**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

NGUYỄN THÀNH TÍN MSSV: N17DCCN153 CHUYÊN NGÀNH: CÔNG NGHỆ PHẦN MỀM 2017-2022 Lớp: D17CQCP02-N

**Gáy**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

------------------------------



**ĐỒ ÁN**

**TỐT NGHIỆP ĐẠI HỌC**

***Đề tài*: “Xây dựng Game 2D Soldier's Avenger”**

**Người hướng dẫn : Ths. Huỳnh Trung Trụ**

**Sinh viên thực hiện : Nguyễn Thành Tín**

**Mã số sinh viên : N17DCCN153**

**Lớp : D17CQCP02-N**

**Hệ** **: Chính Quy**

TP. HCM

2021

**TP. HỒ CHÍ MINH, NĂM 2021**

**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

------------------------------



**ĐỒ ÁN**

**TỐT NGHIỆP ĐẠI HỌC**

***Đề tài*:**

**“Xây dựng Game 2D Soldier's Avenger”**

**Người hướng dẫn : Ths. Phan Thanh Hy**

**Sinh viên thực hiện : Lê Gia Bảo Phú**

**Mã số sinh viên : N17DCCN131**

**Lớp : D17CQCP02-N**

**Hệ** **: Chính Quy**

**TP. HỒ CHÍ MINH, NĂM 2021**

BM.HV-KT.01/06

|  |  |
| --- | --- |
| HỌC VIỆN | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM** |
| CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG | **Độc lập – Tự do – Hạnh phúc** |
| CƠ SỞ TẠI TP. HỒ CHÍ MINH |  |
| **Khoa: Công nghệ thông tin 2** | *TP. Hồ Chí Minh, ngày 21 tháng 09 năm 2021* |

ĐỀ TÀI ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC

Căn cứ Quyết định số: 402/QĐ-HVCS, ngày 21 tháng 09 năm 2021 của Phó Giám đốc Học viện – Phụ trách Cơ sở tại TP. Hồ Chí Minh về việc “phê duyệt danh sách giáo viên hướng dẫn và giao đề tài đồ án tốt nghiệp cho sinh viên Đại học chính quy Khóa 2017-2022 Ngành Công nghệ thông tin, An toàn thông tin và Công nghệ đa phương tiện”;

Khoa Công nghệ thông tin 2 giao nhiệm vụ thực hiện Đồ án tốt nghiệp cho sinh viên:

1. **Họ và tên sv : Nguyễn Thành Tín Mã SV : N17DCCN153**

**Lớp : D17CQCP02-N Khóa : 2017-2022**

**Ngành đào tạo : Công nghệ thông tin Hệ đào tạo : Đại học Chính quy**

1. **Tên đề tài tốt nghiệp: Xây dựng Game 2D Soldier's Avenger**
2. **Nội dung chính của Đồ án:**

**Lý thuyết:**

Tìm hiểu nghiệp vụ, ngôn ngữ và công nghệ sử dụng:

-Tìm hiểu và xây dựng game trên Engine Unity3D trên thiết bị Mobile và PC

-Tìm hiểu ngôn ngữ C# , JavaScript

-Tìm hiểu NodeJS, WebSocket

-Database: MongoDb

**Thực hành:**

-Thiết kế Cơ sở dữ liệu hệ thống.

-Thiết kế và xây dựng hệ thống UI : Home, Inventory, Daily Reward, List Mission, Leader Board, GameSetting , GamePlay,.....

- Xử lí đăng nhập : login , register.

- Xây dựng hệ thống cốt truyện chính.

- Xây dựng và thiết kế hệ thống Player.

- Xây dựng và thiết kế hệ thống Enemy.

- Xây dựng và thiết kế Map, Trap.

- Xây dựng và thiết kế Bonus Items.

- Xây dựng hệ thống effect chiến đấu, nhạc nền, âm thanh.

- Xây dựng GamePlay :

+ Map : xây dựng Map cho mỗi màn chơi, hệ thống bẫy tương tác với Player.

+ Player : chiến đấu với Enemy, nhận vàng, kinh nghiệm nâng cấp sức mạnh, kĩ năng, vũ khí,... qua từng màn chơi.

+ Enemy: chiến đấu với Player , xử lí AI né tránh, giữ khoảng cách, tấn công Player.

- Đưa game lên CHPlay.

1. **Cơ sở dữ liệu ban đầu:**
2. **Giáo viên hướng dẫn: ThS. Huỳnh Trung Trụ**
3. **Ngày giao đề tài: 27/09/2021**
4. **Ngày nộp quyển: 07/12/2021**

**TRƯỞNG KHOA CNTT**

**Nơi nhận:**

* *Sinh viên có tên tại khoản 1;*
* *Lưu: VP Khoa.*

**TS. Nguyễn Hồng Sơn**

# MỞ ĐẦU

Hiện nay, công nghệ thông tin đã có những bước phát triển mạnh mẽ theo cả chiều rộng và sâu.Máy tính điện tử không còn là một thứ phương tiện quý hiếm mà đang ngày càng trở thành một công cụ làm việc và giải trí thông dụng của con người, không chỉ ở công sở mà còn ngay cả trong gia đình.

Đi cùng với sự tiến hóa công nghệ, nhu cầu giải trí của con người cũng được nâng tầm phát triển, trong đó Game điện tử là một khái niệm được sinh ra và phát triển theo quá trình phát triển mạnh mẽ của máy móc điện tử và Internet.

Game điện tử trên thế giới hiện nay đã đến những bước tiến lớn nhờ công nghệ hiện đại, hướng con người đến trải nghiệm tốt nhất về tất cả các mặt: âm thanh, đồ họa, gameplay, cốt truyện…. trên nhiều nền tảng như mobile, PC, gamepad, playstation,….Trên thế giới, game điên tử đã trở nên phổ biến toàn cầu, trở thành một nền công nghiệp phát triển mạnh mẽ là nơi để đầu tư chất xám hiệu quả. Mỗi giai đoạn phát triển và mỗi nền tảng đều có những game điện tử để lại dấu mốc quan trọng:Mario, HaftLife, GTA, …

Ở Việt Nam cũng có rất nhiều doanh nghiệp cá nhân đã và đang phát triển mạnh về lĩnh vực Game điện tử, nhiều studio cũng như cá nhân đạt nhiều thành tựu nổi tiếng toàn cầu : Emobi Game với game 7554 Chiến dịch Điện Biên Phủ gây tiếng vang lớn trên cộng đồng game thế giới về cốt truyện , hoặc gần đây với cú nổ mang tính lịch sử đối với thể loại game đơn giản và vô tận mang tên Flappy Bird của lập trình viên Nguyễn Hà Đông vang danh Việt Nam trên thị trường game quốc tế. Những phát triển trên đã tạo đà và cả động lực , ý tưởng cho thế hệ mới phát triển mạnh mẽ hơn, tiếp cận với công nghệ tiên tiến trên thế giới.

Để tiếp cận và góp phần đẩy mạnh sự phổ biến của game điên tử ở Việt Nam, em đã quyết định thực hiện đề tài “**Xây dựng Game 2D Soldier's Avenger**”. Việc xây dựng game 2D trên sẽ tạo bước đà đủ lớn để tiếp cận với game điện tử về mọi mặt, phát triển kĩ năng hoàn thiện để tiến tới các mục tiêu lớn hơn trong tương lai.Đồng thời, tạo ra một game 2D thân thiện với người chơi, đơn giản cuốn hút và đặc biệt là tính giải trí cho mọi người sau những khoảng thời gian làm việc và học tập mệt mỏi.

# LỜI CẢM ƠN

Đồ án tốt nghiệp đóng một vai trò rất quan trọng và mang lại nhiều hiệu quả tích cực đối với quá trình học tập và nghiên cứu của sinh viên khi còn ngồi trên ghế giảng đường Đại Học. Đồ án tốt nghiệp giúp cho sinh viên hệ thống hóa lại kiến thức mà sinh viên đã được học, tìm hiểu và nắm bắt được những kiến thức mới cũng như đi sâu vào những vấn đề và lĩnh vực mình yêu thích.

Bản báo cáo này trình bày về quá trình xây dựng và hoàn thiện game 2D cơ bản : xậy dựng bản đồ, nhân vật , kẻ thù cũng như gameplay và các chức năng cần thiết của một game điện tử trên nền tảng PC và mobile.

Trong quá trình thực hiện không thể tránh những thiếu sót, kính mong thầy và các bạn thông cảm.

Em xin chân thành cảm ơn sự giúp đỡ nhiệt tình của thầy **Huỳnh Trung Trụ** trong quá trình em thực hiện và hoàn thành đồ án này. Đồng thời, em xin gửi lời cảm ơn chân thành đến các thầy cô trong **Khoa Công nghệ Thông tin 2** đã tạo mọi điều kiện thuận lợi để em có thể thực hiện được bài báo cáo tốt nghiệp này

**TP.Hồ Chí Minh, ngày 7 tháng 10 năm 2021**

**Sinh viên**

**Nguyễn Thành Tín**

**MỤC LỤC**

[MỞ ĐẦU 4](#_Toc86672838)

[LỜI CẢM ƠN 5](#_Toc86672839)

[DANH SÁCH CÁC BẢNG, SƠ ĐỒ, HÌNH 8](#_Toc86672840)

[KÝ HIỆU CÁC CỤM TỪ VIẾT TẮT 10](#_Toc86672841)

[CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI 11](#_Toc86672842)

[1.1 Lý do chọn đề tài: 11](#_Toc86672843)

[1.2 Mục đích, ý nghĩa chọn đề tài: 11](#_Toc86672844)

[1.3 Phương pháp nghiên cứu: 12](#_Toc86672845)

[1.4 Phạm vi, đối tượng đề tài: 13](#_Toc86672846)

[1.5 Ý nghĩa thực tiễn đề tài 13](#_Toc86672847)

[1.6 Cấu trúc báo cáo 13](#_Toc86672848)

[CHƯƠNG 2: CƠ SỞ LÝ LUẬN VÀ NỀN TẢNG XÂY DỰNG HỆ THỐNG 15](#_Toc86672849)

[2.1. Tổng quan về thương mại điện tử 15](#_Toc86672850)

[2.1.1. Khái niệm thương mại điện tử 15](#_Toc86672851)

[2.1.2. Lợi ích của việc sử dụng thương mại điện tử 15](#_Toc86672852)

[2.1.3. Các yêu cầu trong thương mại điện tử 15](#_Toc86672853)

[2.1.4. Các mô hình trong thương mại điện tử 16](#_Toc86672854)

[2.2. Xây dựng hệ thống 17](#_Toc86672855)

[2.2.1 Giới thiệu IntelliJ IDE 17](#_Toc86672856)

[2.2.2 Giới thiệu Visual Studio Code 19](#_Toc86672857)

[2.2.3. Giới thiệu về PostgreSQL 20](#_Toc86672858)

[2.3. Tìm hiểu về Java Spring Boot Framework 23](#_Toc86672859)

[2.3.1 Ngôn ngữ Java 23](#_Toc86672860)

[2.3.2 Tìm hiểu về Spring Boot Framework 26](#_Toc86672861)

[2.3.3 So sánh với các framework khác 29](#_Toc86672862)

[2.4. Tìm hiểu về ReactJS 30](#_Toc86672863)

[2.5. Tổng quan về RestFul API 31](#_Toc86672864)

[2.6 Bảo mật Website 31](#_Toc86672865)

[2.6.1 Định nghĩa lỗ hổng bảo mật 31](#_Toc86672866)

[2.6.2 Bảo mật Website 35](#_Toc86672867)

[2.7 Json Web Token (JWT) 36](#_Toc86672868)

[2.7.1 Khái niệm JWT 36](#_Toc86672869)

[2.7.2 Cấu trúc JWT 36](#_Toc86672870)

[2.7.3 Cách hoạt động Json Web Token (JWT) 38](#_Toc86672871)

[2.7.4 Khi nào nên sử dụng JWT 39](#_Toc86672872)

[2.7.5 Tại sao sử dụng JWT? 39](#_Toc86672873)

[CHƯƠNG 3: XÂY DỰNG WEBSITE KINH DOANH BỘ MÁY TRÒ CHƠI PLAYSTATION TÍCH HỢP CHATBOT TỰ ĐỘNG 41](#_Toc86672874)

[3.1 Khảo sát hệ thống Website kinh doanh bộ máy trò chơi PlayStation 41](#_Toc86672875)

[3.2 Các chức năng của hệ thống 41](#_Toc86672876)

[3.2.1 Yêu cầu chức năng hệ thống 41](#_Toc86672877)

[3.2.2 Yêu cầu phi chức năng: 42](#_Toc86672878)

[3.3 Mô hình chức năng nghiệp vụ (BFD – Bussiness Function Diagram) 42](#_Toc86672879)

[3.4 Mô hình luồng dữ liệu (DFD – Data Flow Diagram) 43](#_Toc86672880)

[3.4.1 Biểu đồ DFD mức ngữ cảnh 44](#_Toc86672881)

[3.5 Mô hình Usecase 44](#_Toc86672882)

[3.5.1 Danh sách các Actor 44](#_Toc86672883)

[3.5.2 Xác định các usecase 45](#_Toc86672884)

[3.5.3 Biểu đồ Usecase tổng quát của hệ thống 47](#_Toc86672885)

[3.5.4 Đặc tả các usecase 47](#_Toc86672886)

[3.6 Thiết lập mô hình thực thể 58](#_Toc86672887)

[3.6.1 Xác định các thực thể 58](#_Toc86672888)

[3.6.2 Xác định mối quan hệ giữa các thực thể 59](#_Toc86672889)

[3.6.3 Sơ đồ ERD (Entity Relationship Diagram) 60](#_Toc86672890)

[3.6.4 Mô hình dữ liệu quan hệ 60](#_Toc86672891)

[3.7 Thiết kế dữ liệu 60](#_Toc86672892)

[3.7.1 Từ điển dữ liệu 60](#_Toc86672893)

[3.7.2 Mô hình diagram 66](#_Toc86672894)

[3.8 Xây dựng dự án 66](#_Toc86672895)

[3.8.1 Xây dựng BackEnd với Spring Boot 66](#_Toc86672896)

[3.8.2 Xây dựng FrontEnd với ReactJS 73](#_Toc86672897)

[3.8.3 Thực hiện thanh toán trực tuyến PayPal 77](#_Toc86672898)

[CHƯƠNG 4: ĐÁNH GIÁ KẾT QUẢ VÀ HƯỚNG PHÁT TRIỂN CỦA ĐỀ TÀI 82](#_Toc86672899)

[4.1 Kết quả 82](#_Toc86672900)

[4.1.1 Thiết kế giao diện cho khách hàng 82](#_Toc86672901)

[4.1.2 Thiết kế giao diện cho quản lý 85](#_Toc86672902)

[4.2 Đánh giá 95](#_Toc86672903)

[4.2.1 Những kết quả đạt được: 95](#_Toc86672904)

[4.2.2 Hạn chế: 96](#_Toc86672905)

[4.2.3 Hướng phát triển nghiên cứu 96](#_Toc86672906)

[4.2.4 Kết luận 96](#_Toc86672907)

[DANH MỤC TÀI LIỆU THAM KHẢO 97](#_Toc86672908)

# DANH SÁCH CÁC BẢNG, SƠ ĐỒ, HÌNH

# KÝ HIỆU CÁC CỤM TỪ VIẾT TẮT

|  |  |
| --- | --- |
|  | |
| CSDL hoặc DB | Cơ sở dữ liệu: DataBase |
| NSD = User = Account | Quyền làm việc được cấp cho NGƯỜI SỬ DỤNG trên hệ thống App, quyền này thông thường được đảm bảo bởi tên đăng nhập (username) và mật khẩu (password); cũng có thể gọi là “tài khoản người dùng” [“tài khoản”] |
|  |  |

# CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

## Lý do chọn đề tài:

Hiện nay, Game điện tử đối với nhân loại là một món ăn tinh thần không thể thiếu với mọi lứa tuổi, mọi giới tính,…Game điện tử trở thành nơi giải trí hiệu quả sau những giờ học tập và làm việc căng thẳng, nơi để mọi người giao lưu, kết bạn, giải trí, rèn luyện và làm mọi việc mà trên thực tế chúng ta chưa đủ kinh tế hoặc thời gian dài để thực hiện.Đối với lập trình viên, đây là mảnh đất màu mỡ để đầu tư chất xám hiệu quả nhất, nơi ta có thể xây dựng mọi thứ bằng trí tưởng tượng của chúng ta và các dòng code.

Việc chọn đề tài trên cũng là bước đầu tiên trên con đường tiếp cận các công nghệ làm game cũng như hướng tới việc xây dựng thế giới ảo trong Game.Việc xây dựng một game là xây dựng một thế giới ảo với những quy luật mà do chính chúng ta, game’s master đề ra, nói đơn giản, chúng ta xây dựng game chính là xây dựng 1 thế giới mơ ước của chính chúng ta.

## 1.2 Mục đích, ý nghĩa chọn đề tài:

* Xây dựng được một thế giới game 2D với những quy luật, nơi mà chúng ta vào vai một người lính tiến vào công cuộc trả thù những con quái vật và cả những kẻ thù ác.
* Xây dựng một gameplay thân thiện, người chơi sẽ được giải trí và giải đố, phát triển sự nhanh nhạy trong việc xử lí thông tin, phát triển IQ trong quá trình chơi game..
* Là bước đà để tiếp cận công nghệ thế giới.
* Giúp làm quen với việc xây dựng game hướng tới xây dựng một thế giới game rộng lớn.

## 1.3 Phương pháp nghiên cứu:

Tìm hiểu “**Xây dựng Game 2D Soldier's Avenger**”

* Khảo sát hệ thống

- Khảo sát các game có nội dung hoặc gameplay tương tự với phong cách phiêu lưu chiến đấu và cả giải đố ví dụ: Mario,…

* Phân tích chức năng hệ thống

- Từ quá trình khảo sát, phân tích và đưa ra bảng các chức năng chính của hệ thống và phân tích chi tiết từng chức năng

* Phân tích thiết kế với UML

- Vẽ và phân tích biểu đồ User Case

- Biểu đồ tuần tự

- Biểu đồ lớp

- Biểu đồ hoạt động

* Phân tích thiết kế CSDL
* Phân tích thiết kế chương trình

**a. Lý Thuyết:**

Tìm hiểu nghiệp vụ, ngôn ngữ và công nghệ sử dụng:

* Tìm hiểu và xây dựng game trên Engine Unity3D trên thiết bị Mobile và PC
* Tìm hiểu ngôn ngữ C# , JavaScript
* Tìm hiểu NodeJS, WebSocket
* Database: MongoDb

**b. Thực Hành:**

* Thiết kế Cơ sở dữ liệu hệ thống.
* Thiết kế và xây dựng hệ thống UI : Home, Inventory, Daily Reward, List Mission, Leader Board, GameSetting , GamePlay,.....
* Xử lí đăng nhập : login , register.
* Xây dựng hệ thống cốt truyện chính.
* Xây dựng và thiết kế hệ thống Player.
* Xây dựng và thiết kế hệ thống Enemy.
* Xây dựng và thiết kế Map, Trap.
* Xây dựng và thiết kế Bonus Items.
* Xây dựng hệ thống effect chiến đấu, nhạc nền, âm thanh.
* Xây dựng GamePlay :
* Map : xây dựng Map cho mỗi màn chơi, hệ thống bẫy tương tác với Player.
* Player : chiến đấu với Enemy, nhận vàng, kinh nghiệm nâng cấp sức mạnh, kĩ năng, vũ khí,... qua từng màn chơi.
* Enemy: chiến đấu với Player , xử lí AI né tránh, giữ khoảng cách, tấn công Player.
* Đưa game lên CHPlay.

