**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG**

**LÝ KIỀU CHÍ**

**KHÓA LUẬN TỐT NGHIỆP**

**PHÁT TRIỂN VÀ TRIỂN KHAI TỰ ĐỘNG HỆ THỐNG BÁN HÀNG TRỰC TUYẾN DỰA TRÊN KIẾN TRÚC MICROSERVICES**

**Development and automated deployment of an online sales system based on Microservices architecture.**

**CỬ NHÂN NGÀNH MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG DỮ LIỆU**

**TP. HỒ CHÍ MINH, 2024**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG**

**LÝ KIỀU CHÍ– 20521131**

**KHÓA LUẬN TỐT NGHIỆP**

**PHÁT TRIỂN VÀ TRIỂN KHAI TỰ ĐỘNG HỆ THỐNG BÁN HÀNG TRỰC TUYẾN DỰA TRÊN KIẾN TRÚC MICROSERVICES**

**Development and automated deployment of an online sales system based on Microservices architecture.**

**CỬ NHÂN NGÀNH MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG DỮ LIỆU**

**GIẢNG VIÊN HƯỚNG DẪN**

**Th.S LÊ ANH TUẤN**

**TP. HỒ CHÍ MINH, 2024**

THÔNG TIN HỘI ĐỒNG CHẤM KHÓA LUẬN TỐT NGHIỆP

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số …………………… ngày ………………….. của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

LỜI CẢM ƠN

MỤC LỤC

[Chương 1. GIỚI THIỆU 3](#_Toc185357569)

[1.1. Giới thiệu khóa luận 3](#_Toc185357570)

[1.2. Những nghiên cứu liên quan 3](#_Toc185357571)

[1.3. Mục tiêu, đối tượng, phạm vi nghiên cứu 3](#_Toc185357572)

[1.3.1. Mục tiêu nghiên cứu 3](#_Toc185357573)

[1.3.2. Đối tượng nghiên cứu 3](#_Toc185357574)

[1.3.3. Phạm vi nghiên cứu 3](#_Toc185357575)

[1.4. Cấu trúc khóa luận tốt nghiệp 3](#_Toc185357576)

[Chương 2. CƠ SỞ LÝ THUYẾT 4](#_Toc185357577)

[2.1. Công nghệ xây dựng website 4](#_Toc185357578)

[2.1.1. Kiến trúc Microservice 4](#_Toc185357579)

[2.1.2. Công nghệ Frontend 4](#_Toc185357580)

[2.1.2.1. ReactJS 4](#_Toc185357581)

[2.1.2.2. NextJS 4](#_Toc185357582)

[2.1.2.3. Tailwind CSS 4](#_Toc185357583)

[2.1.3. Công nghệ Backend 4](#_Toc185357584)

[2.1.3.1. NodeJS 4](#_Toc185357585)

[2.1.3.2. Express FrameWork 4](#_Toc185357586)

[2.1.4. Hệ quản trị cơ sở dữ liệu MongoDb 4](#_Toc185357587)

[2.2. Công nghệ triển khai 4](#_Toc185357588)

[2.2.1. Docker 4](#_Toc185357589)

[2.2.2. Kubernetes 4](#_Toc185357590)

[2.2.3. CI/CD pipeline 4](#_Toc185357591)

[2.3. Recommender System với mô hình Machine Learning 4](#_Toc185357592)

[2.4. Giải pháp thanh toán paypal và stripe 4](#_Toc185357593)

[2.4.1. Nền tảng thanh toán paypal 4](#_Toc185357594)

[2.4.2. Nền tảng thanh toán Stripe 4](#_Toc185357595)

[Chương 3. THIẾT KẾ MÔ HÌNH HỆ THỐNG 5](#_Toc185357596)

[3.1. Mô hình hệ thống website 5](#_Toc185357597)

[3.2. Mô hình hệ thống Recommender System 5](#_Toc185357598)

[3.2.1. Quá trình xây dựng mô hình 5](#_Toc185357599)

[3.2.2. Kiểm thử mô hình 5](#_Toc185357600)

[3.3. Mô hình triển khai của hệ thống 5](#_Toc185357601)

[3.3.1. Triển khai trên Kubernetes Cluster 5](#_Toc185357602)

[3.3.1.1. Xây dựng môi trường 5](#_Toc185357603)

