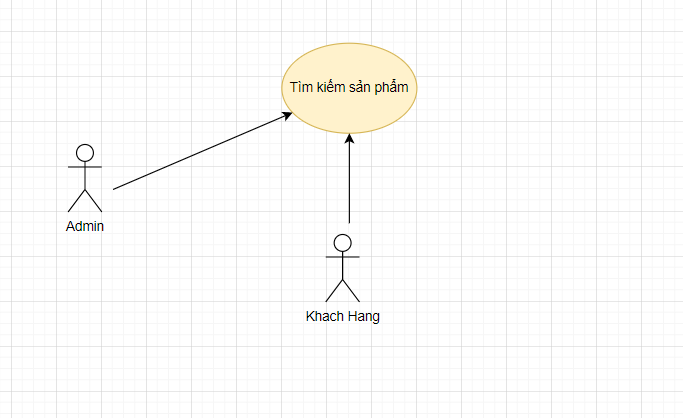
Hình 2.: *Biểu đồ Usecase Khách hàng*

* + - 1. Diagram

         Description automatically generated**UseCase Người Quản trị**

Hình 2.3*: Biểu đồ Usecase Người Quản trị*

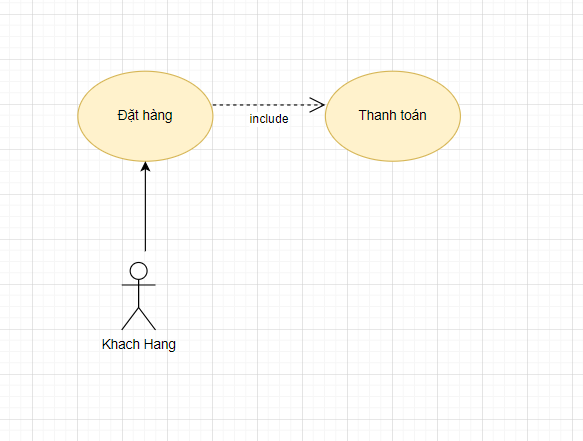
* + - 1. **Use case Tìm kiếm sản phẩm**

****

Hình 2.5: *Biểu đồ Usecase Tìm kiếm phòng*

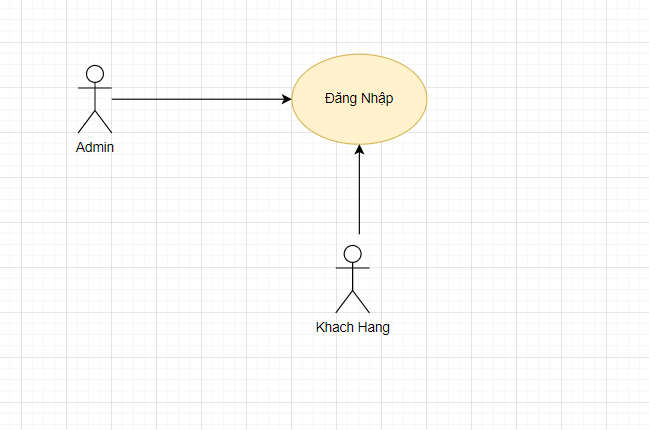
Hình 2.6*: Biểu đồ Usecase Đặt phòng*

* + - 1. **Use case Thanh toán đơn**

****

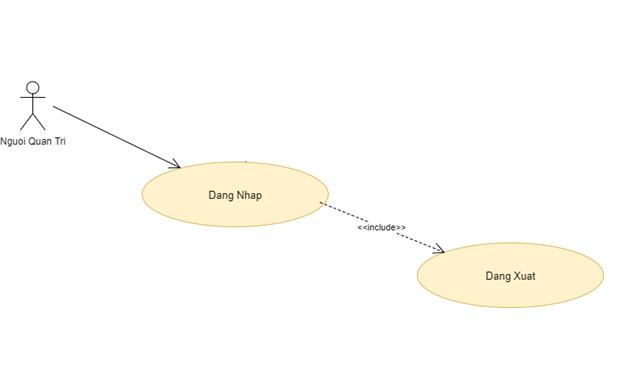
Hình 2.7*: Biểu đồ Usecase Thanh toán đơn đặt phòng*

* + - 1. **Use case Đăng nhập hệ thống**

****

Hình 2.8*: Biểu đồ Usecase Đăng nhập*

* + - 1. **Use case Đăng xuất hệ thống**

****

Hình 2.9*: Biểu đồ Usecase Đăng xuất*

* + - 1. Graphical user interface

         Description automatically generated**Use case Quản lý sản phẩm**

Hình 2.10*: Biểu đồ Usecase Quản lý phòng*

* + - 1. Graphical user interface

         Description automatically generated**Use case Quản lý dịch vụ**

Graphical user interface, application

Description automatically generatedHình 2.11: *Biểu đồ Usecase Quản lý dịch vụ*

* + - 1. **Use case Quản lý phân quyền**

Hình 2.12: *Biểu đồ Usecase Quản lý phân quyền*

* + - 1. **Use case Quản lý nhân viên**

Graphical user interface, application

Description automatically generated

Hình 2.13: *Biểu đồ Usecase Quản lý nhân viên*

* + - 1. **Use case Quản lý hóa đơn**

Graphical user interface, application

Description automatically generated**Hình 2.14: *Biểu đồ Usecase Quản lý hóa đơn***

