|  |  |
| --- | --- |
|  | **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **---------------------------------------** |
| NGUYỄN ĐỨC HUY | ĐỒ ÁN TỐT NGHIỆP NGÀNH KỸ THUẬT PHẦN MỀM |
| **XÂY DỰNG ỨNG DỤNG THƯƠNG MẠI ĐIỆN TỬ TRÊN NỀN TẢNG DI ĐỘNG BẰNG FLUTTER** |
|  | **CBHD : ThS. Nguyễn Thái Cường** |
|  | **Sinh viên : Nguyễn Đức Huy** |
|  | **Mã số sinh viên: 2021601675** |
| KỸ THUẬT PHẦN MỀM |  |
|  | Hà Nội – Năm 2025 |

MỤC LỤC

[MỤC LỤC 2](#_Toc197521056)

[DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ CÁI VIẾT TẮT 4](#_Toc197521057)

[DANH MỤC HÌNH VẼ 4](#_Toc197521058)

[DANH MỤC BẢNG BIỂU 4](#_Toc197521059)

[LỜI CẢM ƠN 5](#_Toc197521060)

[MỞ ĐẦU 6](#_Toc197521061)

[1.Lý do chọn đề tài 6](#_Toc197521062)

[2.Mục tiêu, đối tượng và phạm vi nghiên cứu 6](#_Toc197521063)

[3.Phương pháp nghiên cứu 7](#_Toc197521064)

[CHƯƠNG 1. TỔNG QUAN VỀ NỘI DUNG NGHIÊN CỨU 8](#_Toc197521065)

[1.1.Giới thiệu chung 8](#_Toc197521066)

[1.2.Cơ sở lý thuyết 8](#_Toc197521067)

[1.2.1.Tổng quan về phát triển ứng dụng di động 8](#_Toc197521068)

[1.2.2.Giới thiệu Flutter 10](#_Toc197521069)

[1.2.3.Giới thiệu Node.js 11](#_Toc197521070)

[1.2.4.Giới thiệu MongoDB 12](#_Toc197521071)

[1.2.5. Mô hình MVVM kết hợp Provider trong Flutter 12](#_Toc197521072)

[1.3. Tổng quan về chức năng của hệ thống thương mại điện tử 13](#_Toc197521073)

[1.4. Lý do lựa chọn công nghệ và mô hình kiến trúc 14](#_Toc197521074)

[CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 16](#_Toc197521075)

[2.1. Khảo sát người dùng 16](#_Toc197521076)

[2.2. Giới thiệu về hệ thống 17](#_Toc197521077)

[2.3. Phân tích yêu cầu 18](#_Toc197521078)

[2.3.1. Về hệ thống 18](#_Toc197521079)

[2.3.2. Về người sử dụng 18](#_Toc197521080)

[2.3.3. Yêu cầu về chức năng 19](#_Toc197521081)

[2.3.4. Yêu cầu phi chức năng: 20](#_Toc197521082)

[2.4. Thiết kế hệ thống 21](#_Toc197521083)

[2.4.1. Mô hình hóa Usecase 21](#_Toc197521084)

[2.4.1.1. Xác định các tác nhân 21](#_Toc197521085)

[2.4.1.2. Xác định các Usecase 21](#_Toc197521086)

[2.4.1.3. Biểu đồ Usecase 22](#_Toc197521087)

[2.4.2. Mô tả chi tiết các Usecase 27](#_Toc197521088)

[2.4.2.1. Mô tả Usecase <Đăng ký tài khoản> 27](#_Toc197521089)

[2.4.2.2. Mô tả Usecase <Đăng nhập> 29](#_Toc197521090)

[2.4.2.3. Mô tả Usecase <Quên mật khẩu> 31](#_Toc197521091)

[2.4.2.4. Mô tả Usecase <Tìm kiếm sản phẩm> 33](#_Toc197521092)

[2.4.2.5. Mô tả Usecase <Thêm vào giỏ hàng> 35](#_Toc197521093)

[2.4.2.7. Mô tả Usecase <Đánh giá sản phẩm> 39](#_Toc197521094)

[2.4.2.8. Mô tả Usecase <Thống kê doanh thu> 41](#_Toc197521095)

[2.4.2.9. Mô tả Usecase <Quản lý thông tin sản phẩm> 43](#_Toc197521096)

[2.4.2.10. Mô tả Usecase <Quản lý khách hàng> 47](#_Toc197521097)

[2.4.2.11. Mô tả Usecase <Xuất hóa đơn> 50](#_Toc197521098)

[2.4.2.12. Mô tả Usecase <Gửi thông tin đơn hàng qua Email> 53](#_Toc197521099)

[2.4.3. Biểu đồ lớp hệ thống 54](#_Toc197521100)

[2.4.3.1. Lớp Model 55](#_Toc197521101)

[2.4.3.2. Lớp ViewModel 55](#_Toc197521102)

[2.4.3.3. Lớp View (Màn hình) 56](#_Toc197521103)

[2.4.4. Thiết kế giao diện 56](#_Toc197521104)

[2.4.4.1. Giao diện Đăng nhập và Đăng ký 56](#_Toc197521105)

[2.4.4.2. Giao diện Trang chủ và Danh sách sản phẩm 58](#_Toc197521106)

[2.4.4.3. Giao diện Giỏ hàng và Thanh toán 60](#_Toc197521107)

[2.4.4.4. Giao diện Quản lý đơn hàng 62](#_Toc197521108)

[2.4.4.5. Giao diện Đánh giá sản phẩm 64](#_Toc197521109)

[2.4.4.6. Giao diện Quản trị viên 64](#_Toc197521110)

[2.4.5. Thiết kế cơ sở dữ liệu 66](#_Toc197521111)

[CHƯƠNG 3. KẾT QUẢ 69](#_Toc197521112)

[3.1.Cài đặt môi trường 69](#_Toc197521113)

[3.2. Cài đặt chương trình 69](#_Toc197521114)

[3.3. Kết quả đạt được 71](#_Toc197521115)

[3.4. Kết quả kiểm thử 71](#_Toc197521116)

[KẾT LUẬN 74](#_Toc197521117)

[1. Đánh giá kết quả đạt được 74](#_Toc197521118)

[2. Hướng phát triển ứng dụng trong tương lai 75](#_Toc197521119)

[TÀI LIỆU THAM KHẢO 76](#_Toc197521120)

DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ CÁI VIẾT TẮT

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

LỜI CẢM ƠN

Trong suốt thời gian học tập và thực hiện đồ án tốt nghiệp em luôn nhận được sự hướng dẫn nhiệt tình từ phía nhà trường và các thầy cô giáo. Vì thế, lời đầu tiên em xin chân thành cảm ơn trường Đại học Công nghiệp Hà Nội, thầy cô giáo khoa Công nghệ thông tin đã tạo điều kiện thuận lợi và truyền dạy kiến thức cho em trong thời gian qua để em hoàn thành đề tài một cách tốt nhất. Và hơn hết em xin bày tỏ lòng biết ơn sâu sắc nhất tới thầy giáo hướng dẫn **ThS. Nguyễn Thái Cường** đã tận tình giúp đỡ, định hướng em rất nhiều trong suốt quá trình tìm hiểu nghiên cứu và hoàn thành đồ án tốt nghiệp.

Khi thực hiện nghiên cứu đề tài, em đã cố gắng hoàn thành tốt nhất báo cáo đồ án tốt nghiệp. Tuy nhiên kiến thức chuyên ngành của bản thân còn nhiều hạn chế. Vì vậy trong báo cáo không tránh khỏi những thiếu sót, em rất mong nhận được sự đóng góp của tất cả các thầy cô giáo để đồ án của em được đầy đủ và hoàn chỉnh hơn.

Em xin chân thành cảm ơn!

MỞ ĐẦU

1.Lý do chọn đề tài

Trong bối cảnh nền kinh tế số phát triển mạnh mẽ, thương mại điện tử đã và đang trở thành một phần không thể thiếu trong đời sống hiện đại. Đặc biệt, với sự phổ biến của các thiết bị di động thông minh, người tiêu dùng ngày càng có xu hướng mua sắm trực tuyến thông qua các ứng dụng di động. Điều này đặt ra yêu cầu cấp thiết cho các doanh nghiệp cũng như nhà phát triển phần mềm phải xây dựng những ứng dụng thương mại điện tử tiện lợi, thân thiện, đáp ứng tốt nhu cầu của cả khách hàng và nhà quản trị.

Nhận thấy tiềm năng to lớn cũng như những yêu cầu thực tế trên, em đã lựa chọn đề tài “Xây dựng ứng dụng thương mại điện tử trên nền tảng di động bằng Flutter”. Đề tài tập trung phát triển một ứng dụng di động đa nền tảng, tích hợp đầy đủ các chức năng cơ bản như đăng ký, đăng nhập, mua hàng, thanh toán, quản lý sản phẩm, quản lý khách hàng, thống kê doanh thu, xuất hóa đơn, gửi thông tin đơn hàng qua email và đánh giá sản phẩm. Ứng dụng được xây dựng dành cho cả người dùng (user) và quản trị viên (admin), bảo đảm đáp ứng nhu cầu sử dụng thực tế và dễ dàng mở rộng trong tương lai.

2.Mục tiêu, đối tượng và phạm vi nghiên cứu

a. Mục tiêu nghiên cứu

* Phân tích, thiết kế và xây dựng một ứng dụng thương mại điện tử đa nền tảng (Android, iOS) bằng Flutter.
* Triển khai các chức năng cơ bản và cần thiết cho một ứng dụng thương mại điện tử hiện đại: đăng ký, đăng nhập, quên mật khẩu, tìm kiếm sản phẩm, mua hàng, thanh toán, quản lý sản phẩm, quản lý khách hàng, thống kê doanh thu, xuất hóa đơn, gửi email xác nhận đơn hàng và đánh giá sản phẩm.
* Đảm bảo ứng dụng có giao diện thân thiện, dễ sử dụng cho người dùng và thuận tiện cho quản trị viên trong việc quản lý hệ thống.

b. Đối tượng nghiên cứu

* Ứng dụng thương mại điện tử trên nền tảng di động, dành cho cả khách hàng và quản trị viên.
* Các công nghệ chính: Flutter, Node.js, MongoDB, tích hợp các dịch vụ email.

c. Phạm vi nghiên cứu

* Xây dựng và triển khai ứng dụng thương mại điện tử với các chức năng cơ bản, tập trung vào trải nghiệm người dùng và hiệu quả quản trị.
* Không đi sâu vào các công nghệ thanh toán nâng cao (ví dụ: tích hợp nhiều cổng thanh toán quốc tế), bảo mật đa lớp phức tạp hay các giải pháp tối ưu hóa hiệu năng quy mô lớn.
* Ứng dụng chủ yếu phát triển và thử nghiệm trên nền tảng Android.

3.Phương pháp nghiên cứu

* Nghiên cứu lý thuyết:
  + Thu thập, tổng hợp và phân tích các tài liệu liên quan đến phát triển ứng dụng di động bằng Flutter, xây dựng backend với Node.js, quản lý dữ liệu bằng MongoDB, cùng các kiến thức về thương mại điện tử.
  + Tìm hiểu các tiêu chuẩn thiết kế giao diện, trải nghiệm người dùng (UI/UX) trong lĩnh vực thương mại điện tử.
* Nghiên cứu thực nghiệm:
  + Phân tích các ứng dụng thương mại điện tử phổ biến hiện nay để tham khảo chức năng và quy trình nghiệp vụ.
  + Thiết kế, xây dựng và triển khai ứng dụng mẫu theo mô hình đã chọn.
  + Thực hiện kiểm thử các chức năng, đánh giá hiệu quả, thu thập phản hồi để hoàn thiện sản phẩm.

CHƯƠNG 1. TỔNG QUAN VỀ NỘI DUNG NGHIÊN CỨU

1.1.Giới thiệu chung

Hiện nay, thương mại điện tử đang ngày càng phát triển mạnh mẽ, trở thành xu hướng tất yếu trong thời đại số hóa. Sự phổ biến của các thiết bị di động thông minh cùng với sự tiến bộ vượt bậc của các công nghệ lập trình đa nền tảng đã mở ra những cơ hội mới cho các doanh nghiệp tiếp cận khách hàng, nâng cao hiệu quả kinh doanh cũng như tối ưu hóa trải nghiệm người dùng. Đặc biệt, sau đại dịch Covid-19, hành vi của người tiêu dùng có sự thay đổi rõ rệt khi ưu tiên sử dụng các dịch vụ mua sắm trực tuyến, đặt hàng qua điện thoại di động thay cho các hình thức truyền thống.

Trước yêu cầu thực tiễn đó, việc xây dựng các ứng dụng thương mại điện tử trên nền tảng di động với đầy đủ chức năng, giao diện thân thiện, vận hành ổn định và bảo mật cao là một hướng đi quan trọng. Đề tài "Xây dựng ứng dụng thương mại điện tử trên nền tảng di động bằng Flutter" được thực hiện nhằm mục tiêu nghiên cứu, phân tích, thiết kế và xây dựng một ứng dụng mẫu đáp ứng các yêu cầu cơ bản của hệ thống thương mại điện tử hiện đại.

1.2.Cơ sở lý thuyết

1.2.1.Tổng quan về phát triển ứng dụng di động

Hiện nay có 3 hướng chính xây dựng và phát triển 1 ứng dụng di động là: Native App, Web App và Hybrid App. Mỗi hướng đều có ưu và nhược điểm và kỹ năng riêng.

1. **Native app**

Native App là những ứng dụng được viết riêng cho một loại nền tảng như iOS, Android, Windows Phone bằng các ngôn ngữ tương ứng của mọi nền tảng đó ví dụ Java trên Android, Swift trên iOS.

Mỗi Native App chỉ chạy được trên một nền tảng và không thể mang sang các nền tảng khác.

* Ưu điểm:
  + Tận dụng được tính năng có sẵn trên thiết bị như: GPS, Camera, thiết bị thu âm...
  + Hiệu năng cao vì code native chạy trực tiếp trên máy.
  + Có thể chạy được ở chế độ offline.
* Nhược điểm:
  + Không thể kết hợp nhiều nền tảng. Mỗi một ứng dụng chỉ chạy trên 1 nền tảng nhất định.
  + Mỗi hệ điều hành cần phải viết ứng dụng riêng, khó đồng bộ giữa các ứng dụng.
  + Việc bảo trì hay nâng cấp sẽ làm mất nhiều thời gian do phải sửa chữa từng app trên từng hệ điều hành.
  + Xây dựng ứng dụng cần dùng các phần mềm riêng biệt theo hệ điều hành (dùng Android Studio để phát triển ứng dụng Android, Xcode để phát triển ứng dụng iOS).

1. **Web app**

Web App được phát triển khi đã có sẵn một website đang hoạt động. Ta tạo thêm một trang web riêng cho các thiết bị di động sử dụng HTML, CSS và một số thư viện khác hỗ trợ.

Web App được thiết kế chạy trên nền tảng web hoặc các trình duyệt của thiết bị di động cho phép người dùng thao tác như thao tác trang web giống như thao tác ứng dụng.

* Ưu điểm:
  + Có thể chạy trên tất cả trình duyệt của mobile hỗ trợ phiên bản HTML và JavaScript.
  + Không cần cài đặt trên máy miễn là máy có trình duyệt web.
  + Một phiên bản duy nhất cho tất cả, nên giảm chi phí và thời gian cho phát triển, bảo trì, cũng như nâng cấp sau này.
  + Phiên bản được cập nhật liên tục không cần phải cập nhật trên chợ ứng dụng.
* Nhược điểm:
  + Hiệu năng không được tốt như native app và luôn phải chạy online.
  + Không thể dùng được các tính năng tích hợp của di động: Notification, chụp hình, nghiêng máy, định vị GPS, các sensor...
  + Với một số máy đời cũ, Web App sẽ bị vỡ giao diện, hiển thị sai, hoặc JavaScript không chạy.

