MỤC LỤC

[MỤC LỤC 4](#_Toc102747834)

[DANH SÁCH CÁC THUẬT NGỮ 6](#_Toc102747835)

[DANH SÁCH BẢNG BIỂU 7](#_Toc102747836)

[DANH SÁCH HÌNH VẼ 8](#_Toc102747837)

[CHƯƠNG 1: MỞ ĐẦU 9](#_Toc102747838)

[1.1 Lý do chọn đồ án 9](#_Toc102747839)

[1.2 Mục tiêu của đồ án 9](#_Toc102747840)

[1.3 Giới hạn và phạm vi của đồ án 10](#_Toc102747841)

[1.4 Nội dung thực hiện 10](#_Toc102747842)

[1.5 Phương pháp tiếp cận 10](#_Toc102747843)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 11](#_Toc102747844)

[2.1 Quy trình phát triển phần mềm 11](#_Toc102747845)

[2.2 Phương pháp phân tích thiết kế hướng đối tượng 14](#_Toc102747846)

[2.3 Thư viện hỗ trợ kết nối và trao đổi dữ liệu 19](#_Toc102747847)

[2.4 Công nghệ áp dụng 23](#_Toc102747848)

[CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 24](#_Toc102747849)

[3.1 Xác định yêu cầu bài toán 24](#_Toc102747850)

[3.2 Đặc tả yêu cầu ứng dụng 26](#_Toc102747851)

[3.3 Thiết kế hệ thống 31](#_Toc102747852)

[CHƯƠNG 4: TRIỂN KHAI WEBSITE 46](#_Toc102747853)

[4.1 Triển khai các chức năng cho ứng dụng 46](#_Toc102747854)

[4.2 Kiểm thử và triển khai ứng dụng 56](#_Toc102747855)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 58](#_Toc102747856)

[Kết quả đạt được 58](#_Toc102747857)

[Hạn chế của đề tài 58](#_Toc102747858)

[Hướng phát triển của đề tài 58](#_Toc102747859)

[TÀI LIỆU THAM KHẢO 59](#_Toc102747860)

DANH SÁCH CÁC THUẬT NGỮ

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Từ viết tắt | Cụm từ tiếng anh | Diễn giải |
| 1 | IT | Information Technology | Công nghệ Thông tin |
| 2 | DB | DataBase | Cơ sở dữ liệu |
| 3 | SQL | Structured Query Language | Ngôn ngữ truy vấn cấu trúc |

DANH SÁCH BẢNG BIỂU

[Bảng 1 1: Yêu cầu khách hàng 25](#_Toc102751593)

[Bảng 1 2: Yêu cầu chức năng 27](#_Toc102751594)

[Bảng 1 3 Yêu cầu phi chức năng 27](#_Toc102751595)

[Bảng 2. 1 Bảng tài khoản 33](#_Toc102751713)

[Bảng 2. 2 Bảng đơn hàng 33](#_Toc102751714)

[Bảng 2. 3 Bảng giỏ hàng 33](#_Toc102751715)

[Bảng 2. 4 Bảng chi tiết hóa đơn 34](#_Toc102751716)

[Bảng 2. 5 Bảng sản phẩm 34](#_Toc102751717)

[Bảng 2. 6 Bảng chi tiết sản phẩm 34](#_Toc102751718)

[Bảng 2. 7 Bảng danh mục 35](#_Toc102751719)

DANH SÁCH HÌNH VẼ

[Hình 2. 2: Mô hình thác nước 13](#_Toc102748803)

[Hình 2. 3 : Mô hình chữ V 14](#_Toc102748804)

[Hình 2. 4 : Mô hình xoắn ốc 15](#_Toc102748805)

[Hình 2. 5: Các View trong OOAD sử dụng UML 17](#_Toc102748806)

[Hình 2. 6: Các bản vẽ trong OOAD sử dụng UML 18](#_Toc102748807)

[Hình 2. 7: Ký hiệu về Use Case 19](#_Toc102748808)

[Hình 2. 8: Ký hiệu về Class 19](#_Toc102748809)

[Hình 2. 9: Ký hiệu về Actor 19](#_Toc102748810)

[Hình 3. 1: Biểu đồ Use Case chức năng QL nhân viên 27](file:///E:\Báo%20cáo%20đồ%20án%204\11_10119639_Lương%20Trung%20Đức_Đồ%20Án%204.docx#_Toc102749394)

[Hình 3. 2: Biểu đồ Use Case chức năng QL Thông tin cá nhân 28](#_Toc102749395)

[Hình 3. 3 Biểu đồ Use Case chức năng Đặt hàng 28](#_Toc102749396)

[Hình 3. 4 Biểu đồ Use Case chức năng QL Tài khoản 29](#_Toc102749397)

[Hình 3. 5 Biểu đồ Use Case chức năng QL Sản Phẩm 29](#_Toc102749398)

[Hình 3. 6 Biểu đồ Use Case chức năng QL Đơn Hàng 30](#_Toc102749399)

[Hình 3. 7 Biểu đồ lớp thực thể 31](#_Toc102749400)

[Hình 3. 8 : Mô hình cơ sở dữ liệu quan hệ 32](#_Toc102749401)

[Hình 3. 9 : Biểu đồ lớp VOPC QL tài khoản 35](#_Toc102749402)

[Hình 3. 10: Biểu đồ lớp VOPC QL sản phẩm 36](#_Toc102749403)

[Hình 3. 11 Biểu đồ lớp VOPC QL đơn hàng 36](#_Toc102749404)

[Hình 3. 12 : Biểu đồ tuần tự đăng nhập 37](#_Toc102749405)

[Hình 3. 13: Biểu đồ tuần tự giỏ hàng 38](#_Toc102749406)

[Hình 3. 14 : Biểu đồ tuần tự giỏ hàng 39](#_Toc102749407)

[Hình 3. 15 : Biểu đồ chi tiết QL sản phẩm 39](#_Toc102749408)

[Hình 3. 16: Biểu đồ chi tiết QL đơn hàng 40](#_Toc102749409)

# MỞ ĐẦU

* 1. Lý do chọn đồ án

Ngày nay, nhu cầu vui chơi giải trí ngày càng phát triển. Trong đó nhu cầu ăn uống cũng phát triển không kém, đặc biệt là giới trẻ. Trong điều kiện kinh tế thị trường cạnh tranh đòi hỏi các nhà quản lý phải có thông tin chính xác nắm bắt thông tin kịp thời về sở thích, mong muốn của người dùng….Từ đố đưa ra các kế hoạch và chiến lược kinh doanh hợp lý để giảm chi phí, nâng cao hiệu quả kinh doanh và có cơ hội phát triển hơn.

Để làm được việc đó, việc ứng dụng tin học hóa vào vấn đề quản lý là rất cần thiết. Trước đây chưa có phần mềm quản lý đều gặp nhiều khó khăn đòi hỏi phải có một lượng nhân lực lớn mà hiệu quả không cao. Giờ đây nhờ các ứng dụng quản lý, tin học hóa được thao tác thủ công mà làm việc ở các cửa hàng trở nên thuận lợi.

Chính vì những lý do trên nên em quyết định chọn đề tài để nghiên cứu là: “Xây dựng ứng dụng bán trà sữa”.

1.2 Mục tiêu của đồ án

1.2.1 Mục tiêu tổng quát

Xây dựng ứng dụng bán trà sữa

1.2.2 Mục tiêu cụ thể

Xây dựng ứng dụng bán sách gồm các chức năng:

* Đăng nhập, đăng ký tài khoản
* Xem các loại danh mục trà sữa của cửa hàng
* Đặt mua hàng
* Tìm kiếm và thanh toán
* Xem được trạng thái của đơn hàng

1.3 Giới hạn và phạm vi của đồ án

1.3.1 Đối tượng nghiên cứu

* Đối tượng nghiên cứu: Mọi người, chú trọng đến các bạn học sinh, sinh viên

1.3.2 Phạm vi nghiên cứu

* Phạm vi không gian: Cửa hàng trà sữa tại Hà Nội
* Phạm vi thời gian: Từ tháng 2 đến tháng 5

1.4 Nội dung thực hiện

* Bước 1: Tìm hiểu và nghiên cứu các ứng dụng bán trà sữa và các ứng dụng liên quan
* Bước 2: Phỏng vấn các bạn học sinh, sinh viên tại Hà Nội
* Bước 3: Thiết kế cơ sở dữ liệu, giao diện, các đối tượng của chương trình
* Bước 4: Lập trình ứng dụng
* Bước 5: Kiểm thử ứng dụng

1.5 Phương pháp tiếp cận

* Khảo sát yêu cầu của người dùng
* Tìm hiểu các thông tin liên quan, nhu cầu người dùng

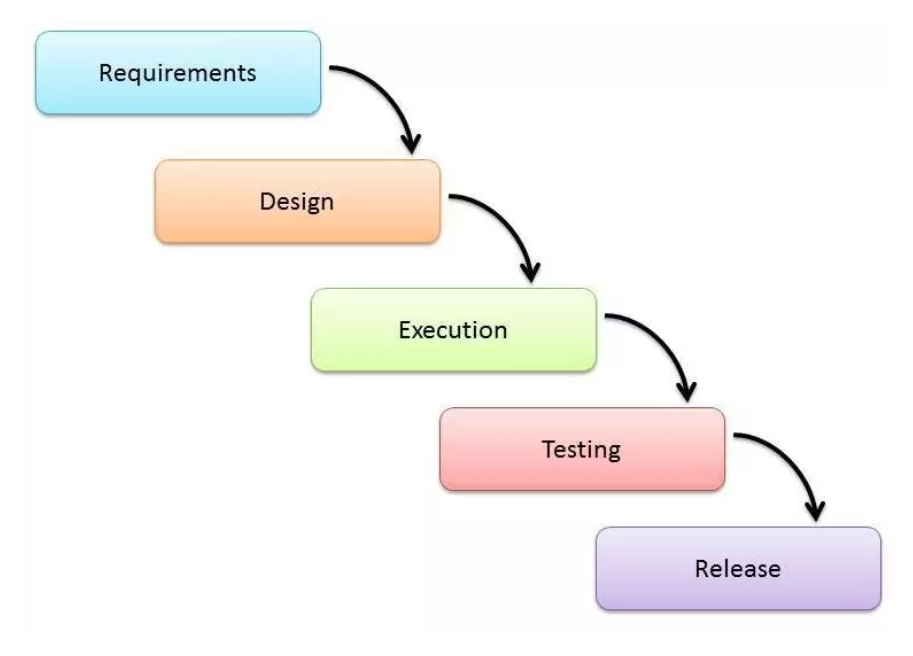
# CƠ SỞ LÝ THUYẾT

* 1. Quy trình phát triển phần mềm
     1. Giới thiệu Quy trình phát triển phần mềm

Quy trình phát triển phần mềm là một cấu trúc bao gồm tập hợp các thao tác và kết quả tương quan sử dụng trong việc phát triển để sản xuất ra một phần mềm

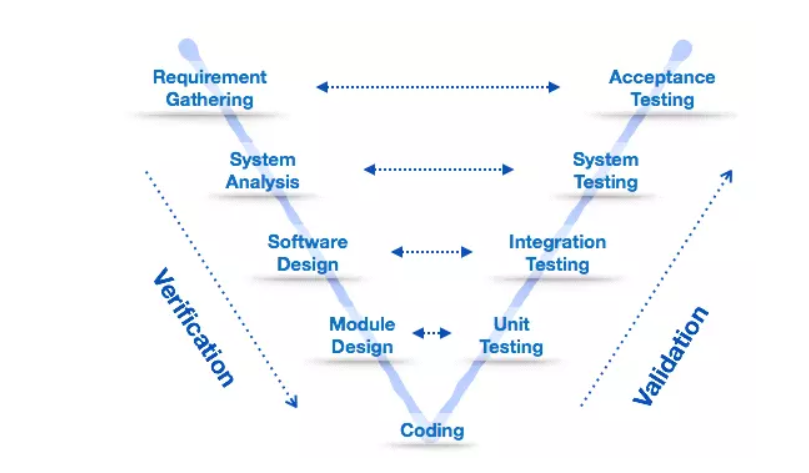
Chúng ta có thể chia quy trình phát triển phền mềm ra làm 4 giai đoạn:

* Đặc tả phần mềm: Là tiến trình để hiểu và xác định những dịch vụ nào cần có trong hệ thống, nhưng ràng buộc đối với việc phát triển và hức năng của hệ thống. Tiến trình này sẽ sinh ra các tài liệu yêu cầu.
* Thiết kế và thực thi phần mềm: liên quan đến việc chuyển những yêu cầu phần mềm thành hệ thống có thể thực thi được
* Thiết kề phần mềm: Là việc mô tả các trúc phần mềm, dữ liệu của hệ thống, giao diện giao tiếp giữa các thành phần, thuật toán được sử dụng.
* Thực thi phần mềm: Các lập trình viên dùng các ngôn ngữ lập trình để viết lệnh (source) thực thi để tạo ra hệ thống dựa trên các bản đặc tả thiết kế chi tiết, đồng thời tiến hành các thử nghiệm (test case) với các dữ liệu giả định.
* Kiểm thử phần mềm: Là quá trình vận hành chương trình để tìm ra lỗi
* Cài đặt và bảo trì phần mềm:
* Cài đặt và triển khai hệ thống vừa phát triển để người dùng có thể sử dụng được
* Bảo trì phần mềm: Điều chỉnh các lỗi chưa được phát triển trong các giai đoạn trước, nâng cao tính năng sử dụng và an toàn vận hành của phần mềm, đảm bảo cho việc phần mềm được cập nhật khi môi trường và yêu cầu của người sử dụng thay đổi
  + 1. Một số mô hình quy trình phát triển phần mềm
* Một số mô hình quy trình phát triển phần mềm:



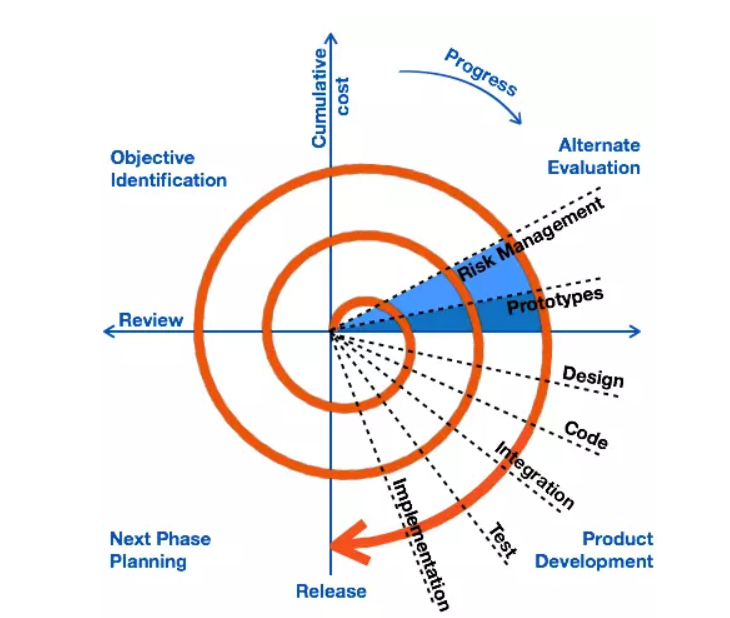
Hình 2. : Mô hình thác nước

* Mô hình thác nước: là một mô hình quy trình phát triển phần mềm, trong đó quy trình phát triển trông giống như một dòng chảy, với các pha được thực hiện theo một cách nghiêm ngặt và không có sự quay lui hay nhảy vượt pha là: Phân tích yêu cầu, thiết kế, triển khai thực hiện, kiểm thử, liên kết và bảo trì.



Hình 2. : Mô hình chữ V

* Mô hình V hiện nay là một trong những mô hình quy trình phát triển phần mềm được sử dụng rộng rãi nhất. Trong mô hình V việc thực hiện kiểm tra được diễn ra ngay từ giai đoạn lấy yêu cầu. V mô hình cũng được gọi là mô hình xác minh(verification) và mô hình xác nhận(validation).



Hình 2. : Mô hình xoắn ốc

* Mô hình xoắn ốc là quy trình phát triển định hướng rủi ro cho các dự án phần mềm. Kết hợp thế mạnh của các mô hình khác và giải quyết khó khăn của các mô hình trước còn tồn tại. Dựa trên các mô hình rủi ro riêng biệt của mỗi dự án, mô hình xoắn ốc đưa ra cách áp dụng các yếu tố của một hoặc nhiều mô hình xử lý, chẳng hạn như mô hình gia tốc, mô hình thác nước hoặc mô hình tạo mẫu tiến hóa.
  1. Phương pháp phân tích thiết kế hướng đối tượng
     1. Giới thiệu ngôn ngữ UML

**Khái niệm về UML (Unified Modeling Language)**

UML là ngôn ngữ mô hình hóa hợp nhất dùng để biểu diễn hệ thống. Nói một cách đơn giản là nó dùng để tạo ra các bản vẽ nhằm mô tả thiết kế hệ thống. Các bản vẽ này được sử dụng để các nhóm thiết kế trao đổi với nhau cũng như dùng để thi công hệ thống (phát triển), thuyết phục khách hàng, các nhà đầu tư v.v.. (Giống như trong xây dựng người ta dùng các bản vẽ thiết kế để hướng dẫn và kiểm soát thi công, bán hàng  căn hộ v.v..)

**Tại sao lại là OOAD và UML?**

OOAD cần các bản vẽ để mô tả hệ thống được thiết kế, còn UML là ngôn ngữ mô tả các bản vẽ nên cần nội dung thể hiện.  Do vậy, chúng ta phân tích và thiết kế theo hướng đối tượng và sử dụng UML để biểu diễn các thiết kế đó nên chúng thường đi đôi với nhau.

**OOAD sử dụng UML**

UML sử dụng để vẽ cho nhiều lĩnh vực khác nhau như phần mềm, cơ khí, xây dựng v… trong phạm vi các bài viết này chúng ta chỉ nghiên cứu cách sử dụng UML cho phân tích và thiết kế hướng đối tượng trong ngành phần mềm. OOAD sử dụng UML bao gồm các thành phần sau:

–          View (góc nhìn)

–          Diagram (bản vẽ)

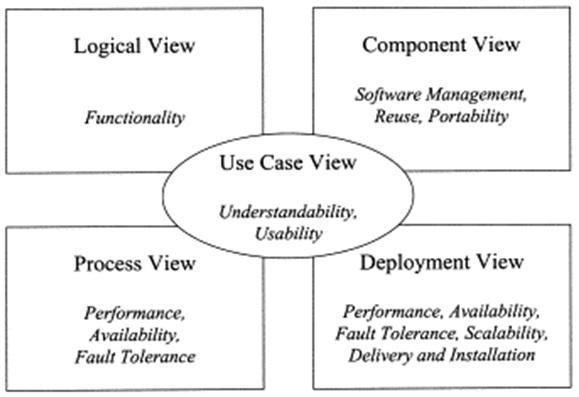
–          Notations (ký hiệu)

–          Mechanisms (qui tắc, cơ chế)

Chúng ta sẽ tìm hiểu kỹ hơn các thành phần trên.

***View (góc nhìn)***

Mỗi góc nhìn như thầy bói xem voi, nó không thể hiện hết hệ thống nhưng thể hiện rõ hệ thống ở một khía cạnh. Chính vì thế trong xây dựng có bản vẽ kiến trúc (nhìn về mặt kiến trúc), bản vẽ kết cấu (nhìn về mặt kết cấu), bản vẽ thi công (nhìn về mặt thi công). Trong phần mềm cũng như vậy, OOAD sử dụng UML có các góc nhìn sau:



Hình 2. : Các View trong OOAD sử dụng UML

Trong đó,

–          ***Use Case View***: cung cấp góc nhìn về các ca sử dụng giúp chúng ta hiểu hệ thống có gì? ai dùng và dùng nó như thế nào.

–          ***Logical View***: cung cấp góc nhìn về cấu trúc hệ thống, xem nó được tổ chức như thế nào. Bên trong nó có gì.

–          ***Process View:***cung cấp góc nhìn động về hệ thống, xem các thành phần trong hệ thống tương tác với nhau như thế nào.

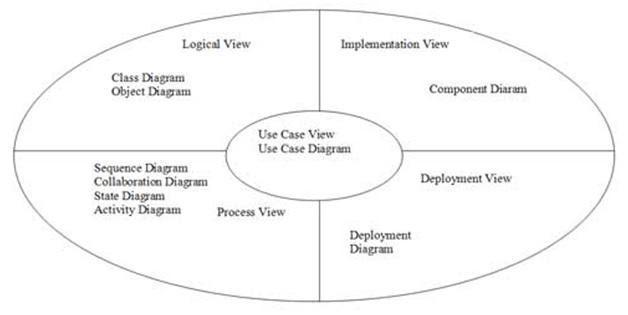
–          ***Component View:***Cũng là một góc nhìn về cấu trúc giúp chúng ta hiểu cách phân bổ và sử dụng lại các thành phần trong hệ thống ra sao.

–          ***Deployment View***: cung cấp góc nhìn về triển khai hệ thống, nó cũng ảnh hưởng lớn đến kiến trúc hệ thống.

Tập hợp các góc nhìn này sẽ giúp chúng ta hiểu rõ hệ thống cần phân tích, thiết kế. Trong hình 1 chúng ta thấy góc nhìn Use Case View nằm ở giữa và chi phối tất cả các góc nhìn còn lại. Chính vì thế chúng ta thường thấy các tài liệu nói về 4 view + 1 chứ không phải 5 view nhằm nhấn mạnh vai trò của Use Case View.

***Diagram (Bản vẽ)***

Diagram các bạn có thể dịch là sơ đồ. Tuy nhiên ở đây chúng ta sử dụng từ bản vẽ cho dễ hình dung. Các bản vẽ được dùng để thể hiện các góc nhìn của hệ thống.



Hình 2. : Các bản vẽ trong OOAD sử dụng UML

Trong đó,

–          **Use Case Diagram**: bản vẽ mô tả về ca sử dụng của hệ thống. Bản vẽ này sẽ giúp chúng ta biết được ai sử dụng hệ thống, hệ thống có những chức năng gì. Lập được bản vẽ này bạn sẽ hiểu được yêu cầu của hệ thống cần xây dựng.

–          **Class Diagram**:bản vẽ này mô tả cấu trúc của hệ thống, tức hệ thống được cấu tạo từ những thành phần nào. Nó mô tả khía cạnh tĩnh của hệ thống.

–          **Object Diagram**:Tương tự như Class Diagram nhưng nó mô tả đến đối tượng thay vì lớp (Class).

–          **Sequence Diagarm**: là bản vẽ mô tả sự tương tác của các đối tượng trong hệ thống với nhau được mô tả tuần tự các bước tương tác theo thời gian.

–          **Collaboration Diagram**:tương tự như sequence Diagram nhưng nhấn mạnh về sự tương tác thay vì tuần tự theo thời gian.