* + - 1. **Use case Quản lý khách hàng**

**Graphical user interface

Description automatically generated**

Hình 2.15*: Biểu đồ Usecase Quản lý khách hàng*

* + - 1. Graphical user interface

         Description automatically generated**Use case Quản lý tỉnh thành**

Hình 2.16*: Biểu đồ Usecase Quản lý tỉnh thành*

* + - 1. **Use case Quản lý khách sạn**

Graphical user interface

Description automatically generated

Hình 2.17*: Biểu đồ Usecase Quản lý khách sạn*

**2.2.3. Biểu đồ trình tự**

**2.2.3.1. Quản lý thuê phòng**

Hình 2.18*: Biểu đồ trình tự Quản lý thuê phòng*

**2.2.3.2. Quản lý Phòng**

Hình 2.19*: Biểu đồ trình tự Quản lý phòng*

**2.2.3.3. Quản lý khách hàng**



Hình 2.20: *Biểu đồ trình tự Quản lý khách hàng*

**2.2.3.4. Quản lý tài khoản**



Hình 2.21*:* Biểu đồ trình tự Quản lý tài khoản

**2.2.3.5. Quản lý nhân viên**



Hình 2.22: *Biểu đồ trình tự Quản lý nhân viên*

**2.2.3.6. Lập hóa đơn**



Hình 2.23*: Biểu đồ trình tự Lập hóa đơn*

**2.2.3.7. Quản lý dịch vụ**

Hình 2.24: *Biểu đồ trình tự Quản lý dịch vụ*

**2.2.3.8. Quản lý thuê dịch vụ**

Hình 2.25*: Biểu đồ trình tự Quản lý thuê dịch vụ*

**2.2.4. Biểu đồ hoạt động**

**2.2.4.1. Quản lý thuê phòng**

Hình 2.26*: Biểu đồ hoạt động Quản lý thuê phòng*

**2.2.4.2. Quản lý Phòng**

Hình 2.27*: Biểu đồ hoạt động Quản lý Phòng*

**2.2.4.3. Quản lý khách hàng**



Hình 2.28*: Biểu đồ hoạt động Quản lý khách hàng*

**2.2.4.4. Quản lý tài khoản**



Hình 2.29*: Biểu đồ hoạt động Quản lý tài khoản*

**2.2.4.5. Quản lý nhân viên**



Hình 2.30*: Biểu đồ hoạt động Quản lý nhân viên*

**2.2.4.6. Lập hóa đơn**



Hình 2.31*: Biểu đồ hoạt động Lập hóa đơn*

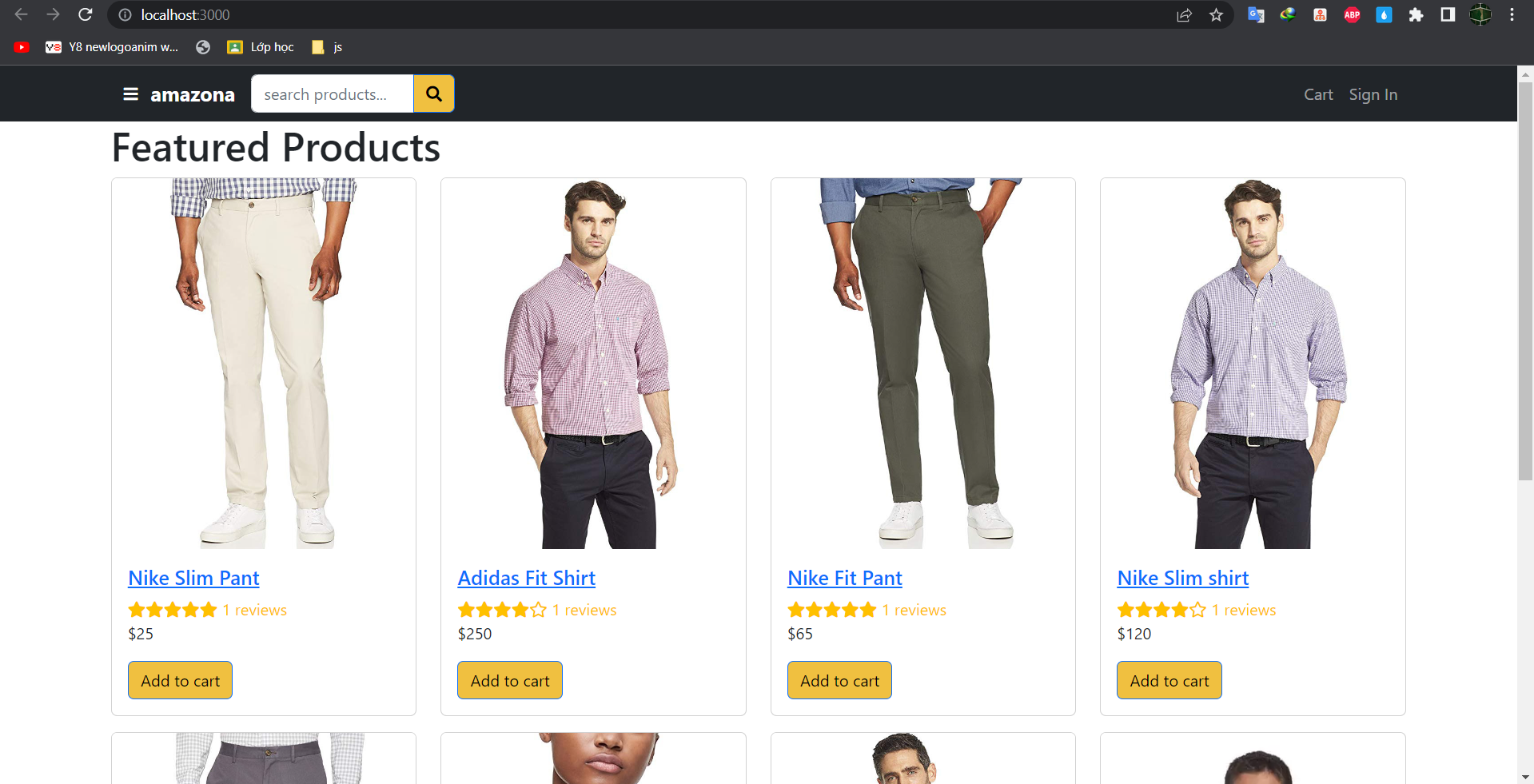
**2.2.4.7. Quản lý thuê dịch vụ**



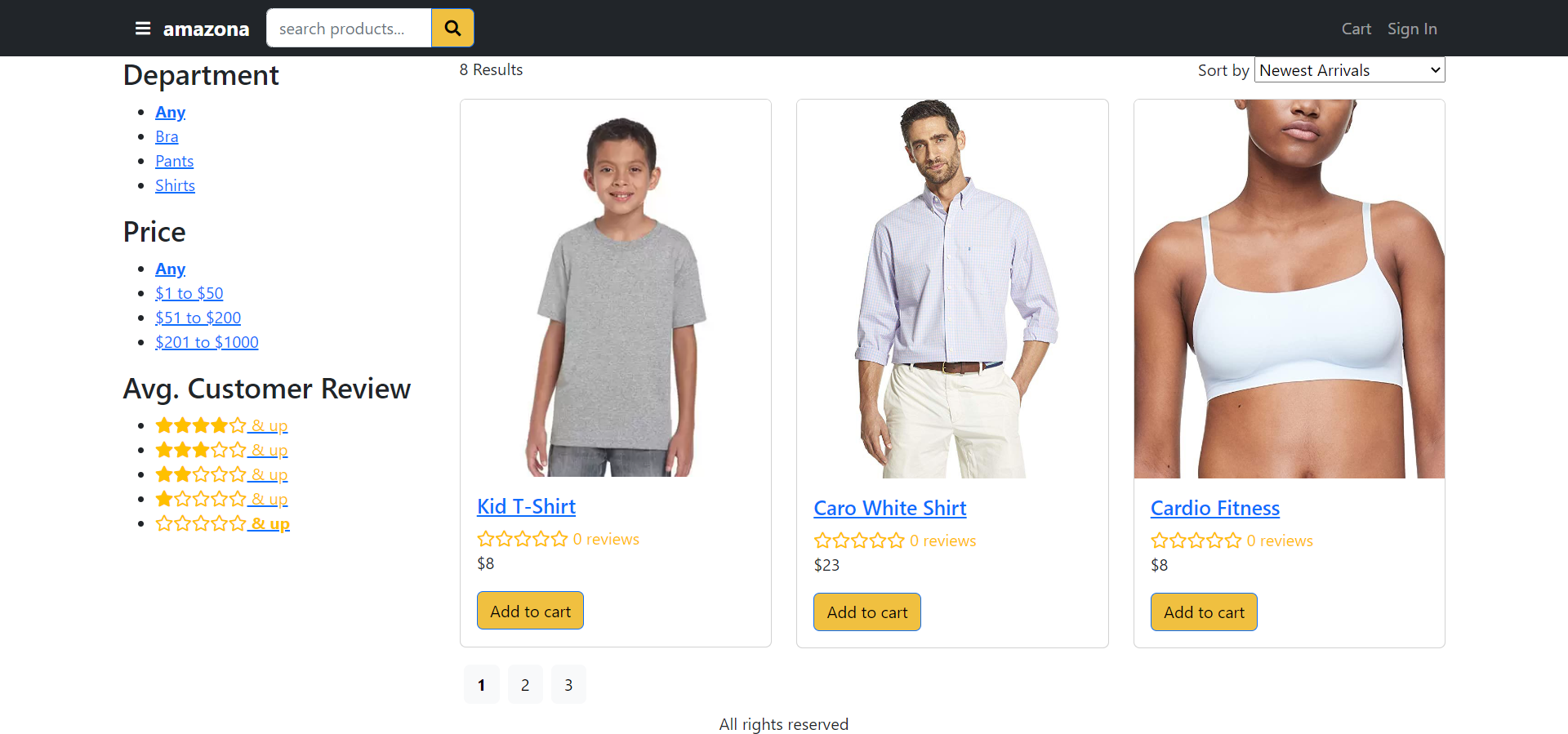
Hình 2.32*: Biểu đồ hoạt động Quản lý dịch vụ*