[Chương 4. TRIỂN KHAI HỆ THỐNG 6](#_Toc185357604)

[THỰC NGHIỆM 6](#_Toc185357605)

[4.1.1. Thiết lập thực nghiệm 6](#_Toc185357606)

[4.1.1.1. Môi trường thực nghiệm 6](#_Toc185357607)

[4.1.1.2. Chỉ số đánh giá 6](#_Toc185357608)

[4.1.1.3. Ngữ cảnh thực nghiệm 6](#_Toc185357609)

[4.1.2. Kết quả thực nghiệm 6](#_Toc185357610)

[4.1.2.1. Kết quả đo từ mô hình 6](#_Toc185357611)

[4.1.2.2. DOF Ro thật 9](#_Toc185357612)

[Chương 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 12](#_Toc185357613)

[5.1. KẾT LUẬN 12](#_Toc185357614)

[5.2. Hướng phát triển 12](#_Toc185357615)

DANH MỤC HÌNH

DANH MỤC BẢNG

DANH MỤC TỪ VIẾT TẮT

|  |  |
| --- | --- |
| **Từ** | **Nội dung** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

TÓM TẮT KHÓA LUẬN

Trong khóa luận này, tôi đề xuất một giải pháp xây dựng website bán hàngsử dụng kiến trúc microservice, triển khai trên Kubernetes, và tích hợp hệ thống gợi ý sản phẩm (Recommender System) dựa trên AI để giải quyết bài toán đề xuất sản phẩm phù hợp cho người dùng.

Hệ thống được phát triển dựa trên kiến trúc microservice nhằm tách biệt các chức năng chính như dịch vụ người dùng, dịch vụ sản phẩm, dịch vụ thanh toán, dịch vụ gợi ý và dịch vụ quản lý đơn hàng. Việc triển khai trên nền tảng Kubernetes giúp hệ thống có khả năng mở rộng linh hoạt, tự động cân bằng tải và đảm bảo độ tin cậy cao khi vận hành.

Khóa luận tập trung vào phát triển hệ thống gợi ý thông minh bằng cách áp dụng các thuật toán Content-based Filtering và Collaborative Filtering kết hợp với AI/ML để gợi ý sản phẩm dựa trên đặc điểm của đồng hồ và hành vi mua hàng của người dùng. Dữ liệu hệ thống được lưu trữ và quản lý thông qua MongoDB Atlas.

Giao diện website được xây dựng bằng ReactJS/NextJS và Tailwind CSS nhằm đảm bảo tính thân thiện, linh hoạt và trải nghiệm người dùng tốt nhất. Đồng thời, hệ thống còn tích hợp các phương thức thanh toán hiện đại như PayPal và Stripe để đáp ứng nhu cầu thanh toán đa dạng của khách hàng.

Trong quá trình phát triển, hệ thống được triển khai thử nghiệm trên môi trường Kubernetes, áp dụng CI/CD để tự động hóa quy trình phát triển và triển khai. Dữ liệu thu thập từ người dùng được xử lý và đánh giá thông qua các tiêu chí như tỷ lệ gợi ý chính xác, tỷ lệ phản hồi tích cực từ người dùng và thời gian đáp ứng của hệ thống gợi ý.

Kết quả cho thấy hệ thống hoạt động ổn định, đáp ứng được nhu cầu mở rộng và cung cấp gợi ý sản phẩm chính xác, hiệu quả. Khóa luận đã chứng minh tiềm năng ứng dụng của AI trong hệ thống recommender system và khả năng mở rộng của kiến trúc microservice khi kết hợp với Kubernetes trong việc phát triển các ứng dụng web hiện đại.

# GIỚI THIỆU

## Giới thiệu khóa luận

Sự bùng nổ của thương mại điện tử đã làm thay đổi hoàn toàn hành vi mua sắm của người tiêu dùng. Chỉ với vài cú click chuột, người dùng có thể dễ dàng tìm kiếm và mua sắm hàng hóa từ khắp nơi trên thế giới. Tuy nhiên, sự tăng trưởng nhanh chóng của các nền tảng thương mại điện tử cũng đặt ra những thách thức lớn về mặt kỹ thuật. Việc quản lý và vận hành một website thương mại điện tử với hàng triệu lượt truy cập mỗi ngày đòi hỏi các giải pháp linh hoạt, hiệu quả và dễ dàng mở rộng.