–          **State Diagram:**bản vẽ mô tả sự thay đổi trạng thái của một đối tượng. Nó được dùng để theo dõi các đối tượng có trạng thái thay đổi nhiều trong hệ thống.

–          **Activity Diagram:** bản vẽ mô tả các hoạt động của đối tượng, thường được sử dụng để hiểu về nghiệp vụ của hệ thống.

–          **Component Diagram:**bản vẽ mô tả về việc bố trí các thành phần của hệ thống cũng như việc sử dụng các thành phần đó.

–          **Deployment Diagram:** bản vẽ mô tả việc triển khai của hệ thống như việc kết nối, cài đặt, hiệu năng của hệ thống v.v…

Chúng ta sẽ bàn kỹ các bản vẽ này trong các bài tiếp theo. Vì nó chính là hạt nhân của loạt bài này.

**Lưu ý**: Ở đây chúng ta sử dụng từ hệ thống tương đương với sản phẩm phần mềm.

***Notations (các ký hiệu)***

Notations là các ký hiệu để vẽ, nó như từ vựng trong ngôn ngữ tự nhiên. Bạn phải biết từ vựng thì mới ghép thành câu, thành bài được. Chúng ta sẽ tìm hiểu kỹ các notations trong từng bản vẽ sau này. Dưới đây là vài ví dụ về notation.



Hình 2. : Ký hiệu về Use Case



Hình 2. : Ký hiệu về Class



Hình 2. : Ký hiệu về Actor

Và còn rất nhiều ký hiệu nữa.

***Mechanisms (Rules)***

Mechanisms là các qui tắc để lập nên bản vẽ, mỗi bản vẽ có qui tắc riêng và bạn phải nắm được để tạo nên các bản vẽ thiết kế đúng. Các qui tắc này chúng ta sẽ bàn kỹ trong các bài về các bản vẽ.

* + 1. Các bước phân tích thiết kế hướng đối tượng
  1. Thư viện hỗ trợ kết nối và trao đổi dữ liệu
     1. Một số thư viện hỗ trợ kết nối và trao đổi dữ liệu

*<*

(ZmikiSoft.com) – Ngôn ngữ lập trình server là những ngôn ngữ chạy phía server ( server-side). Đây là những ngôn ngữ được các lập trình viên sử dụng để viết các chương trình, tiện ích chạy trên các server.

Server là các máy chủ, với cấu hình máy cao ( Ram, CPU, SSD,…). Các máy server chạy 24/24 để đảm bảo cho các ứng dụng server-side chạy liên tục, trừ những lúc cần bảo trì, nâng cấp server. Các máy server cũng có những hệ điều hành riêng biệt cho nó, phổ biến nhất là hệ điều hành linux-server và windows – server.

Như vậy ngôn ngữ lập trình Server là các ngôn ngữ có thể viết được các ứng dụng chạy trên các hệ điều hành cài đặt trên các máy Server, cụ thể ở đây là Linux và Windows.



Windows-server: là hệ điều hành đặc thù do microsoft xây dựng, hệ điều hành này có thể chạy được hầu hết các ngôn ngữ, đặc biệt là các ngôn ngữ đặc thù trong  bộ .net của microsoft. Đây là hệ điều hành độc quyền của microsoft và phải mua mới sử dụng được. Windows mạnh về giao diện người dùng, dễ thao tác, được hỗ trợ mạnh mẽ từ microsoft.

Các ngôn ngữ trong bộ .net ( C#, VB.net,…): lập trình các ứng dụng chạy ngầm trên server, webservice, web application, lập trình socket, giao thức….

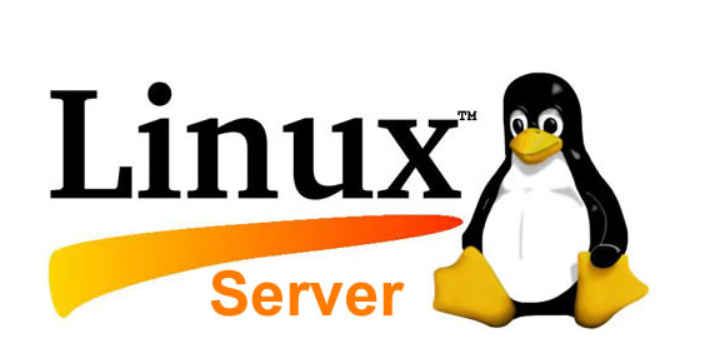
Java: lập trình các ứng dụng webservice, web application, cấu hình server, các ứng dụng chạy nền phía server, lập trình các giao thức….

PHP: ngôn ngữ lập trình các ứng dụng web, webservice và web application…

Ruby: ngôn ngữ lập trình webservice, web application…

Python: ngôn ngữ lập trình webservice, web application, cấu hình server…

C/C++: lập trình các giao thức server, các module tối ưu hệ thống, lập trình server…



Linux-server: Linux là hệ điều hành nguồn mở, ngày nay có rất nhiều hệ điều hành phát triển từ nhân Linux như Ubuntu, debian, Suse, centos …. rất nhiều, đây là các hệ điều hành gần như miễn phí cho người dùng cài đặt. Linux không nặng về giao diện nên việc thực thi các tác vụ nhanh hơn, nhưng thay vào đó việc sử dụng lại khá phức tạp, thường chỉ dành cho các nhà chuyên môn, lập trình viên. Việc mua, thuê các server linux cũng rẻ hơn so với Windows. Linux chạy được hầu hết các ngôn ngữ chạy trên Windows (Java/PHP/Ruby/Python/C/C++…) trừ bộ ngôn ngữ đặc thù trong bộ .net của microsoft.

* + 1. Giới thiệu
* **Web API là gì?**

**Web API** là một phương thức dùng để cho phép các ứng dụng khác nhau có thể giao tiếp, trao đổi dữ liệu qua lại. Dữ liệu được Web API trả lại thường ở dạng [JSON](https://topdev.vn/blog/json-la-gi/) hoặc XML thông qua giao thức HTTP hoặc HTTPS.

Những điểm nổi bật của Web API

Web API hỗ trợ restful đầy đủ các phương thức: Get/Post/put/delete dữ liệu. Nó giúp bạn xây dựng các HTTP service một cách rất đơn giản và nhanh chóng. Nó cũng có khả năng hỗ trợ đầy đủ các thành phần HTTP: URI, request/response headers, caching, versioning, content format.

**Tự động hóa sản phẩm**

Với **web API**, chúng ta sẽ tự động hóa quản lý công việc, cập nhật luồng công việc, giúp tăng năng suất và tạo hiệu quả công việc cao hơn.

**Khả năng tích hợp linh động**

API cho phép lấy nội dung từ bất kỳ website hoặc ứng dụng nào một cách dễ dàng nếu được cho phép, tăng trải nghiệm người dùng. API hoạt động như một chiếc cổng, cho phép các công ty chia sẻ thông tin được chọn nhưng vẫn tránh được những yêu cầu không mong muốn.

**Cập nhật thông tin thời gian thực**

API có chức năng thay đổi và cập nhật thay đổi theo thời gian thực. Với công nghệ này, dữ liệu sẽ được truyền đi tốt hơn, thông tin chính xác hơn, dịch vụ cung cấp linh hoạt hơn.

**Có tiêu chuẩn chung dễ sử dụng**

Bất kỳ người dùng, công ty nào sử dụng cũng có thể điều chỉnh nội dung, dịch vụ mà họ sử dụng.

* + - * + Hỗ trợ đầy đủ các thành phần MVC như: routing, controller, action result, filter, model binder, IoC container, [dependency injection](https://topdev.vn/blog/dependency-injection-la-gi/), unit test.
    1. Ưu điểm
* Web API được sử dụng hầu hết trên các ứng dụng desktop, ứng dụng mobile và ứng dụng website.
* Linh hoạt với các định dạng dữ liệu khi trả về client: Json, XML hay định dạng khác.
* Nhanh chóng xây dựng HTTP service: URI, request/response headers, caching, versioning, content formats và có thể host trong ứng dụng hoặc trên IIS.
* Mã nguồn mở, hỗ trợ chức năng RESTful đầy đủ, sử dụng bởi bất kì client nào hỗ trợ XML, Json.
* Hỗ trợ đầy đủ các thành phần MVC như: routing, controller, action result, filter, model binder, IoC container, dependency injection, unit test.
* Giao tiếp hai chiều được xác nhận trong các giao dịch, đảm bảo độ tin cậy

cao.

* + 1. Nhược điểm
* Do web API còn khá mới nên chưa thể đánh giá nhiều về nhược điểm của mô hình nay. Tuy nhiên, có hai nhược điểm dễ dàng nhận thấy:
* Web API chưa hoàn toàn phải là RESTful service, mới chỉ hỗ trợ mặc định GET, POST
* Để sử dụng hiệu quả cần có kiến thức chuyên sâu, có kinh nghiệm backend tốt
* Tốn thời gian và chi phí cho việc phát triển, nâng cấp và vận hành
* Có thể gặp vấn đề về bảo mật khi hệ thống bị tấn công nếu không giới hạn điều kiện kỹ.
  1. Công nghệ áp dụng

**Web API hoạt động như thế nào?**

1. Đầu tiên là xây dựng URL API để bên thứ ba có thể gửi request dữ liệu đến máy chủ cung cấp nội dung, dịch vụ thông qua giao thức HTTP hoặc HTTPS.
2. Tại web server cung cấp nội dung, các ứng dụng nguồn sẽ thực hiện kiểm tra xác thực nếu có và tìm đến tài nguyên thích hợp để tạo nội dung trả về kết quả.
3. Server trả về kết quả theo định dạng JSON hoặc XML thông qua giao thức HTTP/HTTPS.
4. Tại nơi yêu cầu ban đầu là ứng dụng web hoặc ứng dụng di động , dữ liệu JSON/XML sẽ được parse để lấy data. Sau khi có được data thì thực hiện tiếp các hoạt động như lưu dữ liệu xuống Cơ sở dữ liệu, hiển thị dữ liệu…

# PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

* 1. Xác định yêu cầu bài toán
     1. Khảo sát bài toán

|  |  |
| --- | --- |
| **STT** | **Tên yêu cầu** |
| **1** | Đăng nhập vào hệ thống |
| **2** | Hiển thị thông tin loại sản phẩm |
| **3** | Hiển thị nội dung của ứng dụng như sản phẩm được ưa dùng |
| **4** | Hiển thị các topping theo kèm khi đặt hàng |
| **5** | Hiển thị thông tin kết quả tìm kiếm |
| **6** | Quản lý giỏ hàng khi khách hàng thêm sản phẩm vào giỏ hàng |
| **7** | Chức năng đặt hàng |
| **8** | Hiển thị trạng thái của đơn hàng |
| **9** | Quản lý dữ liệu khách hàng |