1. **Hybrid app**

Hybrid App là ứng dụng kết hợp của cả Mobile Web App và Native App. Theo đó, các phần cơ bản của ứng dụng vẫn viết bằng ngôn ngữ web, nhưng được đặt trong native container, nên vẫn có thể đưa lên các App Store.

* Ưu điểm:
  + Chỉ cần có kiến thức về HTML, CSS, JavaScript...
  + Viết một lần dùng được nhiều nơi.
  + Tận dụng được các chức năng của hệ thống.
  + Có thể chạy được ở chế độ offline.
* Nhược điểm:
  + Hiệu năng chậm hơn so với Native App.
  + Không ổn định do khó debug. Framework sẽ dịch code thành code native, sửa lỗi khá khó khăn vì không biết được dịch như thế nào.

1.2.2.Giới thiệu Flutter

Flutter là một framework mã nguồn mở do Google phát triển, cho phép xây dựng giao diện người dùng cho ứng dụng di động, web và desktop chỉ với một cơ sở mã nguồn duy nhất. Flutter sử dụng ngôn ngữ lập trình Dart, được giới thiệu lần đầu vào năm 2017 và nhanh chóng trở thành một trong những công nghệ phát triển ứng dụng đa nền tảng phổ biến nhất hiện nay.

* Ưu điểm của Flutter:
* Phát triển nhanh với tính năng Hot Reload, giúp lập trình viên thấy ngay kết quả thay đổi mã nguồn mà không cần biên dịch lại.
* Hệ thống widget phong phú, dễ dàng tùy biến giao diện theo ý muốn.
* Hiệu năng cao gần tương đương với Native App do Flutter biên dịch trực tiếp thành mã máy.
* Hỗ trợ đa nền tảng, bao gồm Android, iOS, web và desktop với một cơ sở mã nguồn duy nhất.
* Cộng đồng phát triển mạnh mẽ, tài liệu đầy đủ, dễ tiếp cận cho người mới.
* Một số framework phổ biến trong Flutter:
* Flutter SDK: Cung cấp các công cụ cơ bản để xây dựng ứng dụng Flutter.
* Flutter UI: Cung cấp các widget cơ bản và Material Design, Cupertino cho iOS.
* Provider: Package quản lý trạng thái đơn giản, hiệu quả, được khuyến nghị bởi đội phát triển Flutter.
* Dio: Thư viện HTTP client mạnh mẽ cho việc kết nối API.
* Flutter Bloc/Provider/GetX: Các giải pháp quản lý trạng thái phổ biến trong Flutter.

1.2.3.Giới thiệu Node.js

Node.js là một nền tảng xây dựng ứng dụng phía máy chủ sử dụng ngôn ngữ JavaScript, được phát triển dựa trên V8 JavaScript Engine của Google Chrome. Node.js áp dụng mô hình lập trình bất đồng bộ, hướng sự kiện, giúp xây dựng các ứng dụng có khả năng mở rộng cao.

* Ưu điểm của Node.js:
* Hiệu năng cao, khả năng xử lý đồng thời nhiều kết nối.
* Sử dụng JavaScript cho cả frontend và backend, thuận tiện cho phát triển.
* Hệ sinh thái npm phong phú với hàng nghìn package hỗ trợ.
* Phù hợp cho các ứng dụng thời gian thực, xử lý dữ liệu lớn.
* Cộng đồng lớn, hỗ trợ mạnh mẽ.
* Một số framework và thư viện phổ biến trong Node.js:
* Express.js: Framework phổ biến nhất để xây dựng API RESTful.
* Mongoose: Thư viện ODM (Object Data Modeling) cho MongoDB.
* JWT (JSON Web Token): Giải pháp xác thực và phân quyền.
* Nodemailer: Thư viện gửi email từ ứng dụng Node.js.
* Socket.io: Thư viện hỗ trợ giao tiếp thời gian thực.

1.2.4.Giới thiệu MongoDB

MongoDB là hệ quản trị cơ sở dữ liệu NoSQL dạng document, lưu trữ dữ liệu dưới dạng các tài liệu BSON (mở rộng của JSON). MongoDB phù hợp với các ứng dụng có cấu trúc dữ liệu linh hoạt, khả năng mở rộng cao, đáp ứng tốt cho nhu cầu xử lý dữ liệu lớn, phân tán như trong thương mại điện tử.

* Đặc điểm của MongoDB:
  + Lưu trữ dữ liệu dưới dạng document, không cần định nghĩa schema cứng nhắc.
  + Hỗ trợ truy vấn phong phú, index đa dạng.
  + Khả năng mở rộng theo chiều ngang thông qua sharding.
  + Hiệu năng cao, đáp ứng tốt cho các ứng dụng có lượng truy cập lớn.
  + Tích hợp tốt với Node.js thông qua thư viện Mongoose.
* Ưu điểm của MongoDB trong ứng dụng thương mại điện tử:
  + Linh hoạt trong lưu trữ thông tin sản phẩm với các thuộc tính khác nhau.
  + Dễ dàng mở rộng khi số lượng người dùng và sản phẩm tăng lên.
  + Hỗ trợ tốt cho các truy vấn phức tạp như tìm kiếm, lọc, phân loại sản phẩm.
  + Hiệu năng tốt khi xử lý dữ liệu lớn như lịch sử đơn hàng, đánh giá sản phẩm.

1.2.5. Mô hình MVVM kết hợp Provider trong Flutter

Mô hình MVVM (Model - View - ViewModel) là một kiến trúc phổ biến trong phát triển ứng dụng, giúp tách biệt rõ ràng giữa giao diện, logic xử lý và dữ liệu. Khi kết hợp với Provider – một package quản lý trạng thái trong Flutter, MVVM giúp ứng dụng có cấu trúc rõ ràng, dễ bảo trì và mở rộng.

Model: Định nghĩa cấu trúc dữ liệu và các thao tác với dữ liệu. Trong ứng dụng thương mại điện tử, Model bao gồm các đối tượng như User, Product, Order, Cart...

View: Là các widget Flutter, chịu trách nhiệm hiển thị giao diện và nhận tương tác từ người dùng. View không chứa logic xử lý, chỉ hiển thị dữ liệu và gửi sự kiện tới ViewModel.

ViewModel: Là lớp trung gian, chứa logic xử lý, kết nối dữ liệu từ Model tới View và ngược lại. ViewModel không biết về View cụ thể nào đang sử dụng nó, giúp tăng tính tái sử dụng.

Provider: Là một giải pháp quản lý trạng thái trong Flutter, đóng vai trò cầu nối giữa ViewModel và View. Provider giúp chia sẻ dữ liệu và trạng thái giữa các widget, tự động rebuild UI khi dữ liệu thay đổi.

* Ưu điểm của MVVM kết hợp Provider:
  + Tách biệt rõ ràng các thành phần, dễ bảo trì, mở rộng.
  + Khả năng kiểm thử cao, có thể test ViewModel mà không cần View.
  + Tái sử dụng code tốt, một ViewModel có thể dùng cho nhiều View.
  + Provider giúp quản lý trạng thái hiệu quả, tránh rebuild không cần thiết.
  + Cấu trúc rõ ràng, dễ hiểu cho cả team phát triển.

Hình 1.1: Mô hình MVVM kết hợp Provider trong Flutter

1.3. Tổng quan về chức năng của hệ thống thương mại điện tử

Một hệ thống thương mại điện tử hiện đại cần đáp ứng đầy đủ các chức năng cơ bản sau:

* Quản lý người dùng:
  + Đăng ký tài khoản mới.
  + Đăng nhập, xác thực người dùng.
  + Quên mật khẩu, khôi phục thông tin đăng nhập.
  + Quản lý thông tin cá nhân, địa chỉ giao hàng.
* Quản lý sản phẩm:
  + Hiển thị danh sách sản phẩm theo danh mục.
  + Tìm kiếm, lọc sản phẩm theo nhiều tiêu chí.
  + Xem chi tiết sản phẩm, thông số kỹ thuật.
  + Thêm, sửa, xóa thông tin sản phẩm (đối với admin).
  + Quản lý tồn kho, giá bán.
* Quản lý đơn hàng:
  + Thêm sản phẩm vào giỏ hàng.
  + Cập nhật số lượng, xóa sản phẩm khỏi giỏ hàng.
  + Thanh toán với nhiều phương thức khác nhau.
  + Theo dõi trạng thái đơn hàng.
  + Xem lịch sử đơn hàng đã đặt.
  + Đánh giá và phản hồi:
  + Đánh giá sản phẩm sau khi mua.
* Thống kê và báo cáo:
  + Thống kê doanh thu theo ngày, tháng, năm.
  + Báo cáo sản phẩm bán chạy, tồn kho.
  + Phân tích hành vi người dùng, xu hướng mua sắm.
* Thông báo và liên lạc:
  + Gửi thông báo đơn hàng qua email.
  + Thông báo trạng thái đơn hàng, giao hàng.
  + Xuất hóa đơn, gửi hóa đơn qua email.

1.4. Lý do lựa chọn công nghệ và mô hình kiến trúc

Việc lựa chọn Flutter, Node.js, MongoDB và mô hình MVVM kết hợp Provider dựa trên các lý do sau:

* Lý do chọn Flutter:
  + Phát triển nhanh, tiết kiệm thời gian và chi phí khi xây dựng ứng dụng cho nhiều nền tảng.
  + Giao diện đẹp, linh hoạt với hệ thống widget phong phú.
  + Hiệu năng cao, gần tương đương với Native App.
  + Cộng đồng lớn, tài liệu đầy đủ, dễ tiếp cận.
* Lý do chọn Node.js:
  + Hiệu năng cao, khả năng xử lý đồng thời nhiều kết nối.
  + Sử dụng JavaScript, dễ dàng cho lập trình viên đã quen với frontend.
  + Hệ sinh thái npm phong phú, nhiều thư viện hỗ trợ.
  + Tích hợp tốt với MongoDB thông qua Mongoose.
* Lý do chọn MongoDB:
  + Linh hoạt trong lưu trữ dữ liệu, không cần schema cứng nhắc.
  + Dễ mở rộng khi lượng dữ liệu tăng.
  + Hiệu năng tốt với các truy vấn phức tạp.
  + Tích hợp tốt với Node.js.
* Lý do chọn MVVM kết hợp Provider:
  + Tách biệt rõ ràng các thành phần, dễ bảo trì, mở rộng.
  + Khả năng kiểm thử cao.
  + Provider là giải pháp quản lý trạng thái đơn giản, hiệu quả.
  + Cấu trúc rõ ràng, dễ hiểu cho team phát triển.

CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

2.1. Khảo sát người dùng

Để đảm bảo ứng dụng thương mại điện tử đáp ứng đúng nhu cầu của người dùng, em đã tiến hành khảo sát người dùng thông qua phiếu khảo sát trực tuyến. Kết quả khảo sát từ 50 người dùng tiềm năng đã cung cấp những thông tin quan trọng về nhu cầu và mong muốn của người dùng đối với ứng dụng thương mại điện tử trên nền tảng di động.

* Câu 1: Bạn thường sử dụng ứng dụng thương mại điện tử nào trên điện thoại?

Trả lời:

* + Shopee: 82%
  + Lazada: 45%
  + Tiki: 62%
  + Sendo: 30%
  + Khác: 15%
* Câu 2: Bạn gặp khó khăn gì khi sử dụng các ứng dụng thương mại điện tử hiện tại?

Trả lời:

* + Giao diện phức tạp, khó tìm sản phẩm: 45%
  + Thông tin sản phẩm không đầy đủ: 58%
  + Quá trình thanh toán phức tạp: 36%
  + Thông báo quá nhiều, gây phiền nhiễu: 52%
  + Tốc độ ứng dụng chậm: 40%
  + Khó khăn trong việc theo dõi đơn hàng: 32%
* Câu 3: Tính năng nào bạn cho là quan trọng nhất trong một ứng dụng thương mại điện tử?

Trả lời:

* + Tìm kiếm sản phẩm nhanh chóng, chính xác: 75%
  + Thanh toán đơn giản, nhiều phương thức: 68%
  + Theo dõi đơn hàng dễ dàng: 60%
  + Đánh giá sản phẩm từ người dùng khác: 52%
  + Khuyến mãi, giảm giá hấp dẫn: 65%
  + Giao diện đơn giản, dễ sử dụng: 70%
* Câu 4: Bạn thường gặp vấn đề gì trong quá trình thanh toán trên ứng dụng thương mại điện tử?

Trả lời:

* + Quá nhiều bước xác nhận: 48%
  + Không hỗ trợ phương thức thanh toán mong muốn: 35%
  + Lỗi kỹ thuật trong quá trình thanh toán: 42%
  + Thông tin thanh toán không rõ ràng: 38%
  + Không gặp vấn đề gì: 20%
* Câu 5: Bạn mong muốn ứng dụng thương mại điện tử cải thiện điều gì?

Trả lời:

* + Giao diện đơn giản, trực quan hơn: 65%
  + Tốc độ ứng dụng nhanh hơn: 58%
  + Thông báo thông minh, không gây phiền nhiễu: 50%
  + Quy trình thanh toán đơn giản hơn: 62%
  + Hệ thống đánh giá sản phẩm minh bạch hơn: 48%
  + Dịch vụ khách hàng tốt hơn: 55%
* Câu 6: Bạn thích nhận thông báo về đơn hàng qua hình thức nào?

Trả lời:

* + Thông báo trên ứng dụng: 70%
  + Email: 45%
  + SMS: 30%
  + Cả ba hình thức: 25%

2.2. Giới thiệu về hệ thống

Dựa trên kết quả khảo sát và phân tích nhu cầu thực tế, em đề xuất xây dựng ứng dụng thương mại điện tử trên nền tảng di động với tên gọi "ShopEase". Ứng dụng được phát triển bằng Flutter cho phần giao diện người dùng, Node.js cho backend và MongoDB cho cơ sở dữ liệu, nhằm tạo ra một hệ thống hoàn chỉnh, hiệu quả và đáp ứng tốt nhu cầu của người dùng.

ShopEase sẽ cung cấp các chức năng cơ bản của một ứng dụng thương mại điện tử hiện đại, bao gồm: đăng ký, đăng nhập, quên mật khẩu, tìm kiếm sản phẩm, mua hàng, thanh toán, quản lý thông tin sản phẩm, quản lý khách hàng, thống kê doanh thu, xuất hóa đơn, gửi thông tin đơn hàng qua email và đánh giá sản phẩm. Ứng dụng hướng đến sự đơn giản, trực quan trong giao diện người dùng, kết hợp với hiệu năng cao và độ tin cậy trong xử lý dữ liệu.

2.3. Phân tích yêu cầu

2.3.1. Về hệ thống

Ứng dụng ShopEase hướng tới một hệ thống có giao diện đẹp, dễ sử dụng, tốc độ nhanh và đảm bảo trải nghiệm tốt nhất cho người dùng. Cụ thể, hệ thống cần đáp ứng các yêu cầu sau:

* Giao diện: Thiết kế giao diện đơn giản, trực quan, thân thiện với người dùng, tuân thủ các nguyên tắc Material Design, dễ dàng điều hướng.
* Hiệu năng: Tối ưu hóa hiệu suất ứng dụng, đảm bảo thời gian phản hồi nhanh, tải trang mượt mà, không bị giật lag khi sử dụng.
* Bảo mật: Đảm bảo an toàn thông tin người dùng, mã hóa dữ liệu nhạy cảm, xác thực người dùng an toàn.
* Khả năng mở rộng: Thiết kế hệ thống có khả năng mở rộng dễ dàng, thêm tính năng mới mà không ảnh hưởng đến các chức năng hiện có.
* Độ tin cậy: Hệ thống hoạt động ổn định, xử lý lỗi hiệu quả, sao lưu dữ liệu định kỳ.