Nghiên cứu này tập trung vào việc ứng dụng kiến trúc microservice và nền tảng Kubernetes để giải quyết vấn đề quá tải và nâng cao hiệu năng của các hệ thống thương mại điện tử. Bằng cách chia nhỏ ứng dụng thành các dịch vụ nhỏ, độc lập, chúng ta có thể dễ dàng quản lý, cập nhật và mở rộng từng dịch vụ một. Ngoài ra, Kubernetes sẽ giúp tự động hóa quá trình triển khai, quản lý và cân bằng tải cho các dịch vụ này. Bên cạnh đó, việc tích hợp các mô hình học máy vào hệ thống Recommender System sẽ giúp nâng cao trải nghiệm người dùng bằng cách đưa ra những gợi ý sản phẩm phù hợp và cá nhân hóa.

Nghiên cứu này nhằm mục tiêu chứng minh rằng kiến trúc microservice kết hợp với Kubernetes và các mô hình học máy là một giải pháp hiệu quả để xây dựng và vận hành các hệ thống thương mại điện tử hiện đại, đáp ứng được nhu cầu ngày càng cao của người tiêu dùng.

## Những nghiên cứu liên quan

## Mục tiêu, đối tượng, phạm vi nghiên cứu

### Mục tiêu nghiên cứu

* Phát triển một hệ thống bán hàng trực tuyến dựa trên kiến trúc Microservices để tối ưu hóa hiệu suất, khả năng mở rộng và tính bảo trì của hệ thống.
* Phân tích và xây dựng các services độc lập cho các chức năng quản lý sản phẩm, đơn hàng, người dùng và thanh toán.
* Xây dựng và triển khai hệ thống khuyến nghị dựa trên thuộc tính của sản phẩm, giúp gợi ý các sản phẩm tương tự với những sản phẩm mà người dùng đã quan tâm.
* Tích hợp quy trình CI/CD để tự động hóa việc kiểm thử, triển khai và quản lý hệ thống.
* Đảm bảo khả năng mở rộng, dễ bảo trì, tự khắc phục lỗi, và phân bổ tài nguyên linh hoạt thông qua nền tảng ảo hóa và điều phối container; tích hợp hệ thống giám sát các dịch vụ và tài nguyên hệ thống.
* Tối ưu hóa hiệu suất hệ thống để đảm bảo tính ổn định và khả năng xử lý khối lượng giao dịch lớn.

### Đối tượng nghiên cứu

* **Hệ thống:**

Hệ thống bán hàng trực tuyến bao gồm các services quản lý sản phẩm, đơn hàng, người dùng, thanh toán được cấu hình giao tiếp với các dịch vụ khác thông qua API và triển khai tự động.

* **Công nghệ:**

+ Công nghệ hệ thống bán hàng: ReactJS, NodeJ, và MongoDB.

+ Hệ thống khuyến nghị: Phương pháp Content-Based Filtering, Collaborative Filtering, Neural Collaborative Filtering.

+ Kiến trúc Microservices, công nghệ container (Docker), nền tảng quản lý Kubernetes,

+ Các công cụ triển khai CI/CD pipelines (GitHub Actions, SonarQube, Trivy) và giám sát hệ thống (Prometheus, Grafana).

### Phạm vi nghiên cứu

**• Không gian:**

Nghiên cứu sẽ được triển khai trên nền tảng Docker, Kubernetes cluster để kiểm thử và đánh giá hệ thống.

**• Thời gian:**

Từ 09/2024 đến 12/2024, nghiên cứu sẽ hoàn thành, bao gồm việc phân tích thiết kế, phát triển, triển khai và đánh giá hệ thống.

**• Nội dung:**

Tập trung vào phát triển các Microservices cho hệ thống bán hàng và tích hợp quy trình tự động hóa triển khai.

## Cấu trúc khóa luận tốt nghiệp

Tôi xin trình bày nội dung của Khoá luận theo cấu trúc như sau:

* Chương 1: Giới thiệu tổng quan về đề tài của Khóa luận và những nghiên cứu liên quan.
* Chương 2: Trình bày cơ sở lý thuyết và kiến thức nền tảng liên quan đến đề tài.
* Chương 3: Trình bày chi tiết thiết kết mô hình website và chi tiết mô hình webite trên môi trường kubernetes với quy trình triển khai tự động.
* Chương 4: Trình bày thực nghiệm và đánh giá.
* Chương 5: Kết luận và hướng phát triển của đề tài.