# CHƯƠNG 3: CÀI ĐẶT

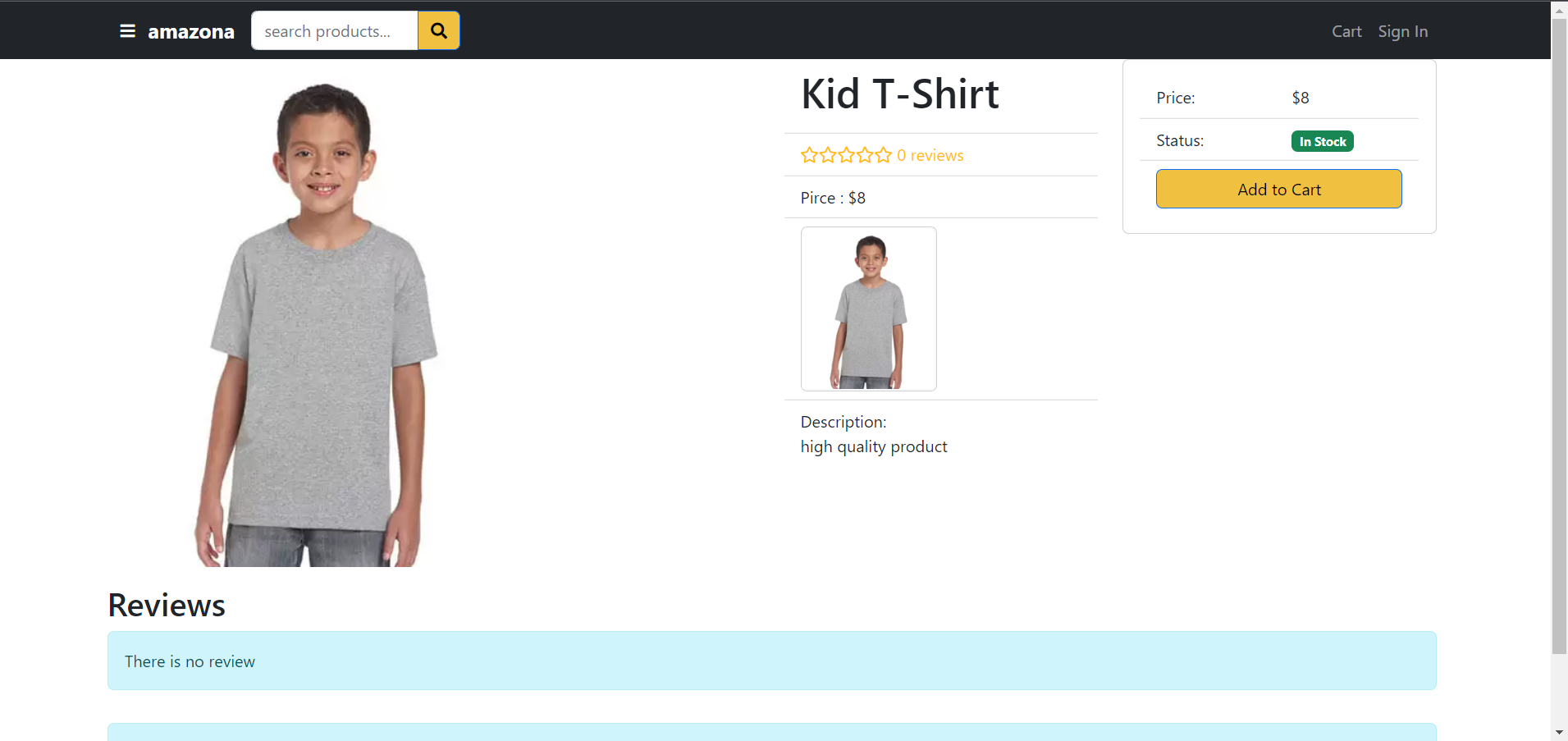
## 3.1 Giao diện người dùng



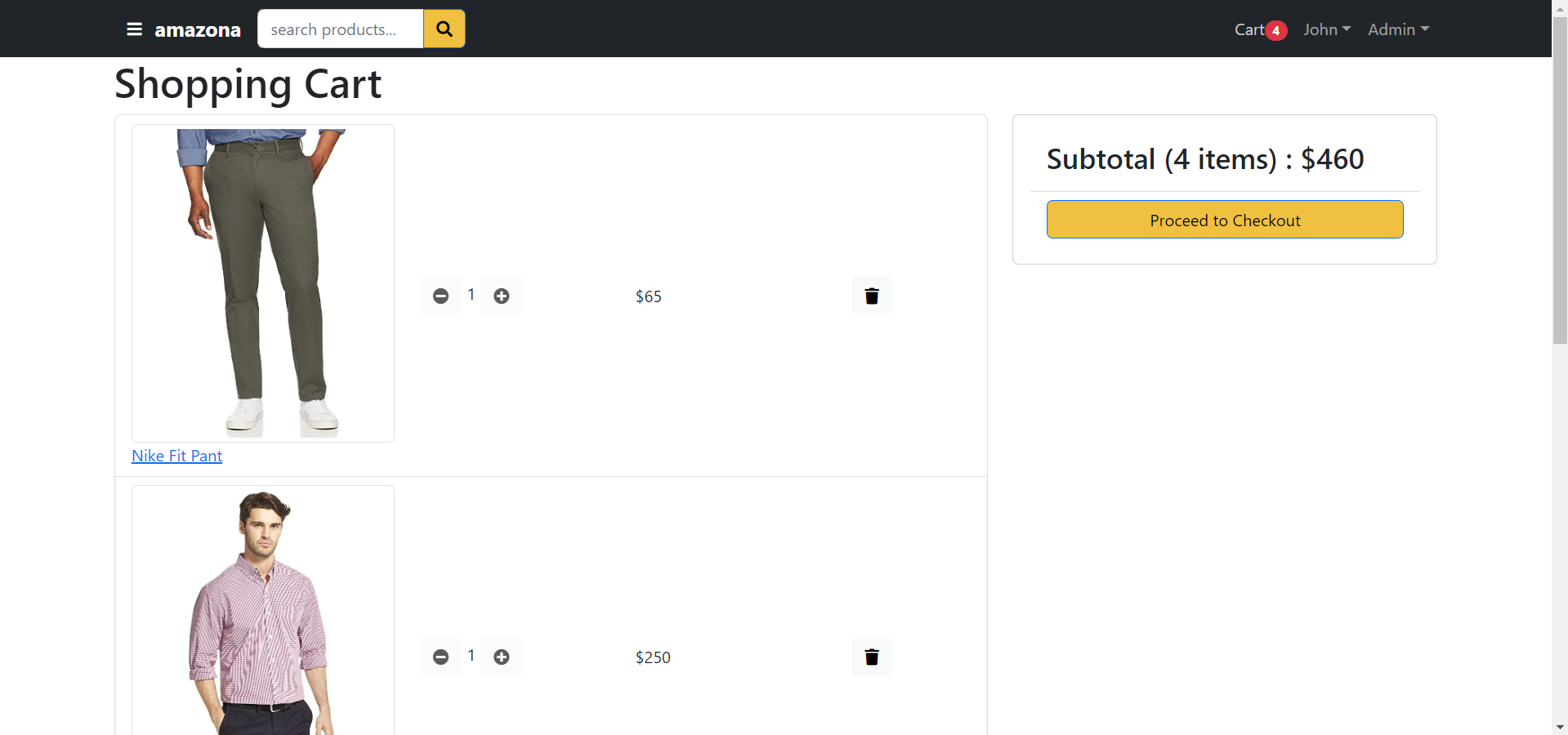
*Hình 3.1.1 Giao diện trang chủ*

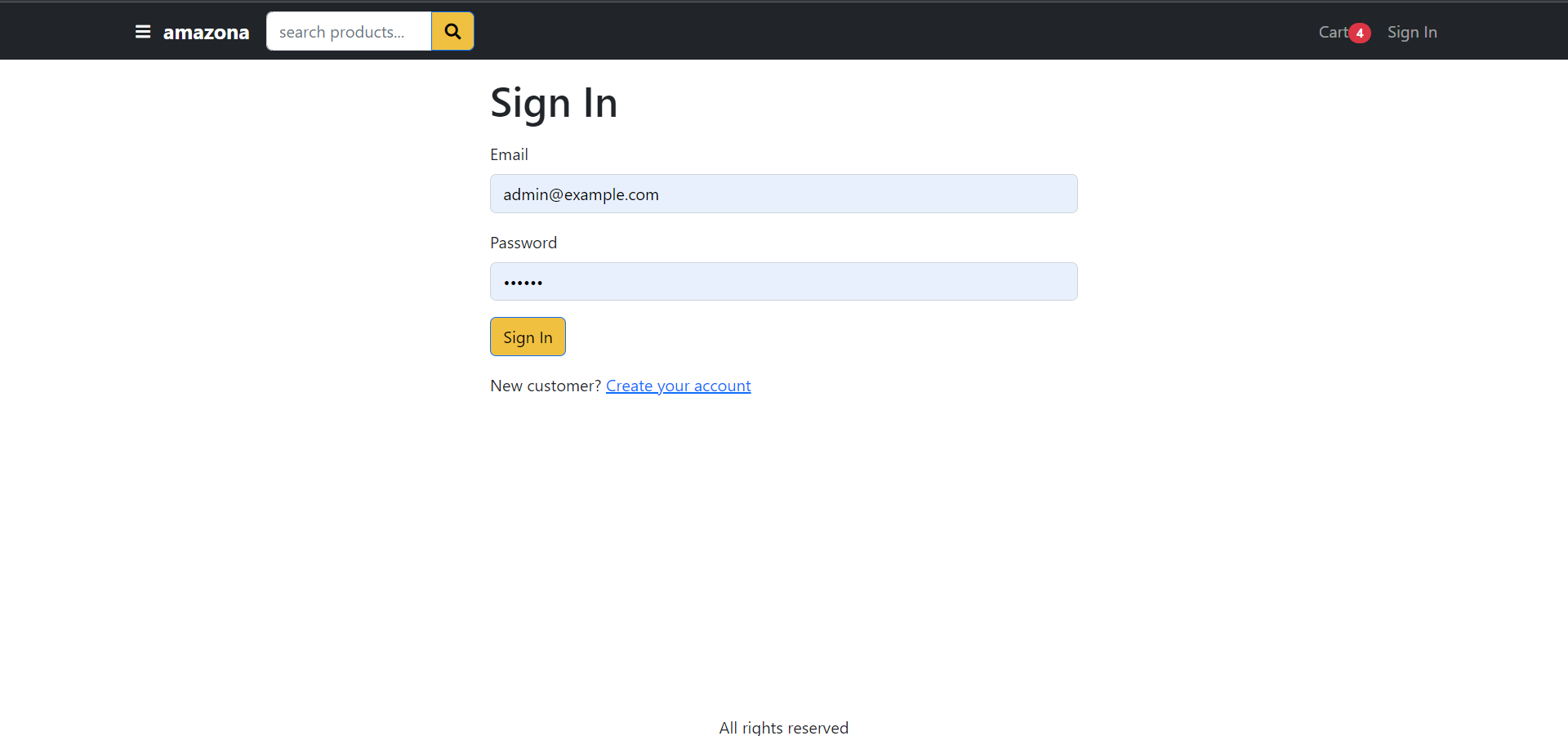


*Hình 3.1.2 Giao diện danh mục sản phẩm.*



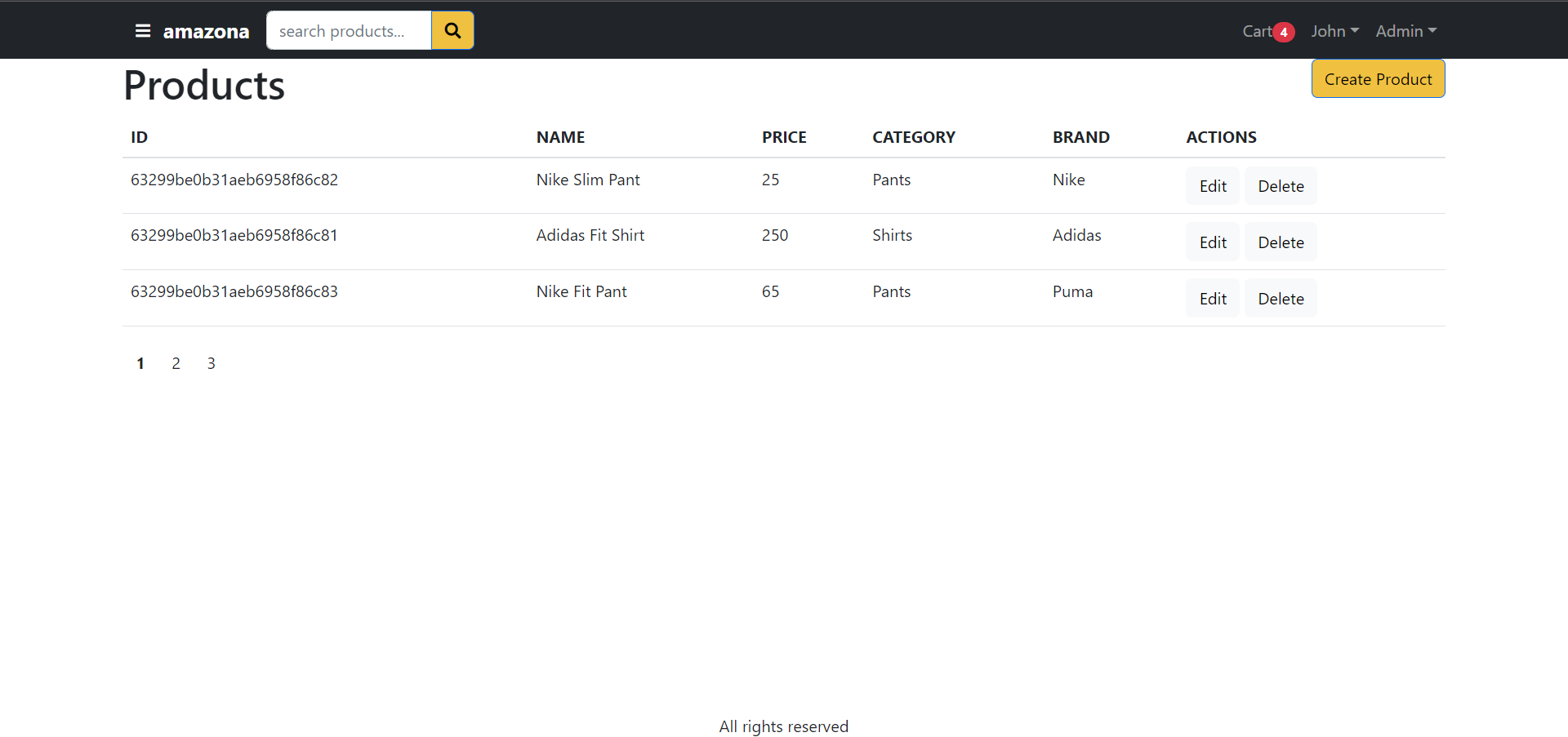
*Hình 3.1.3 Giao diện sản phẩm.*

*Hình 3.1.4. Giao diện giỏ hàng.*

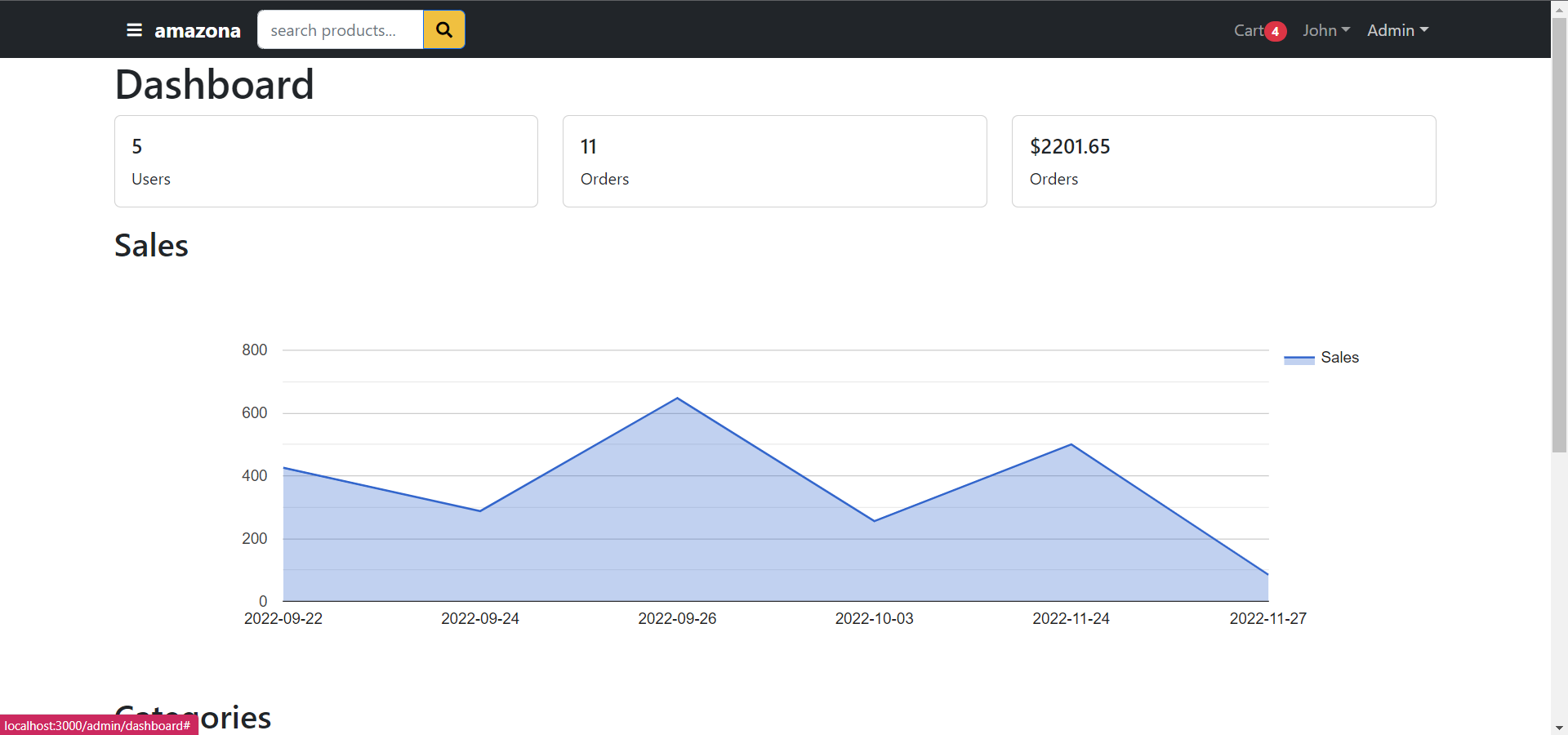
**

*Hình 3.1.6. Giao diện Đăng nhập/ Đăng ký*