Bảng 1 : Yêu cầu khách hàng

* + 1. Phân tích yêu cầu bài toán

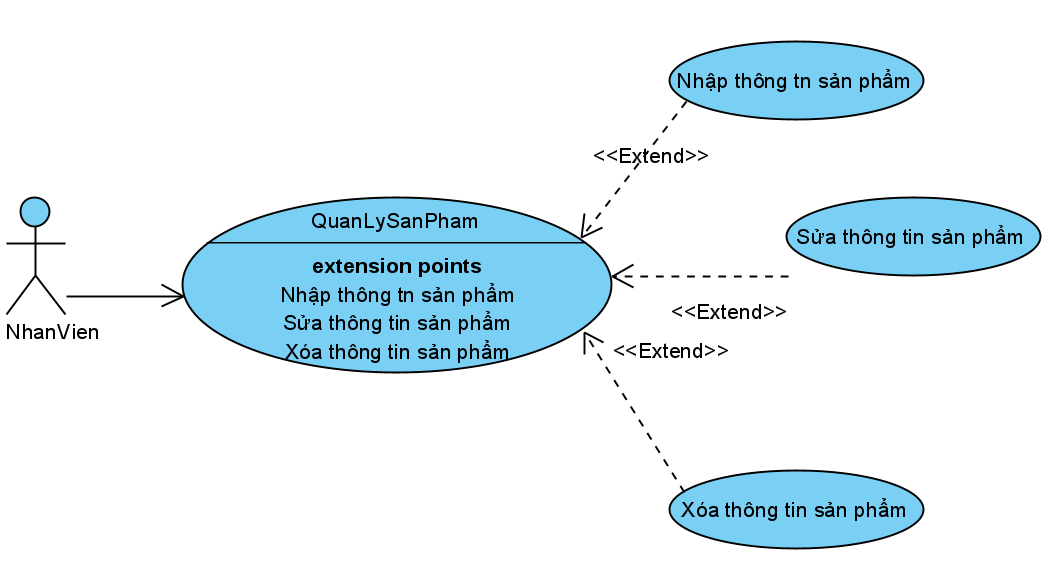
|  |  |  |
| --- | --- | --- |
| **STT** | **Tên yêu cầu** | **Mô tả yêu cầu** |
| **I** | **Các yêu cầu chức năng nghiệp vụ** |  |
| 1 | Chức năng đăng nhập | Để sử dụng toàn bộ các chức năng của hệ thống , học sinh có 1 tài khoản để đăng nhập vào hệ thống. Mỗi tài khoản được khách hàng đăng ký trước bào gồm tài khoản và mật khẩu. |
| 2 | Chức năng mua hàng | + Trang chủ: Hiển thị danh mục các sản phẩm của cửa hàng  + Trang sản phẩm: Khách hàng có thể mua thêm topping đi kèm theo sản phẩm. |
| 3 | Chức năng xem trạng thái của đơn hàng | Khách hàng của thể vào xem tình trạng đơn hàng đã đặt: Chờ xác nhận, Đã giao, Đã hủy. |
| 4 | Quản lý tài khoản | Khách hàng có thể xem lại thông tin cá nhân của mình: Email, mật khẩu |
| II | **Yêu cầu chức năng hệ thống** |  |
| 1 | Quản lý hệ thống | + Quản lý thông tin tài khoản  + Đổi mật khẩu |
| 2 | Quản lý sản phẩm | + Quản lý loại sản phẩm  + Quản lý sản phẩm |
| 3 | Quản lý đơn hàng | + Quản lý tình trạng đơn hàng |
| 4 | Quản lý tài khoản | + Quản lý người dùng |

Bảng 1 : Yêu cầu chức năng

|  |  |  |
| --- | --- | --- |
| II | **Yêu cầu phi chức năng** |  |
| 1 | Giao diện | Giao diện hệ thống phải dễ sử dụng, trực quan, thân thiện với mọi người dùng. |
| 2 | Tốc độ xử lý | Hệ thống phải xử lý nhanh chóng và chính xác. |
| 3 | Bảo mật dữ liệu | Tính bảo mật và độ an toàn cao. |

Bảng 1 Yêu cầu phi chức năng

* 1. Đặc tả yêu cầu ứng dụng
     1. Các yêu cầu chức năng của hệ thống
        1. Chức năng của phân hệ quản trị nội dung (nếu có)

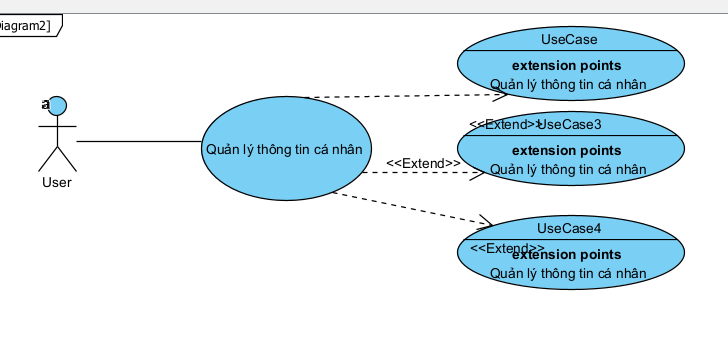


Hình .. Biểu đồ Use Case tổng quát

Hình 3. : Biểu đồ Use Case chức năng QL nhân viên

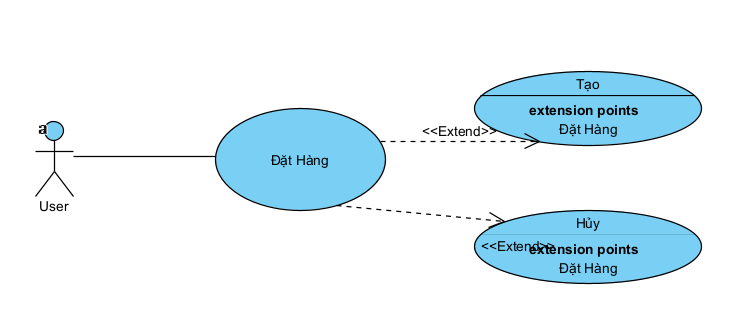
* + - 1. Chức năng của phân hệ người dùng (nếu có)

**1.** Biểu đồ Use Case chức năng QL Thông tin cá nhân



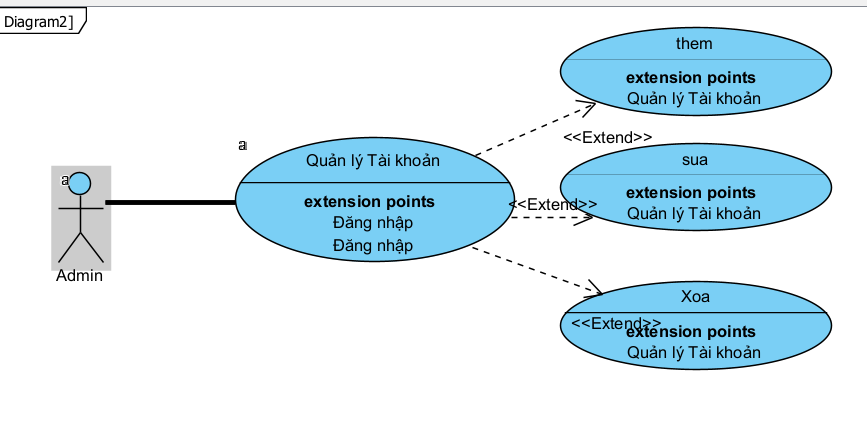
Hình 3. : Biểu đồ Use Case chức năng QL Thông tin cá nhân

**2.** Biểu đồ Use Case chức năng Đặt hàng



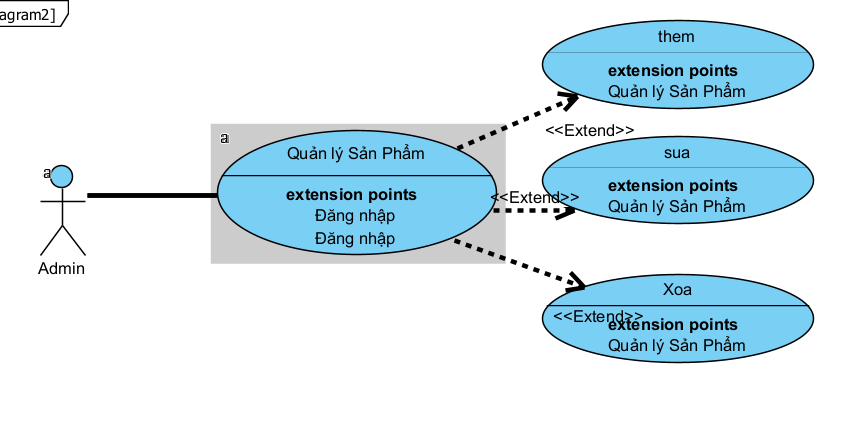
Hình 3. Biểu đồ Use Case chức năng Đặt hàng

**3.**Biểu đồ Use Case chức năng QL Tài khoản



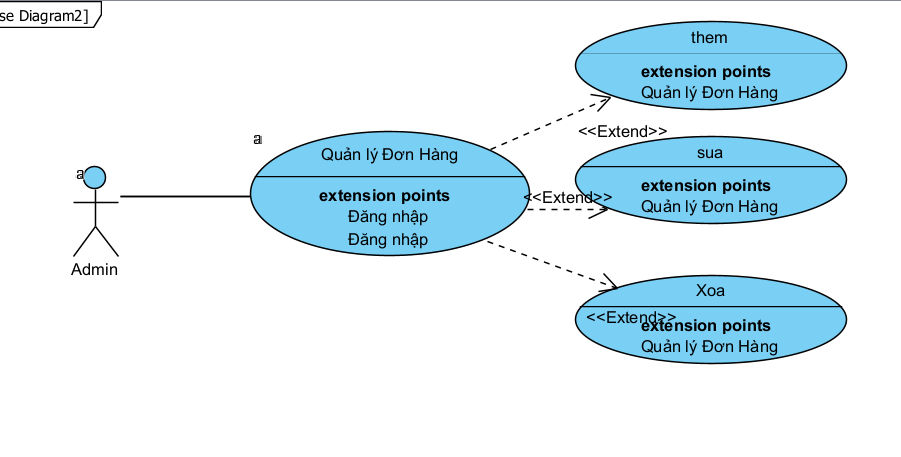
Hình 3. Biểu đồ Use Case chức năng QL Tài khoản

**4.** Biểu đồ Use Case chức năng QL Sản Phẩm



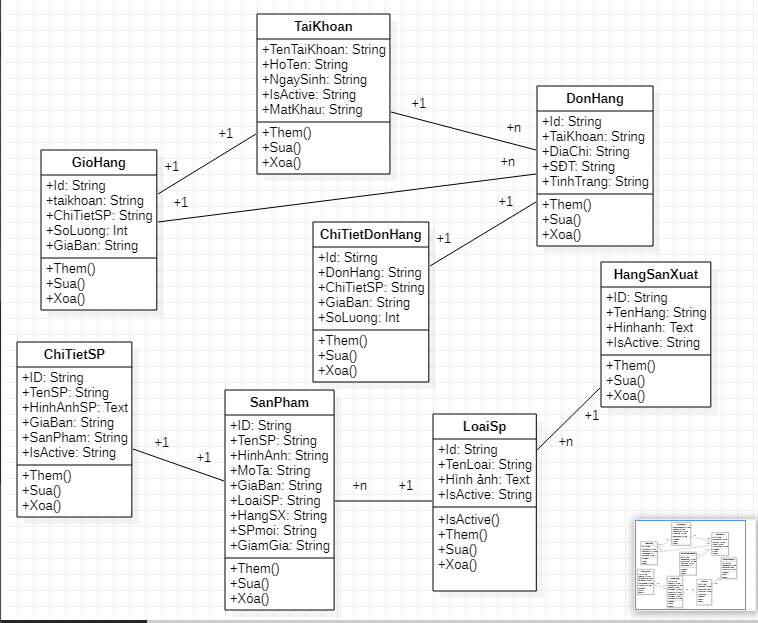
Hình 3. Biểu đồ Use Case chức năng QL Sản Phẩm