# CƠ SỞ LÝ THUYẾT

## Công nghệ xây dựng website

### Kiến trúc Microservice

Microservices là một cách tiếp cận kiến ​​trúc và tổ chức đối với phát triển phần mềm, trong đó phần mềm bao gồm các dịch vụ độc lập nhỏ giao tiếp qua các API được xác định rõ ràng. Các dịch vụ này thuộc sở hữu của các nhóm nhỏ, độc lập.

Kiến trúc Microservices giúp ứng dụng dễ mở rộng quy mô hơn và phát triển nhanh hơn, cho phép đổi mới và rút ngắn thời gian đưa các tính năng mới ra thị trường.

Đặc điểm của Microservices:

**Tách biệt chức năng**: Mỗi dịch vụ tập trung vào một nhiệm vụ cụ thể và độc lập.

**Triển khai độc lập:** Các dịch vụ có thể được triển khai và cập nhật riêng mà không ảnh hưởng đến các dịch vụ khác.

**Khả năng mở rộng linh hoạt**: Mỗi dịch vụ có thể mở rộng độc lập dựa trên nhu cầu sử dụng.

**Giao tiếp nhẹ và tiêu chuẩn hóa:** Giao tiếp giữa các dịch vụ thường thông qua API RESTful, gRPC, hoặc Message Broker.

**Tự động hóa và DevOps:** Microservices phù hợp với các công cụ CI/CD, Docker, và Kubernetes để triển khai tự động hóa.

**Công nghệ đa dạng:** Mỗi dịch vụ có thể được viết bằng các công nghệ khác nhau (Node.js, Python, Java, Go,...) phù hợp với mục tiêu của nó

### Công nghệ Frontend

#### ReactJS

React là thư viện JavaScript được phát triển và duy trì bởi Meta (Facebook), dùng để xây dựng giao diện người dùng (UI) theo hướng component-based. React giúp việc tạo các ứng dụng web hiện đại trở nên nhanh chóng và hiệu quả thông qua các đặc điểm chính sau:

Đặc điểm chính của React:

**Component-based**: Ứng dụng được chia thành các components nhỏ, tái sử dụng được. Mỗi component chịu trách nhiệm một phần của giao diện.

**Virtual DOM:** React sử dụng Virtual DOM để tăng hiệu suất. Khi trạng thái thay đổi, React sẽ so sánh Virtual DOM với DOM thực tế và chỉ cập nhật những phần thay đổi.

**Declarative UI:** React sử dụng cách tiếp cận declarative để viết UI. Bạn chỉ cần định nghĩa giao diện mong muốn, React sẽ quản lý việc cập nhật nó khi dữ liệu thay đổi.

**One-Way Data Flow:** Dữ liệu trong React di chuyển theo một hướng từ cha sang con thông qua props, giúp dễ dàng quản lý trạng thái ứng dụng.

#### NextJS

Next.js là một framework được xây dựng trên React để mở rộng các khả năng của React như server-side rendering (SSR), static site generation (SSG) và routing. Next.js giúp tăng hiệu suất và tối ưu SEO cho các ứng dụng React.

Các tính năng chính của Next.js:

* **Server-Side Rendering (SSR):** Next.js cho phép render trang web trên server trước khi gửi về trình duyệt. Điều này giúp tải trang nhanh hơn và tối ưu hóa SEO.
* **Static Site Generation (SSG):** Next.js hỗ trợ tạo trang tĩnh trong quá trình build, giúp cải thiện tốc độ tải trang.
* **File-based Routing**: Hệ thống routing trong Next.js dựa vào cấu trúc file. Ví dụ, file pages/about.js sẽ tương ứng với đường dẫn /about.
* **API Routes:** Next.js cho phép tạo các API endpoints thông qua các file trong thư mục pages/api.
* **Image Optimization:** Tính năng tối ưu hình ảnh giúp giảm kích thước và cải thiện hiệu suất tải trang.