2.3.2. Về người sử dụng

Ứng dụng ShopEase phục vụ hai nhóm người dùng chính:

* Khách hàng: Người dùng cuối sử dụng ứng dụng để tìm kiếm, mua sắm sản phẩm. Họ cần một giao diện đơn giản, dễ sử dụng, tìm kiếm nhanh chóng, thanh toán đơn giản, theo dõi đơn hàng dễ dàng.
* Quản trị viên: Người quản lý hệ thống, cần các công cụ để quản lý sản phẩm, khách hàng, đơn hàng, thống kê doanh thu.

2.3.3. Yêu cầu về chức năng

* Các chức năng chính của hệ thống:
  + Quản lý tài khoản người dùng:
    - Đăng ký tài khoản mới
    - Đăng nhập hệ thống
    - Quên mật khẩu
    - Quản lý thông tin cá nhân
  + Quản lý sản phẩm:
    - Hiển thị danh sách sản phẩm
    - Tìm kiếm sản phẩm
    - Xem chi tiết sản phẩm
    - Thêm, sửa, xóa sản phẩm (Admin)
    - Quản lý danh mục sản phẩm
  + Mua hàng và thanh toán:
    - Thêm sản phẩm vào giỏ hàng
    - Quản lý giỏ hàng
    - Đặt hàng
    - Thanh toán đơn hàng
  + Quản lý đơn hàng:
    - Xem danh sách đơn hàng
    - Xem chi tiết đơn hàng
    - Cập nhật trạng thái đơn hàng (Admin)
    - Xuất hóa đơn
    - Gửi thông tin đơn hàng qua email
  + Quản lý khách hàng:
    - Xem danh sách khách hàng (Admin)
    - Xem chi tiết thông tin khách hàng (Admin)
    - Thống kê hành vi mua hàng (Admin)
  + Đánh giá sản phẩm:
* Đánh giá, nhận xét sản phẩm
  + Thống kê doanh thu:
    - Thống kê doanh thu theo thời gian

2.3.4. Yêu cầu phi chức năng:

* Giao diện đồ họa:
  + Giao diện đẹp, hiện đại, phù hợp với xu hướng thiết kế.
  + Bố cục hợp lý, dễ sử dụng trên các kích thước màn hình khác nhau.
  + Hỗ trợ chế độ sáng/tối (light/dark mode).
* Hoạt động của hệ thống:
  + Thời gian phản hồi nhanh, dưới 2 giây cho các thao tác thông thường.
  + Khả năng xử lý đồng thời nhiều người dùng.
  + Hoạt động ổn định, ít lỗi.
* Ngôn ngữ:
  + Hỗ trợ tiếng Anh.
* Độ tin cậy của ứng dụng:
  + Tỷ lệ lỗi thấp, dưới 1%.
  + Khả năng phục hồi sau lỗi nhanh chóng.
  + Sao lưu dữ liệu định kỳ.
* Hiệu năng của ứng dụng:
  + Tải trang nhanh, dưới 3 giây.
  + Sử dụng tài nguyên hợp lý, không gây lag trên thiết bị cấu hình trung bình.
  + Tối ưu hóa sử dụng dữ liệu mạng.
* Bảo mật:
  + Mã hóa thông tin nhạy cảm.
  + Xác thực người dùng an toàn.
  + Tuân thủ các quy định về bảo vệ dữ liệu cá nhân.

2.4. Thiết kế hệ thống

2.4.1. Mô hình hóa Usecase

2.4.1.1. Xác định các tác nhân

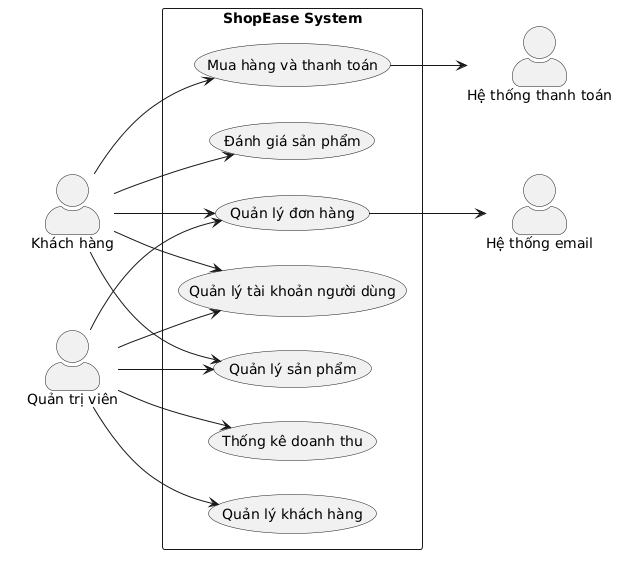
* Khách hàng: Người dùng đã đăng ký tài khoản, có thể mua sắm, đánh giá sản phẩm, quản lý thông tin cá nhân.
* Quản trị viên: Người quản lý hệ thống, có quyền quản lý sản phẩm, khách hàng, đơn hàng, thống kê doanh thu.
* Hệ thống email: Hệ thống gửi email thông báo, xác nhận đơn hàng, hóa đơn.
* Hệ thống thanh toán: Cổng thanh toán tích hợp với ứng dụng.

2.4.1.2. Xác định các Usecase

* Quản lý tài khoản người dùng:
  + Đăng ký tài khoản
  + Đăng nhập
  + Quên mật khẩu
  + Quản lý thông tin cá nhân
* Quản lý sản phẩm:
  + Xem danh sách sản phẩm
  + Tìm kiếm sản phẩm
  + Xem chi tiết sản phẩm
  + Thêm sản phẩm mới
  + Cập nhật thông tin sản phẩm
  + Xóa sản phẩm
  + Quản lý danh mục sản phẩm
* Mua hàng và thanh toán:
  + Thêm vào giỏ hàng
  + Quản lý giỏ hàng
  + Đặt hàng
  + Thanh toán
* Quản lý đơn hàng:
  + Xem danh sách đơn hàng
  + Xem chi tiết đơn hàng
  + Cập nhật trạng thái đơn hàng
  + Xuất hóa đơn
  + Gửi thông tin đơn hàng qua email
* Quản lý khách hàng:
  + Xem danh sách khách hàng
  + Xem chi tiết khách hàng
* Đánh giá sản phẩm:
  + Đánh giá sản phẩm
* Thống kê doanh thu:
  + Thống kê doanh thu theo thời gian

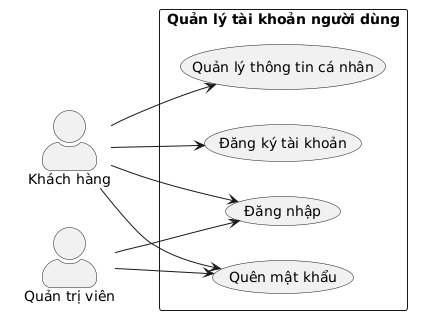
2.4.1.3. Biểu đồ Usecase

* **Biểu đồ Usecase chính**



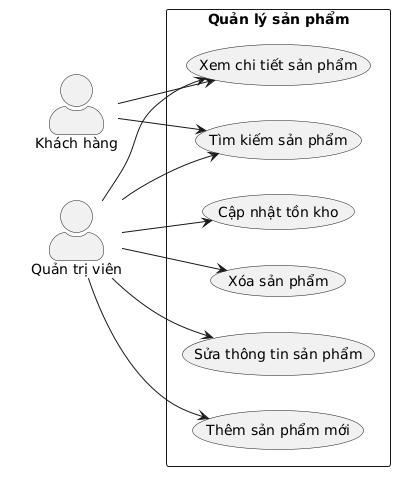
Hình 2.1: Biểu đồ Usecase chính

* **Biểu đồ Usecase thứ cấp**
  + Phân rã Usecase <Quản lý tài khoản người dùng>



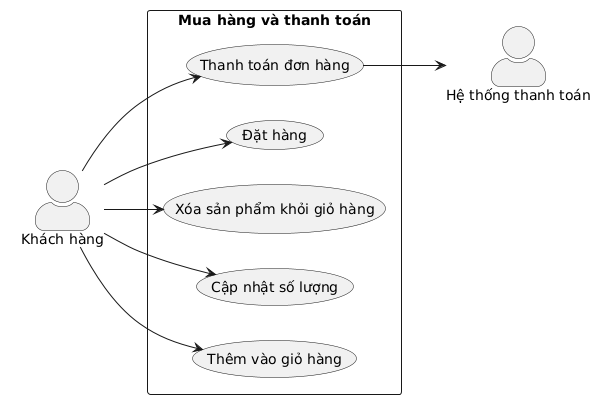
Hình 2.2: Usecase Quản lý tài khoản người dùng

* + Phân rã Usecase <Quản lý sản phẩm>



Hình 2.3: Usecase Quản lý sản phẩm

* + Phân rã Usecase <Mua hàng và thanh toán>



Hình 2.4: Usecase Mua hàng và thanh toán

* + Phân rã Usecase <Quản lý đơn hàng>



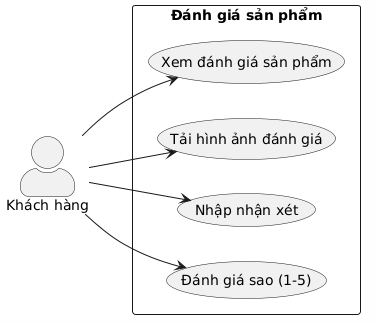
Hình 2.5: Usecase Quản lý đơn hàng

* + Phân rã Usecase <Quản lý khách hàng>



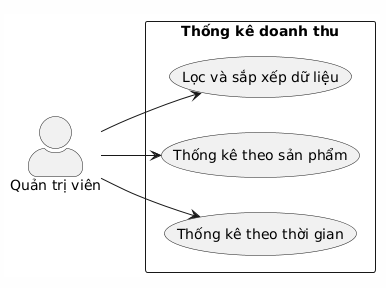
Hình 2.6: Usecase Quản lý khách hàng

* + Phân rã Usecase <Đánh giá sản phẩm>



Hình 2.7: Usecase Đánh giá sản phẩm

* + Phân rã Usecase <Thống kê doanh thu>



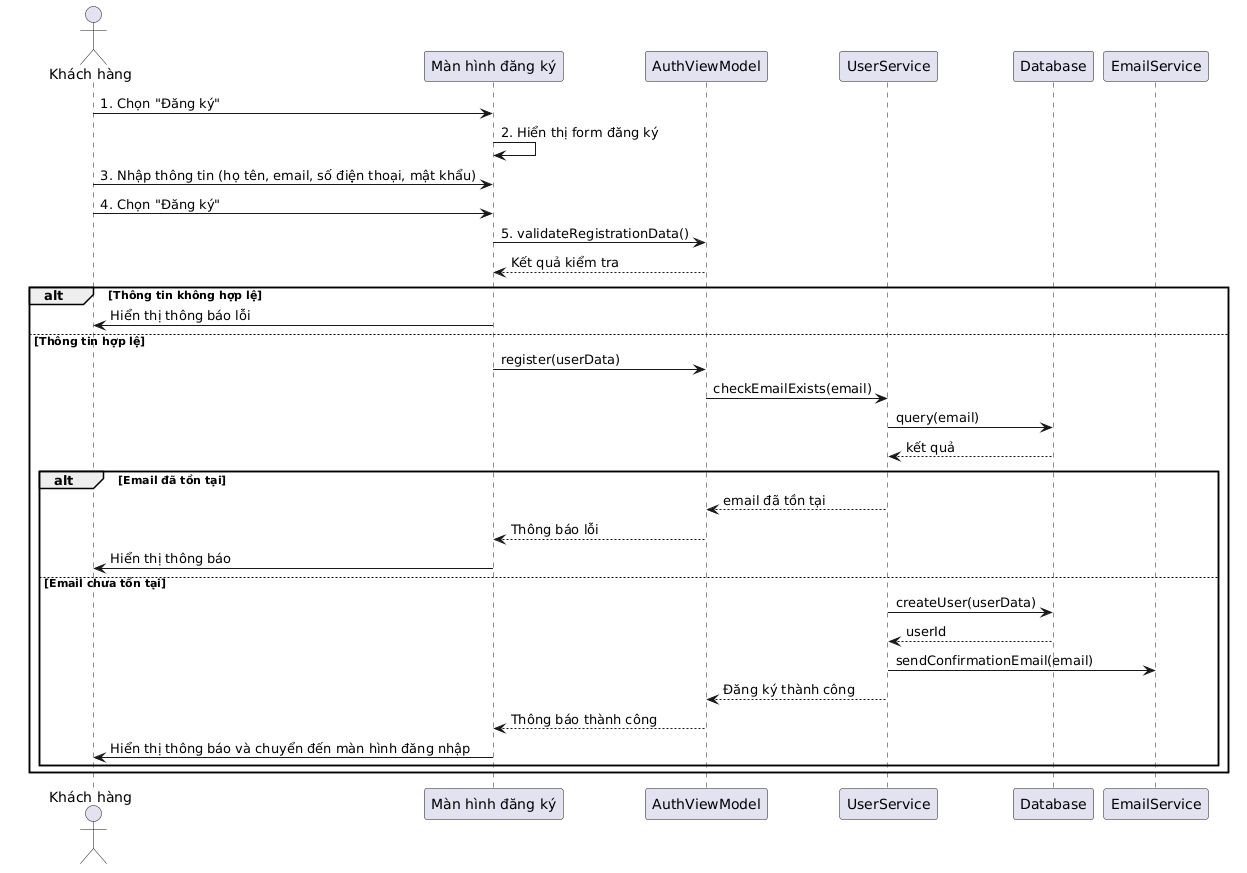
Hình 2.8: Usecase Thống kê doanh thu

2.4.2. Mô tả chi tiết các Usecase

2.4.2.1. Mô tả Usecase <Đăng ký tài khoản>

|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép người dùng đăng ký tài khoản mới để sử dụng ứng dụng |
| Tác nhân | | Khách hàng |
| Tiền điều kiện | | Người dùng chưa có tài khoản trong hệ thống |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi người dùng chọn "Đăng ký" trên màn hình đăng nhập.  2. Hệ thống hiển thị form đăng ký với các trường: Họ tên, Email, Số điện thoại, Mật khẩu, Xác nhận mật khẩu.  3. Người dùng nhập thông tin vào form.  4. Người dùng chọn "Đăng ký".  5. Hệ thống kiểm tra tính hợp lệ của thông tin:  - Kiểm tra định dạng email  - Kiểm tra định dạng số điện thoại  - Kiểm tra độ mạnh của mật khẩu  - Kiểm tra mật khẩu và xác nhận mật khẩu trùng khớp  6. Hệ thống kiểm tra email và số điện thoại đã tồn tại trong hệ thống chưa.  7. Hệ thống tạo tài khoản mới và lưu vào cơ sở dữ liệu.  8. Hệ thống gửi email xác nhận đến email đăng ký.  9. Hệ thống hiển thị thông báo đăng ký thành công và chuyển đến màn hình đăng nhập.  10. Usecase kết thúc. |
| Luồng rẽ nhánh | Luồng A1: Thông tin không hợp lệ  - Ở bước 5, nếu thông tin không hợp lệ, hệ thống hiển thị thông báo lỗi tương ứng và yêu cầu người dùng nhập lại.  Luồng A2: Email hoặc số điện thoại đã tồn tại  - Ở bước 6, nếu email hoặc số điện thoại đã tồn tại, hệ thống hiển thị thông báo và yêu cầu người dùng sử dụng email/số điện thoại khác.  Luồng A3: Người dùng hủy đăng ký  - Ở bất kỳ bước nào, người dùng có thể chọn "Hủy" để quay lại màn hình đăng nhập. |
| Hậu điều kiện | | Tài khoản người dùng mới được tạo và lưu trữ trong cơ sở dữ liệu |