**c. Công cụ hỗ trợ:**

* Visual Studio Code
* Unity3D

## 1.4 Phạm vi, đối tượng đề tài:

* Độ tuổi: 10+ (trên 10 tuổi, đủ nhận thức về trò chơi và giải đố trong trò chơi)
* Giới tính: phù hợp với tất cả giới tính.
* Độ phủ sóng: game sẽ được đưa lên các cửa hàng điện tử có thể tiếp cận dễ dàng với người chơi thông qua internet.
* Thiết bị: game đơn giản phù hợp với mọi loại thiết bị trên nhiều nền tảng.
* Nền tảng : PC và Mobile : điều khiển gamepad và keyboard
* Thể loại: game Phiêu lưu , giải đố phù hợp với nhu cầu giải trí.

## 1.5 Ý nghĩa thực tiễn đề tài

* Game giúp người chơi hóa thân vào nhân vật mô phỏng, tham gia các trận chiến và giải đố, chiến đấu giúp người chơi giải trí cũng như tăng cường khả năng xử lí tình huống, phát triển IQ.
* Việc xây dựng game sẽ giúp ta tiếp cận và học nhiều kĩ năng trong việc xây dựng thế giới game trong tương lai.

## 1.6 Cấu trúc báo cáo

Phần 1: Mở đầu

Phần 2: Nội dung

- Chương I: Tổng quan đề tài

- Chương II: Cơ sở lý luận và nền tảng xây dựng hệ thống

- Chương III: Phân tích và thiết kế hệ thống

- Chương IV: Xây dựng GamePlay

- Chương V: Đánh giá kết quả và hướng phát triển đề tài

Phần 3: Tài liệu tham khảo

# CHƯƠNG 2: CƠ SỞ LÝ LUẬN VÀ NỀN TẢNG XÂY DỰNG HỆ THỐNG

## 2.1. Tổng quan về Game điện tử

### 2.1.1. Khái niệm Game điện tử

Trò chơi điện tử là trò chơi sử dụng thiết bị điện tử để tạo ra một hệ thống tương tác với người chơi nhằm mục đích chính là giải trí.

Game điện tử phải được hiểu đầy đủ là bao gồm: Các trò chơi dùng điện báo đánh chữ (Teletype games), Máy chơi game cầm tay (Electronic handhelds), Máy bắn bi và các thiết bị tương tự (Pinball machines and similar devices), Trò chơi đổi thưởng (Redemption games), Máy đánh bạc (Slot machines), Trò chơi âm thanh (Audio games) và Trò chơi video (Video games).

Đề tài trên đi vào xây dựng video game, thuộc Game điện tử cũng chính là xu hướng hiện đại phát triển nhất của Game điện tử.

### 2.1.2. Lợi ích của Game điện tử

**2.1.2.1. Ngành dịch vụ đang hot nhất trên thị trường**

Game điện tử nhất là video game hiện tại đang phát triển mạnh mẽ do sự phát triển của công nghệ cùng với nhu cầu giải trí càng cao của con người. Do đó đây luôn là môi trường màu mỡ cho việc đầu tư chất xám cũng như lợi nhuận đáng kể do ngành dịch vụ này mang lại do có thị trường người dùng trải dài trên mọi lứa tuổi, mọi giới tính, mọi nơi…

Người chơi còn có thể đầu tư vào game, buôn bán vật phẩm giá trị trong game để tạo thu nhập, từ đó đem lại lợi nhuận cho cả những người chơi giỏi.

**2.1.2.2. Tính giải trí**

Điều tiên quyết trong một game điện tử là tính giải trí. Một game điện tử phát triển được phải đảm bảo được nhu cầu giải trí của con người. Trong thời đại ngày nay, nhu cầu giải trí ngày càng cao cũng như chất lượng phải đảm bảo,do đó các game điện tử phải đảm bảo tính giải trí phù hợp với thời đại để tránh bị lạc hậu, lãng quên.

**2.1.2.3. Phát triển bản thân người chơi**

Game điện tử không chỉ đơn giản để giải trí, mà còn đảm bảo tính tò mò, thích thú và thử thách người chơi, và để hoàn thành game, người chơi phải hoàn thiện khả năng phản xạ, tính logic và khả năng giải đố tùy thuộc vào thể loại game. Do đó game nâng cao khả năng của người chơi, tăng cường IQ, luyện phản xạ cho tay, mắt…

Một số game ra đời nhằm giúp đỡ trong cả ngành y học, trong vật lí trị liệu giúp đỡ bệnh nhân đi đứng, lấy lại cảm giác từ các giác quan qua các game thực tế ảo hoặc các game phản xạ cơ bản.

**2.1.2.4. Tính xã hội**

Ngày nay với sự phát triển của internet đã tạo nên một cộng đồng game lớn mạnh trên toàn thế giới, các hội nhóm game hoặc các tổ chức ngày càng lón được tạo ra, xây dựng các giải đấu dành cho những game thủ chuyên nghiệp. Game điện tử trở thành môn thể thao điện tử không thể thiếu trong các kì vận hội trong tương lai.

Game online giúp các cá nhân tương tác với nhau như một xã hội thu nhỏ, giúp kết bạn mới hoặc kết hợp chơi với nhau tạo nên tính đoàn kết trong cộng đồng.

### 2.1.3. Các yêu cầu trong một game điện tử hay

Mỗi trò chơi đều có những đặc tính riêng tạo nên sự khác biệt so với những trò chơi khác. Nhưng đặc biệt một số game lại sở hữu các “quyền năng” vô hình nào đó khiến game thủ không thể nào dứt ra khỏi mà cứ chơi mãi ngày này qua ngày khác, trong khi có những game lại bị bỏ xó không thương tiếc, chờ ngày gỡ cài đặt.

Dưới đây là các yêu cầu của 1 game điện tử hay :

**2.1.3.1.Các nấc thang trình độ**

Đây có lẽ là khía cạnh quan trọng nhất làm nên sự hấp dẫn nhất của một trò chơi. Điểm cốt lõi của eSports và competitive games chính là sự cải thiện liên tục, thay đổi liên tục, đòi hỏi người chơi phải luôn luôn nâng cao kỹ năng của bản thân, tạo nên những thách thức để người chơi vượt qua, từ đó làm nên tính cạnh tranh và lôi cuốn của trò chơi.Dù vẫn có những tựa game cực kỳ khó như Mega man Classic hoặc game chiến thuật nhóm tuyệt hay như MOBA, nhưng tựu chung thì cách phổ biến nhất để lôi cuốn người chơi đó là cung cấp các cách thức rõ ràng giúp họ nâng cao trình độ. Nếu trong game không có tính năng đấu PvP (Player versus Player - người chơi đấu với người chơi) thì hãy sử dụng leaderboard (bảng xếp hạng), giúp người chơi có thể dễ dàng so sánh mình với những người chơi khác. Thêm vào đó, hãy nhớ bao gồm cả tính năng thử thách hàng ngày (daily challenges) để tạo thành thói quen cho người chơi.

**2.1.3.2.** **Cốt truyện hấp dẫn**

Khi được yêu cầu đưa ra ví dụ về một game có cốt truyện hấp dẫn, nhiều người sẽ nghĩ ngay đến những tựa game nặng tính kể chuyện như game JRPG (Japanese Role Playing Game - thể loại game nhập vai Nhật Bản) hay những game phương Tây có chế độ một người chơi (single player). Tuy nhiên, hiếm có khi nào mà một trò chơi lại bị “dở đi” bởi các yếu tố kể chuyện.Các yếu tố thuộc về kiến thức cũng khá cần thiết, thậm chí những game eSports như Liên minh huyền thoại(LMHT) cũng đã giải quyết được vấn đề này. Mỗi lượt chơi có thể không liên quan nhiều với nhau nhưng bằng việc cùng nhau giải quyết các vấn đề trong thế giới giả tưởng của game, LMHT đã tạo được sự liên kết và cộng hưởng giữa nhiều game thủ trong game. Xây dựng thế giới trong game và tạo nên tình huống truyện hấp dẫn không phải là những cách duy nhất để thỏa mãn yêu cầu của người chơi về mặt cốt truyện. Nhiều tình tiết trong game MMO(Massively Multiplayer Online) hiện nay còn hấp dẫn hơn cả câu chuyện gốc do những nhà văn chuyên nghiệp viết cho các nhà phát triển game. Các yếu tố trên được đưa vào trong mỗi trò chơi thông qua những hình ảnh trực quan, khơi gợi cảm xúc từ phía người chơi và trở thành câu chuyện riêng của mỗi người.

**2.1.3.3. Khả năng thành thục**

Hầu hết thời điểm chúng ta vui vẻ nhất khi chơi một trò chơi đó là khi chúng ta đã tương đối thành thạo nó và có thể chơi tự nhiên mà không cần phải suy nghĩ nhiều. Không cần phải “cày” cật lực để kiếm XP hay khám phá những hang động trong Spelunky đến cả nghìn lần, đôi khi niềm vui đơn giản đến từ những lúc CHỈ chơi game mà thôi.Điều này có thể áp dụng như nhau ở cả dòng game phổ thông và core game. Đôi khi sự mệt mỏi của những chuyến bay dài có thể bị đánh tan bởi những đường chạy gần như vô tận (endless run) của Banana kong hay Temple run. Những game này vẫn đòi hỏi người chơi có kỹ năng nhưng chúng gần như đem lại trạng thái thư giãn tuyệt đối, giúp chúng ta quên đi quãng thời gian dài dằng dặc và sự mệt mỏi của những chuyến bay.

**2.1.3.4. Sự bí ẩn**

Mong muốn tìm được câu trả lời cho những câu hỏi kéo dài không có lời giải đáp hoặc biết cho bằng được kết thúc của một chuyện gì đó hẳn là tác nhân mạnh mẽ nhất khiến một người kiên nhẫn theo dõi câu chuyện cho đến cuối cùng. Đôi khi chúng ta cứ nấn ná xem cho bằng hết một bộ phim chỉ vì muốn biết kết thúc của nó ra sao dù cho phim dở tệ. Game cũng vậy, dù cho rất hay chăng nữa nhưng nếu trong game không có những tình tiết bí mật nhằm kích thích trí tò mò của người chơi thì cũng khó mà giữ chân họ được lâu.

**2.1.3.5. Giảm bớt/Tối thiểu hóa hình phạt khi người chơi mất mạng trong game**

Trong nhiều game, sau khi bị “chết”, người chơi đã quen với việc có những hình đồ họa sặc sỡ màu mè thể hiện những câu kiểu như “Hết mạng. Đi lối này!” như thể không có chúng thì chúng ta không biết đường để... thoát game vậy. Thực ra không nhất thiết phải buộc người chơi rời khỏi game mỗi khi mất mạng như thế. Thay vì như vậy các nhà phát triển game nên áp dụng những cách thức nhẹ nhàng và tế nhị hơn để giảm thiểu hình phạt cho việc “chết” trong game, có thể chỉ đơn giản là một menu được thiết kế hợp lý, hoặc đến cách thức phức tạp hơn như là hệ thống tính năng autosave. Game Buff knight của studio Hàn Quốc Buff đã thực hiện điều này rất tốt khi dẫn dắt người chơi đến cửa hàng nhằm upgrade nhân vật của mình sau khi vừa bị mất mạng.

**2.1.3.6. Tính mới lạ**

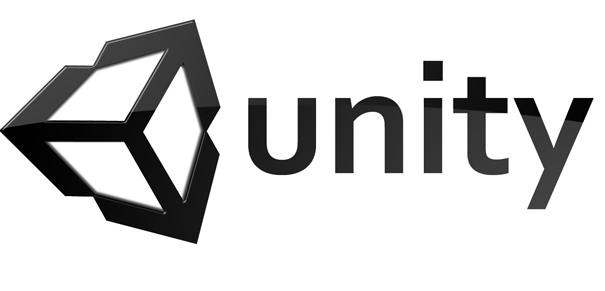
Chắc chắn là người chơi nào cũng sẽ có cảm giác thích thú phấn khích khi được chơi một trò chơi "không giống ai". Khi Flappy bird mới ra lò, rất nhiều người chơi nhanh chóng trở nên hào hứng khi chưa bao giờ gặp trò chơi nào khó như vậy!Đồ họa đẹp mắt, thế giới trong game sinh động mới lạ, nhân vật kì quặc hay cốt truyện bất ngờ v.v. hãy cố gắng tạo ra bất cứ thứ gì có thể khiến cho trò chơi của bạn khác lạ so với những game đã xuất hiện nhan nhản trên thị trường, chỉ có thế thì mới khiến người chơi có hứng thú được. Và tất nhiên sự mới mẻ này cũng chỉ tồn tại được trong một thời gian ngắn mà thôi, vì chỉ cần có game nào trở nên thành công một tẹo thì sẽ nhanh chóng có một lô một lốc những game ăn theo nó ngay, đơn cử như Flappy bird!

**2.1.3.7. Thử thách trí tuệ**

Người chơi luôn thích được thử thách, có thể chỉ đơn giản là một câu đố, hoặc một thử thách phức tạp cần phải “động não” nhiều hơn. Các thể loại thử thách trí tuệ trải dài từ chọn cánh cửa nào để mở trong các survival horror game cho đến việc hoàn thành một cuộc bố ráp như trong các game MMO. Cảm giác giải mã được câu đố hay tìm ra được những bước thích hợp để hoàn thành nhiệm vụ quả thật khiến người chơi thấy thích thú và thỏa mãn. Những nhà phát triển game Nhật Bản từ lâu đã khai thác yếu tố này một cách rất hiệu quả, với series Level 5’s Professor Layton và nhiều game khác trong dòng game survival horror.

## 2.2. Xây dựng hệ thống

### 2.2.1 Giới thiệu Unity3D

****

***Hình 2.1. Phần mềm Unity3D***

(Nguồn: *https://teky.edu.vn/blog/unity-3d-la-gi*/)

Hiện nay, với 27 nền tảng phát triển liên tục 15 năm (từ năm 2005), Unity đang được đánh giá là một trong những phần mềm đa nền tảng mạnh mẽ nhất cho những lập trình viên game.

Unity hỗ trợ được hầu hết trên các nền tảng từ IOS, Android, hay Windowa, MacOs,…, sử dụng các ngôn ngữ lập trình C# và Javascript. Điều đó đã tạo nên thế mạnh để Unity trở thành chìa khóa để phát triển phần mềm game tốt nhất hiện nay.

Ngoài ra, việc cho phép lập trình game theo thời gian thực của Unity đã cho phép các Designer có thêm nhiều lợi thế:

* Không cần Render.
* Thiết kế được Map, Character, Interface Graphic,.. từ những phần mềm khác (như Modo, Lightwave,…)
* Tối ưu hóa định dạng file theo dạng “.FBX” hay “.exe”. Vì vậy, thiết kế có thể dễ dàng chia sẻ trên các thiết bị khác.

Bên cạnh đó, Unity có khả năng cung cấp các tính năng tích hợp quan trọng. Lập trình với Unity ngày càng trở nên đơn giản với các công cụ:

* Cho phép lập trình viên có thể chia sẻ các bản thiết kế.
* Cho phép sử dụng các tài nguyên hiệu ứng có sẵn trên Unity. Hoặc bạn có thể mua hoặc bán sản phẩm cho story của Unity.
* Cung cấp hệ thống toàn diện, dễ sử dụng cho lập trình viên.

Được coi là chìa khóa để phát triển phần mềm game, Unity 3D sở hữu nhiều ưu điểm.

* Về quy mô sử dụng phần mềm.

Theo một số thống kê, hiện nay có tới 47% số các nhà phát triển game sử dụng Unity 3D. Nó đang được coi là nền tảng thiết kế game phổ biến nhất trên toàn thế giới.

Cộng đồng Unity rất lớn mạnh. Với quy mô trên toàn thế giới, cộng đồng designer được thành lập ở nhiều nơi. Trong đó chia sẻ các kiến thức và kinh nghiệm về Unity 3D. Nhờ đó, kho sưu tập tài nguyên hiệu ứng có thể dử dụng trong Unity 3D ngày càng trở nên phong phú, đa dạng,…

* Chi phí sử dụng phần mềm.

Chính sách về chi phí của Unity được nới lỏng đến mức tối đa. Phần mềm được sử dụng miễn phí cho các cá nhân và doanh nghiệp có doanh thu thấp. Nếu muốn nâng cấp và sử dụng bản Pro, chi phí phải bỏ ra cũng rất nhỏ. Chỉ từ 1.500 USD cho một năm sử dụng Unity 3D.

* Cách sử dụng phần mềm đơn giản.

Là một phần mềm lập trình đa nền tảng, có thể dử dụng trên nhiều thiết bị khác nhau. Unity 3D cung cấp một hệ thống toàn diện cho cả nhiệm vụ soạn thảo và sửa lỗi. Điều này giúp phần mềm lập trình này có thể tối ưu hóa mọi công việc cho lập trình viên. Vì vậy, Unity 3D đảm bảo dễ sử dụng cho cả newbie.

* Nền tảng được phát triển liên tục, không ngừng nâng cấp phần mềm.

Phát triển Unity trong vòng 15 năm (từ năm 2015 đến nay). Unity Technologies đã liên tục cho ra đời các phiên bản khác nhau của Unity. Các phiên bản không ngừng được chú trọng phát triển tạo được những tính năng ấn tượng.

### 2.2.2 Giới thiệu Visual Studio 2019



**Hình 2.2. Phần mềm Visual Studio 2019**

(Nguồn: *https://lapcamerahanoi.com/key-visual-studio-2019/)*

* Microsoft Visual Studio là một môi trường phát triển tích hợp (IDE) từ Microsoft.
* Visual Studio bao gồm một trình soạn thảo mã hỗ trợ IntelliSense cũng như cải tiến mã nguồn. Trình gỡ lỗi tích hợp hoạt động cả về trình gỡ lỗi mức độ mã nguồn và gỡ lỗi mức độ máy. Công cụ tích hợp khác bao gồm một mẫu thiết kế các hình thức xây dựng giao diện ứng dụng, thiết kế web, thiết kế lớp và thiết kế giản đồ cơ sở dữ liệu.
* Visual Studio hỗ trợ nhiều ngôn ngữ lập trình khác nhau và cho phép trình biên tập mã và gỡ lỗi để hỗ trợ (mức độ khác nhau) hầu như mọi ngôn ngữ lập trình. Các ngôn ngữ tích hợp gồm có C, C++ và C++/CLI , VB.NET , C# và F#. Hỗ trợ cho các ngôn ngữ khác như J++/J#, Python và Ruby thông qua dịch vụ cài đặt riêng rẽ. Nó cũng hỗ trợ XML/XSLT, HTML/XHTML, JavaScript và CSS
* Một số tính năng Visual Studio Code:
* Cải thiện trải nghiệm tìm kiếm nâng cao

Một trong những tính năng hữu ích trong Visual Studio 2019 là hộp tìm kiếm được cập nhật. Trong một phiên bản trước, nó được gọi là Khởi động nhanh. Trải nghiệm tìm kiếm mới trong Visual Studio 2019 nhanh hơn và hoạt động hiệu quả để có kết quả tốt nhất.