#### Tailwind CSS

Tailwind CSS là một utility-first CSS framework (framework CSS ưu tiên tiện ích) dùng để xây dựng giao diện nhanh chóng và linh hoạt. Không giống như các framework CSS khác như Bootstrap hay Foundation, Tailwind không cung cấp các components sẵn có (như button, card), thay vào đó cung cấp các class tiện ích (utility classes) giúp bạn tạo giao diện tùy chỉnh trực tiếp trong mã HTML.

Đặc điểm của Tailwind CSS:

* **Utility-First**: Tailwind CSS cung cấp các class tiện ích nhỏ, đơn giản để bạn xây dựng giao diện một cách trực tiếp và nhanh chóng mà không cần viết CSS tùy chỉnh.
* **Tùy chỉnh dễ dàng**: Tailwind cung cấp tệp cấu hình tailwind.config.js để bạn có thể tùy chỉnh theme, màu sắc, khoảng cách, font chữ,... phù hợp với dự án của mình.
* **Không cần viết CSS thủ công:** Tailwind giúp giảm thiểu việc viết CSS thủ công bằng cách sử dụng class tiện ích. Bạn chỉ cần sử dụng các class đã có để xây dựng giao diện.
* **Responsive Design**: Tailwind hỗ trợ responsive design bằng cách sử dụng các breakpoints có sẵn.
* **Tối ưu hóa kích thước file CSS**: Tailwind kết hợp với PurgeCSS để loại bỏ các class CSS không sử dụng, giúp giảm dung lượng file CSS trong môi trường production.
* **Hỗ trợ Dark Mode:** Tailwind cung cấp cơ chế hỗ trợ Dark Mode dễ dàng thông qua cấu hình.

### Công nghệ Backend

#### NodeJS

Node.js là một môi trường chạy JavaScript phía server-side được xây dựng trên V8 JavaScript Engine (công cụ thực thi JavaScript của Google Chrome). Node.js cho phép bạn sử dụng JavaScript để phát triển các ứng dụng phía server, thay vì chỉ dùng trên trình duyệt.

#### Express FrameWork

Express.js là một web framework nhỏ gọn và mạnh mẽ cho Node.js, được sử dụng để xây dựng các ứng dụng web và API một cách nhanh chóng và đơn giản. Được ra mắt lần đầu vào năm 2010, Express cung cấp một tập hợp các tính năng mạnh mẽ cho các ứng dụng web backend và các dịch vụ RESTful API.

Các đặc điểm chính của Express.js:

* **Minimal và Unopinionated:** Express có thiết kế tối giản, chỉ cung cấp các tính năng cơ bản nhất để xây dựng ứng dụng. Điều này giúp lập trình viên tự do lựa chọn các thư viện và công cụ phù hợp.
* **Routing**: Express cung cấp hệ thống định tuyến linh hoạt để xử lý các **endpoint** khác nhau dựa trên phương thức HTTP (GET, POST, PUT, DELETE,...) và URL.
* **Middleware:** Middleware trong Express là các hàm trung gian giúp xử lý yêu cầu trước khi gửi phản hồi. Middleware được sử dụng để:

+ Thực hiện xác thực (authentication).

+ Ghi log (logging).

+ Phân tích dữ liệu từ request body.

+ Xử lý lỗi.

* **Xử lý dữ liệu với Body-parser**: Express hỗ trợ đọc dữ liệu từ requestbody thông qua middleware express.json() hoặc express.urlencoded().
* **Template Engines**: Express hỗ trợ các templateengines để tạo giao diện động như:
* EJS (Embedded JavaScript)
* Pug (trước đây là Jade)
* Handlebars
* **Xây dựng RESTful API:** Express là công cụ lý tưởng để xây dựng các dịch vụ RESTfulAPI, với khả năng xử lý nhanh chóng các phương thức HTTP như GET, POST, PUT và DELETE.

### Hệ quản trị cơ sở dữ liệu MongoDb

MongoDB là một hệ quản trị cơ sở dữ liệu NoSQL mã nguồn mở, được thiết kế để lưu trữ và truy vấn dữ liệu theo dạng tài liệu (document-oriented). Thay vì lưu trữ dữ liệu dưới dạng bảng như các cơ sở dữ liệu quan hệ (SQL), MongoDB lưu trữ dữ liệu dưới dạng JSON hoặc BSON (Binary JSON).

MongoDB được phát triển bởi MongoDB Inc. và được sử dụng rộng rãi trong các ứng dụng hiện đại nhờ khả năng linh hoạt, mở rộng và hiệu suất cao.