* Biểu đồ trình tự:

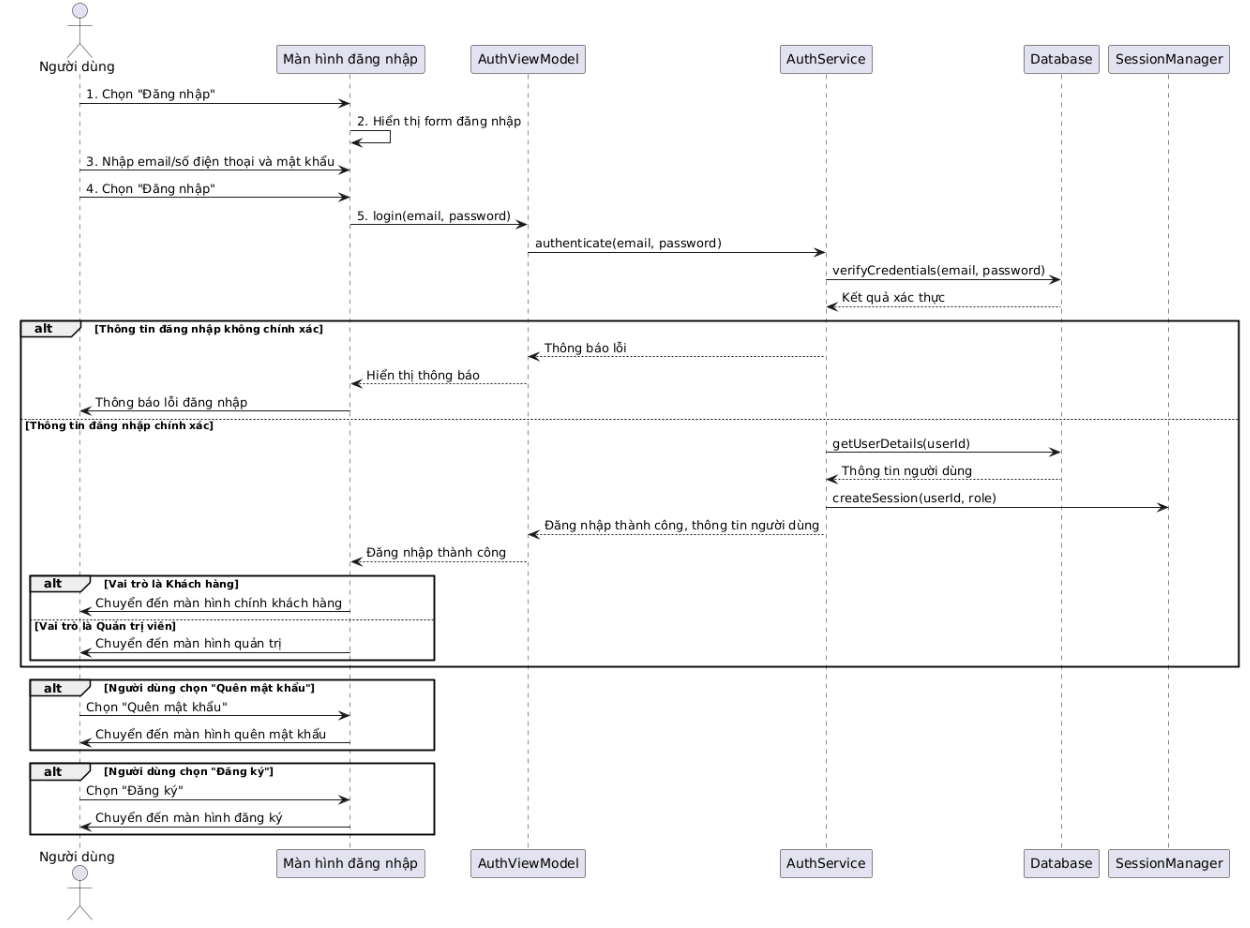


Hình 2.9: Biểu đồ trình tự Đăng ký tài khoản

2.4.2.2. Mô tả Usecase <Đăng nhập>

|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép người dùng đăng nhập vào hệ thống để sử dụng các chức năng của ứng dụng. |
| Tác nhân | | Khách hàng, Quản trị viên |
| Tiền điều kiện | | Người dùng đã có tài khoản trong hệ thống. |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi người dùng chọn "Đăng nhập" trên màn hình chính.  2. Hệ thống hiển thị form đăng nhập với các trường: Email/Số điện thoại, Mật khẩu.  3. Người dùng nhập thông tin vào form.  4. Người dùng chọn "Đăng nhập".  5. Hệ thống kiểm tra tính hợp lệ của thông tin:  - Kiểm tra định dạng email hoặc số điện thoại  - Kiểm tra mật khẩu  6. Hệ thống xác thực thông tin đăng nhập với cơ sở dữ liệu.  7. Nếu thông tin hợp lệ, hệ thống tạo phiên đăng nhập và lưu token xác thực.  8. Hệ thống chuyển đến màn hình chính phù hợp với vai trò (Khách hàng hoặc Quản trị viên).  9. Usecase kết thúc. |
| Luồng rẽ nhánh | Luồng A1: Thông tin đăng nhập không chính xác  - Ở bước 6, nếu thông tin đăng nhập không chính xác, hệ thống hiển thị thông báo lỗi và yêu cầu người dùng nhập lại.  Luồng A2: Quên mật khẩu  - Ở bước 3, người dùng có thể chọn "Quên mật khẩu" để thực hiện usecase Quên mật khẩu.  Luồng A3: Đăng ký tài khoản mới  - Ở bước 3, người dùng có thể chọn "Đăng ký" để thực hiện usecase Đăng ký tài khoản. |
| Hậu điều kiện | | Người dùng được đăng nhập vào hệ thống và có thể sử dụng các chức năng tương ứng với vai trò. |

* Biểu đồ trình tự:

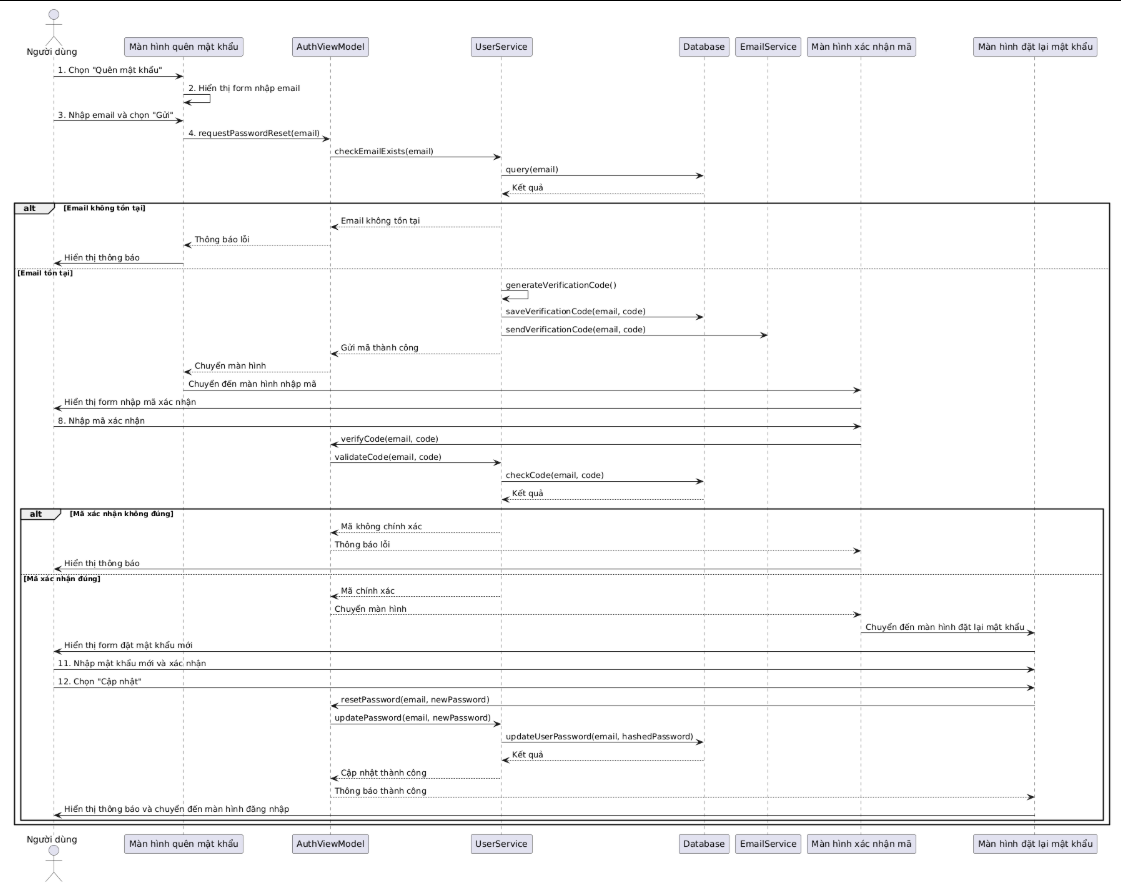


Hình 2.10: Biểu đồ trình tự Đăng nhập

2.4.2.3. Mô tả Usecase <Quên mật khẩu>

|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép người dùng khôi phục mật khẩu khi quên thông tin đăng nhập. |
| Tác nhân | | Khách hàng, Quản trị viên |
| Tiền điều kiện | | Người dùng đã có tài khoản trong hệ thống. |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi người dùng chọn "Quên mật khẩu" trên màn hình đăng nhập.  2. Hệ thống hiển thị form yêu cầu nhập email đã đăng ký.  3. Người dùng nhập email và chọn "Gửi".  4. Hệ thống kiểm tra email có tồn tại trong hệ thống không.  5. Hệ thống tạo mã xác nhận và lưu vào cơ sở dữ liệu.  6. Hệ thống gửi email chứa mã xác nhận đến email người dùng.  7. Hệ thống hiển thị form nhập mã xác nhận.  8. Người dùng nhập mã xác nhận và chọn "Xác nhận".  9. Hệ thống kiểm tra mã xác nhận.  10. Hệ thống hiển thị form đặt lại mật khẩu.  11. Người dùng nhập mật khẩu mới, xác nhận mật khẩu và chọn "Cập nhật".  12. Hệ thống kiểm tra mật khẩu hợp lệ và trùng khớp.  13. Hệ thống cập nhật mật khẩu mới vào cơ sở dữ liệu.  14. Hệ thống hiển thị thông báo thành công và chuyển đến màn hình đăng nhập.  15. Usecase kết thúc. |
| Luồng rẽ nhánh | Luồng A1: Email không tồn tại  - Ở bước 4, nếu email không tồn tại, hệ thống hiển thị thông báo lỗi và yêu cầu người dùng nhập lại.  Luồng A2: Mã xác nhận không chính xác  - Ở bước 9, nếu mã xác nhận không chính xác, hệ thống hiển thị thông báo lỗi và yêu cầu người dùng nhập lại.  Luồng A3: Mật khẩu không hợp lệ  - Ở bước 12, nếu mật khẩu không hợp lệ hoặc không trùng khớp, hệ thống hiển thị thông báo lỗi và yêu cầu người dùng nhập lại. |
| Hậu điều kiện | | Mật khẩu của người dùng đã được cập nhật. |

* Biểu đồ trình tự:

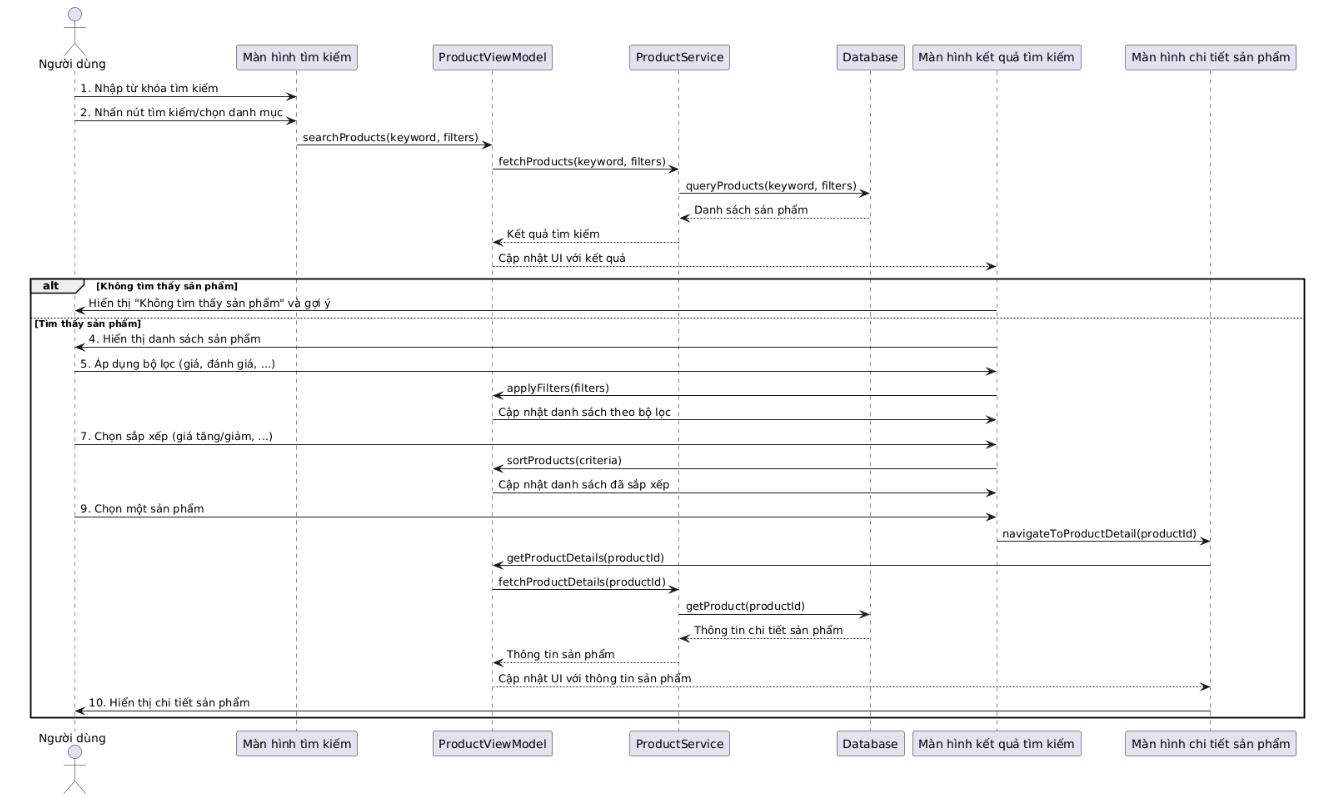


Hình 2.11: Biểu đồ trình tự Quên mật khẩu

2.4.2.4. Mô tả Usecase <Tìm kiếm sản phẩm>

|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép người dùng tìm kiếm sản phẩm theo từ khóa, danh mục hoặc các bộ lọc. |
| Tác nhân | | Khách hàng, Quản trị viên |
| Tiền điều kiện | | Người dùng đã đăng nhập vào hệ thống. |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi người dùng nhập từ khóa vào ô tìm kiếm hoặc chọn danh mục sản phẩm.  2. Người dùng nhấn nút tìm kiếm hoặc chọn một danh mục.  3. Hệ thống thực hiện tìm kiếm trong cơ sở dữ liệu dựa trên từ khóa hoặc danh mục.  4. Hệ thống hiển thị danh sách sản phẩm phù hợp với từ khóa hoặc danh mục.  5. Người dùng có thể áp dụng các bộ lọc (giá, đánh giá, thương hiệu...).  6. Hệ thống cập nhật danh sách sản phẩm theo bộ lọc.  7. Người dùng có thể chọn sắp xếp kết quả (giá tăng/giảm, đánh giá cao nhất...).  8. Hệ thống sắp xếp danh sách sản phẩm theo tiêu chí đã chọn.  9. Người dùng chọn một sản phẩm để xem chi tiết.  10. Hệ thống hiển thị trang chi tiết sản phẩm.  11. Usecase kết thúc. |
| Luồng rẽ nhánh | Luồng A1: Không tìm thấy sản phẩm  - Ở bước 4, nếu không có sản phẩm nào phù hợp với từ khóa hoặc bộ lọc, hệ thống hiển thị thông báo "Không tìm thấy sản phẩm" và gợi ý các từ khóa liên quan.  Luồng A2: Lỗi khi tìm kiếm  - Ở bước 3, nếu xảy ra lỗi khi tìm kiếm, hệ thống hiển thị thông báo lỗi và đề xuất người dùng thử lại sau. |
| Hậu điều kiện | | Danh sách sản phẩm phù hợp với từ khóa hoặc bộ lọc được hiển thị. |