Bây giờ kết quả tìm kiếm sẽ hiển thị các đề xuất khi bạn nhập vào hộp tìm kiếm và nó sẽ tự động trình bày kết quả. Bạn cũng có thể sử dụng các phím tắt trong khi tìm kiếm và điều này sẽ giúp bạn nhớ các truy vấn tìm kiếm để sử dụng trong tương lai và nhớ lại bất cứ khi nào bạn xây dựng các ứng dụng web mới.

* Tái cấu trúc được cải thiện

Tái cấu trúc C # mới giúp tổ chức mã của bạn dễ dàng hơn. Chỉ cần gọi các phép tái cấu trúc bằng cách nhấn Ctrl +. và chọn hành động bạn muốn thực hiện.

* IntelliCode tốt hơn

Visual Studio IntelliCode là một phần mở rộng giúp tăng cường nỗ lực phát triển phần mềm của bạn bằng cách sử dụng trí tuệ nhân tạo (AI).

Nó cung cấp các đề xuất và các đối tượng mã hóa trong quá trình phát triển. Intellicode sẽ đạt được sức mạnh khi sử dụng 2000 dự án nguồn mở khác Quy trình làm việc trên nền tảng đám mây

Cửa sổ bắt đầu trình bày cho bạn một số tùy chọn để giúp bạn mã hóa nhanh chóng. Trước tiên, chúng tôi đã đặt tùy chọn để sao chép hoặc kiểm tra mã từ repo.

* Chia sẻ trực tiếp

Visual Studio Live Share là một dịch vụ dành cho nhà phát triển cho phép bạn chia sẻ một cơ sở mã và bối cảnh của nó với đồng đội và nhận được sự cộng tác hai chiều ngay lập tức từ bên trong Visual Studio. Với Live Share, đồng đội có thể đọc, điều hướng, chỉnh sửa và gỡ lỗi một dự án mà bạn đã chia sẻ với họ và thực hiện một cách liền mạch và an toàn.

* Đánh giá mã tích hợp

Chúng tôi đang giới thiệu một tiện ích mở rộng mới mà bạn có thể tải xuống để sử dụng với Visual Studio 2019. Với tiện ích mở rộng mới này, bạn có thể xem lại, chạy và thậm chí gỡ lỗi các yêu cầu kéo từ nhóm của mình mà không cần rời khỏi Visual Studio. Chúng tôi hỗ trợ mã trong cả kho GitHub và Azure DevOps.

* Tính năng tìm kiếm trong khi gỡ lỗi

Bạn đã biết rằng gỡ lỗi là quá trình xác định lỗi và loại bỏ các lỗi đó trong phần mềm. Tìm kiếm các đối tượng và các giá trị phù hợp trong quá trình gỡ lỗi rất hữu ích cho các nhà phát triển.

Trong Visual Studio 2019 tìm kiếm trong khi gỡ lỗi được thêm Địa phương, Ô tô và cửa sổ Xem được sử dụng để tìm các giá trị và đối tượng. Dưới đây hoạt hình sẽ giải thích quá trình thực tế gỡ lỗi cửa sổ tìm kiếm trong VS2019.

* Hiệu suất tốt hơn

Cho dù bạn đang mã hóa trong C ++ hay .NET Core, các điểm dừng dữ liệu có thể là một lựa chọn tốt để chỉ đặt các điểm dừng thông thường. Điểm dừng dữ liệu cũng tuyệt vời cho các tình huống như tìm kiếm nơi một đối tượng toàn cầu đang được sửa đổi hoặc được thêm hoặc xóa khỏi danh sách.

* Dọn mã trong một lần nhấp

Lệnh dọn mã mới trong Visual Studio 2019 là phát hiện các cảnh báo và đề xuất bằng nút một lần bấm. Lệnh này sẽ giúp định dạng mã và thực hiện các thay đổi thành định dạng mã hóa được đề xuất bởi các tệp và cài đặt .editorconfig. Nó cũng cho phép lưu các bộ sưu tập các bản sửa lỗi làm hồ sơ mặc định

* Hỗ trợ Xamarin

Visual Studio 2019 hỗ trợ Xamarin.Forms 4.0 cùng với các tính năng mới bao gồm các mẫu dự án Shell mới và các điều khiển CollectionView mới. Các mẫu Shell giúp người dùng thiết kế các ứng dụng dành cho thiết bị di động với giao diện được cải tiến và được liên kết chặt chẽ với Material Design. Xamarin CollectionView là một giải pháp thay thế tốt hơn và linh hoạt hơn cho các điều khiển ListView.

### 2.2.3. Giới thiệu về MongoDB



**Hình 2.3. Database MongoDB**

(Nguồn*: https://viblo.asia/p/mongodb-la-gi-co-so-du-lieu-phi-quan-he-bJzKmgoPl9N.html*)

* MongoDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở, là CSDL thuộc NoSql và được hàng triệu người sử dụng.
* MongoDB là một database hướng tài liệu (document), các dữ liệu được lưu trữ trong document kiểu JSON thay vì dạng bảng như CSDL quan hệ nên truy vấn sẽ rất nhanh.
* Với CSDL quan hệ chúng ta có khái niệm bảng, các cơ sở dữ liệu quan hệ (như MySQL hay SQL Server...) sử dụng các bảng để lưu dữ liệu thì với MongoDB chúng ta sẽ dùng khái niệm là collection thay vì bảng
* So với RDBMS thì trong MongoDB collection ứng với table, còn document sẽ ứng với row , MongoDB sẽ dùng các document thay cho row trong RDBMS.
* Các collection trong MongoDB được cấu trúc rất linh hoạt, cho phép các dữ liệu lưu trữ không cần tuân theo một cấu trúc nhất định.
* Thông tin liên quan được lưu trữ cùng nhau để truy cập truy vấn nhanh thông qua ngôn ngữ truy vấn MongoDB

**2.2.3.1. Ưu điểm của MongoDB**

* Do 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, linh hoạt trong việc lưu trữ dữ liệu, nên bạn muốn gì thì cứ insert vào thoải mái.
* Dữ liệu trong MongoDB không có sự ràng buộc lẫn nhau, không có join như trong RDBMS nên khi insert, xóa hay update nó không cần phải mất thời gian kiểm tra xem có thỏa mãn các ràng buộc dữ liệu như trong RDBMS.
* MongoDB rất dễ mở rộng (Horizontal Scalability). Trong MongoDB có một khái niệm cluster là cụm các node chứa dữ liệu giao tiếp với nhau, khi muốn mở rộng hệ thống ta chỉ cần thêm một node với vào cluster:
* Trường dữ liệu “\_id” luôn được tự động đánh index (chỉ mục) để tốc độ truy vấn thông tin đạt hiệu suất cao nhất.
* Khi có một truy vấn dữ liệu, bản ghi được cached lên bộ nhớ Ram, để phục vụ lượt truy vấn sau diễn ra nhanh hơn mà không cần phải đọc từ ổ cứng.
* Hiệu năng cao: Tốc độ truy vấn (find, update, insert, delete) của MongoDB nhanh hơn hẳn so với các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS). Với một lượng dữ liệu đủ lớn thì thử nghiệm cho thấy tốc độ insert của MongoDB có thể nhanh tới gấp 100 lần so với MySQL.

**2.2.3.2. Nhược điểm của PostgreSQL:**

* Một ưu điểm của MongoDB cũng chính là nhược điểm của nó. MongoDB không có các tính chất ràng buộc như trong RDBMS nên khi thao tác với mongoDB thì phải hết sức cẩn thận.
* Tốn bộ nhớ do dữ liệu lưu dưới dạng key-value, các collection chỉ khác về value do đó key sẽ bị lặp lại. Không hỗ trợ join nên dễ bị dữ thừa dữ liệu.
* Khi insert/update/remove bản ghi, MongoDB sẽ chưa cập nhật ngay xuống ổ cứng, mà sau 60 giây MongoDB mới thực hiện ghi toàn bộ dữ liệu thay đổi từ RAM xuống ổ cứng điêù này sẽ là nhược điểm vì sẽ có nguy cơ bị mất dữ liệu khi xảy ra các tình huống như mất điện..

## 2.3. Tìm hiểu về .NET

### 2.3.1 Ngôn ngữ C#

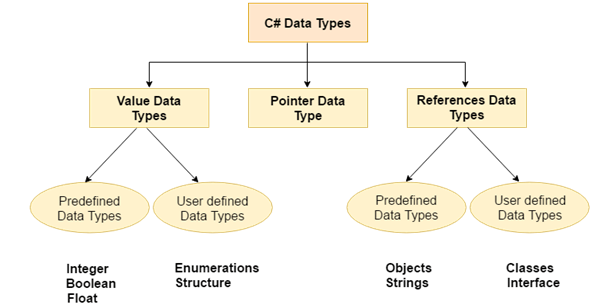
**2.3.1.1 Lịch sử phát triển của C#**



**Hình 2.4. Lịch sử phát triển C#**

(Nguồn: [https://kcntt.duytan.edu.vn/Home/ArticleDetail/vn/168/3633/lich-su-phat-trien-cua-c)](https://vi.wikipedia.org/wiki/Java_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh))

**2.3.1.2 Những đặc trưng của ngôn ngữ C#**



**Hình 2.5. Đặc trưng ngôn ngữ C#**

- Ngôn ngữ lập trình C# có những đặc trưng cơ bản sau:

* Đơn giản(Simple)
* Ngôn ngữ lập trình hiện đại(Modern programming language)
* Lập trình hướng đối tượng(Object oriented)
* Kiểu an toàn(Type safe)
* Khả năng tương tác(Interoperability)
* Có thể mở rộng và cập nhật(Scalable and Updateable)
* Hướng thành phần(Component oriented)
* Ngôn ngữ lập trình có cấu trúc(Structured programming language)
* Nhiều thư viện(Rich Library)
* Tốc độ nhanh(Fast speed)

### 2.3.2 Tìm hiểu về .NET Framework

**2.3.2.1 .NET Framework**

* .NET Framework được Microsoft đưa ra chính thức từ năm 2002. .NET Framework chỉ hoạt động trên Windows. Những nền tảng ứng dụng như WPF, Winforms, ASP.NET(1-4) hoạt động dựa trên .NET Framework.
* Mono là phiên bản cộng đồng nhằm mang .NET đến những nền tảng ngoài Windows. Mono được phát triển chủ yếu nhằm xây dựng những ứng dụng với giao diện người dùng và được sử dụng rất rộng rãi: Unity Game, Xamarin…
* Cho đến năm 2013, Microsoft định hướng đi đa nền tảng và phát triển .NET core. .NET core hiện được sử dụng trong các ứng dụng Universal Windows platform và ASP.NET Core. Từ đây, C# có thể được sử dụng để phát triển các loại ứng dụng đa nền tảng trên các hệ điều hành khác nhau (Windows, Linux, MacOS,…)

## 2.4. Tổng quan về RestFul API



**Hình 2.9. Tổng quan ResFul API**

(Nguồn: <https://movan.vn/rest-api-gioi-thieu-rest-api/>)

- REST (REpresentational State Transfer) được đưa ra vào năm 2000, trong luận văn tiến sĩ của Roy Thomas Fielding (đồng sáng lập giao thức HTTP). Nó là một dạng chuyển đổi cấu trúc dữ liệu, là một phong cách kiến ​​trúc cho việc thiết kế các ứng dụng có kết nối. Nó sử dụng HTTP đơn giản để tạo cho giao tiếp giữa các máy. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, DELETE, vv đến một URL để xử lý dữ liệu.

- API (Application Programming Interface) là giao diện lập trình ứng dụng giúp tạo ra các phương thức kết nối với các thư viện và ứng dụng khác nhau.

- REST API là một ứng dụng chuyển đổi cấu trúc dữ liệu có các phương thức để kết nối với các thư viện và ứng dụng khác. REST API không được xem là một công nghệ, nó là một giải pháp để tạo ra các ứng dụng web services thay thế cho các kiểu khác như SOAP, WSDL (Web Service Definition Language),…

- Các ưu điểm của REST:

* Giúp cho ứng dụng trở nên rõ ràng hơn.
* REST URL đại diện cho resource chứ không phải là hành động.
* Dữ liệu được trả về với nhiều định dạng khác nhau như: xml, html, rss, json …
* Code đơn giản và ngắn gọn.
* REST chú trọng vào tài nguyên hệ thống.

# CHƯƠNG 3: Xây dựng game 2D Soldier's Avenger

## 3.1 Khảo sát hệ thống Game 2D Soldier's Avenger

- Xây dựng hệ thống player:

* Hệ thống di chuyển
* Hệ thống tương tác: Chiến đấu với quái vật, kẻ thù, nhặt vật phẩm, sử dụng vật phẩm.
* Hệ thống

## 3.2 Các chức năng của hệ thống

### 3.2.1 Yêu cầu chức năng hệ thống

- Công việc của nhân viên:

* Thêm/Xóa/Sửa các danh mục
* Xử lý nhập kho, tiếp nhận và xử lý đơn hàng

- Công việc khách hàng:

* Đăng ký và Đăng nhập
* Xem thông tin sản phẩm, chấm điểm và đánh giá bình luận
* Đặt hàng và thanh toán

- Đối tướng tham gia hệ thống:

* Khách hàng
* Nhân viên cửa hàng

### 3.2.2 Yêu cầu phi chức năng:

• Hệ thống đáng tin cậy, chính xác, giao diện thân thiện, dễ sử dụng, truy cập dữ liệu nhanh chóng.

• Đảm bảo tính bảo mật cho người điều hành hệ thống.

• Phải có tính linh hoạt cao

## 3.3 Mô hình chức năng nghiệp vụ (BFD – Bussiness Function Diagram)

Chức năng là công việc mà tổ chức cần làm và được phân theo nhiều mức từ tổng hợp đến chi tiết.

Mỗi chức năng có một tên duy nhất, các chức năng khác nhau phải có tên khác nhau. Để xác định tên cho các chức năng một cách chính xác và cụ thể, có thể bàn luận và nhất trí với người sử dụng.

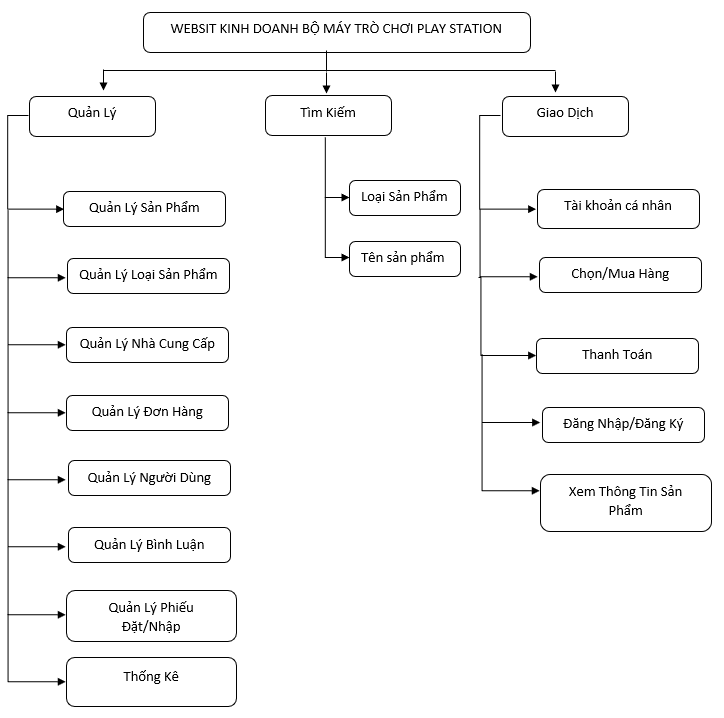
Hình thức biểu diễn: Hình chữ nhật:



Mỗi chức năng được phân rã thành các chức năng con. Các chức năng con có quan hệ phân cấp với chức năng cha.

Biểu diễn mối quan hệ phân cấp chức năng như sau:





***Hình 3.1. Mô hình phân rã chức năng được biểu diễn thành hình cây phân cấp* *của website kinh doanh máy Playstation***

## 3.4 Mô hình luồng dữ liệu (DFD – Data Flow Diagram)

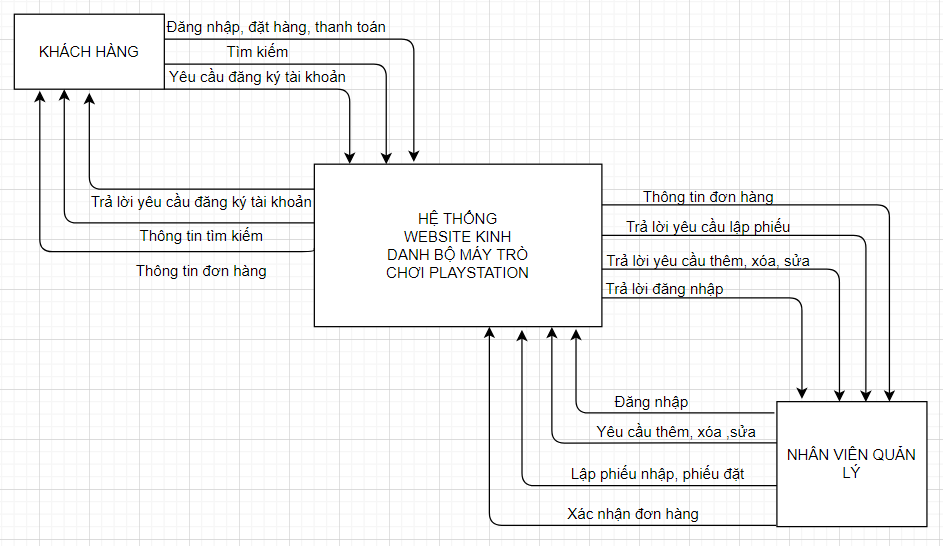
- Mô hình luồng dữ liệu là một công cụ mô tả quan hệ thông tin giữa các công việc.

Biểu đồ luồng dữ liệu gồm có 5 yếu tố chính:



***Hình 3.2. Các đối tượng cơ bản sử dụng để thiết kế biểu đồ luồng dữ liệu***

### 3.4.1 Biểu đồ DFD mức ngữ cảnh



**Hình 3.3. Biểu đồ DFD mức ngữ cảnh Quản lý cửa hàng máy Playstation**

## 3.5 Mô hình Usecase

### 3.5.1 Danh sách các Actor

- Trong chương trình quản lý Website bao gồm Admin và nhân viên, khách hàng.