**5.** Biểu đồ Use Case chức năng QL Đơn Hàng



Hình 3. Biểu đồ Use Case chức năng QL Đơn Hàng

* + 1. Biểu đồ lớp thực thể



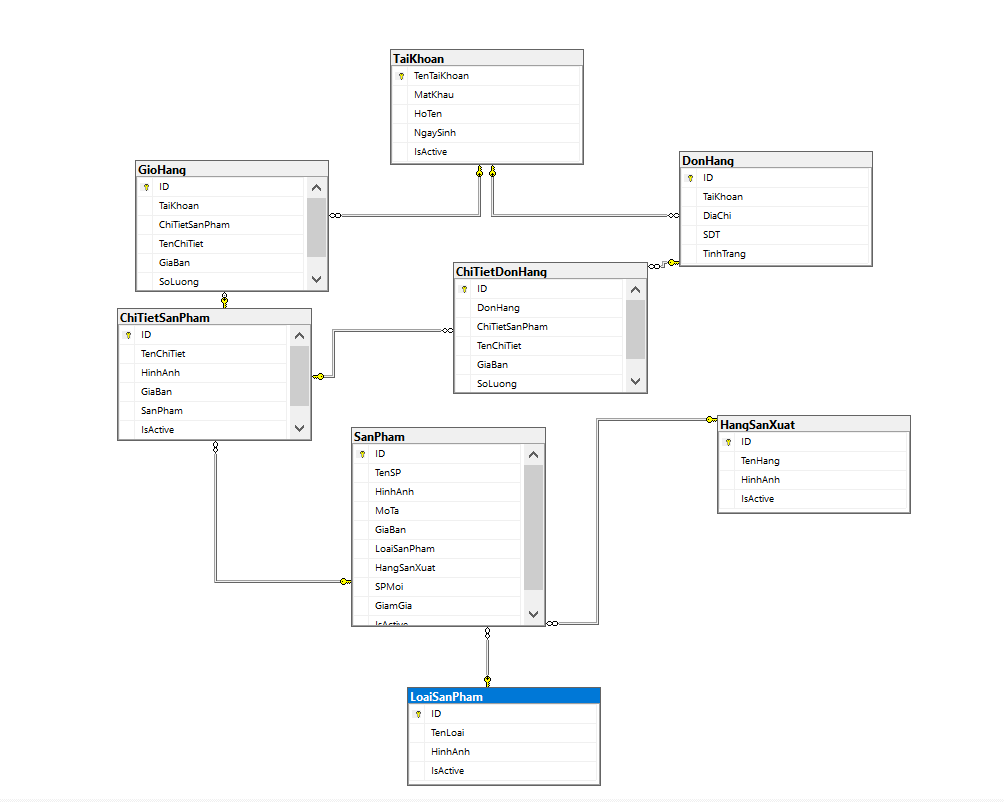
Hình 3. Biểu đồ lớp thực thể

* + 1. Các yêu cầu phi chức năng của hệ thống

|  |  |  |
| --- | --- | --- |
| II | **Yêu cầu phi chức năng** |  |
| 1 | Giao diện | Giao diện hệ thống phải dễ sử dụng, trực quan, thân thiện với mọi người dùng. |
| 2 | Tốc độ xử lý | Hệ thống phải xử lý nhanh chóng và chính xác. |
| 3 | Bảo mật dữ liệu | Tính bảo mật và độ an toàn cao. |

Bảng 1. : Yêu cầu phi chức năng

* 1. Thiết kế hệ thống
     1. Thiết kê cơ sở dữ liệu
* Mô hình cơ sở dữ liệu quan hệ



Hình 3. : Mô hình cơ sở dữ liệu quan hệ

* **Mô tả cấu trúc các bảng**

+ Bảng tài khoản

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | Tentaikhoan | Nvarchar(50) | Primary key | Tên tài khoản |
| 2 | Matkhau | Nvarchar(30) |  | Mật khẩu |
| 3 | Hoten | Nvarchar(30) |  | Họ tên |
| 4 | NgaySinh | Date |  | Ngày sinh |
| 5 | IsActive | Ntext |  | Trạng thái |

Bảng 2. Bảng tài khoản

+ Bảng đơn hàng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | ID | Varchar | Primary key |  |
| 2 | Tentaikhoan | Nvarchar(50) |  | Tên tài khoản |
| 3 | DiaChi | Nvarchar(100) |  | Địa chỉ |
| 4 | SĐT | Varchar(11) |  | Số điện thoại |
| 5 | Tinhtrang | Ntext |  | Tình trạng |

Bảng 2. Bảng đơn hàng

+ Bảng giỏ hàng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | ID | Nvarchar(30) | Primarykey |  |
| 2 | taikhoan | Nvarchar(100) |  | Tài khoản |
| 3 | ChiTietSP | Varchar(200) |  | Chi tiết sản phẩm |
| 4 | GiaBan | Int |  | Giá bán |
| 5 | SoLuong | Int |  | Số lượng |

Bảng 2. Bảng giỏ hàng

+ Bảng chi tiết hóa đơn

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | ID | Nvarchar(30) | Primary key |  |
| 2 | Donhang | Nvarchar(50) |  | Đơn hàng |
| 3 | ChiTietSP | Varchar(200) |  | Chi tiết sản phẩm |
| 4 | Giaban | Int |  | Giá bán |
| 5 | SoLuong | Int |  | Số lượng |

Bảng 2. Bảng chi tiết hóa đơn

+ Bảng sản phẩm

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | Id | Nvarchar(30) | Primary key |  |
| 2 | TenSP | Nvarchar(30) |  | Tên sản phẩm |
| 3 | Hinhanh | ntext |  | Hình ảnh sản phẩm |
| 4 | MoTa | Nvarchar(200) |  | Mô tả sản phẩm |
| 5 | GiaBan | Int |  | Giá bán |
| 6 | Loạigiaban | Int |  | Loại sản phẩm |

Bảng 2. Bảng sản phẩm

+ Bảng chi tiết sản phẩm

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | Id | Nvarchar(30) | Primary key |  |
| 2 | TenSP | Nvarchar(50) |  | Tên sản phẩm |
| 3 | Hinhanh | Ntext |  | Hình ảnh sản phẩm |
| 4 | MoTa | Nvarchar(200) |  | Mô tả sản phẩm |
| 5 | GiaBan | Int |  | Giá bán |

Bảng 2. Bảng chi tiết sản phẩm

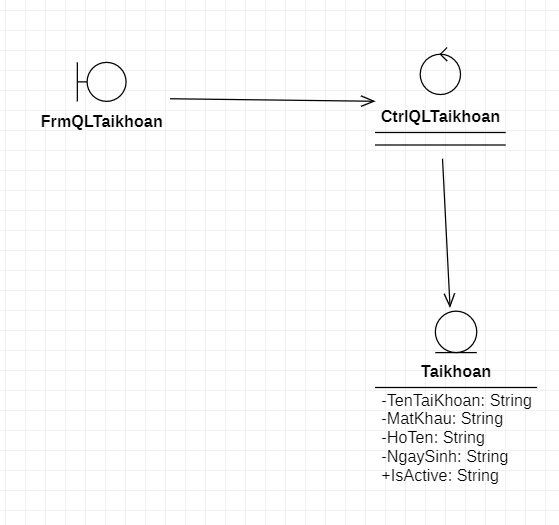
+ Bảng danh mục

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu dữ liệu | Ràng buộc | Mô tả |
| 1 | ID | Nvarchar(30) | Primary key |  |
| 2 | tenLoai | Nvarchar(30) |  | Tên loại sản phẩm |
| 3 | HinhAnh | ntext |  | Hình ảnh sản phẩm |
| 4 | UsActive | ntext |  | Tình trạng |

Bảng 2. Bảng danh mục

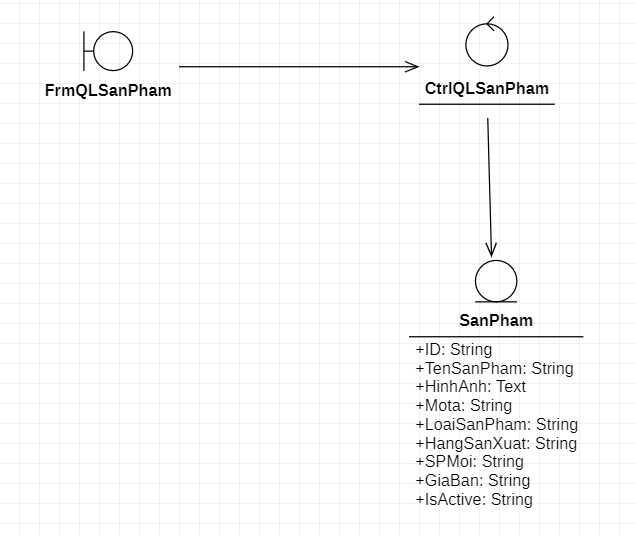
* + 1. Thiết kế các lớp và các đối tượng
       1. Biểu đồ lớp VOPC của các ca sử dụng

**1**.QL tài khoản:



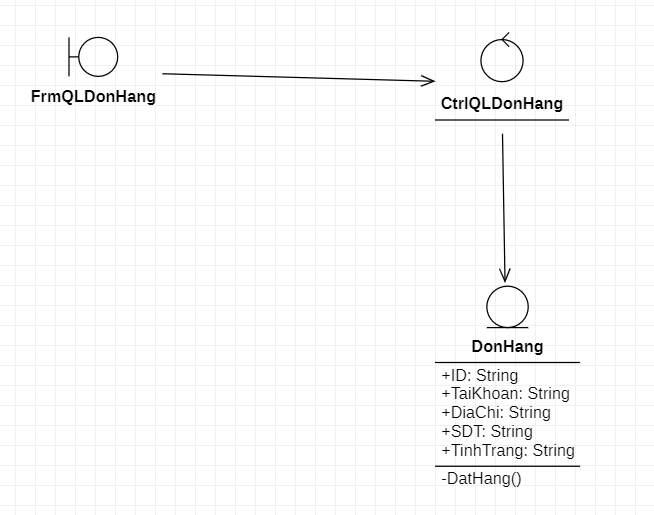
Hình 3. : Biểu đồ lớp VOPC QL tài khoản

**2.**QL sản phẩm:



Hình 3. : Biểu đồ lớp VOPC QL sản phẩm

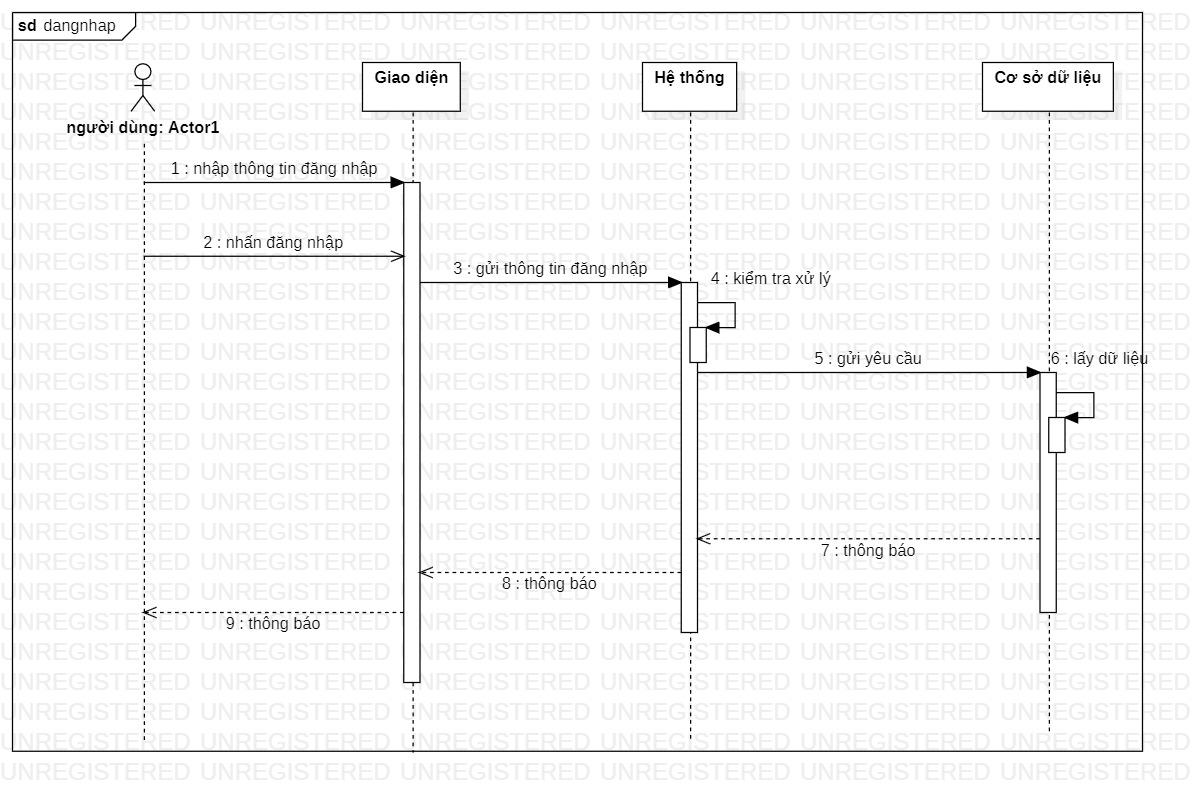
**3.**QL đơn hàng:



Hình 3. Biểu đồ lớp VOPC QL đơn hàng

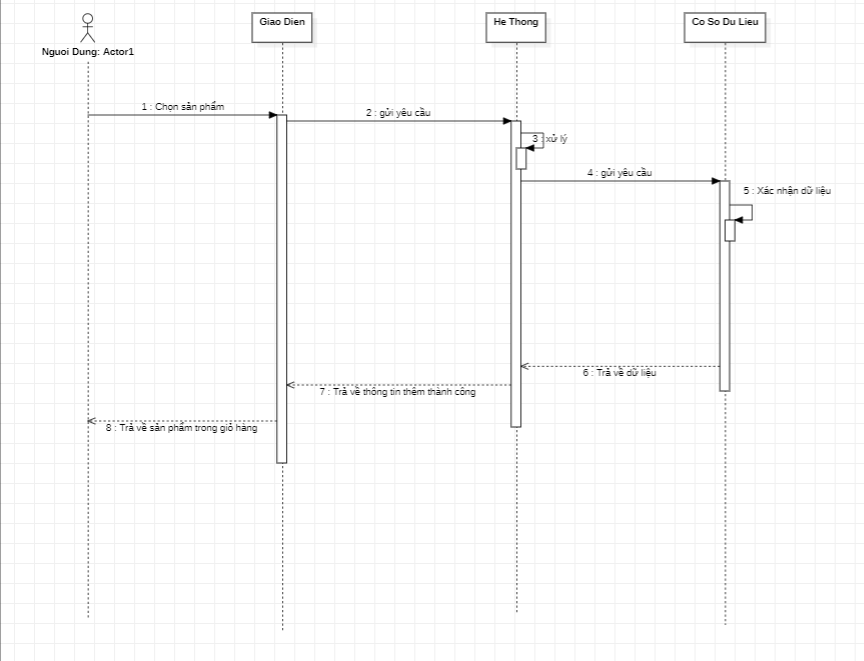
* + - 1. Biểu đồ tuần tự

**1.**Chức năng đăng nhập:



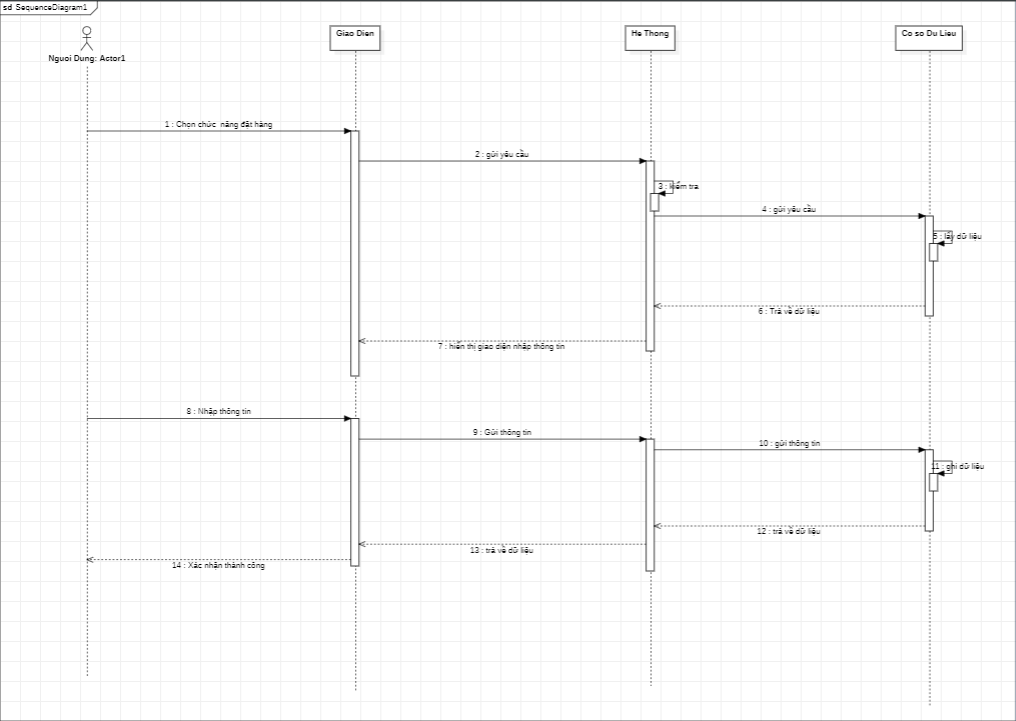
Hình 3. : Biểu đồ tuần tự đăng nhập

**2.**Chức năng giỏ hàng:



Hình 3. : Biểu đồ tuần tự giỏ hàng

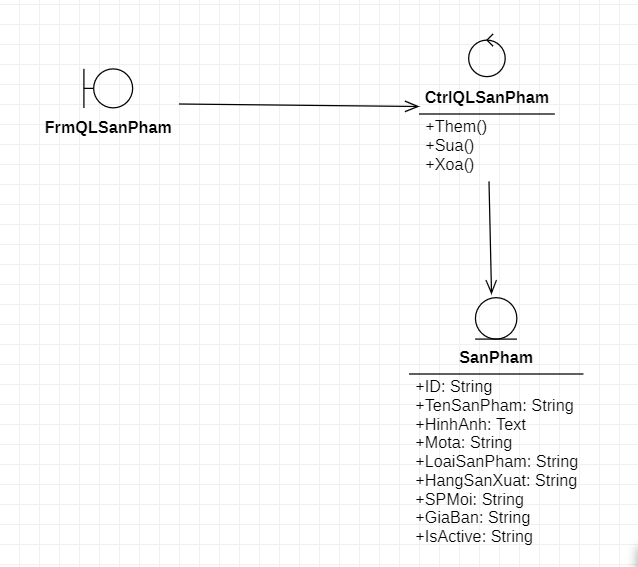
**3.**Đặt hàng:



Hình 3. : Biểu đồ tuần tự giỏ hàng

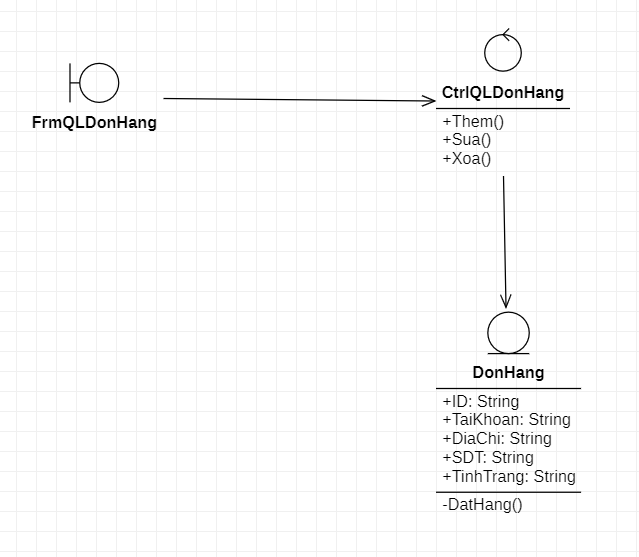
* + - 1. Biểu đồ lớp chi tiết

**1.**QL sản phẩm:



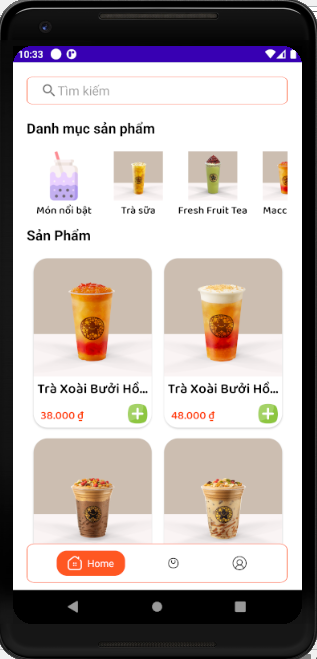
Hình 3. : Biểu đồ chi tiết QL sản phẩm

**2.**QL đơn hàng:

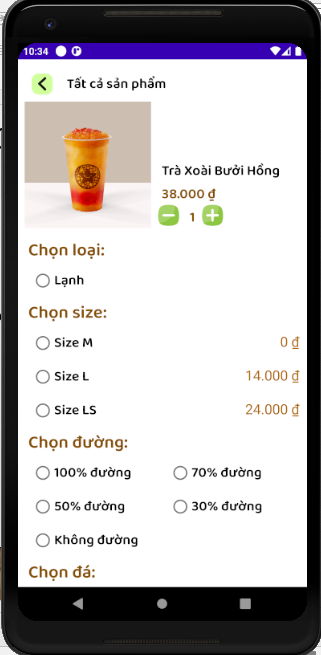


Hình 3. : Biểu đồ chi tiết QL đơn hàng

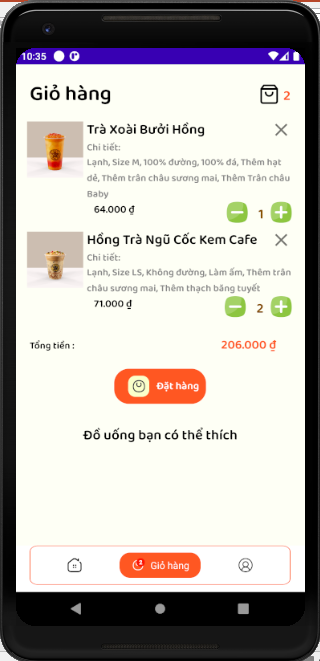
* + 1. Thiết kế giao diện



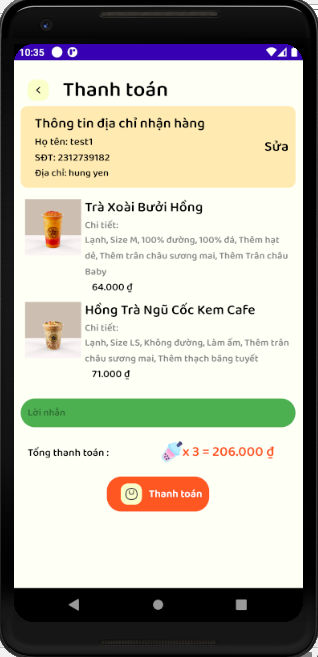
Hình 4. : Giao diện chính ứng dụng



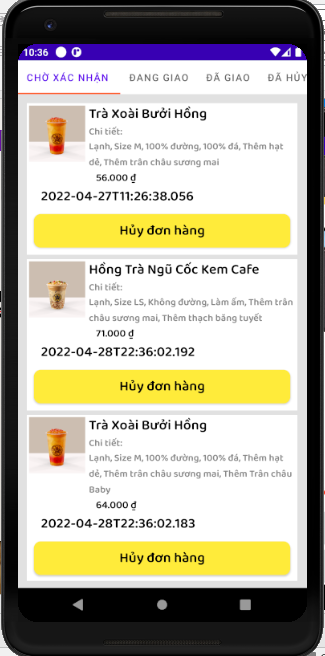
Hình 4. : Giao diện đặt hàng



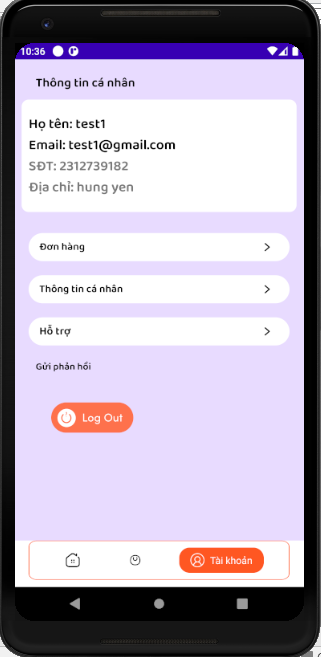
Hình 4. : Giao diện giỏ hàng



Hình 4. : Giao diện thanh toán



Hình 4. : Giao diện trạng thái đơn hàng



Hình 4. : Giao diện thông tin cá nhân

# TRIỂN KHAI WEBSITE

* 1. Triển khai các chức năng cho ứng dụng

Ứng dụng có các chức năng chính:

* + - * + Chức năng đăng kí, đăng nhập
        + Chức năng thêm vào giỏ hàng
        + Chức năng đặt hàng
        + Chức năng xem tình trạng của đơn hàng
    1. **Các chức năng nghiệp vụ**

**+ Chức năng tìm kiếm:**

**public class** Timkiem **extends** Fragment {  
  
 EditText **timkiem**;  
 TextView **tb**;  
 RecyclerView **recy\_1**;  
 ImageView **image**;  
 sp\_adapter **sp\_adapter**;  
 ArrayList<san\_pham> **arrayList**=**new** ArrayList<>();  
 **public** Timkiem() {  
 *// Required empty public constructor* }  
  
  
 @Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 *// Inflate the layout for this fragment* **return** inflater.inflate(R.layout.***fragment\_timkiem***, container, **false**);  
 }  
  
 @Override  
 **public void** onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {  
 **super**.onViewCreated(view, savedInstanceState);  
 *chipNavigationBar*.setVisibility(View.***GONE***);  
 **timkiem**=view.findViewById(R.id.***timkiem***);  
 **tb**=view.findViewById(R.id.***tb***);  
 **recy\_1**=view.findViewById(R.id.***recy\_1***);  
 **image**=view.findViewById(R.id.***image***);  
  
 **image**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 *chipNavigationBar*.setVisibility(View.***VISIBLE***);  
 Navigation.*findNavController*(view).navigate(R.id.***action\_timkiem2\_to\_trang\_chu***);  
 }  
 });  
 Home home= (Home) getActivity();  
 *on\_back1*(home,**this**, R.id.***action\_timkiem2\_to\_trang\_chu***);  
 **timkiem**.addTextChangedListener(**new** TextWatcher() {  
 @Override  
 **public void** beforeTextChanged(CharSequence charSequence, **int** i, **int** i1, **int** i2) {  
  
 }  
  
 @Override  
 **public void** onTextChanged(CharSequence charSequence, **int** i, **int** i1, **int** i2) {  
  
 }  
  
 @Override  
 **public void** afterTextChanged(Editable editable) {  
 getdata(**timkiem**.getText().toString().trim());  
 }  
 });  
  
 }  
  
 **private void** getdata(String tikiem) {  
 API.***API***.gettimkiem(tikiem).enqueue(**new** Callback<List<san\_pham>>() {  
 @Override  
 **public void** onResponse(Call<List<san\_pham>> call, Response<List<san\_pham>> response) {  
 **if** (response.isSuccessful()){  
 **tb**.setVisibility(View.***GONE***);  
 *//xóa dữu liệu ở arraylist* **arrayList**.clear();  
 *// add dữu liệu mới vào arraylist* **arrayList**= (ArrayList<san\_pham>) response.body();  
 **sp\_adapter**=**new** sp\_adapter(**arrayList**,Timkiem.**this**,2);  
 **recy\_1**.setLayoutManager(**new** GridLayoutManager(getContext(),2));  
 *// set dữu liệu hiển thị trên recy\_1* **recy\_1**.setAdapter(**sp\_adapter**);  
 *// tự động load lại dữu liệu hiển thị* **recy\_1**.post(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **sp\_adapter**.notifyDataSetChanged();  
 }  
 });  
 }**else** {  
 **arrayList**.clear();  
 **sp\_adapter**=**new** sp\_adapter(**arrayList**,Timkiem.**this**,2);  
 **recy\_1**.setAdapter(**sp\_adapter**);  
 **recy\_1**.post(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **sp\_adapter**.notifyDataSetChanged();  
 }  
 });  
  
 **tb**.setVisibility(View.***VISIBLE***);  
 }  
 }  
  
 @Override  
 **public void** onFailure(Call<List<san\_pham>> call, Throwable t) {  
 Log.*e*(**"Lỗi tìm kiếm"**,**""**+t);  
 }  
 });  
 }

* + 1. **Các chức năng hệ thống** 
       - * **Chức năng đăng kí:**

**public class** Dang\_ky **extends** Fragment {  
 ImageView **image**;  
 EditText **hoten**, **email**, **mk**;  
 Button **btndk**;  
 TextView **dangnhap**;  
 Dialog **dialog**;  
  
 **public** Dang\_ky() {  
 *// Required empty public constructor* }  
  
  
 @Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 *// Inflate the layout for this fragment* **return** inflater.inflate(R.layout.***fragment\_dang\_ky***, container, **false**);  
 }  
  
 @Override  
 **public void** onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {  
 **super**.onViewCreated(view, savedInstanceState);  
 anhxa(view);  
 onclick();  
 }  
  
 **private void** onclick() {  
 **image**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 Navigation.*findNavController*(view).navigate(R.id.***action\_dang\_ky\_to\_welcome***);  
 }  
 });  
 **btndk**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **dialog**.show();  
 String email1 = **email**.getText().toString().trim();  
 String hoten1 = **hoten**.getText().toString().trim();  
 String mk1 = **mk**.getText().toString().trim();  
 **if** (hoten1.isEmpty() && hoten1.isEmpty() && mk1.isEmpty()) {  
 **dialog**.dismiss();  
 Toast.*makeText*(getContext(), **"Vui lòng ko để trống thông tin"**, Toast.***LENGTH\_SHORT***).show();  
 } **else** {  
 API.***API***.kiemtra(email1, mk1).enqueue(**new** Callback<List<taikhoan>>() {  
 @Override  
 **public void** onResponse(Call<List<taikhoan>> call, Response<List<taikhoan>> response) {  
 **if** (!response.isSuccessful()) {  
 taikhoan taikhoan = **new** taikhoan(0, email1, hoten1, mk1,**"Thêm số điện thoại"**,**"Thêm địa chỉ"**);  
 API.***API***.post\_tk(taikhoan).enqueue(**new** Callback<com.example.project\_tra\_sua.models.taikhoan>() {  
 @Override  
 **public void** onResponse(Call<com.example.project\_tra\_sua.models.taikhoan> call, Response<com.example.project\_tra\_sua.models.taikhoan> response) {  
 **if** (response.isSuccessful()) {  
 **dialog**.dismiss();  
 Toast.*makeText*(getContext(), **"Đăng ký thành công"**, Toast.***LENGTH\_SHORT***).show();  
 Navigation.*findNavController*(view).navigate(R.id.***action\_dang\_ky\_to\_dang\_nhap***);  
  
  
  
 } **else** {  
 **dialog**.dismiss();  
 Toast.*makeText*(getContext(), **"Đăng ký thất bại"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
  
 @Override  
 **public void** onFailure(Call<com.example.project\_tra\_sua.models.taikhoan> call, Throwable t) {  
 **dialog**.dismiss();  
 Log.*e*(**"lỗi đăng ký"**, **""** + t);  
 }  
 });  
 } **else** {  
 **dialog**.dismiss();  
 Toast.*makeText*(getContext(), **"Tài khoản tồn tại"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
  
 @Override  
 **public void** onFailure(Call<List<taikhoan>> call, Throwable t) {  
 **dialog**.dismiss();  
 Log.*e*(**"lỗi đăng ký"**, **""** + t);  
 }  
 });  
 }  
  
  
 }  
 });  
 **dangnhap**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 Navigation.*findNavController*(view).navigate(R.id.***action\_dang\_ky\_to\_dang\_nhap***);  
 }  
 });  
 }

* + - * + **Chức năng đăng nhập:**

@Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 *// Inflate the layout for this fragment* **return** inflater.inflate(R.layout.***fragment\_dang\_nhap***, container, **false**);  
 }  
  
 @Override  
 **public void** onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {  
 **super**.onViewCreated(view, savedInstanceState);  
 anhxa(view);  
 onclick();  
 }  
  
 **private void** onclick() {  
 MainActivity mainActivity= (MainActivity) getActivity();  
 **image**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 Navigation.*findNavController*(view).navigate(R.id.***action\_dang\_nhap\_to\_welcome***);  
 }  
 });  
 *on\_back*(mainActivity,**this**, R.id.***action\_dang\_nhap\_to\_welcome***);  
 **dk**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 Navigation.*findNavController*(view).navigate(R.id.***action\_dang\_nhap\_to\_dang\_ky***);  
 }  
 });  
 **btndn**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **dialog**.show();  
 String email1=**email**.getText().toString().trim();  
 String mk1=**mk**.getText().toString().trim();  
 **if** (email1.isEmpty()||mk1.isEmpty()){  
 **dialog**.dismiss();  
 Toast.*makeText*(getContext(), **"Vui lòng điền đủ thông tin"**, Toast.***LENGTH\_SHORT***).show();  
 }**else** {  
 API.***API***.kiemtra(email1,mk1).enqueue(**new** Callback<List<taikhoan>>() {  
 @Override  
 **public void** onResponse(Call<List<taikhoan>> call, Response<List<taikhoan>> response) {  
 **if** (response.isSuccessful()){  
 **dialog**.dismiss();  
 taikhoan taikhoan=response.body().get(0);  
 Intent intent=**new** Intent(getActivity(),Home.**class**);  
 intent.putExtra(**"idtk"**,taikhoan.getIdtk());  
 startActivity(intent);  
 getActivity().finish();  
 }**else** {  
 **dialog**.dismiss();  
 Toast.*makeText*(getContext(), **"Tài khoản mật khẩu ko tồn tại"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
  
 @Override  
 **public void** onFailure(Call<List<taikhoan>> call, Throwable t) {  
 **dialog**.dismiss();  
 Log.*e*(**"Lỗi đăng nhập"**,**""**+t);  
 }  
 });  
 }  
  
 }  
 });  
 }

* + - * + **Chức năng thêm vào giỏ hàng:**

**public** View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 *// Inflate the layout for this fragment* **return** inflater.inflate(R.layout.***fragment\_gio\_hang***, container, **false**);  
 }  
  