* Biểu đồ trình tự:

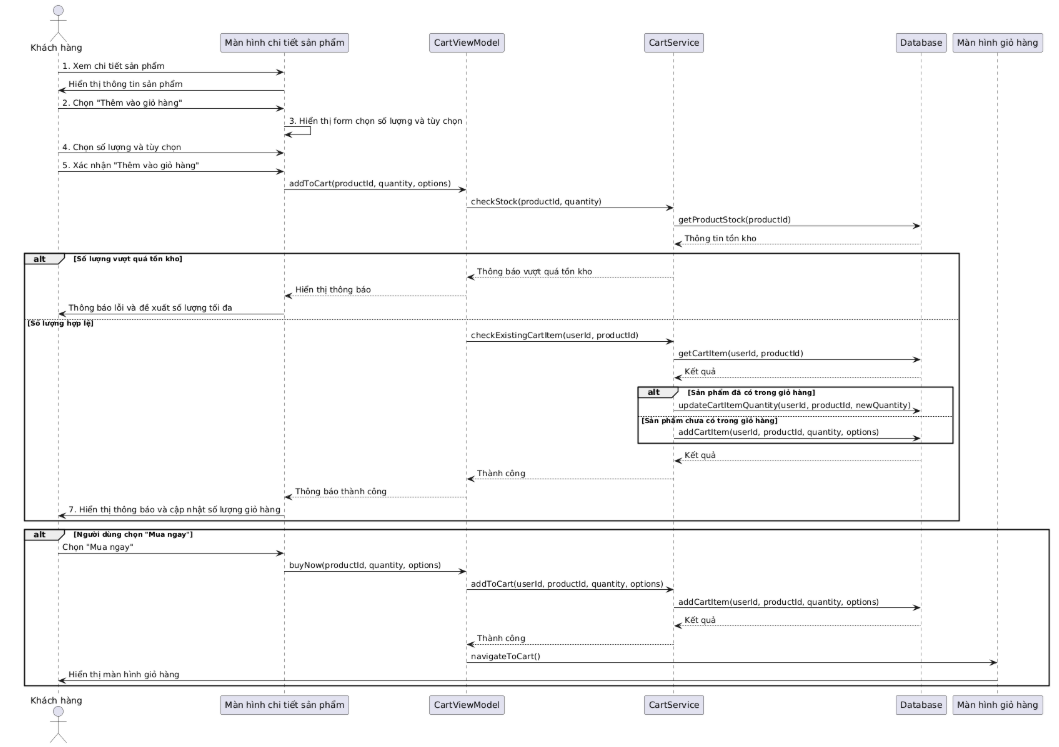


Hình 2.12: Biểu đồ trình tự Tìm kiếm sản phẩm

2.4.2.5. Mô tả Usecase <Thêm vào giỏ hàng>

|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép người dùng thêm sản phẩm vào giỏ hàng. |
| Tác nhân | | Khách hàng |
| Tiền điều kiện | | Người dùng đã đăng nhập vào hệ thống và đang xem thông tin sản phẩm. |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi người dùng đang xem chi tiết sản phẩm và chọn "Thêm vào giỏ hàng".  2. Hệ thống hiển thị form chọn số lượng và các tùy chọn sản phẩm (nếu có).  3. Người dùng chọn số lượng và các tùy chọn sản phẩm.  4. Người dùng xác nhận "Thêm vào giỏ hàng".  5. Hệ thống kiểm tra số lượng tồn kho của sản phẩm.  6. Hệ thống thêm sản phẩm vào giỏ hàng của người dùng.  7. Hệ thống hiển thị thông báo thành công và cập nhật số lượng sản phẩm trong giỏ hàng.  8. Usecase kết thúc. |
| Luồng rẽ nhánh | Luồng A1: Sản phẩm đã có trong giỏ hàng  - Ở bước 6, nếu sản phẩm đã có trong giỏ hàng, hệ thống cập nhật số lượng sản phẩm.  Luồng A2: Số lượng vượt quá tồn kho  - Ở bước 5, nếu số lượng người dùng chọn vượt quá tồn kho, hệ thống hiển thị thông báo và đề xuất số lượng tối đa có thể thêm.  Luồng A3: Người dùng chọn "Mua ngay"  - Ở bước 1, người dùng có thể chọn "Mua ngay" thay vì "Thêm vào giỏ hàng", hệ thống sẽ thêm sản phẩm vào giỏ hàng và chuyển đến trang thanh toán. |
| Hậu điều kiện | | Sản phẩm được thêm vào giỏ hàng của người dùng. |

* Biểu đồ trình tự:

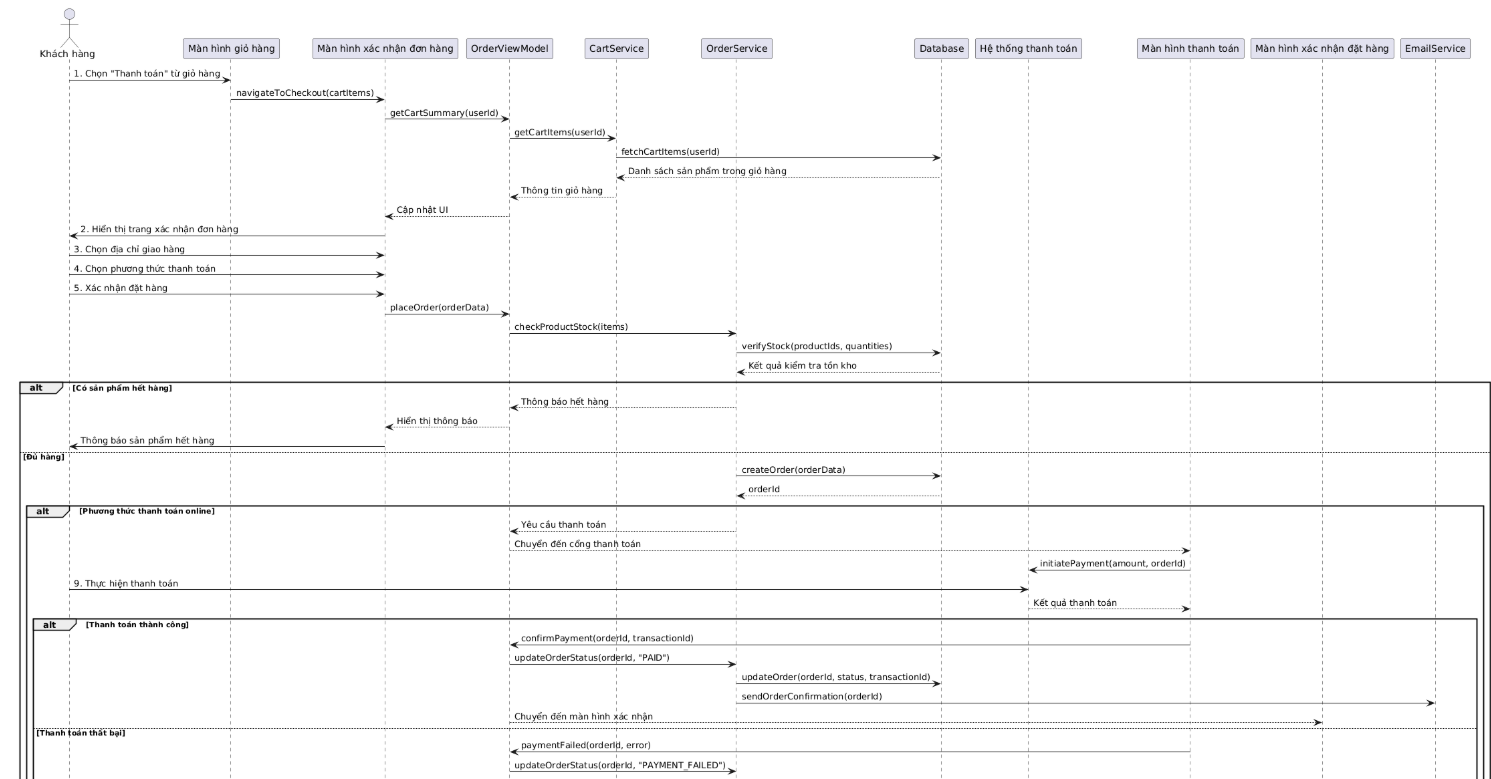


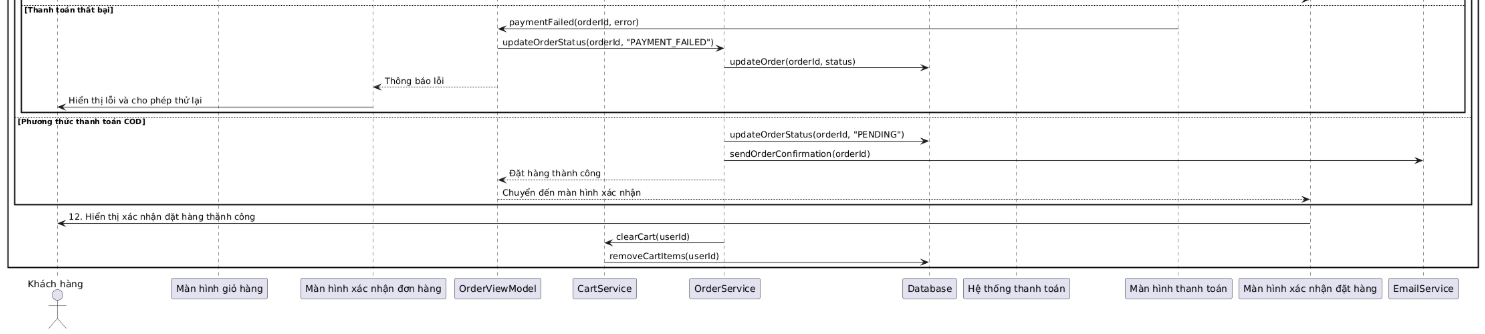
Hình 2.13: Biểu đồ trình tự Thêm vào giỏ hàng

2.4.2.6. Mô tả Usecase <Thanh toán đơn hàng>

|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép người dùng thanh toán đơn hàng đã đặt. |
| Tác nhân | | Khách hàng, Hệ thống thanh toán |
| Tiền điều kiện | | Người dùng đã đăng nhập và có sản phẩm trong giỏ hàng. |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi người dùng chọn "Thanh toán" từ giỏ hàng.  2. Hệ thống hiển thị trang xác nhận đơn hàng với danh sách sản phẩm, số lượng, giá tiền.  3. Người dùng kiểm tra thông tin đơn hàng và chọn địa chỉ giao hàng.  4. Người dùng chọn phương thức thanh toán (COD, thẻ tín dụng, ví điện tử...).  5. Người dùng xác nhận đặt hàng.  6. Hệ thống kiểm tra lại tồn kho của sản phẩm.  7. Hệ thống tạo đơn hàng mới và lưu vào cơ sở dữ liệu.  8. Nếu phương thức thanh toán là online, hệ thống chuyển hướng người dùng đến cổng thanh toán.  9. Người dùng hoàn tất thanh toán trên cổng thanh toán.  10. Hệ thống nhận kết quả thanh toán từ cổng thanh toán.  11. Hệ thống cập nhật trạng thái đơn hàng và gửi email xác nhận đơn hàng.  12. Hệ thống hiển thị trang xác nhận đặt hàng thành công.  13. Usecase kết thúc. |
| Luồng rẽ nhánh | Luồng A1: Sản phẩm hết hàng  - Ở bước 6, nếu có sản phẩm hết hàng, hệ thống thông báo cho người dùng và đề xuất xóa sản phẩm khỏi đơn hàng hoặc chờ nhập hàng.  Luồng A2: Thanh toán không thành công  - Ở bước 10, nếu thanh toán không thành công, hệ thống thông báo lỗi và cho phép người dùng thử lại hoặc chọn phương thức thanh toán khác.  Luồng A3: Thanh toán COD  - Ở bước 8, nếu phương thức thanh toán là COD, hệ thống bỏ qua bước 8-10 và chuyển đến bước 11. |
| Hậu điều kiện | | Đơn hàng được tạo và cập nhật trạng thái thanh toán. |

* Biểu đồ trình tự:



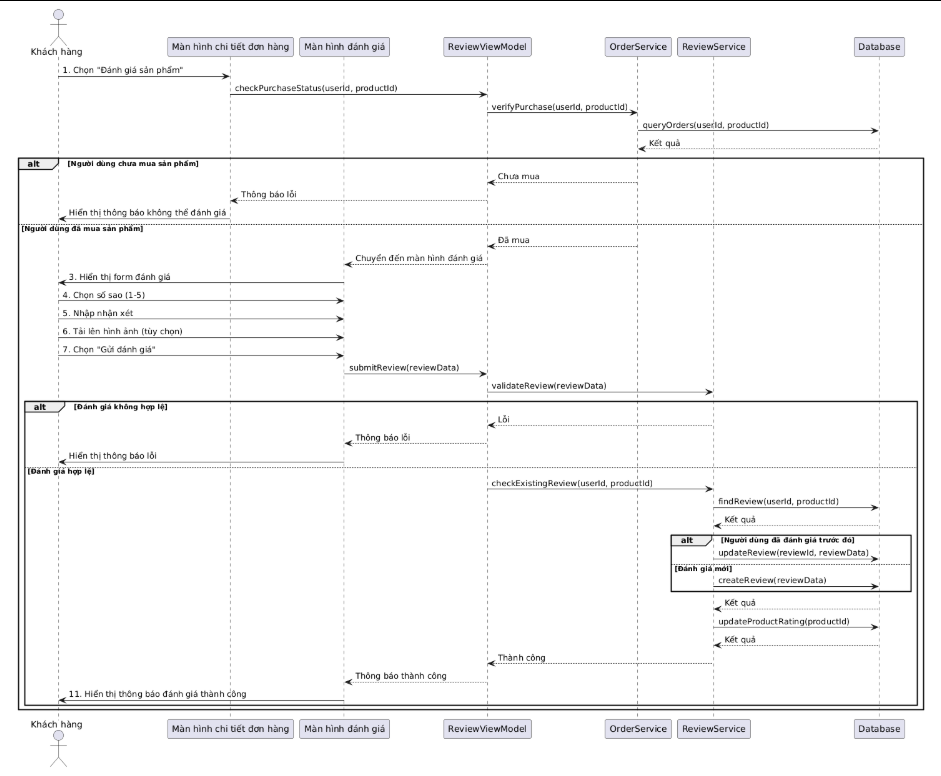


Hình 2.14: Biểu đồ trình tự Thanh toán đơn hàng

2.4.2.7. Mô tả Usecase <Đánh giá sản phẩm>

|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép người dùng đánh giá và nhận xét về sản phẩm đã mua. |
| Tác nhân | | Khách hàng |
| Tiền điều kiện | | Người dùng đã đăng nhập và đã mua sản phẩm. |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi người dùng chọn "Đánh giá sản phẩm" từ chi tiết đơn hàng hoặc trang chi tiết sản phẩm.  2. Hệ thống kiểm tra xem người dùng đã mua sản phẩm này chưa.  3. Hệ thống hiển thị form đánh giá với các trường: Số sao (1-5), Nhận xét, Hình ảnh (tùy chọn).  4. Người dùng chọn số sao đánh giá.  5. Người dùng nhập nhận xét về sản phẩm.  6. Người dùng có thể tải lên hình ảnh sản phẩm (tùy chọn).  7. Người dùng chọn "Gửi đánh giá".  8. Hệ thống kiểm tra tính hợp lệ của đánh giá.  9. Hệ thống lưu đánh giá vào cơ sở dữ liệu.  10. Hệ thống cập nhật điểm đánh giá trung bình của sản phẩm.  11. Hệ thống hiển thị thông báo đánh giá thành công.  12. Usecase kết thúc |
| Luồng rẽ nhánh | Luồng A1: Người dùng chưa mua sản phẩm  - Ở bước 2, nếu người dùng chưa mua sản phẩm, hệ thống hiển thị thông báo và không cho phép đánh giá.  Luồng A2: Đánh giá không hợp lệ  - Ở bước 8, nếu đánh giá không hợp lệ (ví dụ: nhận xét quá ngắn), hệ thống hiển thị thông báo và yêu cầu người dùng chỉnh sửa.  Luồng A3: Người dùng đã đánh giá sản phẩm trước đó  - Ở bước 2, nếu người dùng đã đánh giá sản phẩm trước đó, hệ thống hiển thị đánh giá cũ và cho phép người dùng chỉnh sửa |
| Hậu điều kiện | | Đánh giá của người dùng được lưu và hiển thị trên trang sản phẩm |