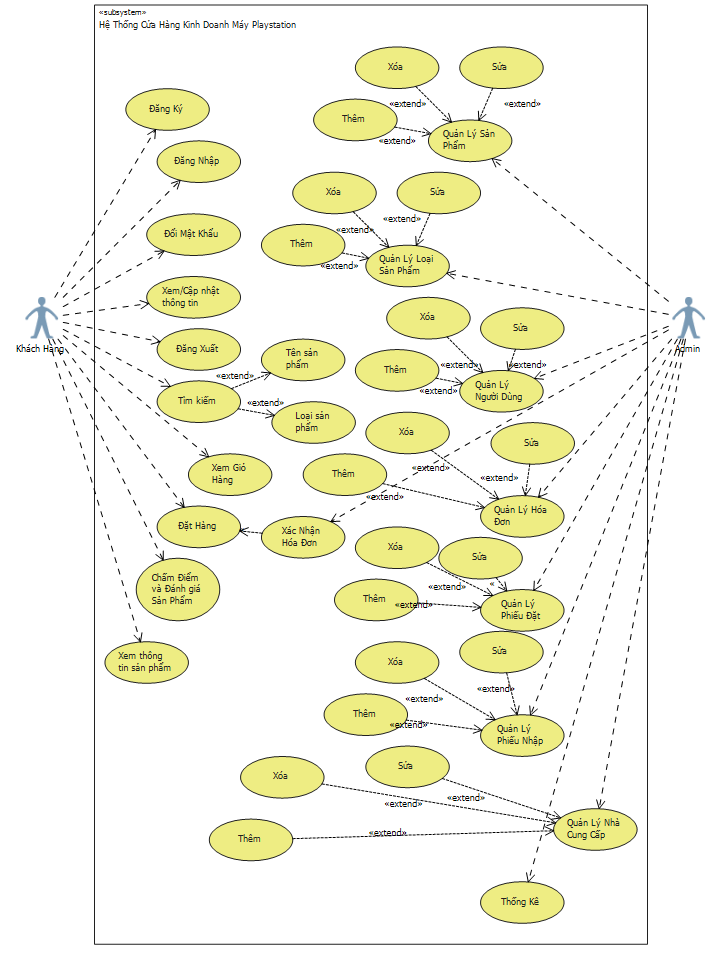
+ Admin và nhân viên: admin là người có quyền cao nhất, và cũng là người đóng vai trò quan trọng của hệ thống sau khi đăng nhập có thể quản lý thông tin sản phẩm, quản lý đơn hàng, quản lý tài khoản, quản lý danh mục, quản lý nhà cung cấp, tìm kiếm, thống kê

+ Khách hàng: có thể xem thông tin cửa hàng, thông tin sản phẩm, tìm kiếm và thêm sản phẩm vào giỏ hàng và đặt mua hàng.

### 3.5.2 Xác định các usecase

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Actor** | **Tên use case** | **Ý nghĩa** |
| 1 | Khách hàng | Đăng ký | Use case này giúp người dùng có quyền truy cập để thực hiện các chức năng của hệ thống cần đến. |
| 2 | Khách hàng | Đăng nhập | Use case này giúp người dùng sử dụng các chức năng của hệ thống cần đến quyền truy cập. |
| 3 | Khách hàng | Đăng xuất | Use case này giúp người dùng thoát khỏi hệ thống |
| 4 | Khách hàng | Đổi mật khẩu | Use case này giúp người dùng thay đổi thông tin cá nhân |
| 6 | Khách hàng | Tìm kiếm sản phẩm | Use case này mô tả chức năng tìm kiếm sản phẩm theo loại sản phẩm |
| 7 | Khách hàng | Đặt hàng | Use case này mô tả chức năng đặt hàng của khách hàng, khách hàng có thể thay đổi số lượng sản phẩm theo ý muốn. Có thể hủy bỏ việc đặt hàng nếu thay đổi ý định. |
| 8 | Khách hàng | Xem giỏ hàng | Chức năng xem giỏ hàng là đựng những mặt hàng mà khách hàng chọn |
| 9 | Khách hàng | Bình luận đánh giá và chấm điểm cho sản phẩm | Use case này giúp người dùng sử dụng thực hiện chức năng đánh giá và bình chọn số điểm cho sản phẩm |
| 10 | Khách hàng | Xem thông tin sản phẩm | Use case này mô tả chức năng xem thông tin sản phẩm của người sử dụng như xem nguồn gốc sản phẩm, thông số kĩ thuật của sản phẩm… |
| 11 | Nhân viên quản lý | Quản lý loại sản phẩm | Use case này mô tả chức năng thêm, xóa hoặc khi thông tin về loại sản phẩm nào đó thay đổi thì nhân viên quản lý sẽ có nhiệm vụ thực hiện chức năng cập nhật lại thông tin đó vào hệ thống. |
| 12 | Nhân viên quản lý | Quản lý sản phẩm | Use case này mô tả chức năng thêm, xóa và cập nhật thông tin sản phẩm vào hệ thống. Khi thông tin của một sản phẩm thay đổi thì nhân viên quản lý là người sẽ cập nhật những thông tin đó vào hệ thống |
| 13 | Nhân viên quản lý | Quản lý người dùng | Use case này mô tả chức năng thêm, xóa và cập nhật thông tin người dùng vào hệ thống |
| 14 | Nhân viên quản lý | Quản lý hóa đơn | Use case này mô tả chức năng quản lý đơn hàng khi khách hàng đặt mua sản phẩm của cửa hàng và xác nhận hóa đơn cho khách hàng |
| 15 | Nhân viên quản lý | Quản lý phiếu đặt | Use case này mô tả chức năng thêm, xóa, sửa phiếu đặt khi cửa hàng đặt thêm sản phẩm từ các nhà cung cấp |
| 16 | Nhân viên quản lý | Quản lý phiếu nhập | Use case này mô tả chức năng thêm, xóa, sửa phiếu nhập khi cửa hàng xác nhận nhập thêm sản phẩm. |
| 17 | Nhân viên quản lý | Quản lý nhà cung cấp | Use case này mô tả chức năng thêm, xóa và khi thông tin về nhà cung cấp nào đó thay đổi thì sẽ thực hiện chức năng cập nhật lại thông tin của nhà cung cấp đó vào hệ thống |
| 18 | Nhân viên quản lý | Quản lý bình luận | Use case này mô tả chức năng thêm, xóa và khi thông tin về bình luận nào đó thay đổi thì sẽ thực hiện chức năng cập nhật lại thông tin của bình luận đó vào hệ thống |
| 19 | Nhân viên quản lý | Thống kê | Thống kê báo cáo doanh thu cửa hàng |

### 3.5.3 Biểu đồ Usecase tổng quát của hệ thống



**Hình 3.4. Mô hình Usecase của hệ thống Cửa hàng Playstation**

### 3.5.4 Đặc tả các usecase

**- Usecase Đăng nhập:**

**+ Tên:** Đăng nhập.

**+ Miêu tả:** Là khách hàng, muốn đăng nhập vào trang web.

**+ Actor:** Khách hàng

+ **Độ ưu tiên:** Phải có.

**+ Trigger:** Khách hàng muốn đăng nhập tài khoản.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã có tài khoản.
* Người dùng đã được phân quyền.

**+ Điều kiện sau:**

* Người dùng đăng nhập thành công.
* Hệ thống ghi nhận đăng nhập thành công.

**+ Luồng cơ bản:**

* Người dùng truy cập trang web
* Người dùng chọn đăng nhập
* Người dùng điền thông tin đăng nhập và tiến hành đăng nhập
* Hệ thống xác thực cho người dùng.
* Hệ thống ghi nhận đăng nhập thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế

**+ Các luồng ngoại lệ:**

* Xác thực thông tin đăng nhập thất bại và thông báo

- **Usecase đăng ký:**

**+ Tên:** Đăng ký.

**+ Miêu tả:** Là khách hàng, muốn đăng ký tài khoản.

**+ Actor:** Khách hàng.

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Khách hàng muốn đăng ký tài khoản.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.

**+ Điều kiện sau:**

* Người dùng đăng ký thành công.
* Hệ thống ghi nhận đăng ký thành công.

**+ Luồng cơ bản:**

* Người dùng truy cập trang web
* Người dùng thực hiện đăng ký
* Người dùng điền thông tin đăng ký và tiến hành đăng ký
* Hệ thống xác thực cho người dùng.
* Hệ thống ghi nhận đăng ký thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Xác thực thông tin đăng ký thất bại và thông báo

**- Usecase đăng xuất:**

**+ Tên:** Đăng xuất.

**+ Miêu tả:** Là khách hàng, đã đăng nhập, muốn đăng xuất.

**+ Actor:** Khách hàng.

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Khách hàng đăng nhập, muốn đăng xuất.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.

**+ Điều kiện sau:**

* Người dùng đăng xuất thành công.
* Hệ thống ghi nhận đăng xuất thành công.

**+ Luồng cơ bản:**

* Người dùng thực hiện đăng xuất.
* Hệ thống xác thực cho người dùng.
* Hệ thống ghi nhận đăng xuất thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Không có luồng ngoại lệ.

**- Usecase đổi mật khẩu:**

**+ Tên:** đổi mật khẩu.

**+ Miêu tả:** Là khách hàng, đã đăng nhập, muốn đổi mật khẩu.

**+ Actor:** Khách hàng.

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Khách hàng đã đăng nhập, muốn đổi mật khẩu.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.

**+ Điều kiện sau:**

* Người dùng đổi mật khẩu thành công.
* Hệ thống ghi nhận đổi mật khẩu thành công.

**+ Luồng cơ bản:**

* Người dùng thực hiện đổi mật khẩu.
* Người nhập mật khẩu mới và xác nhận mật khẩu, tiến hành đổi mật khẩu.
* Hệ thống xác thực cho người dùng.
* Hệ thống ghi nhận đổi mật khẩu thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Xác thực thông tin đổi mật khẩu thất bại và thông báo

**- Usecase tìm kiếm sản phẩm:**

**+ Tên:** tìm kiếm sản phẩm.

**+ Miêu tả:** Là khách hàng, muốn tìm kiếm sản phẩm

**+ Actor:** Khách hàng.

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Khách hàng muốn tìm kiếm sản phẩm.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.

**+ Điều kiện sau:**

* Người dùng tìm kiếm thành công.
* Hệ thống ghi nhận tìm kiếm thành công.
* Hệ thống hiển thị kết quả tìm kiếm.

**+ Luồng cơ bản:**

* Người dùng chọn loại sản phẩm hoặc tên của sản phẩm cần tìm
* Hệ thống xác thực cho người dùng.
* Hệ thống ghi nhận tìm kiếm thành công.
* Hệ thống hiển thị kết quả tìm kiếm.

**+ Luồng thay thế:**

* Không có luồng thay thế

**+ Các luồng ngoại lệ:**

* Không có luồng ngoại lệ.

**- Usecase xem giỏ hàng:**

**+ Tên:** xem giỏ hàng.

**+ Miêu tả:** Là khách hàng, đã đăng nhập, đã thêm sản phẩm vào giỏ hàng

**+ Actor:** Khách hàng.

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Khách hàng muốn xem giỏ hàng

+ **Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.
* Người dùng đã thêm sản phẩm vào giỏ hàng

**+ Điều kiện sau:**

* Người dùng xem giỏ hàng thành công.
* Hệ thống hiển thị danh sách giỏ hàng cho người dùng.

**+ Luồng cơ bản:**

* Người dùng chọn xem giỏ hàng.
* Hệ thống xác thực cho người dùng.
* Hệ thống hiển thị giỏ hàng cho người dùng..

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Không có luồng ngoại lệ

**- Usecase đặt hàng:**

**+ Tên:** đặt hàng**.**.

**+ Miêu tả:** Là khách hàng, đã đăng nhập, đã thêm sản phẩm vào giỏ, muốn đặt hàng.

**+ Actor:** Khách hàng.

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Khách hàng muốn đặt hàng.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.
* Giỏ hàng có sản phẩm.

**+ Điều kiện sau:**

+ Người dùng đặt hàng thành công.

+ Hệ thống ghi nhận đặt hàng thành công.

**+ Luồng cơ bản:**

* Người dùng chọn đặt hàng.
* Hệ thống xác thực cho người dùng.
* Hệ thống ghi nhận đặt hàng thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Hệ thông xác nhận thông tin đặt hàng thất bại, hiển thị thông báo.
* Giỏ hàng trống, đặt hàng thất bại, hiển thị thông báo.

**- Usecase chấm điểm và đánh giá cho sản phẩm:**

**+ Tên:** chấm điểm sản phẩm**.**.

**+ Miêu tả:** Là khách hàng, đã đăng nhập, muốn chấm điểm và đánh giá sản phẩm..

**+ Actor:** Khách hàng.

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Khách hàng muốn chấm điểm và đánh giá cho sản phẩm.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.

**+ Điều kiện sau:**

* Người dùng chấm điểm và đánh giá sản phẩm thành công.

**+ Luồng cơ bản:**

* Người dùng chọn xem thông tin sản phẩm
* Người dùng chấm điểm và đánh giá sản phẩm.
* Hệ thống xác thực cho người dùng.
* Hệ thống thông báo thành công cho người dùng..

**+ Luồng thay thế:**

* Không có luồng thay thế.

**Các luồng ngoại lệ:**

* Người dùng chưa đăng nhập, hiển thị thông báo thất bại.

**- Use case quản lý loại sản phẩm:**

**+ Tên:** quản lý loại sản phẩm**.**.

**+ Miêu tả:** Là nhân viên quản lý, đã đăng nhập

**+ Actor:** Nhân viên quản lý

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Nhân viên quản lý muốn thêm, xóa, sửa loại sản phẩm.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.
* Người dùng thuộc nhóm quyền quản lý.

**+ Điều kiện sau:**

* Người dùng thêm, xóa, sửa loại sản phẩm thành công.
* Hệ thống ghi nhận thêm, xóa, sửa loại sản phẩm thành công.

**+ Luồng cơ bản:**

* Thêm
  + Người dùng chọn thêm loại sản phẩm.
  + Người dùng nhập thông tin loại sản phẩm. thực hiện thao tác thêm.
* Xóa
  + Người dùng chọn loại sản phẩm cần xóa, thực hiện xóa
* Sửa
  + Người dùng chọn loại sản phẩm cần sửa
  + Người dùng nhập thông tin cần sửa. thực hiện thao tác sửa.
* Hệ thống xác thực cho người dùng.
* Hệ thống thông báo thành công cho người dùng.
* Hệ thống ghi nhận thêm loại sản phẩm thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Hệ thống xác thực các thao tác thất bại, hiển thị thông báo thất bại.

**- Use case quản lý sản phẩm:**

**+ Tên:** quản lý sản phẩm**.**.

**+ Miêu tả:** Là nhân viên quản lý, đã đăng nhập

**+ Actor:** Nhân viên quản lý

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Nhân viên quản lý muốn thêm, xóa, sửa sản phẩm.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.
* Người dùng thuộc nhóm quyền quản lý.

**+ Điều kiện sau:**

* Người dùng thêm, xóa, sửa sản phẩm thành công.
* Hệ thống ghi nhận thêm, xóa, sửa sản phẩm thành công.

**+ Luồng cơ bản:**

* Thêm
  + Người dùng chọn thêm sản phẩm.
  + Người dùng nhập thông tin sản phẩm. thực hiện thao tác thêm.
* Xóa
  + Người dùng chọn sản phẩm cần xóa, thực hiện xóa
* Sửa
  + Người dùng chọn sản phẩm cần sửa
  + Người dùng nhập thông tin cần sửa. thực hiện thao tác sửa.
* Hệ thống xác thực cho người dùng.
* Hệ thống thông báo thành công cho người dùng.
* Hệ thống ghi nhận thêm loại sản phẩm thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Hệ thống xác thực các thao tác thất bại, hiển thị thông báo thất bại.

**- Use case quản lý người dùng:**

**+ Tên:** quản lý người dùng**.**.

**+ Miêu tả:** Là nhân viên quản lý, đã đăng nhập

**+ Actor:** Nhân viên quản lý

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Nhân viên quản lý muốn thêm, xóa, sửa thông tin người dùng.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.
* Người dùng thuộc nhóm quyền quản lý.

**+ Điều kiện sau:**

* Người dùng thêm, xóa, sửa thông tin người dùng thành công.
* Hệ thống ghi nhận thêm, xóa, sửa thông tin người dùng thành công.

**+ Luồng cơ bản:**

* Thêm
  + Người dùng chọn thêm thông tin người dùng.
  + Người dùng nhập thông tin thông tin người dùng. thực hiện thao tác thêm.
* Xóa
  + Người dùng chọn thông tin người dùng cần xóa, thực hiện xóa
* Sửa
  + Người dùng chọn thông tin người dùng cần sửa
  + Người dùng nhập thông tin cần sửa. thực hiện thao tác sửa.
* Hệ thống xác thực cho người dùng.
* Hệ thống thông báo thành công cho người dùng.
* Hệ thống ghi nhận thêm thông tin người dùng thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Hệ thống xác thực các thao tác thất bại, hiển thị thông báo thất bại.

**- Use case quản lý hóa đơn:**

**+ Tên:** quản lý hóa đơn**.**.

**+ Miêu tả:** Là nhân viên quản lý, đã đăng nhập

**+ Actor:** Nhân viên quản lý

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Nhân viên quản lý muốn thêm, xóa, sửa và xác nhận hóa đơn.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.
* Người dùng thuộc nhóm quyền quản lý.

**+ Điều kiện sau:**

* Người dùng thêm, xóa, sửa và xác nhận hóa đơn thành công.
* Hệ thống ghi nhận thêm, xóa, sửa và xác nhận hóa đơn thành công.

**+ Luồng cơ bản:**

* Thêm
  + Người dùng chọn thêm hóa đơn.
  + Người dùng nhập thông tin hóa đơn. thực hiện thao tác thêm.
* Xóa
  + Người dùng chọn hóa đơn cần xóa, thực hiện xóa
* Sửa
  + Người dùng chọn hóa đơn cần sửa
  + Người dùng nhập thông tin cần sửa. thực hiện thao tác sửa.
* Xác nhận hóa đơn
* Người dùng chọn hóa đơn cần xác nhận chuyển giao mặt hàng cho khách hàng
* Người dùng thực hiện thao tác xác nhận
* Hệ thống xác thực cho người dùng.
* Hệ thống thông báo thành công cho người dùng.
* Hệ thống ghi nhận thêm hóa đơn thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Hệ thống xác thực các thao tác thất bại, hiển thị thông báo thất bại.

**- Use case quản lý phiếu đặt:**

**+ Tên:** quản lý phiếu đặt**.**.

**+ Miêu tả:** Là nhân viên quản lý, đã đăng nhập

**+ Actor:** Nhân viên quản lý

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Nhân viên quản lý muốn thêm, xóa, sửa phiếu đặt.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.
* Người dùng thuộc nhóm quyền quản lý.