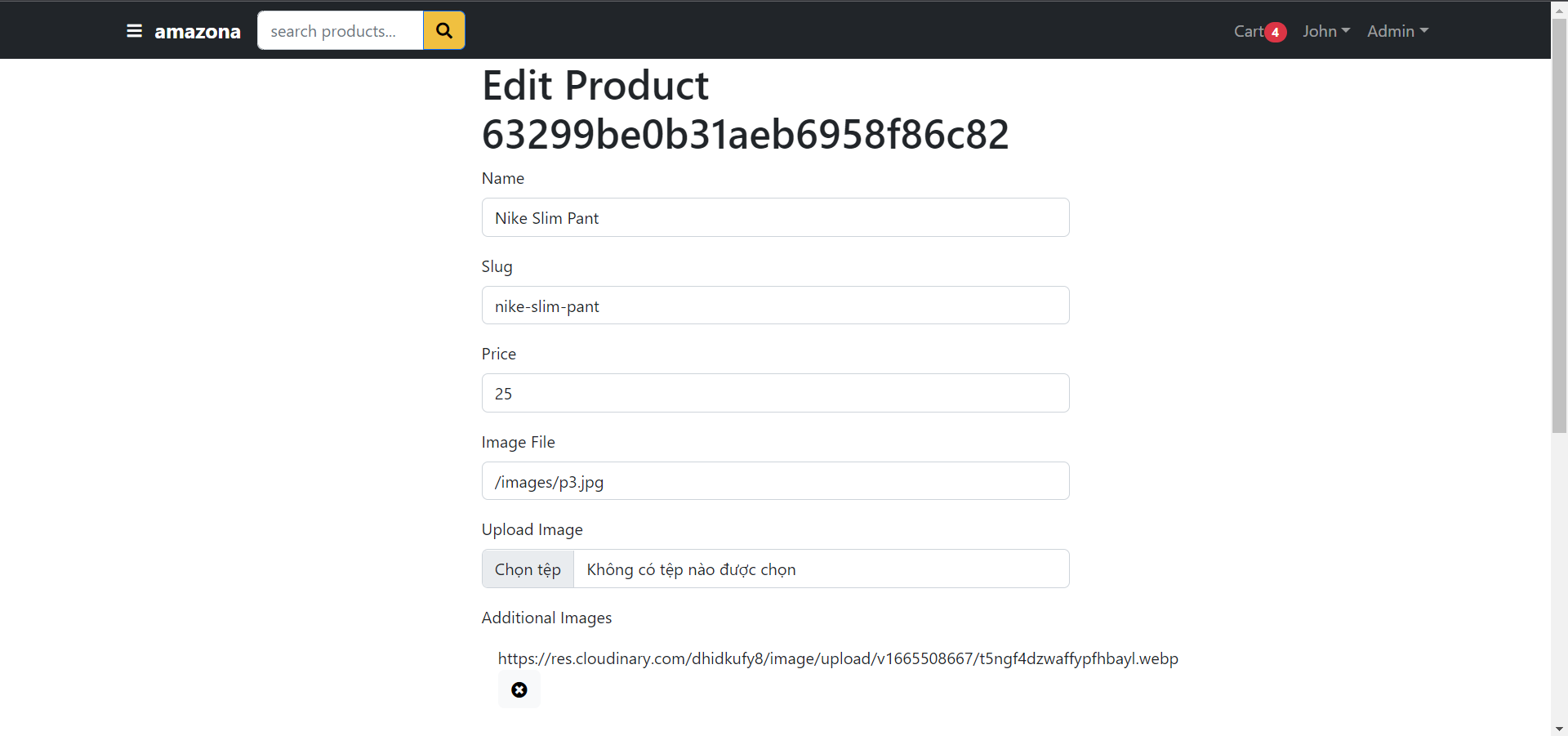
## 3.2. Giao diện quản lý admin



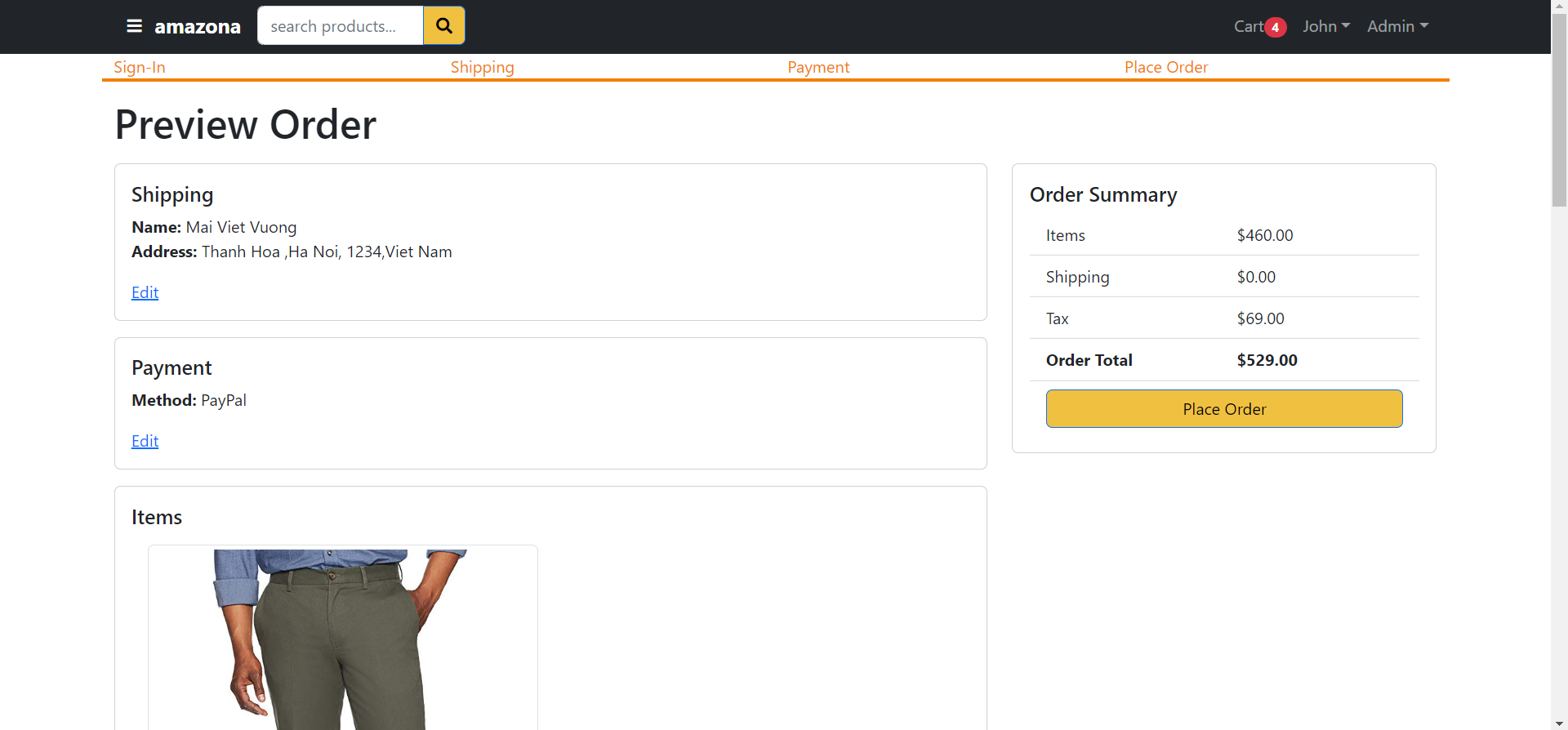
*Hình 3.2.1. Giao diện quản lý sản phẩm*

**

*Hình 3.2.2. Giao diện thống kê*

**

*Hình 3.2.3. Giao diện sản phẩm*

**

*Hình 3.2.4. Giao diện đặt hàng*

# TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN

Sau một thời gian tìm hiểu và nghiên cứu đề tài “**Xây dựng website thời trang Amazona**” em đã phát