* Biểu đồ trình tự:

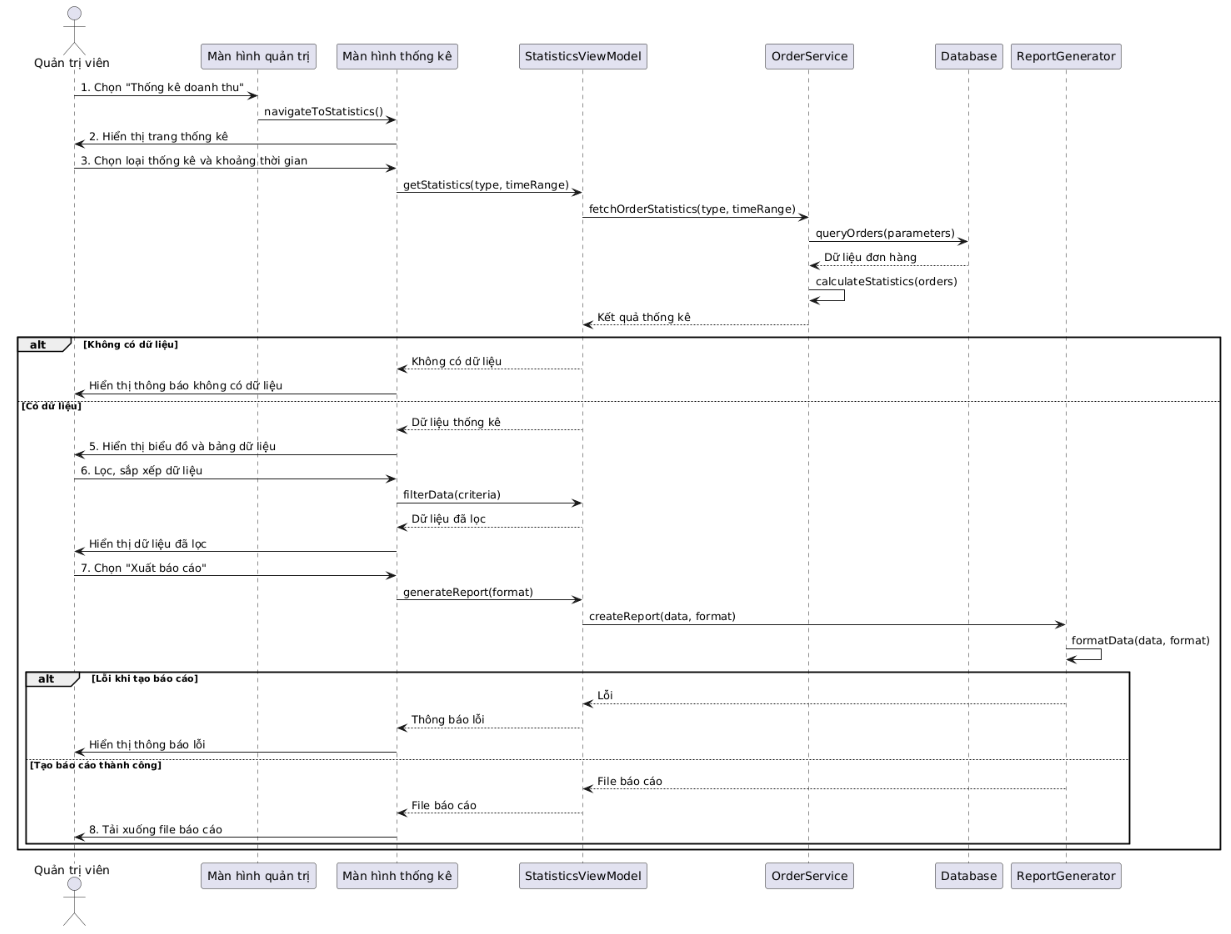


Hình 2.15: Biểu đồ trình tự Đánh giá sản phẩm

2.4.2.8. Mô tả Usecase <Thống kê doanh thu>

|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép quản trị viên xem thống kê doanh thu theo thời gian hoặc sản phẩm. |
| Tác nhân | | Quản trị viên |
| Tiền điều kiện | | Quản trị viên đã đăng nhập vào hệ thống. |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi quản trị viên chọn mục "Thống kê doanh thu" từ trang quản trị.  2. Hệ thống hiển thị trang thống kê với các tùy chọn: Thống kê theo thời gian, Thống kê theo sản phẩm.  3. Quản trị viên chọn loại thống kê và khoảng thời gian (ngày, tuần, tháng, năm).  4. Hệ thống truy vấn dữ liệu từ cơ sở dữ liệu theo các tham số đã chọn.  5. Hệ thống hiển thị kết quả thống kê dưới dạng biểu đồ và bảng dữ liệu.  6. Quản trị viên có thể lọc, sắp xếp dữ liệu theo các tiêu chí khác nhau.  7. Quản trị viên có thể xuất báo cáo dưới dạng PDF hoặc Excel.  8. Hệ thống tạo và tải xuống file báo cáo.  9. Usecase kết thúc. |
| Luồng rẽ nhánh | Luồng A1: Không có dữ liệu  - Ở bước 5, nếu không có dữ liệu thống kê trong khoảng thời gian đã chọn, hệ thống hiển thị thông báo và đề xuất chọn khoảng thời gian khác.  Luồng A2: Lỗi khi xuất báo cáo  - Ở bước 8, nếu xảy ra lỗi khi tạo hoặc tải xuống báo cáo, hệ thống hiển thị thông báo lỗi và đề xuất thử lại sau. |
| Hậu điều kiện | | Dữ liệu thống kê được hiển thị và có thể được xuất ra file. |

* Biểu đồ trình tự:

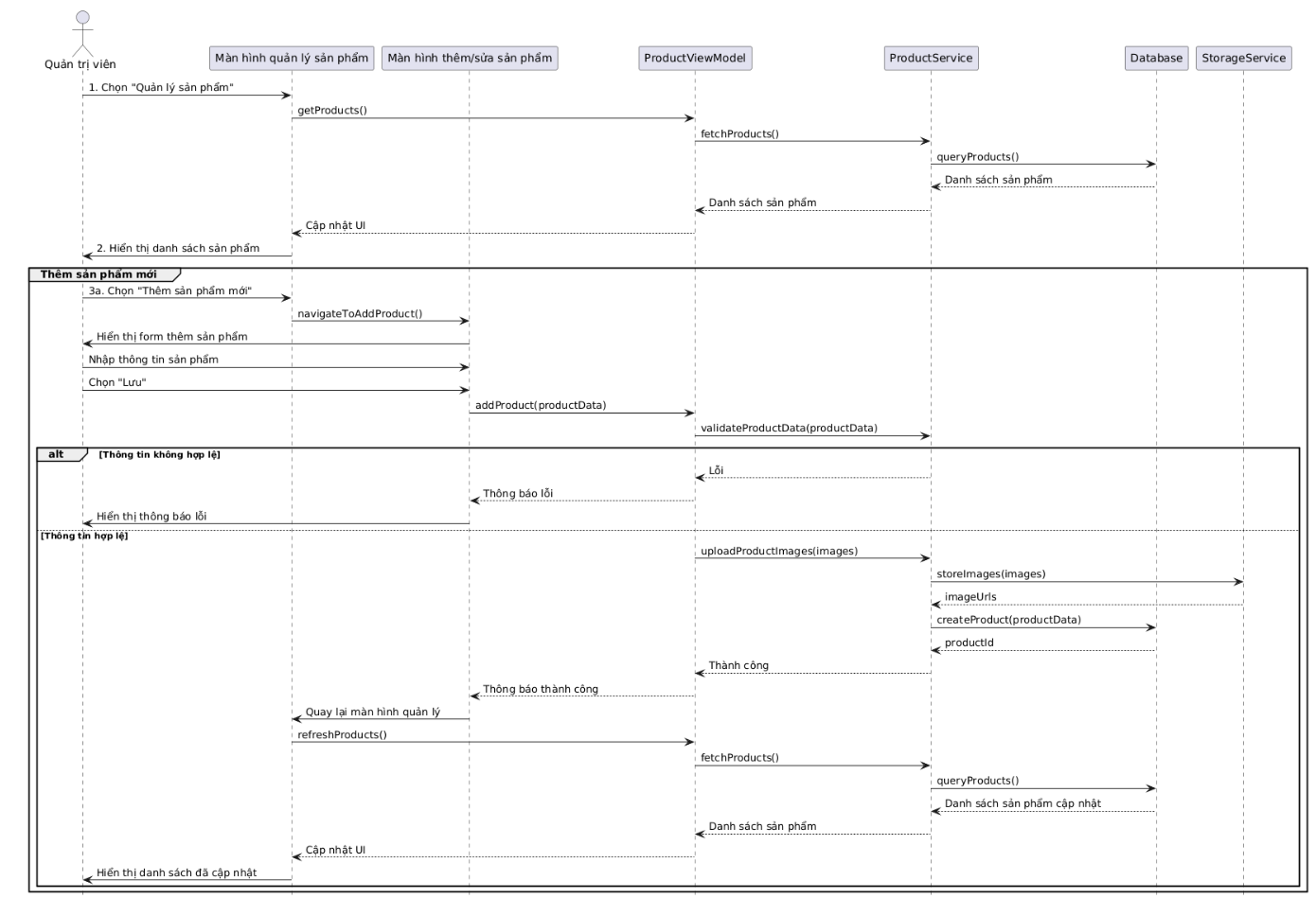


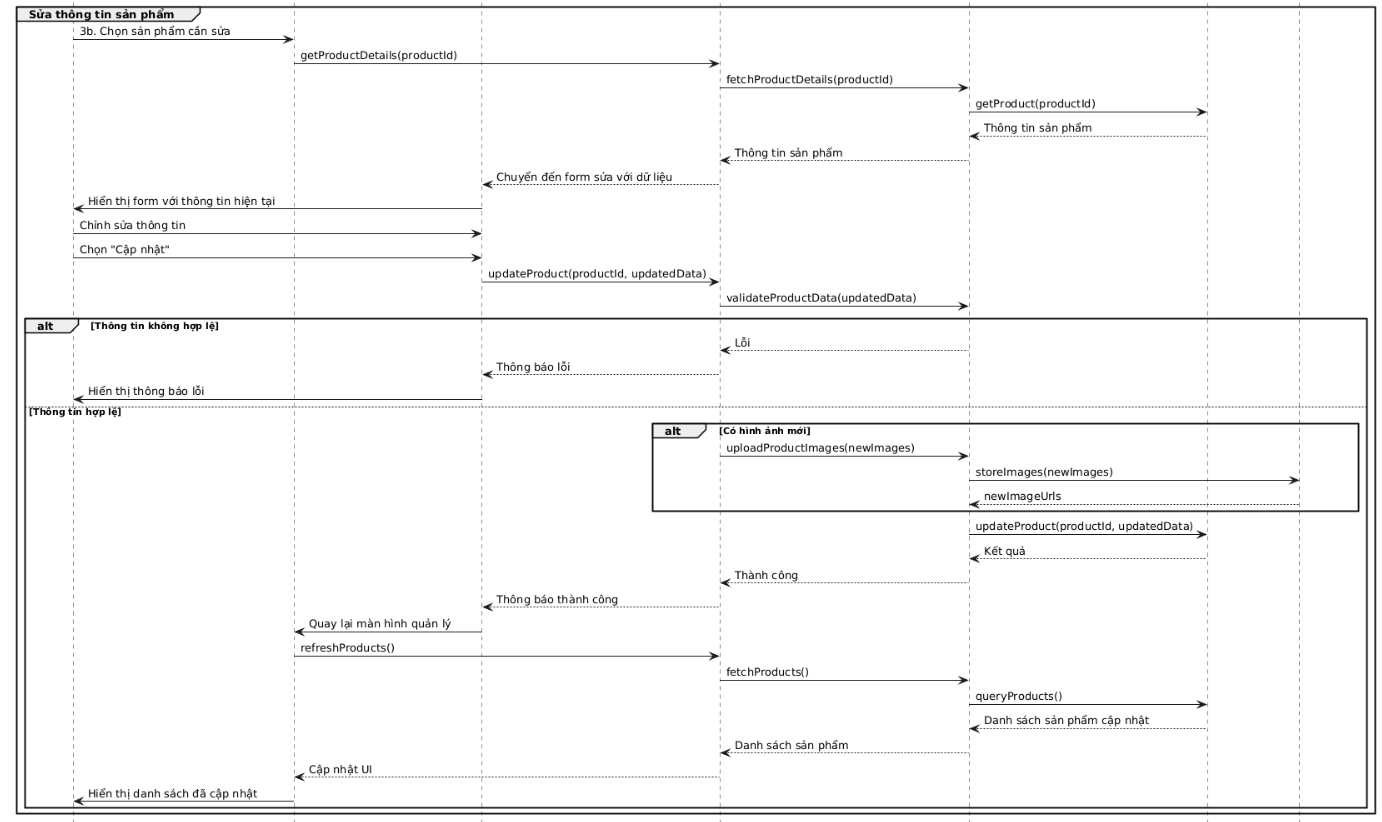
Hình 2.16: Biểu đồ trình tự Thống kê doanh thu