**+ Điều kiện sau:**

* Người dùng thêm, xóa, sửa phiếu đặt thành công.
* Hệ thống ghi nhận thêm, xóa, sửa phiếu đặt thành công.

**+ Luồng cơ bản:**

* Thêm
  + Người dùng chọn thêm phiếu đặt.
  + Người dùng nhập thông tin phiếu đặt. thực hiện thao tác thêm.
* Xóa
  + Người dùng chọn phiếu đặt cần xóa, thực hiện xóa
* Sửa
  + Người dùng chọn phiếu đặt cần sửa
  + Người dùng nhập thông tin cần sửa. thực hiện thao tác sửa.
* Hệ thống xác thực cho người dùng.
* Hệ thống thông báo thành công cho người dùng.
* Hệ thống ghi nhận thêm phiếu đặt thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Hệ thống xác thực các thao tác thất bại, hiển thị thông báo thất bại.

**- Use case quản lý phiếu nhập:**

**+ Tên:** quản lý phiếu nhập**.**.

**+ Miêu tả:** Là nhân viên quản lý, đã đăng nhập

**+ Actor:** Nhân viên quản lý

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Nhân viên quản lý muốn thêm, xóa, sửa phiếu nhập.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.
* Người dùng thuộc nhóm quyền quản lý.

**+ Điều kiện sau:**

* Người dùng thêm, xóa, sửa phiếu nhập thành công.
* Hệ thống ghi nhận thêm, xóa, sửa phiếu nhập thành công.

**+ Luồng cơ bản:**

* Thêm
  + Người dùng chọn thêm phiếu nhập.
  + Người dùng nhập thông tin phiếu nhập. thực hiện thao tác thêm.
* Xóa
  + Người dùng chọn phiếu nhập cần xóa, thực hiện xóa
* Sửa
  + Người dùng chọn phiếu nhập cần sửa
  + Người dùng nhập thông tin cần sửa. thực hiện thao tác sửa.
* Hệ thống xác thực cho người dùng.
* Hệ thống thông báo thành công cho người dùng.
* Hệ thống ghi nhận thêm phiếu nhập thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Hệ thống xác thực các thao tác thất bại, hiển thị thông báo thất bại.

**- Use case quản lý nhà cung cấp:**

**+ Tên:** quản lý nhà cung cấp**.**.

**+ Miêu tả:** Là nhân viên quản lý, đã đăng nhập

**+ Actor:** Nhân viên quản lý

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Nhân viên quản lý muốn thêm, xóa, sửa thông tin nhà cung cấp.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.
* Người dùng thuộc nhóm quyền quản lý.

**+ Điều kiện sau:**

* Người dùng thêm, xóa, sửa thông tin nhà cung cấp thành công.
* Hệ thống ghi nhận thêm, xóa, sửa thông tin nhà cung cấp thành công.

**+ Luồng cơ bản:**

* Thêm
  + Người dùng chọn thêm nhà cung cấp.
  + Người dùng nhập thông tin nhà cung cấp. thực hiện thao tác thêm.
* Xóa
  + Người dùng chọn nhà cung cấp cần xóa, thực hiện xóa
* Sửa
  + Người dùng chọn nhà cung cấp cần sửa
  + Người dùng nhập thông tin cần sửa. thực hiện thao tác sửa.
* Hệ thống xác thực cho người dùng.
* Hệ thống thông báo thành công cho người dùng.
* Hệ thống ghi nhận thêm nhà cung cấp thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Hệ thống xác thực các thao tác thất bại, hiển thị thông báo thất bại.

**- Use case quản lý bình luận:**

**+ Tên:** quản lý bình luận**.**.

**+ Miêu tả:** Là nhân viên quản lý, đã đăng nhập

**+ Actor:** Nhân viên quản lý

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Nhân viên quản lý muốn thêm, xóa, sửa thông tin bình luận.

**+ Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.
* Người dùng thuộc nhóm quyền quản lý.

**+ Điều kiện sau:**

* Người dùng thêm, xóa, sửa thông tin bình luận thành công.
* Hệ thống ghi nhận thêm, xóa, sửa thông tin bình luận thành công.

**+ Luồng cơ bản:**

* Thêm
  + Người dùng chọn thêm bình luận.
  + Người dùng nhập thông tin bình luận. thực hiện thao tác thêm.
* Xóa
  + Người dùng chọn bình luận cần xóa, thực hiện xóa
* Sửa
  + Người dùng chọn bình luận cần sửa
  + Người dùng nhập thông tin cần sửa. thực hiện thao tác sửa.
* Hệ thống xác thực cho người dùng.
* Hệ thống thông báo thành công cho người dùng.
* Hệ thống ghi nhận thêm nhà cung cấp thành công.

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Hệ thống xác thực các thao tác thất bại, hiển thị thông báo thất bại.

**- Usecase thống kê:**

**+ Tên:** thống kê

**+ Miêu tả:** Là nhân viên quản lý, đã đăng nhập

**+ Actor:** Khách hàng.

**+ Độ ưu tiên:** Phải có.

**+ Trigger:** Khách hàng muốn xem giỏ hàng

+ **Điều kiện trước:**

* Thiết bị của người dùng có kết nối internet.
* Người dùng đã đăng nhập trước đó.
* Người dùng thuộc nhóm quyền quản lý.

**+ Điều kiện sau:**

* Người dùng thống kê báo cáo thành công.
* Hệ thống hiển thị thông số thống kê của cửa hàng cho người dùng.

**+ Luồng cơ bản:**

* Người dùng chọn xem thống kê
* Hệ thống xác thực cho người dùng.
* Hệ thống hiển thị thông tin thống kê cho người dùng..

**+ Luồng thay thế:**

* Không có luồng thay thế.

**+ Các luồng ngoại lệ:**

* Không có luồng ngoại lệ

## 3.6 Thiết lập mô hình thực thể

### 3.6.1 Xác định các thực thể

Product (**pr\_id**, pr\_name, price, quantity, description, made\_in, date\_update, image, total\_rating, status)

Category(**ca\_id**, ca\_name)

Supplier (**sup\_id**, sup\_name, sup\_address, sup\_phone)

User (**id**, name, gender, address, email, phone, account, password, active\_status)

Role (**ro\_id**, ro\_name)

Bill (**bil\_id**, total, date\_created, date\_checkout)

Bill\_Status (**bs\_id**, bs\_name)

Bill\_details (**bil\_id**, **pr\_id**, quantity)

Import (**im\_id**, date\_created, total\_price, status)

Place\_Order (**po\_id**, date\_created, total\_price, status)

Product\_Image(**id**, image\_url)

### 3.6.2 Xác định mối quan hệ giữa các thực thể

- Giữa 2 thực thể Category và Product có quan hệ với nhau theo quan hệ 1-nhiều. Một loại sản phẩm có thể có nhiều sản phẩm, một sản phẩm chỉ thuộc một loại sản phẩm.

- Giữa 2 thực thể Supplier và Product có quan hệ với nhau theo quan hệ 1-nhiều. Một nhà cung cấp có thể cung cấp nhiều sản phẩm, một sản phẩm chỉ thuộc một nhà cung cấp.

- Giữa 2 thực thể Supplier và Place\_order có quan hệ với nhau theo quan hệ 1- nhiều. Một nhà cung cấp có thể có nhiều đơn đặt hàng và một đơn đặt hàng chỉ thuộc một nhà cung cấp.

- Giữa 2 thực thể Role và User có quan hệ với nhau theo quan hệ 1-nhiều. Một quyền có thể có nhiều người dùng và một người dùng chỉ thuộc một quyền.

- Giữa 2 thực thể Bill\_status và Bill có quan hệ với nhau theo quan hệ 1-nhiều. Một trang thái hóa đơn có thể có nhiều hóa đơn và một hóa đơn chỉ thuộc một trạng thái tại mỗi thời điểm

- Giữa 2 thực thể User (Customer) và Bill có quan hệ với nhau theo quan hệ 1-nhiều. Một người dùng có thể có nhiều đơn hàng và một đơn hàng chỉ thuộc một người dùng.

- Giữa 2 thực thể Bill và Product có quan hệ với nhau theo quan hệ nhiều-nhiều. Một sản phẩm có thể thuộc nhiều đơn hàng và một đơn hàng có thể có nhiều sản phẩm.

- Giữa 2 thực thể Product và Place\_order có quan hệ với nhau theo quan hệ nhiều-nhiều. Một sản phẩm có thể thuộc nhiều đơn đặt hàng và một đơn đặt hàng có thể có nhiều sản phẩm.

- Giữa 2 thực thể Product và Import có quan hệ với nhau theo quan hệ nhiều-nhiều. Một sản phẩm có thể thuộc nhiều đơn nhập hàng và một đơn nhập hàng có thể có nhiều sản phẩm.

- Giữa 2 thực thể Import và Place\_order có quan hệ với nhau theo quan hệ 1-1. Một đơn nhập hàng chỉ được lập cho một đơn đặt hàng và một đơn đặt hàng chỉ thuộc một đơn đặt hàng.

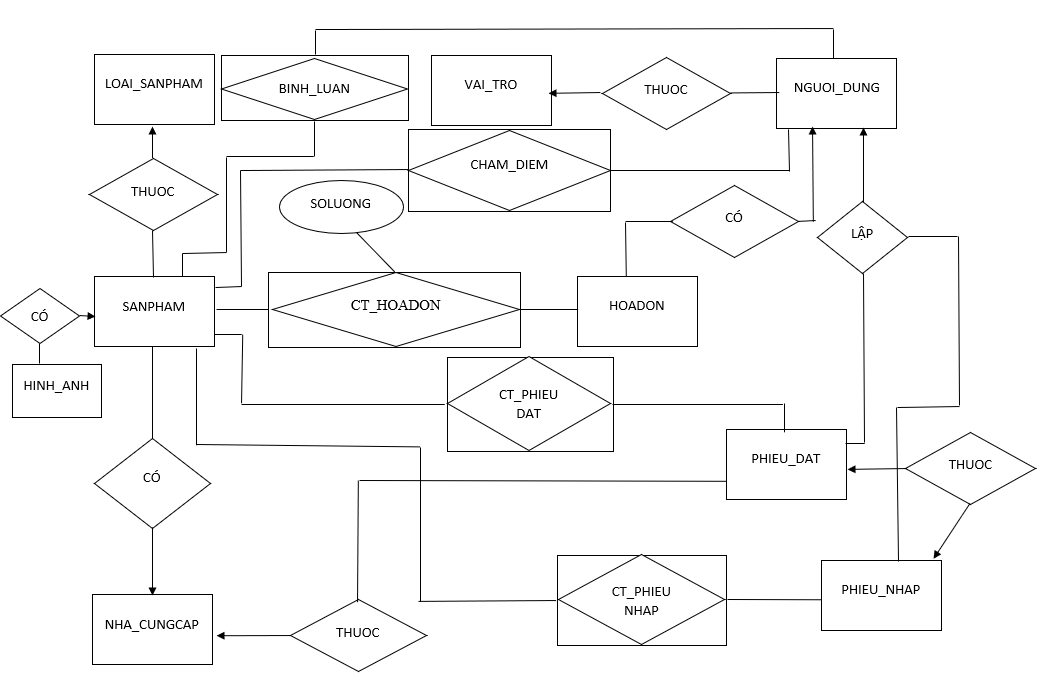
- Giữa 2 thực thể User (Employee) và Place\_order có quan hệ với nhau theo quan hệ 1-nhiều. Một nhân viên có thể lập nhiều đơn đặt hàng và một đơn đặt hàng chỉ thuộc một nhân viên lập.

- Giữa 2 thực thể User (Employee) và Import có quan hệ với nhau theo quan hệ 1-nhiều. Một nhân viên có thể lập nhiều đơn nhập hàng và một đơn nhập hàng chỉ thuộc một nhân viên lập.

- Giữa 2 thực thể User và Product có quan hệ với nhau theo quan hệ nhiều-nhiều. Một người dùng có thể bình chọn điểm và bình luận cho nhiều sản phẩm và một sản phẩm được nhiều người dùng bình chọn điểm và bình luận

- Giữa 2 thực thể Product và Product\_Image có quan hệ với nhau theo quan hệ một-nhiều. Một sản phẩm có thể có nhiều hình ảnh

### 3.6.3 Sơ đồ ERD (Entity Relationship Diagram)



**Sơ đồ 3.1. Sơ đồ ERD Quản lý cửa hàng máy Playstation**

### 3.6.4 Mô hình dữ liệu quan hệ

Product (**pr\_id**, pr\_name, price, quantity, description, made\_in, date\_update, image, total\_rating, status, **ca\_id**, **sup\_id**)

Category(**ca\_id**, ca\_name)

Supplier (**sup\_id**, sup\_name, sup\_address, sup\_phone)

User (**id**, name, gender, address, email, phone, account, password, active\_status, **ro\_id**)

Role (**ro\_id**, ro\_name)

Bill (**bil\_id**, total, date\_created, date\_checkout, **cus\_id**, **bs\_id**)

Bill\_Status (**bs\_id**, bs\_name)

Bill\_details (**bil\_id**, **pr\_id**, quantity)

Import (**im\_id**, date\_created, total\_price, status, **po\_id, emp\_id**)

Place\_Order (**po\_id**, date\_created, total\_price, status, **sup\_id, emp\_id**)

Import\_Details(**im\_id, pr\_id**, quantity, price)

Place\_Order\_Details(**po\_id, pr\_id**, quantity, price)

Rating(**cus\_id, pr\_id**, ratingpoint)

Comment(**id, cus\_id, pr\_id,** date\_comment, content)

Product\_Image(**id**, image\_url, **pr\_id**)

## 3.7 Thiết kế dữ liệu

### 3.7.1 Từ điển dữ liệu

- Bảng Product

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Id | Serial |  | Mã sản phẩm | x |  | x |
| **2** | Name | Varchar | 50 | Tên sản phẩm |  |  | x |
| **3** | Desciption | Varchar | 100 | Mô tả sản phẩm |  |  | x |
| **4** | Quantity | Int |  | Số lượng |  |  | x |
| **5** | Price | Float |  | Giá sản phẩm |  |  | x |
| **6** | Image | Varchar | 500 | Hình ảnh |  |  | x |
| **7** | CreatedDate | Date |  | Ngày thêm sản phẩm |  |  | x |
| **8** | UpdatedDate | Date |  | Ngày cập nhật sản phẩm |  |  | x |
| **9** | Category\_id | Int |  | Mã loại sản phẩm |  | x | x |
| **10** | Supplier\_id | Int |  | Mã nhà cung cấp |  | x | x |
| **11** | TotalRating | Float |  | Tổng số điểm bình chọn |  |  | x |
| **12** | Status | Varchar | 15 | Tình trạng sản phẩm |  |  | x |

**Bảng 3.1. Bảng Product**

- Bảng Category:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Id | Serial |  | Mã loại sản phẩm | x |  | x |
| **2** | Name | Varchar | 50 | Tên loại sản phẩm |  |  | x |
| **3** | Desciption | Varchar | 100 | Mô tả loại sản phẩm |  |  | x |
| **4** | Status | Varchar | 15 | Trạng thái |  |  | x |

**Bảng 3.2. Bảng Category**

- Bảng Role:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Id | Serial |  | Mã vai trò | x |  | x |
| 2 | Desciption | Varchar | 50 | Mô tả vai trò |  |  | x |

**Bảng 3.3. Bảng Role**

- Bảng User:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Id | Serial |  | Mã người dùng | x |  | x |
| **2** | Name | Varchar | 50 | Họ tên người dùng |  |  | x |
| **3** | Gender | Varchar | 10 | Giới tính |  |  | x |
| **4** | Address | Varchar | 100 | Địa chỉ |  |  | x |
| **5** | Email | Varchar | 50 | Email |  |  | x |
| **6** | Phone | Varchar | 15 | Số điện thoại |  |  | x |
| **7** | Account | Varchar | 50 | Tài khoản |  |  | x |
| **8** | Password | Varchar | 500 | Mật khẩu |  |  | x |
| **9** | Active\_Status | Varchar | 20 | Trạng thái tài khoản |  |  | x |
| **10** | Role\_id | Int |  | Vai trò |  | x | x |

**Bảng 3.4. Bảng User**

- Bảng Supplier:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Id | Serial |  | Mã nhà cung cấp | x |  | x |
| **2** | Name | Varchar | 100 | Tên nhà cung cấp |  |  | x |
| **3** | Address | Varchar | 100 | Địa chỉ nhà cung cấp |  |  | x |
| **4** | Phone | Varchar | 20 | Số điện thoại nhà cung cấp |  |  | x |
| **5** | Status | Varchar | 15 |  |  |  | x |

**Bảng 3.5. Bảng Supplier**

- Bảng Rating:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Customer\_id | Int |  | Mã khách hàng | x | x | x |
| **2** | Product\_id | Int |  | Mã sản phẩm | x | x | x |
| **3** | RatingPoint | Float |  | Điểm bình chọn |  |  | x |

**Bảng 3.6. Bảng Rating**

- Bảng BillStatus:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Id | Serial |  | Mã trạng thái hóa đơn | x |  | x |
| **2** | Description | Varchar | 50 | Mô tả trạng thái |  |  | x |

**Bảng 3.7. Bảng BillStatus**

- Bảng Bill:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Id | Serial |  | Mã hóa đơn | x |  | x |
| **2** | Total | Float |  | Tổng tiền |  |  | x |
| **3** | CreatedDate | Date |  | Ngày tạo hóa đơn |  |  | x |
| **4** | UpdatedDate | Date |  | Ngày cập nhật hóa đơn |  |  | x |
| **5** | Customer\_id | Int |  | Mã khách hàng |  | x | x |
| **6** | Status | Varchar | 15 | Trạng thái hóa đơn |  |  | x |

**Bảng 3.8. Bảng Bill**

- Bảng BillDetails:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Bill\_id | Int |  | Mã hóa đơn | x | x | x |
| **2** | Product\_id | Int |  | Mã sản phẩm | x | x | x |
| **3** | Quantity | Int |  | Số lượng |  |  | x |

**Bảng 3.9. Bảng BillDetails**

- Bảng Import:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Id | Serial |  | Mã hóa đơn | x |  | x |
| **2** | Total | Float |  | Tổng tiền |  |  | x |
| **3** | Status | Varchar | 15 | Tình trạng phiếu nhập |  |  | x |
| **4** | CreatedDate | Date |  | Ngày tạo hóa đơn |  |  | x |
| **5** | Place\_Order\_id | Int |  | Mã phiếu đặt |  | x | x |
| **6** | Employee\_id | Int |  | Mã nhân viên |  | x | x |

**Bảng 3.10. Bảng Import**

-Bảng PlaceOrder:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Id | Serial |  | Mã hóa đơn | x |  | x |
| **2** | Total | Float |  | Tổng tiền |  |  | x |
| **3** | Status | Varchar | 15 | Tình trạng phiếu đặt |  |  | x |
| **4** | CreatedDate | Date |  | Ngày tạo hóa đơn |  |  | x |
| **5** | Employee\_id | Int |  | Mã nhân viên |  | x | x |
| **6** | Supplier\_id | Int |  | Mã nhà cung cấp |  | x | x |