 @Override  
 **public void** onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {  
 **super**.onViewCreated(view, savedInstanceState);  
 **list\_item** = view.findViewById(R.id.***list\_item***);  
 **layout1** = view.findViewById(R.id.***layout1***);  
 **sl** = view.findViewById(R.id.***sl***);  
 **tongtien** = view.findViewById(R.id.***tongtien***);  
 **dathang** = view.findViewById(R.id.***dathang***);  
 **muasp** = view.findViewById(R.id.***muasp***);  
 **layout\_giohang** = view.findViewById(R.id.***layout\_giohang***);  
 getdata();  
 getdata1();  
 **dathang**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 Navigation.*findNavController*(view).navigate(R.id.***action\_gio\_hang2\_to\_don\_hang***);  
 }  
 });  
 **muasp**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 *chipNavigationBar*.setItemSelected(R.id.***home***, **true**);  
 }  
 });  
 }  
  
 **private void** getdata1() {  
 API.***API***.tongtien(*idtk*,**"không"**).enqueue(**new** Callback<String>() {  
 @Override  
 **public void** onResponse(Call<String> call, Response<String> response) {  
 **if** (response.isSuccessful()) {  
 **layout1**.setVisibility(View.***VISIBLE***);  
 **layout\_giohang**.setVisibility(View.***GONE***);  
 **tongtien**.setText(*currencyVN*.format(Integer.*parseInt*(response.body())));  
 }**else** {  
 **layout1**.setVisibility(View.***GONE***);  
 **layout\_giohang**.setVisibility(View.***VISIBLE***);  
 }  
  
 }  
  
 @Override  
 **public void** onFailure(Call<String> call, Throwable t) {  
  
 }  
 });  
  
 }  
  
 **private void** getdata() {  
 API.***API***.getallsp(*idtk*,**"không"**).enqueue(**new** Callback<List<giohang>>() {  
 @Override  
 **public void** onResponse(Call<List<giohang>> call, Response<List<giohang>> response) {  
 **if** (response.isSuccessful()) {  
 **sl**.setText(**""** + response.body().size());  
 ArrayList<giohang> arrayList = (ArrayList<giohang>) response.body();  
 **giohangadapter** = **new** giohangadapter(arrayList, gio\_hang.**this**,0);  
 **list\_item**.setLayoutManager(**new** LinearLayoutManager(getContext()));  
 **list\_item**.setAdapter(**giohangadapter**);  
 **giohangadapter**.notifyDataSetChanged();  
 }  
 }  
  
 @Override  
 **public void** onFailure(Call<List<giohang>> call, Throwable t) {  
  
 }  
 });  
 }  
 @Override  
 **public void** onResume() {  
 **super**.onResume();  
 **handler**.postDelayed( **runnable** = **new** Runnable() {  
 **public void** run() {  
 getdata1();  
 API.***API***.getallsp(*idtk*,**"không"**).enqueue(**new** Callback<List<giohang>>() {  
 @Override  
 **public void** onResponse(Call<List<giohang>> call, Response<List<giohang>> response) {  
 **if** (response.isSuccessful()) {  
 **if** (response.body().size()< Integer.*parseInt*(**sl**.getText().toString())){  
 getdata();  
 }  
 }  
 }  
  
 @Override  
 **public void** onFailure(Call<List<giohang>> call, Throwable t) {  
  
 }  
 });  
 **handler**.postDelayed(**runnable**, 2000);  
 }  
 }, 2000);  
 }

* + - * + **Chức năng đặt hàng:**

**public** View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 *// Inflate the layout for this fragment* **return** inflater.inflate(R.layout.***fragment\_don\_hang***, container, **false**);  
}  
  
@Override  
**public void** onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {  
 **super**.onViewCreated(view, savedInstanceState);  
 *chipNavigationBar*.setVisibility(View.***GONE***);  
 **list\_item**=view.findViewById(R.id.***list\_item***);  
 **ttt**=view.findViewById(R.id.***ttt***);  
 **ten**=view.findViewById(R.id.***ten***);  
 **sdt**=view.findViewById(R.id.***sdt***);  
 **diachi**=view.findViewById(R.id.***diachi***);  
 **sua**=view.findViewById(R.id.***sua***);  
 **image**=view.findViewById(R.id.***image***);  
 **layout1**=view.findViewById(R.id.***layout1***);  
 **loinhan**=view.findViewById(R.id.***loinhan***);  
 **dialog**=**new** Dialog(getContext());  
 *dia*(**dialog**);  
 **sua**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 Bundle bundl=**new** Bundle();  
 bundl.putInt(**"ttcn"**,1);  
 Navigation.*findNavController*(view).navigate(R.id.***action\_don\_hang\_to\_thong\_tin\_ca\_nhan***,bundl);  
 }  
 });  
 **image**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 *chipNavigationBar*.setVisibility(View.***VISIBLE***);  
 Navigation.*findNavController*(view).navigate(R.id.***action\_don\_hang\_to\_gio\_hang2***);  
 }  
 });  
 **layout1**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **dialog**.show();  
 **if** (**diachi**.getText().toString().equals(**"Địa chỉ: Thêm địa chỉ"**)){  
 **dialog**.dismiss();  
 Toast.*makeText*(getContext(), **"Vui lòng thêm địa chỉ"**, Toast.***LENGTH\_SHORT***).show();  
 }**else** {  
 API.***API***.getallsp(*idtk*,**"không"**).enqueue(**new** Callback<List<giohang>>() {  
 @RequiresApi(api = Build.VERSION\_CODES.***O***)  
 @Override  
 **public void** onResponse(Call<List<giohang>> call, Response<List<giohang>> response) {  
 **if** (response.isSuccessful()) {  
 **dialog**.dismiss();  
 ArrayList<giohang> arrayList = (ArrayList<giohang>) response.body();  
 **for** (giohang d:arrayList){  
 String ngaygio= String.*valueOf*(java.time.LocalDateTime.*now*());  
 donhang donhang=**new** donhang(0,d.getIdgh(),*idtk*,**loinhan**.getText().toString(),ngaygio,**"Chờ duyệt"**,**""**);  
 API.***API***.Postdonhang(donhang).enqueue(**new** Callback<com.example.project\_tra\_sua.models.donhang>() {  
 @Override  
 **public void** onResponse(Call<com.example.project\_tra\_sua.models.donhang> call, Response<com.example.project\_tra\_sua.models.donhang> response) {  
 **if** (response.isSuccessful()){  
  
 giohang giohang=**new** giohang(d.getIdgh(),d.getIdtk(),d.getIdsp(),d.getSl(),d.getChon(),d.getGiax1(),d.getTongtien(),**"đơn hàng"**);  
 API.***API***.Putgiohang(d.getIdgh(),giohang).enqueue(**new** Callback<com.example.project\_tra\_sua.models.giohang>() {  
 @Override  
 **public void** onResponse(Call<com.example.project\_tra\_sua.models.giohang> call, Response<com.example.project\_tra\_sua.models.giohang> response) {  
  
 }  
  
 @Override  
 **public void** onFailure(Call<com.example.project\_tra\_sua.models.giohang> call, Throwable t) {  
  
 }  
 });  
  
 }**else** {  
 **dialog**.dismiss();  
 Toast.*makeText*(getContext(), **"Đặt hàng thất bại"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
  
 @Override  
 **public void** onFailure(Call<com.example.project\_tra\_sua.models.donhang> call, Throwable t) {  
 **dialog**.dismiss();  
 Log.*e*(**"lỗi đặt hàng"**,t.getMessage());  
 }  
 });  
 }  
 *chipNavigationBar*.setVisibility(View.***VISIBLE***);  
 Toast.*makeText*(getContext(), **"Đặt hàng thành công"**, Toast.***LENGTH\_SHORT***).show();  
 Navigation.*findNavController*(view).navigate(R.id.***action\_don\_hang\_to\_trang\_chu***);  
 *chipNavigationBar*.setItemSelected(R.id.***home***, **true**);  
 }  
 }  
  
 @Override  
 **public void** onFailure(Call<List<giohang>> call, Throwable t) {  
  
 }  
 });  
 }  
  
  
 }  
 });  
 getdata();  
}

* 1. Kiểm thử và triển khai ứng dụng
     1. Kiểm thử

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên testcase** | **Mục đích** | **Thực hiện** | **Kết quả** |
| 1 | Đăng ký tài khoản | Kiểm tra chức năng tạo tài khoản | Nhập thông tin tài khoản: email, mật khẩu, họ tên, ngày sinh, …. Ấn nút tạo tài khoản | Thông báo tạo tài khoản thành công |
| 2 | Nhập tài khoản đã tồn tại | Kiểm tra tài khoản đã tồn tại | Nhập thông tin tài khoản đã tồn tại | Thông báo tài khoản đã tồn tại |
| 3 | Để trống các textbox | Kiểm tra khi textbox rỗng | Không nhập gì | Thông báo không được để trống |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên testcase** | **Mục đích** | **Thực hiện** | **Kết quả** |
| 1 | Đăng  nhập tài khoản chính xác | Kiểm tra chức năng đăng nhập | Nhập thông tin email và mật khẩu | Thông báo đăng nhập thành công, truy cập tới màn hình chính |
| 2 | Đăng nhập tài khoản không chính xác | Kiểm tra chức năng đăng nhập | Nhập sai thông tin email hoặc mật khẩu | Thông báo không thành công |
| 3 | Để trống các textbox | Kiểm tra khi textbox rỗng | Không nhập gì | Thông báo không được để trống |

* + 1. Đóng gói ứng dụng

buildscript **{** repositories **{** google()  
 mavenCentral()  
 **}** dependencies **{** classpath **"com.android.tools.build:gradle:7.0.4"** *// NOTE: Do not place your application dependencies here; they belong  
 // in the individual module build.gradle files* **}  
}**task clean(**type**: Delete) **{** delete rootProject.buildDir  
**}**

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Kết quả đạt được

* Sau thời gian nghiên cứu và phát triển, em đã hoàn thành cơ bản các mục tiêu đề ra của đề tài
* Nắm được quy trình xây dựng một ứng dụng đa phương tiện
* Nội dung ứng dụng được trình bày hợp lý
* Về mặt giao diện
* Thiết kế ứng dụng có giao diện thân thiện với người sử dụng
* Bố cục giao diện ứng dụng cân đối hài hòa, phù hợp với nội dung hiển thị
* Về mặt chức năng
* Tìm kiếm sản phẩm theo tên, xem chi tiết sản phẩm, đặt hàng

Hạn chế của đề tài

* Một số chức năng còn chưa hoàn hảo.
* Cơ sở dữ liệu của hệ thống chưa được đầy đủ và tối ưu.
* Tốc độ chưa hiệu quả.

Hướng phát triển của đề tài

* Với sự nỗ lực, em đã cố gắng hoàn thành yêu cầu đề tài. Do thời gian có hạn, khả năng còn hạn chế nên đề tài mới chỉ đáp ứng được yêu cầu của khách hàng ở mức độ đơn giản và chưa tối ưu.
* Mở rộng bài toán để đáp ứng nhiều hơn các chức năng còn thiếu

TÀI LIỆU THAM KHẢO

1. Tên tác giả (năm XB), Tên sách, NXB, Nơi XB
2. Ed Burnette, *Hello, Android Introducing Google’s Mobile Development Platform*, The Pragmatic Bookshelf, 2010.
3. Marziah Karch, *Android for Work Productivity for Professional*, Press, 2010.
4. J.F. DiMarzio, *Android A Programmer’s Guide*, The McGraw-Hill Companies, 2008.
5. Chris Haseman, *Android Essentials*, Apress, 2008.
6. Sams, *Android Application Development*, Lauren Darcey and Shane Conder, 2010.
7. stackoverflow.com, <https://stackoverflow.com>