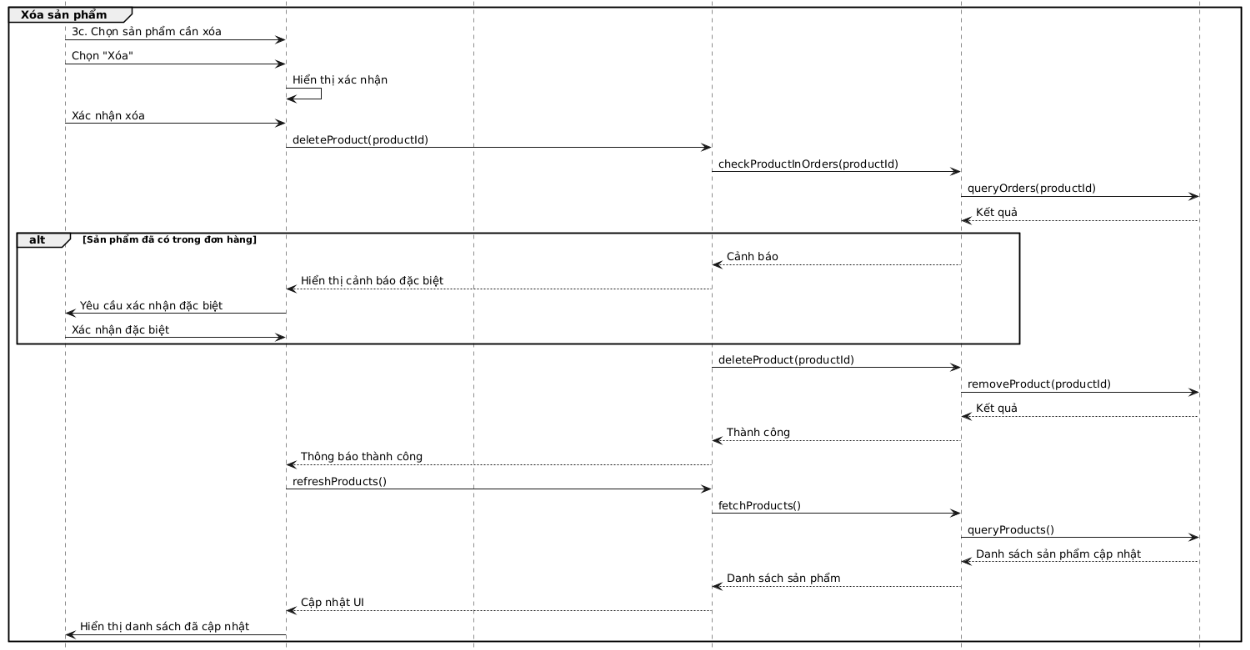
2.4.2.9. Mô tả Usecase <Quản lý thông tin sản phẩm>

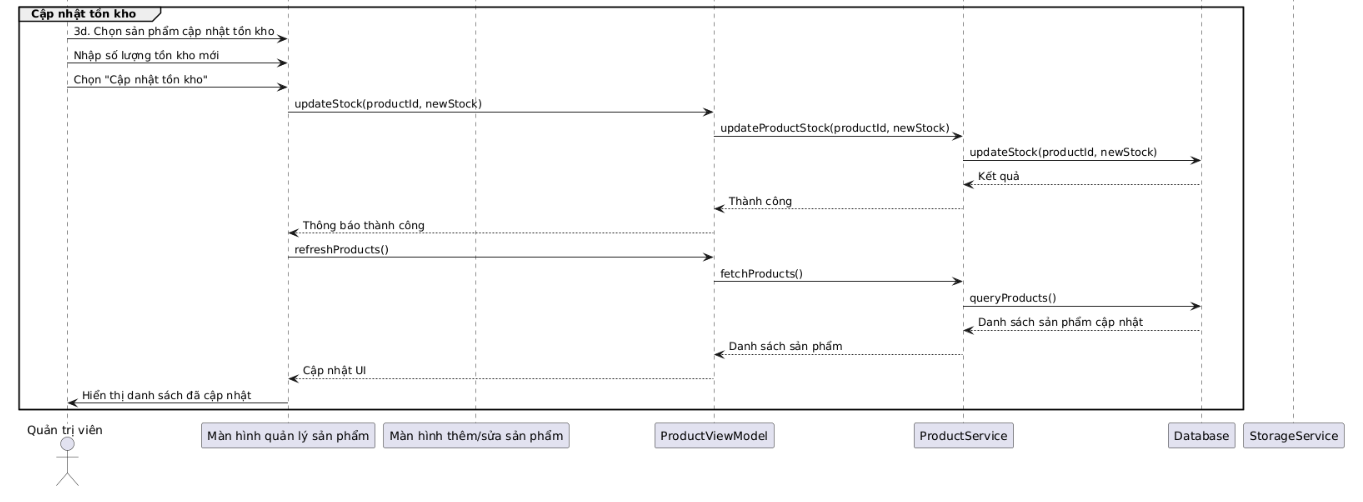
|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép quản trị viên thêm, sửa, xóa và quản lý thông tin sản phẩm trong hệ thống. |
| Tác nhân | | Quản trị viên |
| Tiền điều kiện | | Quản trị viên đã đăng nhập vào hệ thống với quyền quản lý sản phẩm. |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi quản trị viên chọn mục "Quản lý sản phẩm" từ trang quản trị.  2. Hệ thống hiển thị danh sách sản phẩm với các thông tin: hình ảnh, tên, giá, tồn kho, trạng thái.  3. Quản trị viên có thể thực hiện các chức năng:  a. Thêm sản phẩm mới:  - Quản trị viên chọn "Thêm sản phẩm mới".  - Hệ thống hiển thị form nhập thông tin sản phẩm.  - Quản trị viên nhập thông tin sản phẩm: tên, mô tả, giá, tồn kho, danh mục, hình ảnh.  - Quản trị viên chọn "Lưu".  - Hệ thống kiểm tra tính hợp lệ của thông tin.  - Hệ thống lưu thông tin sản phẩm mới vào cơ sở dữ liệu.  b. Sửa thông tin sản phẩm:  - Quản trị viên chọn sản phẩm cần sửa.  - Hệ thống hiển thị form với thông tin hiện tại của sản phẩm.  - Quản trị viên chỉnh sửa thông tin và chọn "Cập nhật".  - Hệ thống kiểm tra tính hợp lệ của thông tin.  - Hệ thống cập nhật thông tin sản phẩm vào cơ sở dữ liệu.  c. Xóa sản phẩm:  - Quản trị viên chọn sản phẩm cần xóa.  - Quản trị viên chọn "Xóa".  - Hệ thống hiển thị thông báo xác nhận.  - Quản trị viên xác nhận xóa.  - Hệ thống xóa sản phẩm khỏi cơ sở dữ liệu.  d. Cập nhật tồn kho:  - Quản trị viên chọn sản phẩm cần cập nhật tồn kho.  - Quản trị viên nhập số lượng tồn kho mới.  - Quản trị viên chọn "Cập nhật tồn kho".  - Hệ thống cập nhật tồn kho trong cơ sở dữ liệu.  4. Usecase kết thúc. |
| Luồng rẽ nhánh | Luồng A1: Thông tin không hợp lệ  - Khi thêm hoặc sửa sản phẩm, nếu thông tin không hợp lệ (ví dụ: giá âm, tên trống), hệ thống hiển thị thông báo lỗi và yêu cầu nhập lại.  Luồng A2: Xóa sản phẩm đã có trong đơn hàng  - Khi xóa sản phẩm đã có trong đơn hàng, hệ thống hiển thị cảnh báo và yêu cầu xác nhận đặc biệt.  Luồng A3: Tải hình ảnh thất bại  - Khi tải hình ảnh sản phẩm thất bại, hệ thống hiển thị thông báo lỗi và cho phép thử lại. |
| Hậu điều kiện | | Thông tin sản phẩm được cập nhật trong cơ sở dữ liệu. |

* Biểu đồ trình tự:







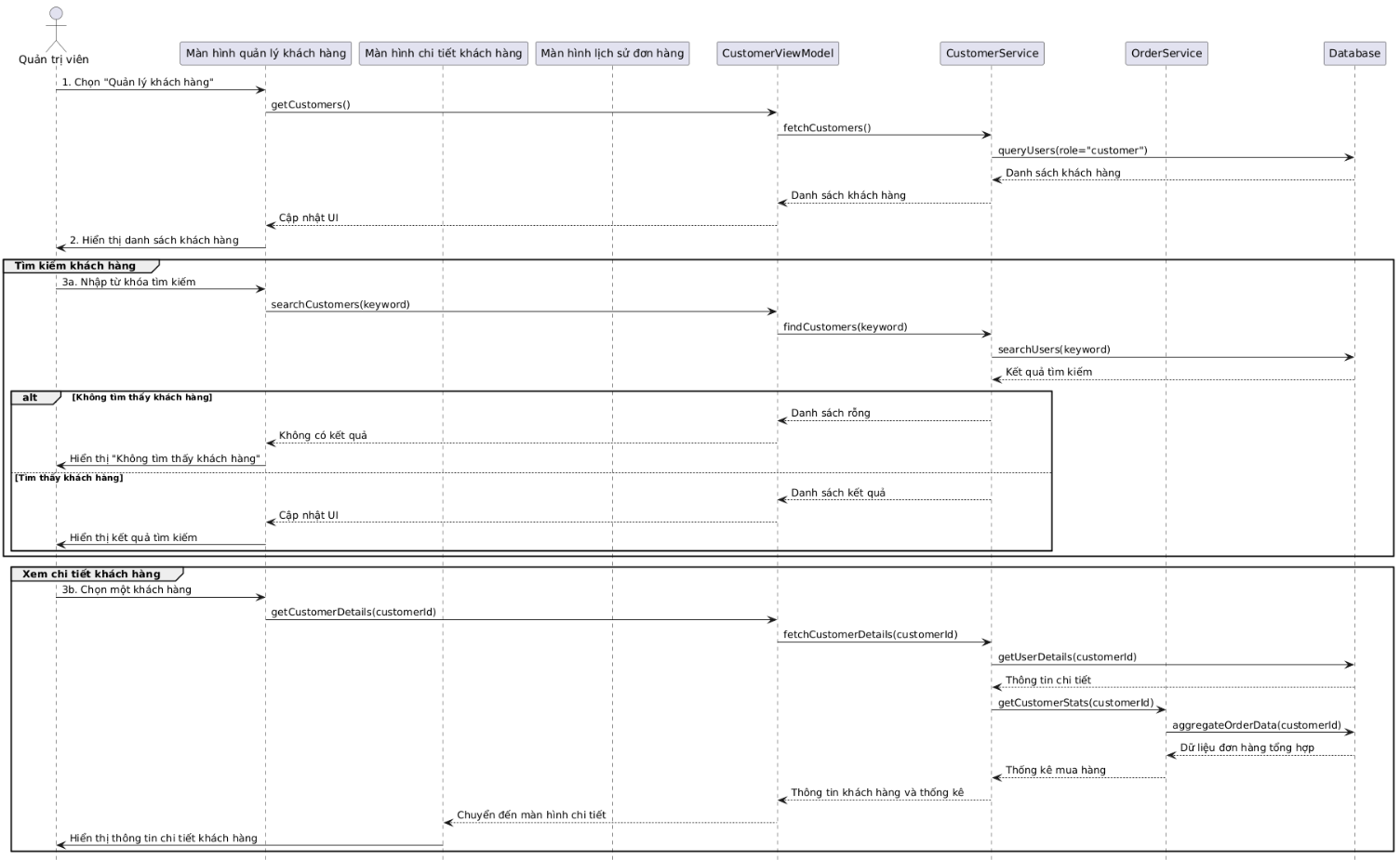


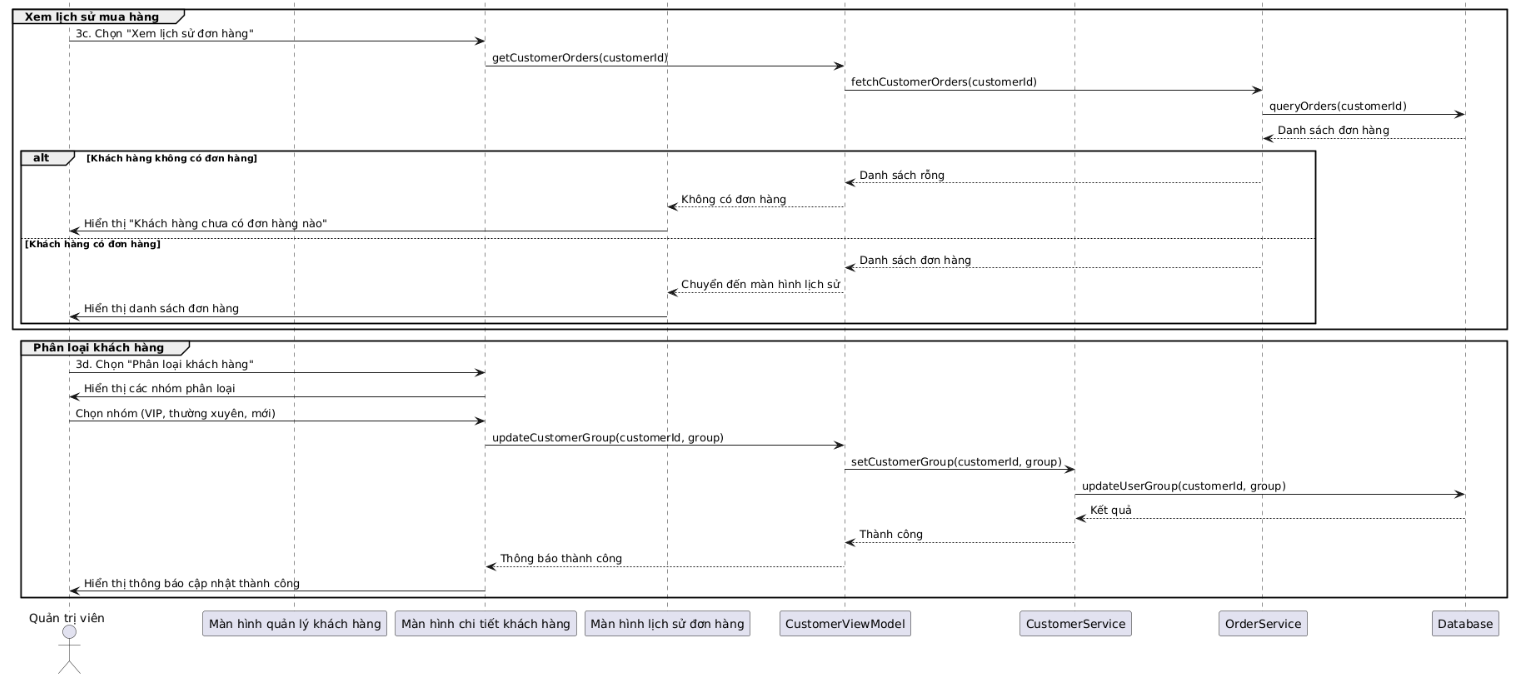
Hình 2.34: Biểu đồ trình tự Quản lý thông tin sản phẩm

2.4.2.10. Mô tả Usecase <Quản lý khách hàng>

|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép quản trị viên xem và quản lý thông tin khách hàng trong hệ thống. |
| Tác nhân | | Quản trị viên |
| Tiền điều kiện | | Quản trị viên đã đăng nhập vào hệ thống với quyền quản lý khách hàng. |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi quản trị viên chọn mục "Quản lý khách hàng" từ trang quản trị.  2. Hệ thống hiển thị danh sách khách hàng với các thông tin: tên, email, số điện thoại, ngày đăng ký.  3. Quản trị viên có thể thực hiện các chức năng:  a. Tìm kiếm khách hàng:  - Quản trị viên nhập từ khóa tìm kiếm (tên, email, số điện thoại).  - Hệ thống hiển thị kết quả tìm kiếm.  b. Xem chi tiết khách hàng:  - Quản trị viên chọn một khách hàng từ danh sách.  - Hệ thống hiển thị thông tin chi tiết của khách hàng: thông tin cá nhân, địa chỉ, lịch sử đơn hàng, tổng chi tiêu.  c. Xem lịch sử mua hàng của khách hàng:  - Quản trị viên chọn "Xem lịch sử đơn hàng" từ trang chi tiết khách hàng.  - Hệ thống hiển thị danh sách đơn hàng của khách hàng.  d. Phân loại khách hàng:  - Quản trị viên chọn "Phân loại khách hàng".  - Quản trị viên chọn nhóm khách hàng (VIP, thường xuyên, mới).  - Hệ thống cập nhật nhóm khách hàng trong cơ sở dữ liệu.  4. Usecase kết thúc. |
| Luồng rẽ nhánh | Luồng A1: Không tìm thấy khách hàng  - Khi tìm kiếm, nếu không có khách hàng nào phù hợp với từ khóa, hệ thống hiển thị thông báo "Không tìm thấy khách hàng".  Luồng A2: Khách hàng không có đơn hàng  - Khi xem lịch sử đơn hàng, nếu khách hàng chưa có đơn hàng nào, hệ thống hiển thị thông báo "Khách hàng chưa có đơn hàng nào". |
| Hậu điều kiện | | Thông tin khách hàng được hiển thị hoặc cập nhật trong cơ sở dữ liệu. |