**Bảng 3.11. Bảng Place Order**

- Bảng Import\_Details:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Import\_id | Int |  | Mã phiếu nhập | x | x | x |
| **2** | Product\_id | Int |  | Mã sản phẩm | x | x | x |
| **3** | Quantity | Int |  | Số lượng |  |  | x |
| **4** | Price | Float |  | Giá |  |  | x |

**Bảng 3.12. Bảng ImportDetails**

- Bảng PlaceOrder\_Details:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | PlaceOrder\_id | Int |  | Mã phiếu đặt | x | x | x |
| **2** | Product\_id | Int |  | Mã sản phẩm | x | x | x |
| **3** | Quantity | Int |  | Số lượng |  |  | x |
| **4** | Price | Float |  | Giá |  |  | x |

**Bảng 3.13. Bảng PlaceOrderDetails**

- Bảng Comment:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Id | Serial |  | Mã bình luận | x |  | x |
| **2** | Date\_Comment | Date |  | Thời gian bình luận |  |  | x |
| **3** | Content | Varchar | 255 | Nội dung |  |  | x |
| **4** | Customer\_id | Int |  | Mã khách hàng |  | x | x |
| **5** | Product\_id | Int |  | Mã sản phẩm |  | x | x |

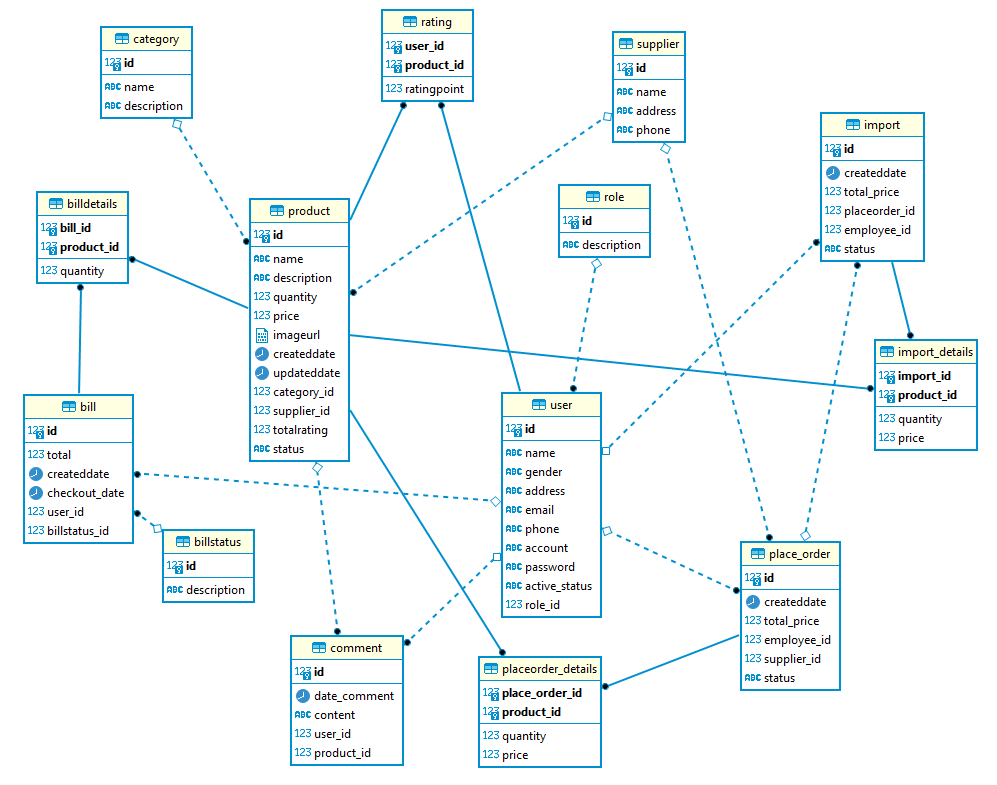
**Bảng 3.14. Bảng Comment**

- Bảng Product\_Image:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | Tên trường | Kiểu | Độ dài | Diễn giải | Khóa chính | Khóa ngoại | Not Null |
| **1** | Id | Serial |  | Mã hình ảnh | x |  | x |
| **2** | Image\_url | Varchar | 500 | Đường dẫn hình ảnh |  |  | x |
| **3** | Product\_id | Int |  | Mã sản phẩm |  | x | x |

**Bảng 3.15 Bảng Product\_Image**

### 3.7.2 Mô hình diagram



**Hình 3.5. Mô hình Diagram cơ sở dữ liệu Quản lý cửa hàng máy Playstation**

## 3.8 Xây dựng dự án

### 3.8.1 Xây dựng BackEnd với Spring Boot

- Khai báo đầy đủ trong file application.properties:

spring.datasource.url=jdbc:postgresql://localhost:5432/NashTechProject  
spring.datasource.username=postgres  
spring.datasource.password=123456  
spring.jpa.show-sql=true  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect  
spring.jpa.hibernate.ddl-auto=update  
*# App Properties*rookies.app.jwtSecret=rookiesSecretKey  
rookies.app.jwtExpirationMs=86400000  
*# Send Email*spring.mail.default-encoding=UTF-8  
spring.mail.host=smtp.gmail.com  
spring.mail.protocol=smtp  
spring.mail.username=ps4gamemachine@gmail.com  
spring.mail.password=ps5console2021  
spring.mail.port=587  
spring.mail.properties.mail.smtp.auth=true  
spring.mail.properties.mail.smtp.starttls.enable=true  
mail.smtp.debug=true

- Kỹ thuật JWT (Json Web Token): (Trích dẫn 1 số phần chính)

+ Lớp JwtUtil.java: thực hiện hai chức năng chính là tạo mới JWT và validate JWT.

@Component  
public class JwtUtils {  
 private static final Logger *logger* = LoggerFactory.*getLogger*(JwtUtils.class);  
  
 private static List<String> *jwtBlackList* = new ArrayList<String>();  
  
 @Value("${rookies.app.jwtSecret}")  
 private String jwtSecret;  
  
 @Value("${rookies.app.jwtExpirationMs}")  
 private int jwtExpirationMs;  
  
 public String generateJwtToken(Authentication authentication) {  
  
 UserDetailsImpl userPrincipal = (UserDetailsImpl) authentication.getPrincipal();  
  
 return Jwts.*builder*()  
 .setSubject((userPrincipal.getUsername()))  
 .setIssuedAt(new Date())  
 .setExpiration(new Date((new Date()).getTime() + jwtExpirationMs))  
 .signWith(SignatureAlgorithm.*HS512*, jwtSecret)  
 .compact();  
 }  
  
 public String getUserNameFromJwtToken(String token) {  
 return Jwts.*parser*().setSigningKey(jwtSecret).parseClaimsJws(token).getBody().getSubject();  
 }  
  
 public boolean validateJwtToken(String authToken) {  
 try {  
 Jwts.*parser*().setSigningKey(jwtSecret).parseClaimsJws(authToken);  
 return true;  
 } catch (SignatureException e) {  
 *logger*.error("Invalid JWT signature: {}", e.getMessage());  
 } catch (MalformedJwtException e) {  
 *logger*.error("Invalid JWT token: {}", e.getMessage());  
 } catch (ExpiredJwtException e) {  
 *logger*.error("JWT token is expired: {}", e.getMessage());  
 } catch (UnsupportedJwtException e) {  
 *logger*.error("JWT token is unsupported: {}", e.getMessage());  
 } catch (IllegalArgumentException e) {  
 *logger*.error("JWT claims string is empty: {}", e.getMessage());  
 }  
  
 return false;  
 }  
  
 public boolean addToBlackList(String jwtToken) throws Exception {  
 if (isInBlacklist(jwtSecret)){  
 return false;  
 }  
 *jwtBlackList*.add(jwtToken);  
 return true;  
 }  
  
 public boolean isInBlacklist(String jwtSecret) throws Exception {  
 if (*jwtBlackList*.contains(jwtSecret)){  
 *//throw new Exception("JWT Token is in blacklist");* return true;  
 }  
 return false;  
 }  
  
 public Boolean isExpire(String jwt){  
 Date expiration = Jwts.*parser*().setSigningKey(jwtSecret).parseClaimsJws(jwt).getBody().getExpiration();  
 Date now = new Date();  
 if(expiration.before(now))  
 return true;  
 return false;  
 }  
  
 @Scheduled(cron="0 0 1 \* \*")  
 public void deleteExpireToken(){  
 for (String jwt : *jwtBlackList*) {  
 if(isExpire(jwt))  
 *jwtBlackList*.remove(jwt);  
 }  
 }  
}

+ Lớp UserDetailServiceImpl: thực hiện implement UserDetails của Spring Boot và định nghĩa override một số method theo nhu cầu.

@Service  
public class UserDetailsServiceImpl implements UserDetailsService {  
  
 private final UserRepository userRepository;  
  
 public UserDetailsServiceImpl (UserRepository userRepository) {  
 this.userRepository = userRepository;  
 }  
  
 @Override  
 @Transactional  
 public UserDetails loadUserByUsername(String username)  
 throws UsernameNotFoundException {  
  
 User user = userRepository.findByAccountAndActivestatusNot(username, "Inactive")  
 .orElseThrow(() ->  
 new UsernameNotFoundException("User Not Found with -> username or email : " + username)  
 );  
  
 return UserDetailsImpl.*build*(user);  
 }  
}

+ Lớp AuthController: phục vụ cho việc đăng nhập, đăng ký, đăng xuất hoặc thay đổi mật khẩu của user.

@CrossOrigin(origins = "\*", maxAge = 3600)  
@RestController  
@RequestMapping("/api/auth")  
public class AuthController {  
 @Autowired  
 private AuthService authService;  
  
 final private AuthenticationManager authenticationManager;  
  
 final private UserRepository userRepository;  
  
 final private RoleRepository roleRepository;  
  
 final private PasswordEncoder encoder;  
  
 final private JwtUtils jwtUtils;  
  
 public AuthController (AuthenticationManager authenticationManager, UserRepository userRepository,  
 RoleRepository roleRepository, PasswordEncoder encoder, JwtUtils jwtUtils) {  
 this.authenticationManager = authenticationManager;  
 this.userRepository = userRepository;  
 this.roleRepository = roleRepository;  
 this.encoder = encoder;  
 this.jwtUtils = jwtUtils;  
 }  
  
 @PostMapping("/signin")  
 public ResponseEntity<?> authenticateUser(@Valid @RequestBody LoginRequest loginRequest) {JwtResponse jp = authService.getJwtResponse(loginRequest);  
  
 return ResponseEntity.*ok*(jp);  
 }  
  
 @PostMapping("/signup")  
 public ResponseEntity<?> registerUser(@Valid @RequestBody SignupRequest signUpRequest) {  
 return authService.getUserSignUp(signUpRequest);  
 }  
  
 @PostMapping("/profile")  
 public ResponseEntity<?> changePasswordUser(@Valid @RequestBody ChangPasswordRequest changPasswordRequest)  
 {  
 return authService.getUserChangePassword(changPasswordRequest);  
 }  
  
 @GetMapping("/logout")  
 public ResponseEntity<?> logoutUser(HttpServletRequest request, HttpServletResponse response) throws Exception {  
 String jwt = JwtAuthTokenFilter.*parseJwt*(request);  
 if (jwt != null && jwtUtils.addToBlackList(jwt)){  
 return ResponseEntity.*ok*(new MessageResponse("User logged out successfully!"));  
 }  
 return ResponseEntity  
 .*badRequest*()  
 .body(new MessageResponse("Error: Cannot add jwt token to blacklist!"));  
 }  
}

- Xây dựng tầng Controller: Trích dẫn minh họa ProductController, tương tự cho các Controller khác

@CrossOrigin(origins = "\*", maxAge = 3600)  
@RestController  
@RequestMapping("api/products")  
@Api(tags = "Products Rest Controller")  
public class ProductController {  
 @Autowired  
 private ProductService productService;  
  
 @Autowired  
 private CategoryService categoryService;  
 @Autowired  
 private SupplierService supplierService;   
 @Autowired  
 private ModelMapper modelMapper;  
  
 @GetMapping  
 @ApiOperation(value = "Get all products")  
 @ApiResponses(value = { @ApiResponse(code = 200, message = "Success"),  
 @ApiResponse(code = 400, message = "Bad request"),  
 @ApiResponse(code = 500, message = "Internal server error") })  
 @ApiImplicitParams(  
 value = {  
 @ApiImplicitParam(name = "page", value = "page number start from 0", dataType = "integer",  
 examples = @Example(@ExampleProperty("1")), paramType = "query"),  
 @ApiImplicitParam(name = "size", value = "maximum number of item in page", dataType = "integer",  
 examples = @Example(@ExampleProperty("40")), paramType = "query"),  
 }  
 )  
 public List<ProductDTO> getAllProducts()  
 {  
 List<ProductDTO> prosDTO = new ArrayList<>();  
 List<Product> products = productService.retrieveProducts();  
 ratingNow(products);  
 for (int i = 0; i < products.size(); i++) {  
 ProductDTO p = convertToDTO(products.get(i));  
 prosDTO.add(p);  
 }  
 return prosDTO.stream().sorted(Comparator.*comparingLong*(ProductDTO::getId)).collect(Collectors.*toList*());  
 }  
  
 @ApiOperation(value = "Create new Product")  
 @ApiResponses(value = { @ApiResponse(code = 200, message = "Success"),  
 @ApiResponse(code = 400, message = "Bad request"),  
 @ApiResponse(code = 500, message = "Internal server error") })  
 @PostMapping()  
 public ProductDTO saveProduct(@RequestBody ProductDTO product)  
 {  
 Category cate = categoryService.getCategory(Long.*valueOf*(product.getCategory\_id()));  
 if (cate == null)  
 {  
 throw new CategoryException(cate.getId());  
 }  
 Supplier sup = supplierService.getSupplier(Long.*valueOf*(product.getSupplier\_id()));  
 if (sup == null)  
 {  
 throw new SupplierException(sup.getId());  
 }  
 if (productService.getProductByName(product.getName()) != null)  
 {  
 throw new ProductException(product.getName());  
 }  
 Product pro = convertToEntity(product);  
 pro.setCreateddate(LocalDateTime.*now*());  
 pro.setUpdateddate(LocalDateTime.*now*());  
 pro.setStatus("On Sale");

return convertToDTO(productService.saveProduct(pro));  
 }  
  
 @ApiOperation(value = "Update Product")  
 @ApiResponses(value = { @ApiResponse(code = 200, message = "Success"),  
 @ApiResponse(code = 400, message = "Bad request"),  
 @ApiResponse(code = 500, message = "Internal server error") })  
 @PutMapping("/{productId}")  
 public ProductDTO updateProduct(@PathVariable(name = "productId") Long productId, @Valid @RequestBody ProductDTO productDetails)  
 {  
 Product product = productService.getProduct(productId);  
 if (product == null)  
 {  
 throw new ProductException(productId);  
 }  
 else  
 {  
 Category cate = categoryService.getCategory(Long.*valueOf*(productDetails.getCategory\_id()));  
 if (cate == null)  
 {  
 throw new CategoryException(cate.getId());  
 }  
 Supplier sup = supplierService.getSupplier(Long.*valueOf*(productDetails.getSupplier\_id()));  
 if (sup == null)  
 {  
 throw new SupplierException(sup.getId());  
 }  
 ProductUpdate(product, productDetails);  
 product.setCategory(cate);  
 product.setSupplier(sup);  
 productService.updateProduct(product);  
 }  
 return convertToDTO(product);  
 }  
  
 @ApiOperation(value = "Delete Product")  
 @ApiResponses(value = { @ApiResponse(code = 200, message = "Success"),  
 @ApiResponse(code = 400, message = "Bad request"),  
 @ApiResponse(code = 500, message = "Internal server error") })  
 @DeleteMapping("/{productId}")  
 public ResponseEntity<?> deleteProduct(@PathVariable(name = "productId") Long productId)  
 {  
 Product product = productService.getProduct(productId);  
 if (product == null)  
 {  
 throw new ProductException(productId);  
 }  
 productService.deleteProduct(productId);

return ResponseEntity.*ok*(new MessageResponse("Delete Successfully"));  
 }  
  
 private ProductDTO convertToDTO(Product p)  
 {  
 ProductDTO prodto = modelMapper.map(p, ProductDTO.class);  
 String cate\_id = String.*valueOf*(p.getCategory().getId());  
 prodto.setCategory\_id(cate\_id);  
 String sup = String.*valueOf*(p.getSupplier().getId());  
 prodto.setSupplier\_id(sup);  
 return prodto;  
 }  
  
 private Product convertToEntity(ProductDTO p)  
 {  
 Product pro = modelMapper.map(p, Product.class);  
 Category c = categoryService.getCategory(Long.*valueOf*(p.getCategory\_id()));  
 pro.setCategory(c);  
 Supplier s = supplierService.getSupplier(Long.*valueOf*(p.getSupplier\_id()));  
 pro.setSupplier(s);  
 return pro;  
 }

private void ProductUpdate(Product product, ProductDTO productDetails)  
 {  
 product.setName(productDetails.getName());  
 product.setDescription(productDetails.getDescription());  
 product.setQuantity(productDetails.getQuantity());  
 product.setPrice(productDetails.getPrice());  
 product.setImageurl(productDetails.getImageurl());  
 product.setTotalrating(productDetails.getTotalrating());  
 product.setUpdateddate(LocalDateTime.*now*());  
 product.setStatus(productDetails.getStatus());  
 }  
}

- Xây dựng tầng Service: Trích dẫn minh họa ProductService, tương tự cho các Service khác

+ Tầng Servive định nghĩa các interface

public interface ProductService {  
 public List<Product> retrieveProducts();  
  
 public Product saveProduct(Product product);  
  
 public void deleteProduct(Long productId);  
  
 public void updateProduct(Product product);

}

+ Class ProductServiceImpl implements ProductService:

@Service  
public class ProductServiceImpl implements ProductService {  
 @Autowired  
 private ProductRepository productRepository;  
  
 @Autowired  
 private ModelMapper modelMapper;  
  
 public void setProductRepository(ProductRepository productRepository)  
 {  
 this.productRepository = productRepository;  
 }  
  
 public List<Product> retrieveProducts()  
 {  
 List<Product> products = productRepository.findAll();  
 return products;  
 }  
  
 @Override  
 public Product saveProduct(Product pro) {  
 return productRepository.save(pro);  
 }  
  
 @Override  
 public void deleteProduct(Long productId) {  
 Product pro = productRepository.findById(productId).get();  
  
 productRepository.delete(pro);  
 }  
  
 @Override  
 public void updateProduct(Product pro) {  
 productRepository.save(pro);  
 }  
}

- Xây dựng tầng Repository: Trích dẫn minh họa ProductRepository, tương tự cho các Repository khác

+ Đây là tầng sẽ có nhiệm vụ chính là thao tác trực tiếp xuống CSDL. Trong phần này, ta sẽ xây dựng các repository cho các thực thể.

public interface ProductRepository extends JpaRepository<Product, Long> {  
 List<Product> findByCategoryIdAndStatus(Long categoryId, String status);  
  
 Page<Product> findByCategoryIdAndStatus(Long categoryId, String status, Pageable pageable);  
  
 List<Product> findByNameContainsAndStatus(String pro\_name, String status);  
  
 Page<Product> findByNameContainsAndStatus(String pro\_name, String status, Pageable pageable);  
  
 Boolean existsByName(String name);  
  
 List<Product> findByStatus(String status);  
  
 Page<Product> findByStatus(String status, Pageable pageable);  
  
 List<Product> findByOrderByTotalratingDesc();  
}

### 3.8.2 Xây dựng FrontEnd với ReactJS

- Xây dựng httpHelper thực hiện gọi API bẳng Axios: GET, POST, PUT, DELETE và những hàm cần xác thực sẽ dùng thêm Authorization trong header

*const endpoint = 'http://localhost:8080/api';*

*const accessToken = localStorage.getItem("accessToken");*

*export function get(url)*

*{ return axios.get(endpoint + url, {*

*headers:{*

*Authorization: `Bearer ${accessToken}`,*

*"Access-Control-Allow-Origin": "\*",*

*} }); }*

*export function post(url, body)*

*{ return axios.post(endpoint + url, body, {*

*headers:{*

*Authorization: `Bearer ${accessToken}`,*

*"Access-Control-Allow-Origin": "\*",*

*} }); }*

*export function put(url, body)*

*{  return axios.put(endpoint + url, body, {*

*headers:{*

*Authorization: `Bearer ${accessToken}`,*

*"Access-Control-Allow-Origin": "\*",*

*} }); }*

*export function del(url)*

*{ return axios.delete(endpoint + url, {*

*headers:{*

*Authorization: `Bearer ${accessToken}`,*

*"Access-Control-Allow-Origin": "\*",*

*}  }); }*

- Xây dựng các Components:

+ Hiển thị danh sách (Trích dẫn minh họa cho Product):

Khai báo mảng danh sách sản phẩm

*constructor(props) {*

*super(props);*

*this.state = {*

*Products: [],*

*};*

*}*

Áp dụng phương thức gọi API

*get(`/products`)*

*.then((response) => {*

*this.setState({*

*Products: response.data,*

*});*

*console.log(response.data)*

*})*

*.catch(error => console.log(error));*

Danh sách sản phẩm được render ra màn hình bằng các thành phần HTML cơ bản như Container, Row, Col với sự hỗ trợ của React.