## Công nghệ triển khai

### Docker

Docker là nền tảng phần mềm mã nguồn mở cho phép dựng, kiểm thử và triển khai ứng dụng một cách nhanh chóng. Docker đóng gói phần mềm vào các đơn vị được tiêu chuẩn hóa được gọi là container có mọi thứ mà phần mềm cần để chạy, bao gồm thư viện, công cụ hệ thống, mã và thời gian hoạt động. Với Docker, có thể quản lý cơ sở hạ tầng giống như cách quản lý ứng dụng giúp nhanh chóng triển khai và thay đổi quy mô ứng dụng vào bất kỳ môi trường nào và biết chắc rằng mã của bạn sẽ chạy được. [1] [2]

### Kubernetes

Kubernetes là một nền tảng nguồn mở, khả chuyển, có thể mở rộng để quản lý các ứng dụng được đóng gói và các service, giúp thuận lợi trong việc cấu hình và tự động hoá việc triển khai ứng dụng. Kubernetes là một hệ sinh thái lớn và phát triển nhanh chóng với các dịch vụ, sự hỗ trợ và công cụ có sẵn rộng rãi.

Kubernetes cung cấp cho bạn một framework để chạy các hệ phân tán một cách mạnh mẽ. Nó đảm nhiệm việc nhân rộng và chuyển đổi dự phòng cho ứng dụng của bạn, cung cấp các mẫu deployment.

### CI/CD pipeline

CI/CD Pipeline (Continuous Integration/Continuous Deployment) là một quy trình tự động hóa trong phát triển phần mềm, giúp tăng tốc độ phát triển, thử nghiệm và triển khai phần mềm một cách liên tục và đáng tin cậy.

CI/CD là một phần quan trọng trong phương pháp phát triển phần mềm DevOps, cho phép các đội ngũ phát triển đưa sản phẩm từ giai đoạn viết mã đến triển khai sản phẩm cuối cùng một cách hiệu quả.

* CI – Continuous Integration (Tích hợp liên tục):
* Là quá trình tích hợp mã nguồn từ nhiều lập trình viên vào nhánh chính (main branch) thường xuyên (hàng ngày hoặc nhiều lần trong ngày).
* Mã sẽ được tự động build, kiểm thử và kiểm tra chất lượng để phát hiện lỗi sớm nhất có thể.
* CD – Continuous Deployment/Delivery (Triển khai/Phân phối liên tục):
* Continuous Delivery: Sau khi CI hoàn tất, phần mềm sẵn sàng để triển khai, nhưng cần sự phê duyệt của con người trước khi đưa lên môi trường production.
* Continuous Deployment: Hoàn toàn tự động đưa phần mềm lên production mà không cần sự can thiệp của con người.

## Recommender System với mô hình Machine Learning

## Giải pháp thanh toán paypal và stripe

### Nền tảng thanh toán paypal

PayPal là một nền tảng thanh toán trực tuyến toàn cầu cho phép người dùng thực hiện các giao dịch thanh toán và chuyển tiền một cách an toàn và nhanh chóng thông qua Internet. PayPal được thành lập vào năm 1998, trụ sở chính tại San Jose, California, Mỹ. Hiện nay, PayPal hoạt động trên hơn 200 quốc gia và vùng lãnh thổ và hỗ trợ nhiều loại tiền tệ.

Các tính năng chính

* **Thanh toán trực tuyến**: Mua hàng trên các website hỗ trợ PayPal.
* **Chuyển tiền quốc tế**: Gửi và nhận tiền giữa các quốc gia.
* **Bảo vệ giao dịch**: Bảo vệ người mua và người bán khỏi gian lận.
* **Tích hợp API**: Hỗ trợ doanh nghiệp tích hợp thanh toán vào website hoặc ứng dụng.

### Nền tảng thanh toán Stripe

Stripe là một nền tảng thanh toán trực tuyến phổ biến, cung cấp các dịch vụ xử lý thanh toán cho các doanh nghiệp và tổ chức trên toàn cầu. Được thành lập vào năm 2010 tại Mỹ, Stripe cho phép các doanh nghiệp tích hợp thanh toán vào website hoặc ứngdụng một cách dễ dàng và an toàn.