triển và hoàn thành về cơ bản theo đúng những yêu cầu về nội dung và thời gian đã định. Trong quá trình nghiên cứu và thực hiện đề tài với quyết tâm cao nhưng do hạn chế về kinh nghiệm và kiến thức nên đề tài của em sẽ không thể tránh khỏi những thiếu xót. Em rất mong nhận được ý kiến đóng góp từ quý thầy cô và các bạn để đề tài được hoàn thiện hơn.

**\* Kết quả đạt được:**

- Hoàn thiện được một số chức năng cơ bản của trang web như tìm kiếm và đặt phòng đối với khách hàng, thêm - sửa- xóa thông tin sản phẩm, thông tin khách hàngthống kê …

- Có cơ hội học tập và nghiên cứu nhiều hơn nữa về bộ môn cũng như các công cụ hỗ trợ, ngôn ngữ khác.

**\* Hạn chế của đề tài:**

Do thời gian làm có hạn nên em chưa thể hoàn thiện đầy đủ chức năng của chương trình. Trên đây là một số chức năng cơ bản nhất của website.

**\* Hướng phát triển:**

Để website hoạt động có hiệu quả hơn, em sẽ cố gắng hoàn thiện và bổ sung, sửa đổi một số giao diện sao cho thân thiện với khách hàng và đặc biệt là dễ dàng sử dụng mà tính bảo mật vẫn được đảm bảo.

Hệ thống cần mở rộng thêm nhiều chức năng có tính tự động hóa cao, thống kê được số lượng khách truy cập.

# TÀI LIỆU THAM KHẢO

[1]. Website tài liệu về NodeJS: <https://www.nodebeginner.org/index-vi.html> .

[2]. Website [W3Schools Online Web Tutorials](https://www.w3schools.com/).