+ Thêm mới (Trích dẫn minh họa cho Product): đối tượng mới thêm sẽ hiển thị lên danh sách

*post(`/products`, {name: newProduct.name.trim(), description: newProduct.description.trim(), quantity: newProduct.quantity,price: newProduct.price, totalrating: 0,imageurl: newProduct.imageurl, category\_id: newProduct.category\_id, supplier\_id: newProduct.supplier\_id})*

*.then((response) => {*

*this.setState({*

*products: [...this.state.products, response.data],*

*});*

*});*

+ Cập nhật (Trích dẫn minh họa cho Product):

*put(`/products/${id}`, {name: this.state.name.trim(), description:this.state.description.trim(), quantity: this.state.quantity, price: this.state.price, totalrating:this.state.totalrating ,imageurl: this.state.imageurl, category\_id: this.state.category\_id, supplier\_id: this.state.supplier\_id, status: this.state.status})*

*.then((response) => {*

*if (response.status === 200)*

*{*

*console.log(response.data);*

*}*

*})*

*.catch((error) => {})*

+ Xóa (Trích dẫn minh họa cho Product): xóa khỏi danh sách products

*del(`/products/${id}`)*

*.then((response) => {*

*this.setState({products: this.state.products.filter(p => p.id !== id)})*

*})*

+ Đăng nhập: Thông tin xác thực đúng và đăng nhập thành công, server sẽ trả về những thông tin cần thiết như tên đăng nhập, chuỗi xác thực jwt hay quyền tương ứng,...

*postLogin('/auth/signin', {username: this.state.username, password: this.state.password})*

*.then((response) => {*

*if (response.status === 200)*

*{*

*localStorage.setItem('accessToken', response.data.accessToken);*

*localStorage.setItem('role', response.data.roles[0])*

*localStorage.setItem('user\_id', response.data.id);*

*localStorage.setItem('username', response.data.username);*

*if (response.data.roles[0] === "ADMIN" || response.data.roles[0] === "STAFF")*

*{*

*this.props.history.push("/admin");*

*window.location.reload();*

*}*

*else if (response.data.roles[0] === "USER")*

*{*

*this.props.history.push("/");*

*window.location.reload();*

*}*

*}*

*})*

*.catch(error => alert('Username or Password is wrong!'));*

+ Đăng ký:

*post(`/auth/signup`, {name: this.state.name, gender: this.state.gender, address: this.state.address,email: this.state.email, phone: this.state.phone, username: this.state.username,password: this.state.password, role: this.state.role})*

*.then((response) => {*

*if (response.status === 200)*

*{*

*localStorage.setItem('accessToken', response.data.accessToken);*

*localStorage.setItem('user\_id', response.data.id);*

*localStorage.setItem('username', response.data.username);*

*}*

*})*

*.catch(error => console.log(error));*

### 3.8.3 Thực hiện thanh toán trực tuyến PayPal

**3.8.3.1 Tổng quan về PayPal Sandbox**

- Paypal được hiểu là một cổng thanh toán trực tuyến, chuyển tiền quốc tế (dịch vụ trung gian) giúp bạn thanh toán khi mua sắm trực tuyến quốc tế, hoặc nhận chi trả từ tài khoản nước ngoài về ngân hàng tại Việt Nam.

- Là công cụ thay thế các phương thức giao dịch chuyển tiền truyền thống như thư/lệnh chuyển tiền, SEC, Western Union…

- Paypal hoạt động trên nền thương mại điện tử thông qua internet, và thu phí khi phát sinh giao dịch chuyển – rút tiền. Với tốc độ phát triển không ngừng của thanh toán trực tuyến, ngày nay Paypal đem lại rất nhiều lợi ích cho người sử dụng như: an toàn, nhanh chóng.

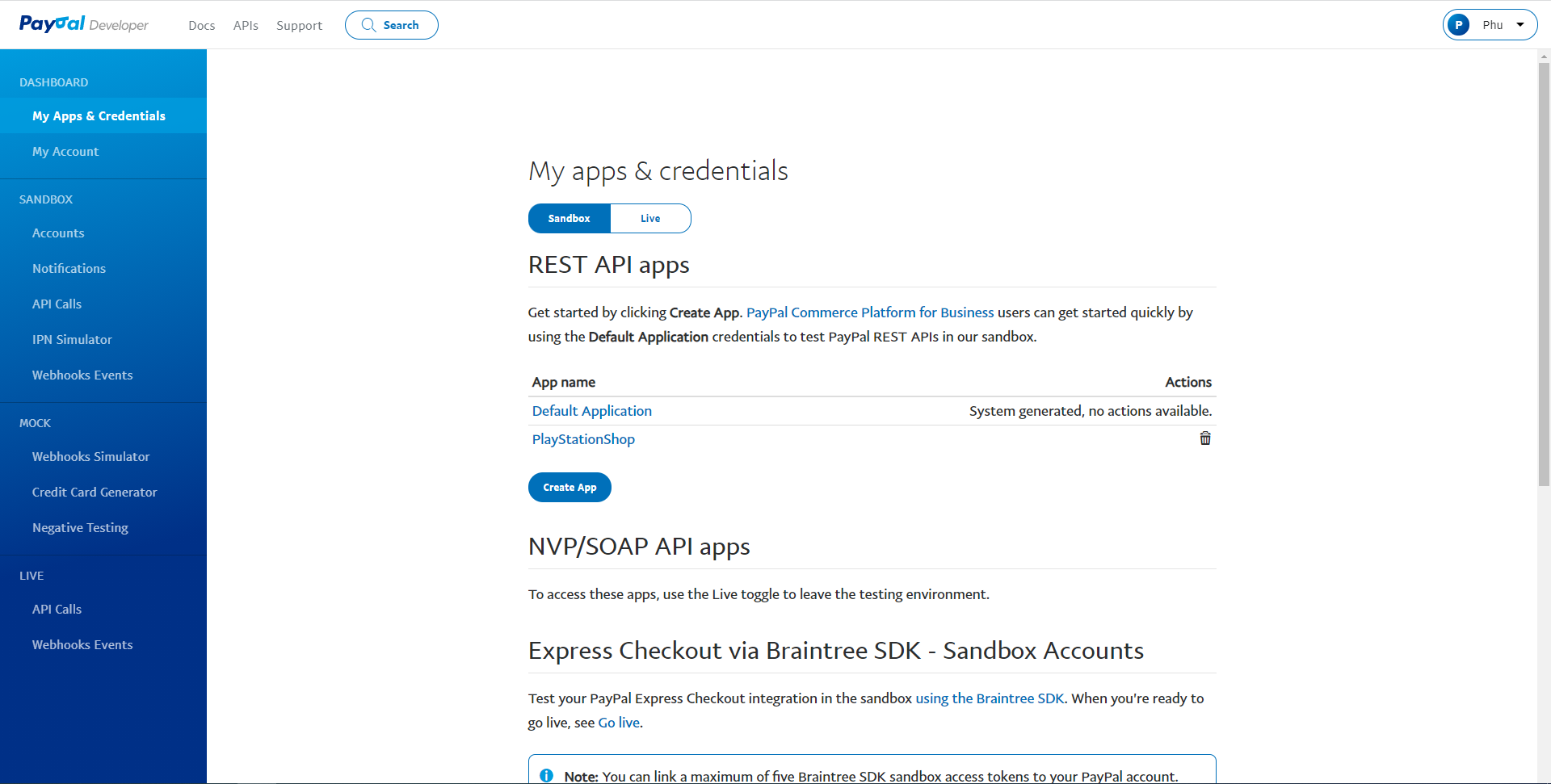
- Sandbox là một môi trường thử nghiệm khép kín, được xây dựng để hoạt động tương tự môi trường sử dụng ngoài thực tế. Tại đây, quí vị có thể bắt đầu xây dựng và theo dõi quá trình hoạt động của ứng dụng mà không ảnh hưởng tới bất kì tài khoản PayPal thực tế.

- Môi trường này sẽ hỗ trợ cho người phát triển phần mềm tiếp cận, làm thử và testing ứng dụng mà không làm ảnh hưởng đến bất kỳ đối tượng nào trong thực tế. Để từ đó, họ đánh giá, so sánh và quyết định lựa chọn dịch vụ mà họ nên tích hợp khi triển khai thực ứng dụng thực sự

**3.8.3.2 Xử lý thanh toán PayPal**

- Đầu tiên, chúng ta phải tạo một tài khoản tại <https://developer.paypal.com/>

- Sau khi tạo thành công được tài khoản, chúng ta đăng nhập vào hệ thống PayPal, click chọn Dashboard. Chúng ta sẽ nhìn thấy giao diện dashboard như hình bên dưới

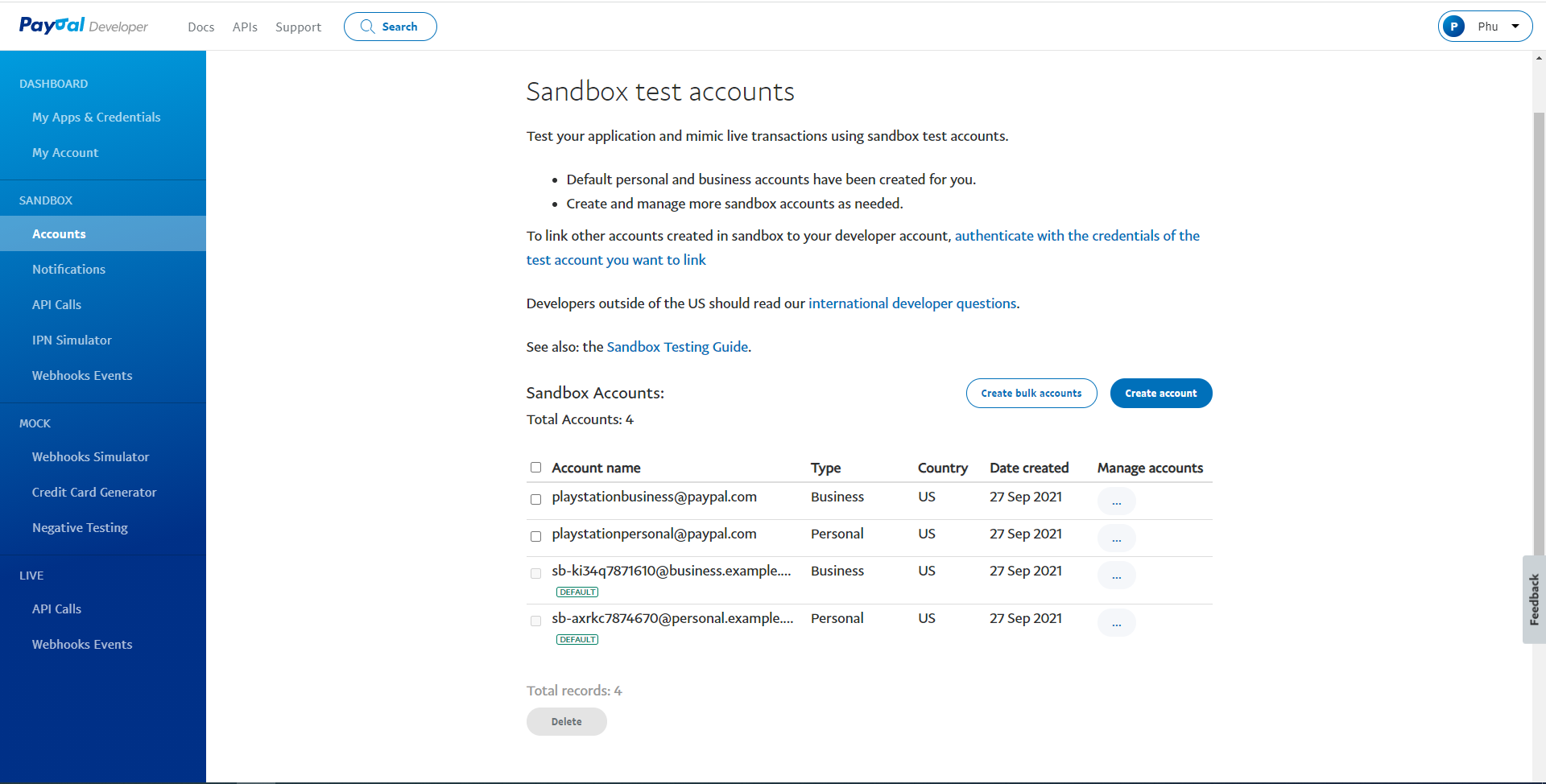


**Hình 3.6.** **Giao diện Dashboard PayPal SandBox**

- Tạo các tài khoản để sử dụng cho việc thực hiện các giao dịch trực tuyến thông qua cổng thanh toán Paypal, bao gồm 2 tài khoản: người mua và người bán.

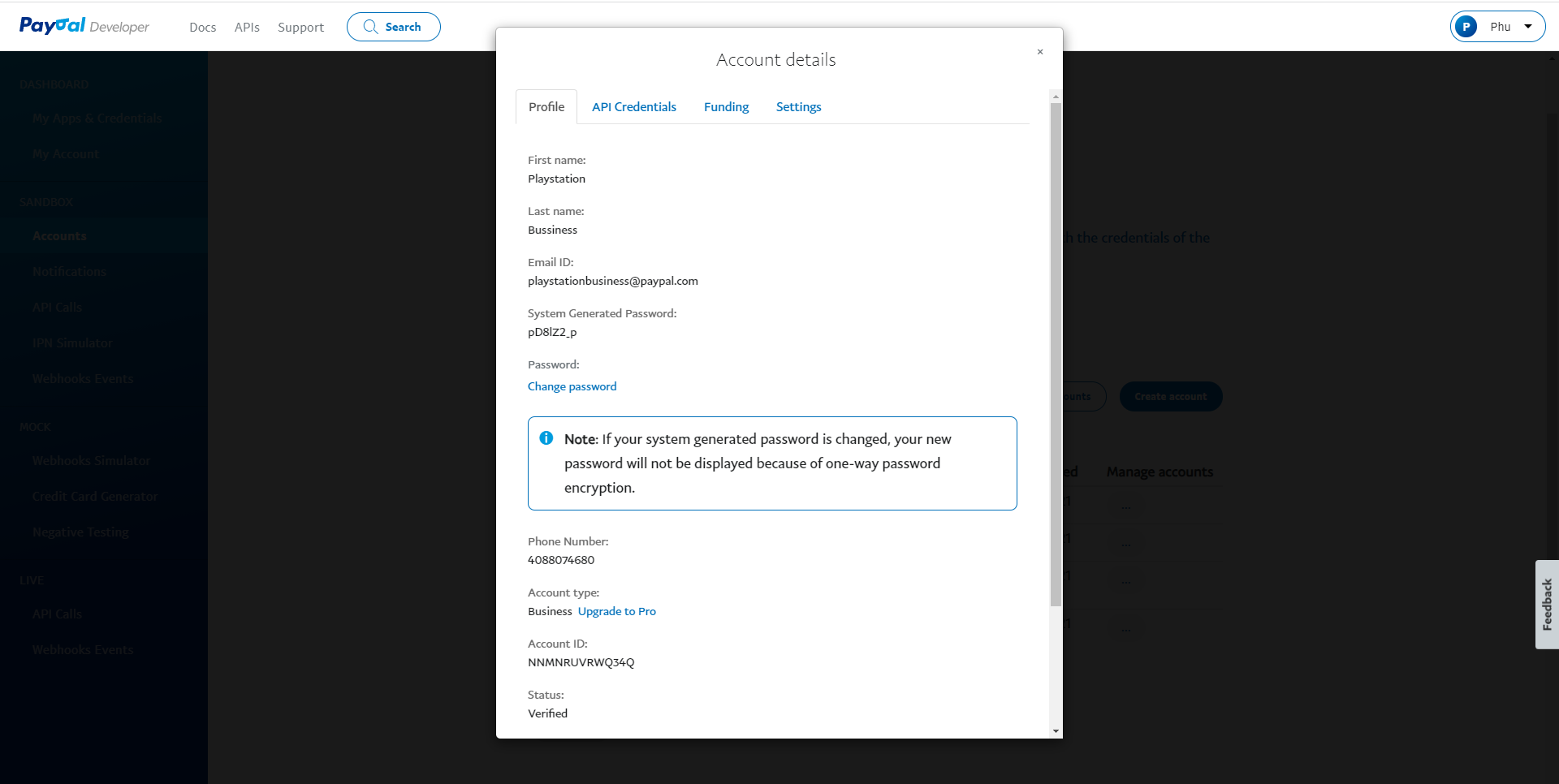
+ Ban đầu PayPal sẽ giả lập sẵn cho chúng ta 2 tài khoản (1 tài khoản mua và 1 tài khoản bán). Ta tạo thêm 2 tài khoản nữa