Các tính năng chính:

* **Xử lý thanh toán toàn cầu**: Hỗ trợ nhiều loại thẻ và phương thức thanh toán quốc tế.
* **Thanh toán định kỳ**: Cho phép thiết lập thanh toán tự động cho các dịch vụ định kỳ.
* **Stripe Checkout**: Cung cấp giao diện thanh toán sẵn có, dễ tích hợp.
* **Hỗ trợ API mạnh mẽ**: Cho phép tùy chỉnh quy trình thanh toán linh hoạt.
* **Bảo mật cao**: Tuân thủ tiêu chuẩn PCI-DSS, mã hóa thông tin thanh toán.

# THIẾT KẾ MÔ HÌNH HỆ THỐNG

## Mô hình hệ thống website

## Mô hình hệ thống Recommender System

### Quá trình xây dựng mô hình

### Kiểm thử mô hình

## Mô hình triển khai của hệ thống

### Triển khai trên Kubernetes Cluster

#### Xây dựng môi trường

# TRIỂN KHAI HỆ THỐNG

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Ở chương này, Tôi đưa ra những kết luận về nghiên cứu, những hạn chế, và đồng thời đưa ra hướng cải thiện và phát triển.

## KẾT LUẬN

Trong khuôn khổ của khóa luận này, tôi đã tập trung nghiên cứu và triển khai thành công hệ thống website bán đồng hồ hiện đại dựa trên kiến trúc microservice, triển khai trên nền tảng Kubernetes, đồng thời tích hợp hệ thống gợi ý sản phẩm thông minh dựa trên AI. Kết quả đạt được chứng minh rằng kiến trúc microservice giúp tăng tính linh hoạt, dễ bảo trì và khả năng mở rộng của hệ thống, trong khi Kubernetes đảm bảo quản lý tài nguyên hiệu quả và tự động hóa triển khai. Hệ thống recommender system đã đáp ứng tốt mục tiêu cá nhân hóa trải nghiệm người dùng thông qua các thuật toán như Content-based Filtering và Collaborative Filtering, đồng thời giải quyết hiệu quả bài toán cold start. Giao diện người dùng được xây dựng bằng ReactJS kết hợp với Tailwind CSS mang lại trải nghiệm hiện đại và thân thiện, trong khi các phương thức thanh toán tích hợp như PayPal, MoMo, và ngân hàng nội địa đáp ứng đa dạng nhu cầu thanh toán. Hệ thống không chỉ mang lại giá trị thực tiễn cao cho các doanh nghiệp kinh doanh trực tuyến mà còn đóng góp ý nghĩa học thuật trong việc áp dụng kiến trúc microservice và AI vào thực tế..

## Hướng phát triển

Trong tương lai, tôi có mong muốn hoàn thiện chuỗi sự kiện mà chúng tôi đề suất. Cụ thể là :

* **Phát triển tính năng tìm kiếm thông minh**: Tích hợp Visual Search sử dụng AI cho phép người dùng tìm kiếm sản phẩm qua hình ảnh. Sử dụng các mô hình nhận diện hình ảnh như ResNet, YOLO, hoặc CLIP để tăng độ chính xác trong tìm kiếm sản phẩm.
* **Mở rộng hệ thống thanh toán**: Tích hợp thêm các hình thức thanh toán phổ biến khác như VNPay, ZaloPay, MOMO, ...hoặc các cổng thanh toán quốc tế mới để đáp ứng đa dạng nhu cầu khách hàng.
* **Mở rộng trải nghiệm người dùng**: Tối ưu giao diện người dùng trên các nền tảng di động và ứng dụng PWA (Progressive Web Apps). Tích hợp chatbot thông minh để hỗ trợ khách hàng 24/7, sử dụng các mô hình NLP tiên tiến như **GPT-based chatbots** hoặc **Dialogflow**.

Nhìn chung, chúng tôi muốn cho sản phẩm này có thể cải thiện nhiều hơn cho trải nghiệm người dùng và cải thiện hiệu quả mà các thông tin thu thập được từ người dùng để tạo nên và tích hợp các máy học phù hợp giúp cho việc mua hàng hóa trên Internet dễ hơn bao giờ hết.

TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | https://docs.docker.com/get-started/docker-overview/ |
| [2] | https://aws.amazon.com/vi/docker/. |
| [3] |
| [4] |
| [5] |
| [6] |