+ Để tạo tài khoản mới, chúng ta click vào Account trong mục Sandbox, sau đó click vào Create Account. Chọn loại tài khoản sau đó click Create. Cần tạo cả 2 loại tài khoản: Personal (người mua) và Business (người bán). Sau khi tạo tài khoản thành công, bạn sẽ nhìn thấy các tài khoản vừa tạo được liệt kê trong bảng dưới đây:



**Hình 3.7. Danh sách tài khoản PayPal**

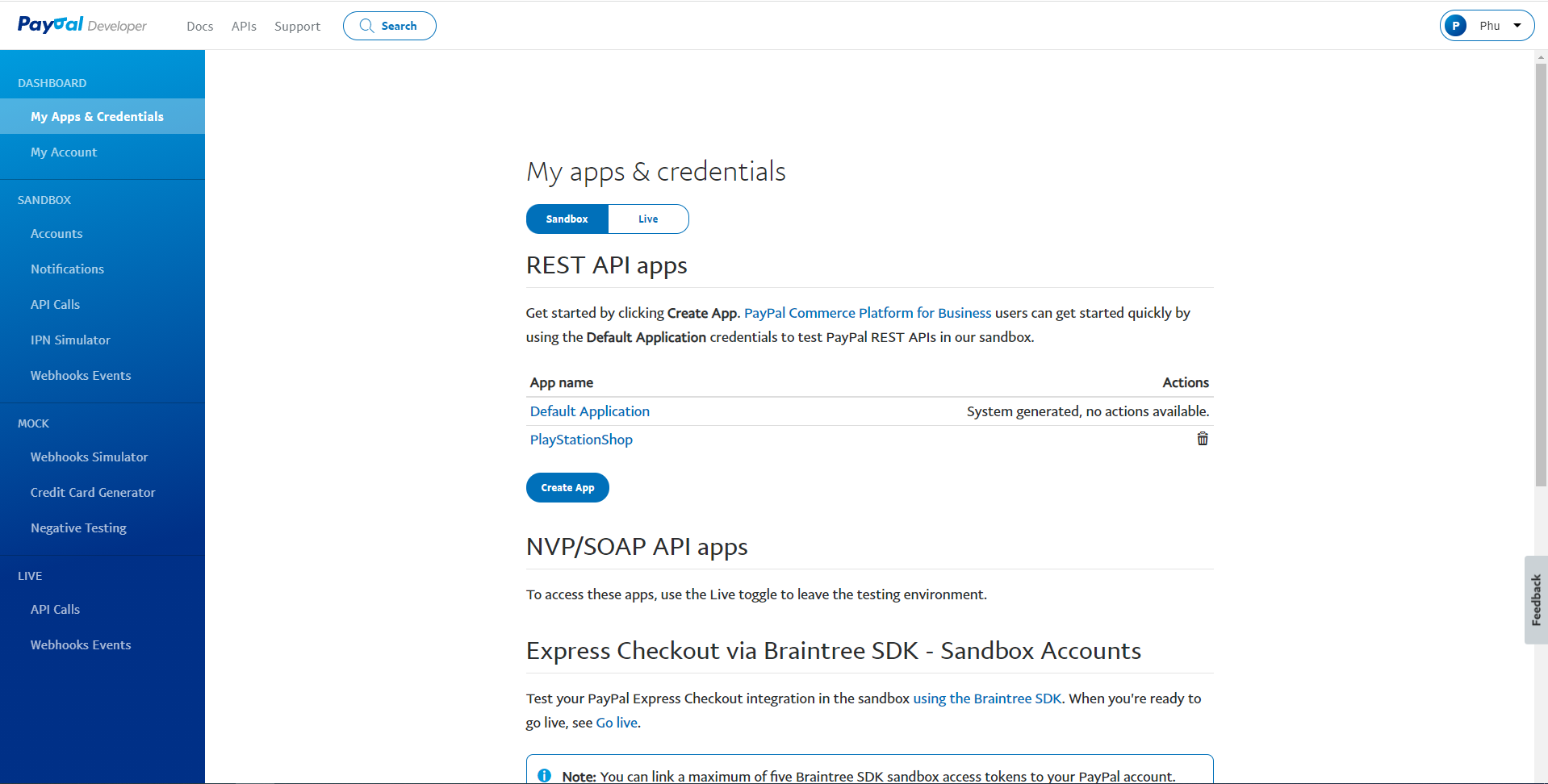
+ Vào Edit/view Account để chỉnh sửa thông tin tài khoản.



**Hình 3.8. Thông tin chi tiết tài khoản PayPal**

- Tạo ứng dụng trên Dashboard

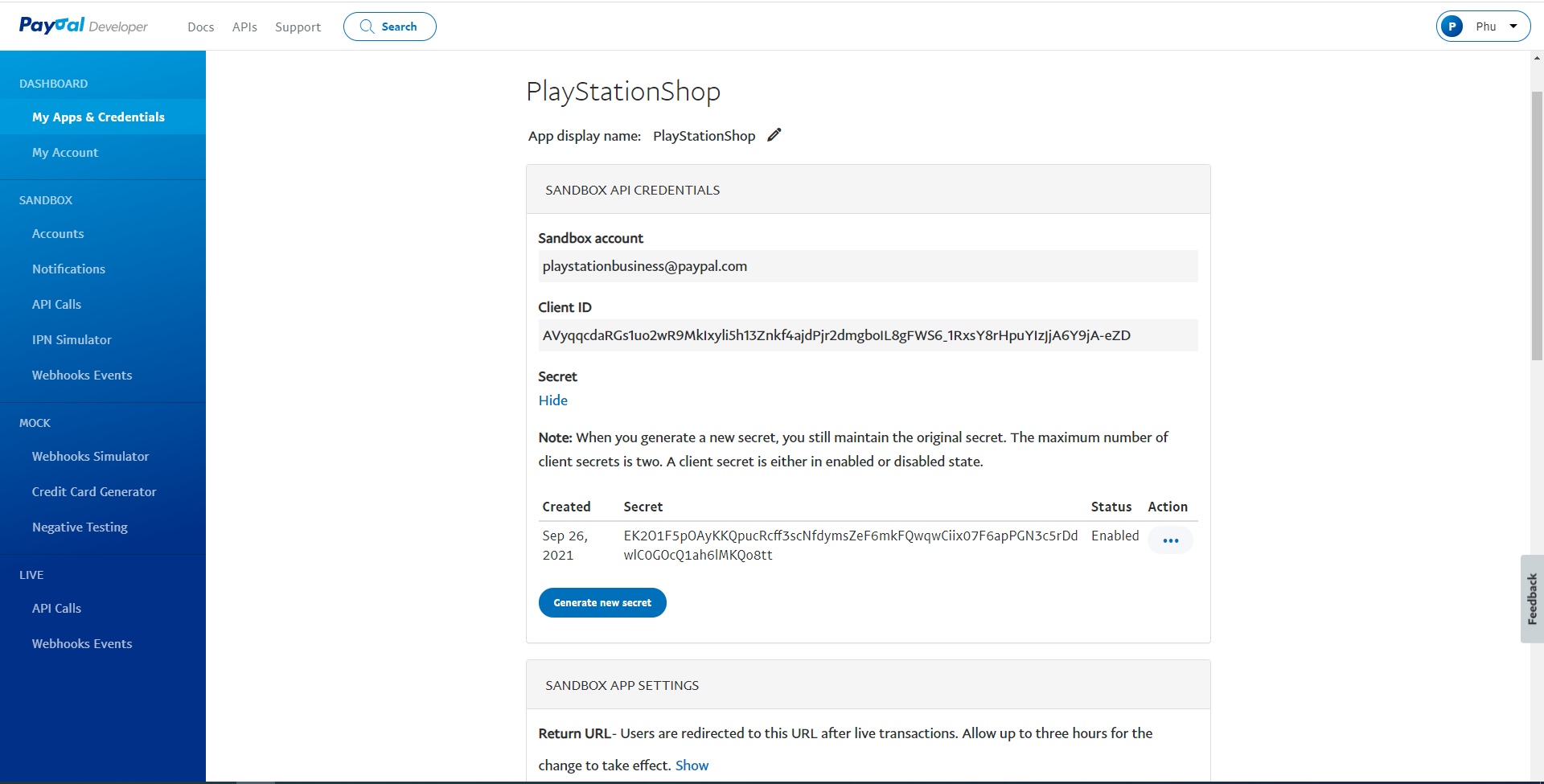
+ Để tạo ứng dụng mới, chúng ta click vào My App & Credentials trên tab dashboard bên trái, sau đó click vào Create App như hình bên dưới



**Hình 3.9. Ứng dụng thực hiện thanh toán trên PayPal**

+ Nhập tên ứng dụng cần tạo và chọn tài khoản để làm vai trò người bán cho ứng dụng này (chọn tài khoản loại Business bạn vừa tạo bên trên).

+ Sau khi tạo ứng dụng thành công, chúng ta click vào tên ứng dụng bạn vừa tạo để lấy Client ID và Secret để thiết lập trong ứng dụng:



**Hình 3.10.** **Thông tin của ứng dụng PayPal**

- Thực hiện import paypal sdk vào Java Spring Boot và tiến hành xây dựng các lớp xử lý

(Trích dẫn code minh họa)

@Controller

public class PaymentController {

public static final String URL\_PAYPAL\_SUCCESS = "pay/success";

public static final String URL\_PAYPAL\_CANCEL = "pay/cancel";

private Logger log = LoggerFactory.getLogger(getClass());

@Autowired

private PaypalService paypalService;

@GetMapping("/")

public String index(){

return "index";

}

@PostMapping("/pay")

public String pay(HttpServletRequest request,@RequestParam("price") double price ){

String cancelUrl = Utils.getBaseURL(request) + "/" + URL\_PAYPAL\_CANCEL;

String successUrl = Utils.getBaseURL(request) + "/" + URL\_PAYPAL\_SUCCESS;

try {

Payment payment = paypalService.createPayment(

price,

"USD",

PaypalPaymentMethod.paypal,

PaypalPaymentIntent.sale,

"payment description",

cancelUrl,

successUrl);

for(Links links : payment.getLinks()){

if(links.getRel().equals("approval\_url")){

return links.getHref();

}

}

} catch (PayPalRESTException e) {

log.error(e.getMessage());

}

return "redirect:/";

}

@GetMapping(URL\_PAYPAL\_CANCEL)

public String cancelPay(){

return "cancel";

}

@GetMapping(URL\_PAYPAL\_SUCCESS)

public String successPay(@RequestParam("paymentId") String paymentId, @RequestParam("PayerID") String payerId){

try {

Payment payment = paypalService.executePayment(paymentId, payerId);

if(payment.getState().equals("approved")){

return "success";

}

} catch (PayPalRESTException e) {

log.error(e.getMessage());

}

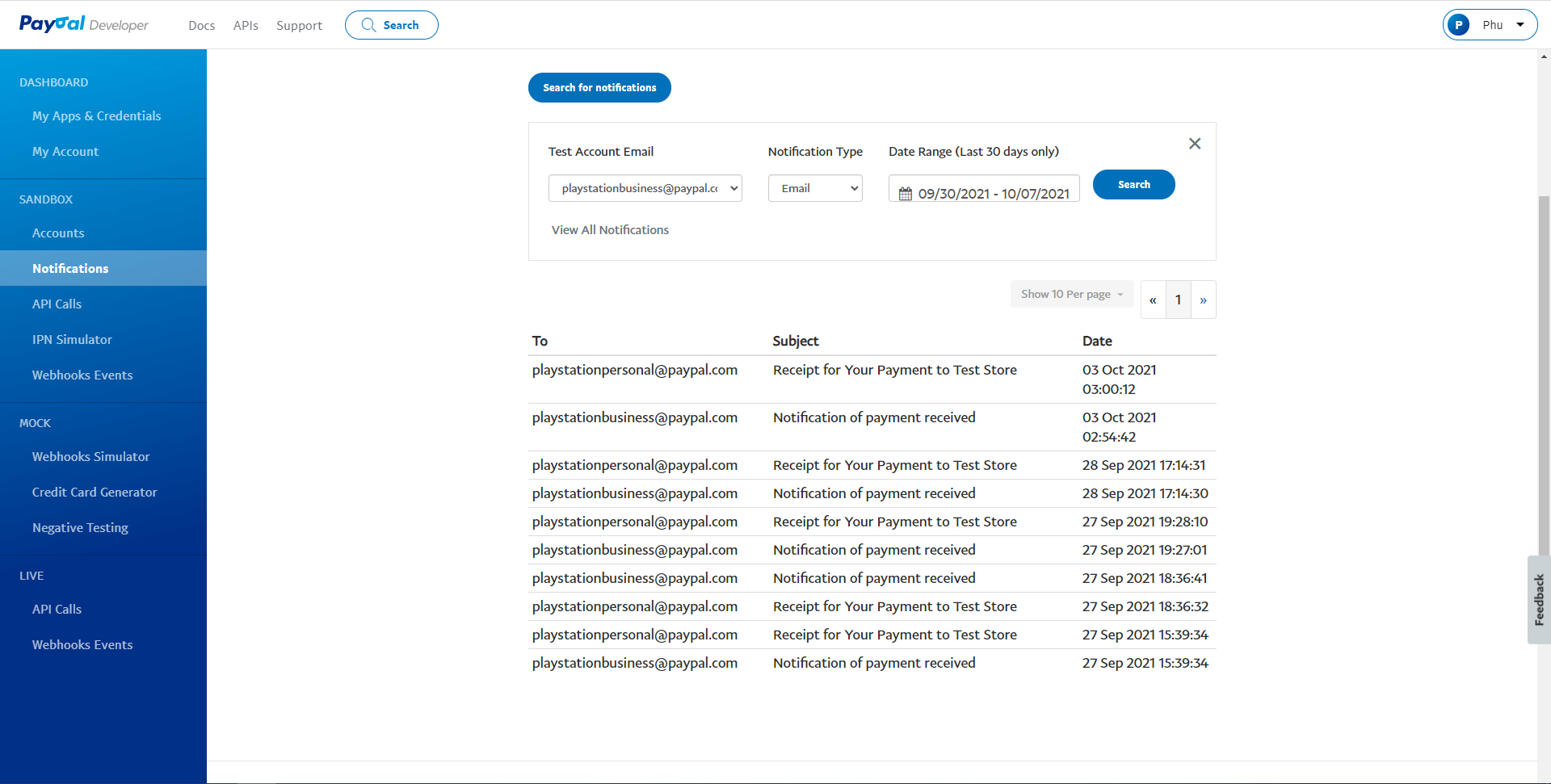
return "redirect:/";

}

}

**3.8.3.3 Hoạt động thanh toán trên PayPal**

- Thanh toán thành công thì chúng ta có thể xem chi tiết mỗi lượt thanh toán được lưu lại trong **Notifications** của mục **Sandbox**:

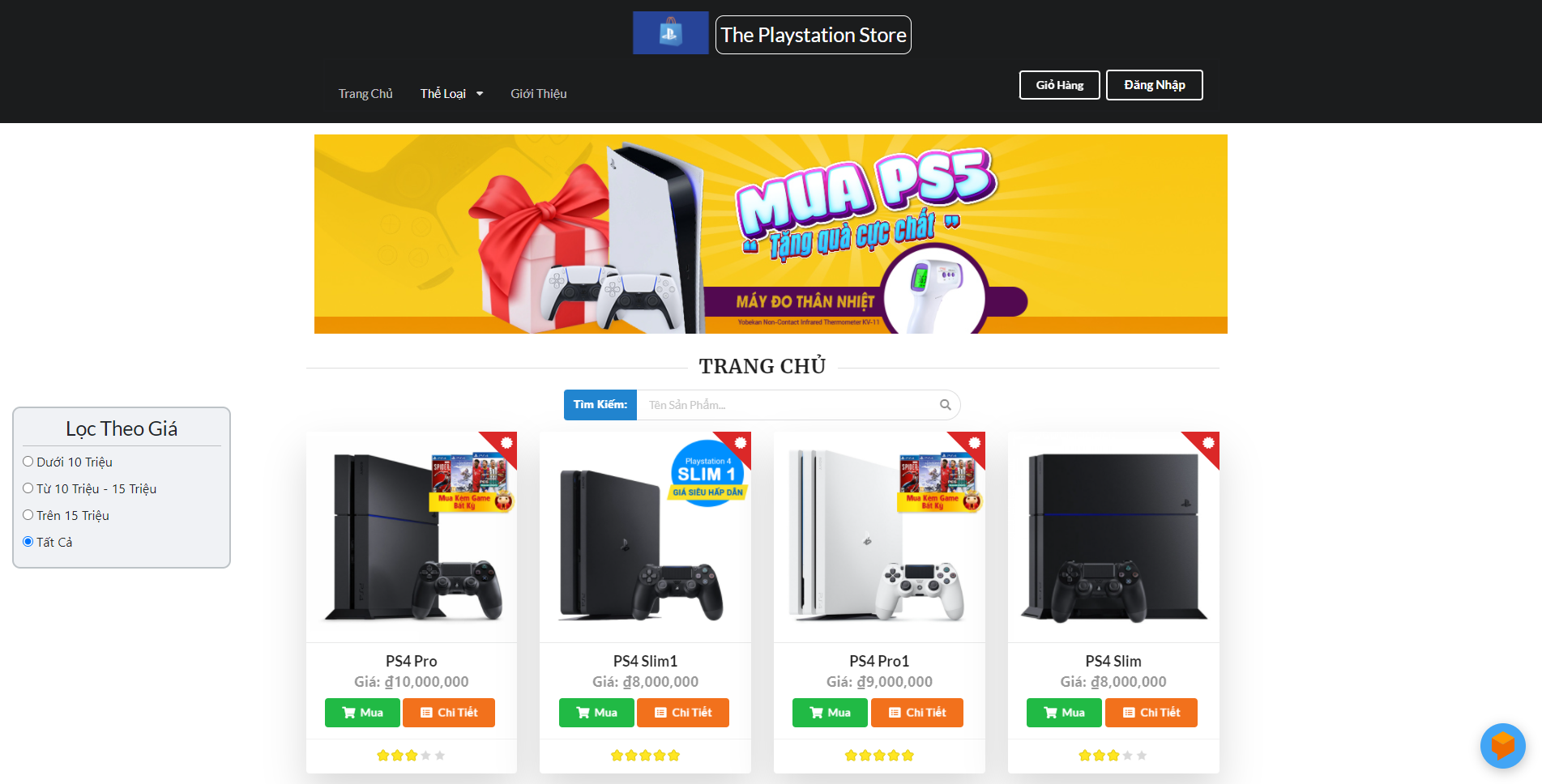


**Hình 3.11. Thông báo thanh toán PayPal**

# CHƯƠNG 4: ĐÁNH GIÁ KẾT QUẢ VÀ HƯỚNG PHÁT TRIỂN CỦA ĐỀ TÀI

## 4.1 Kết quả

**- HomePage (Trang chủ)**



**Hình 4.1. Giao diện trang chủ**