* Biểu đồ trình tự:





Hình 2.35: Biểu đồ trình tự Quản lý khách hàng

2.4.2.11. Mô tả Usecase <Xuất hóa đơn>

|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép quản trị viên hoặc khách hàng xuất hóa đơn cho đơn hàng. |
| Tác nhân | | Quản trị viên, Khách hàng |
| Tiền điều kiện | | Đơn hàng đã được tạo và thanh toán. |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi người dùng chọn "Xuất hóa đơn" từ trang chi tiết đơn hàng.  2. Hệ thống truy xuất thông tin đơn hàng từ cơ sở dữ liệu.  3. Hệ thống tạo hóa đơn với các thông tin: mã đơn hàng, ngày đặt, thông tin khách hàng, danh sách sản phẩm, giá tiền, thuế, tổng cộng.  4. Hệ thống hiển thị xem trước hóa đơn.  5. Người dùng có thể chọn các tùy chọn:  a. Tải xuống dưới dạng PDF:  - Người dùng chọn "Tải xuống PDF".  - Hệ thống tạo file PDF và tải xuống.  b. In hóa đơn:  - Người dùng chọn "In hóa đơn".  - Hệ thống mở hộp thoại in.  - Người dùng xác nhận in.  c. Gửi qua email:  - Người dùng chọn "Gửi qua email".  - Hệ thống gửi hóa đơn qua email đến địa chỉ email của khách hàng.  6. Hệ thống cập nhật trạng thái "Đã xuất hóa đơn" cho đơn hàng.  7. Usecase kết thúc. |
| Luồng rẽ nhánh | Luồng A1: Không thể tạo hóa đơn  - Ở bước 3, nếu không thể tạo hóa đơn do lỗi hệ thống, hệ thống hiển thị thông báo lỗi và đề xuất thử lại sau.  Luồng A2: Gửi email thất bại  - Ở bước 5c, nếu gửi email thất bại, hệ thống hiển thị thông báo lỗi và đề xuất kiểm tra lại địa chỉ email hoặc thử lại sau. |
| Hậu điều kiện | | Hóa đơn được tạo và có thể được tải xuống, in hoặc gửi qua email. |

* Biểu đồ trình tự:



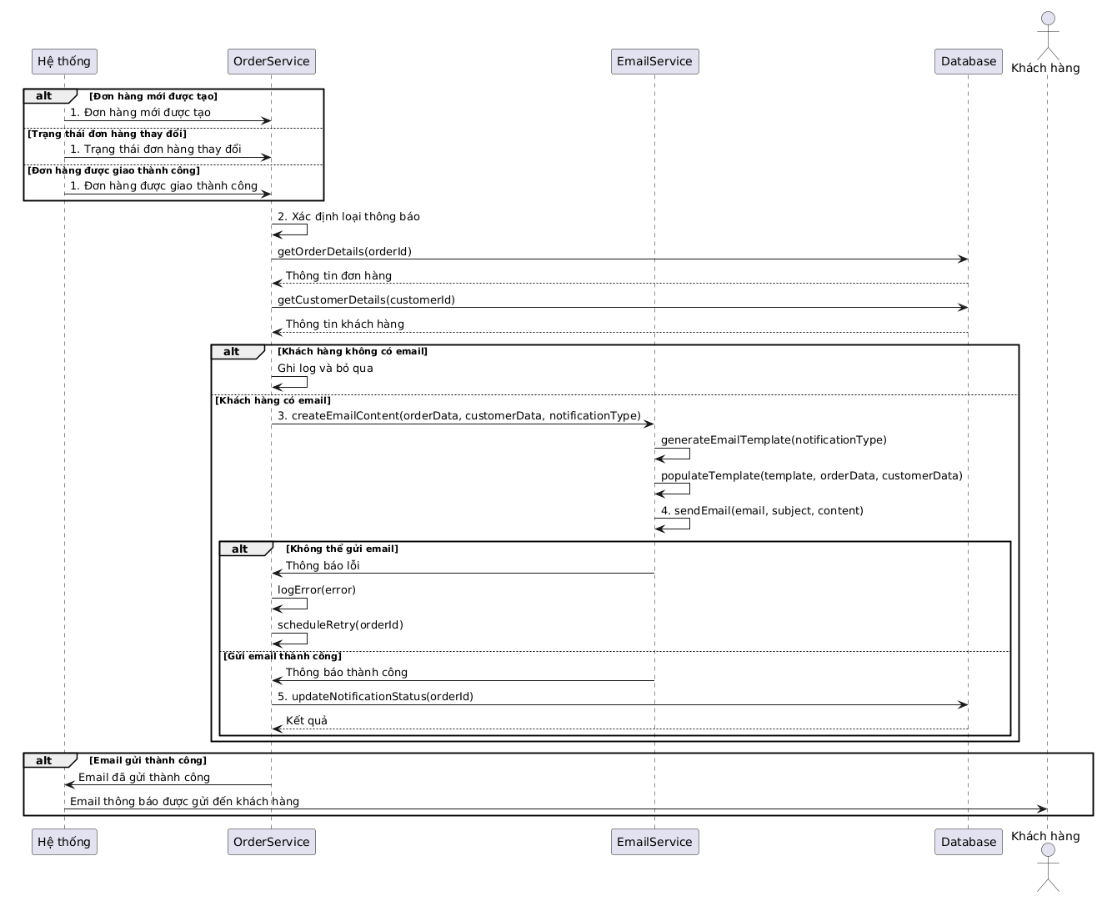


Hình 2.36: Biểu đồ trình tự Xuất hóa đơn

2.4.2.12. Mô tả Usecase <Gửi thông tin đơn hàng qua Email>

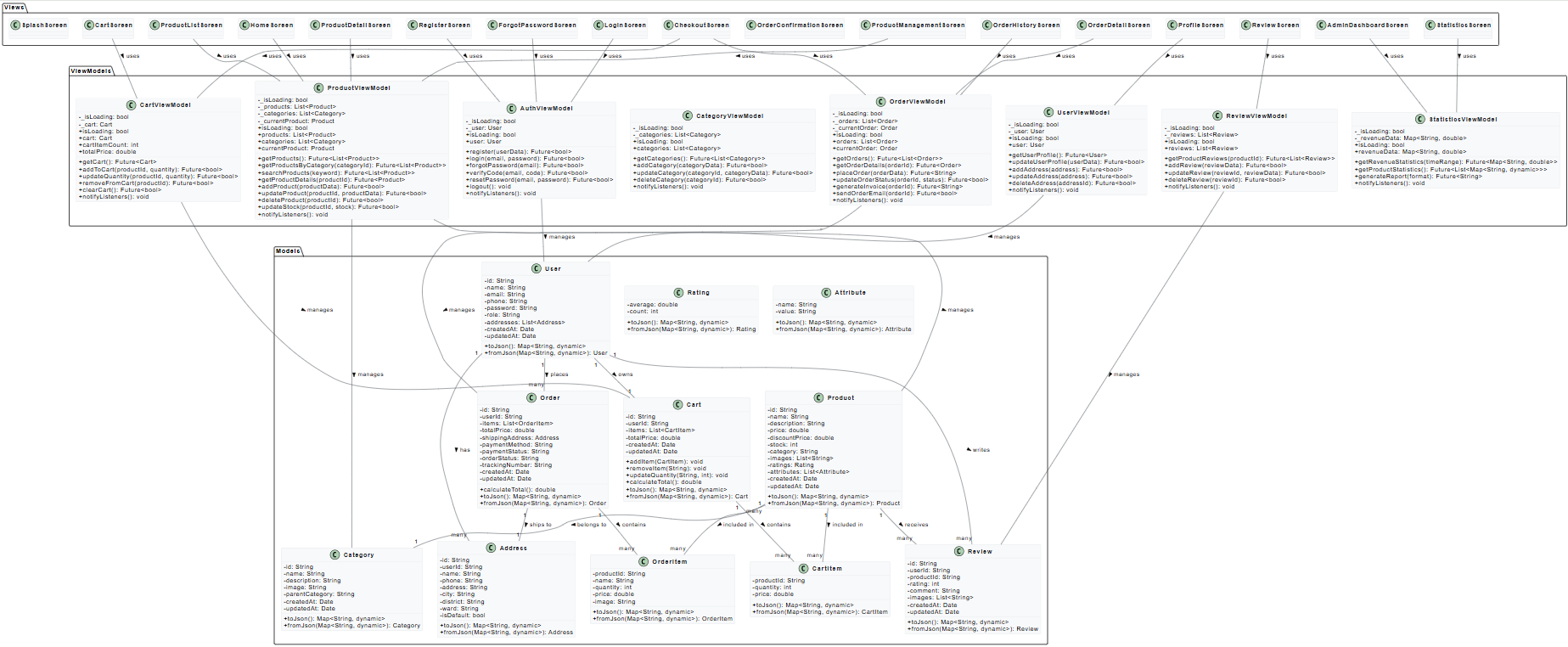
|  |  |  |
| --- | --- | --- |
| Mô tả | | Usecase này cho phép hệ thống gửi thông tin đơn hàng qua email cho khách hàng. |
| Tác nhân | | Hệ thống, Khách hàng |
| Tiền điều kiện | | Đơn hàng đã được tạo và khách hàng có địa chỉ email hợp lệ. |
| Luồng sự kiện | Luồng sự kiện chính | 1. Usecase bắt đầu khi có sự kiện cần gửi thông tin đơn hàng (đơn hàng mới được tạo, trạng thái đơn hàng thay đổi, đơn hàng được giao thành công).  2. Hệ thống truy xuất thông tin đơn hàng và thông tin khách hàng từ cơ sở dữ liệu.  3. Hệ thống tạo nội dung email với các thông tin: mã đơn hàng, ngày đặt, danh sách sản phẩm, tổng tiền, địa chỉ giao hàng, trạng thái đơn hàng.  4. Hệ thống gửi email đến địa chỉ email của khách hàng.  5. Hệ thống cập nhật trạng thái "Đã gửi email" cho đơn hàng.  6. Usecase kết thúc. |
| Luồng rẽ nhánh | Luồng A1: Không thể gửi email  - Ở bước 4, nếu không thể gửi email do lỗi hệ thống hoặc địa chỉ email không hợp lệ, hệ thống ghi log lỗi và đặt lịch gửi lại sau.  Luồng A2: Khách hàng không có email  - Ở bước 2, nếu khách hàng không có địa chỉ email, hệ thống bỏ qua việc gửi email. |
| Hậu điều kiện | | Email chứa thông tin đơn hàng được gửi đến khách hàng. |

* Biểu đồ trình tự:



Hình 2.37: Biểu đồ trình tự Gửi thông tin đơn hàng qua Email

2.4.3. Biểu đồ lớp hệ thống



Hình 2.17: Biểu đồ lớp hệ thống

2.4.3.1. Lớp Model

User: Lưu trữ thông tin người dùng (id, name, email, phone, password, address, role).

Product: Lưu trữ thông tin sản phẩm (id, name, description, price, stock, category, images, ratings).

Category: Lưu trữ thông tin danh mục sản phẩm (id, name, description, image).

Cart: Lưu trữ thông tin giỏ hàng (id, userId, items, totalPrice).

CartItem: Lưu trữ thông tin sản phẩm trong giỏ hàng (productId, quantity, price).

Order: Lưu trữ thông tin đơn hàng (id, userId, items, totalPrice, shippingAddress, paymentMethod, status, createdAt).

OrderItem: Lưu trữ thông tin sản phẩm trong đơn hàng (productId, quantity, price).

Review: Lưu trữ thông tin đánh giá sản phẩm (id, userId, productId, rating, comment, images, createdAt).

Address: Lưu trữ thông tin địa chỉ giao hàng (id, userId, name, phone, address, city, district, ward, isDefault).

2.4.3.2. Lớp ViewModel

AuthViewModel: Xử lý logic đăng nhập, đăng ký, quên mật khẩu.

UserViewModel: Xử lý logic quản lý thông tin người dùng.

ProductViewModel: Xử lý logic hiển thị, tìm kiếm, lọc sản phẩm.

CategoryViewModel: Xử lý logic quản lý danh mục sản phẩm.

CartViewModel: Xử lý logic thêm, sửa, xóa sản phẩm trong giỏ hàng.

OrderViewModel: Xử lý logic đặt hàng, thanh toán, quản lý đơn hàng.

ReviewViewModel: Xử lý logic đánh giá sản phẩm.

StatisticsViewModel: Xử lý logic thống kê doanh thu.

2.4.3.3. Lớp View (Màn hình)

SplashScreen: Màn hình khởi động ứng dụng.

LoginScreen: Màn hình đăng nhập.

RegisterScreen: Màn hình đăng ký.

ForgotPasswordScreen: Màn hình quên mật khẩu.

HomeScreen: Màn hình chính, hiển thị sản phẩm nổi bật, danh mục.

ProductListScreen: Màn hình danh sách sản phẩm.

ProductDetailScreen: Màn hình chi tiết sản phẩm.

CartScreen: Màn hình giỏ hàng.

CheckoutScreen: Màn hình thanh toán.

OrderConfirmationScreen: Màn hình xác nhận đơn hàng.

OrderHistoryScreen: Màn hình lịch sử đơn hàng.

OrderDetailScreen: Màn hình chi tiết đơn hàng.

ProfileScreen: Màn hình thông tin cá nhân.

ReviewScreen: Màn hình đánh giá sản phẩm.

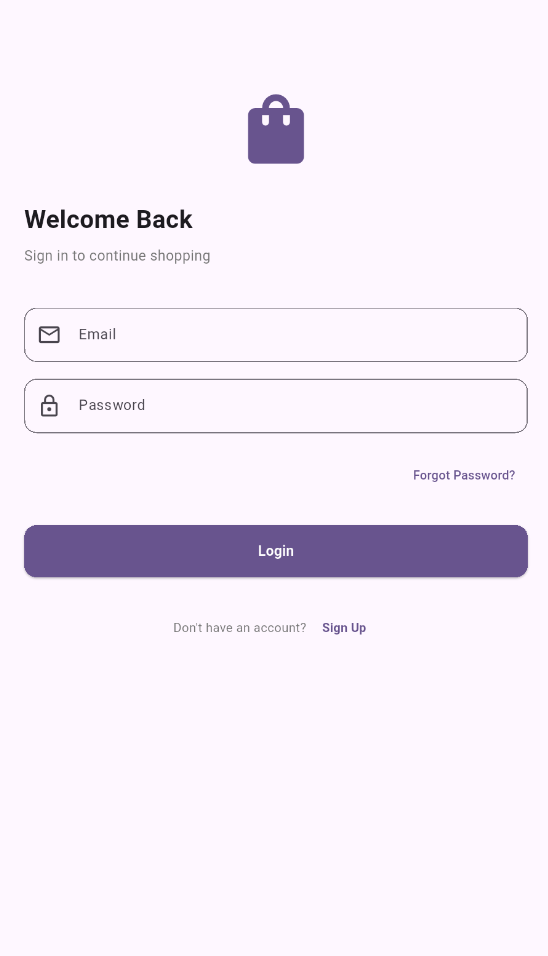
AdminDashboardScreen: Màn hình tổng quan cho quản trị viên.

ProductManagementScreen: Màn hình quản lý sản phẩm.

StatisticsScreen: Màn hình thống kê doanh thu.

2.4.4. Thiết kế giao diện

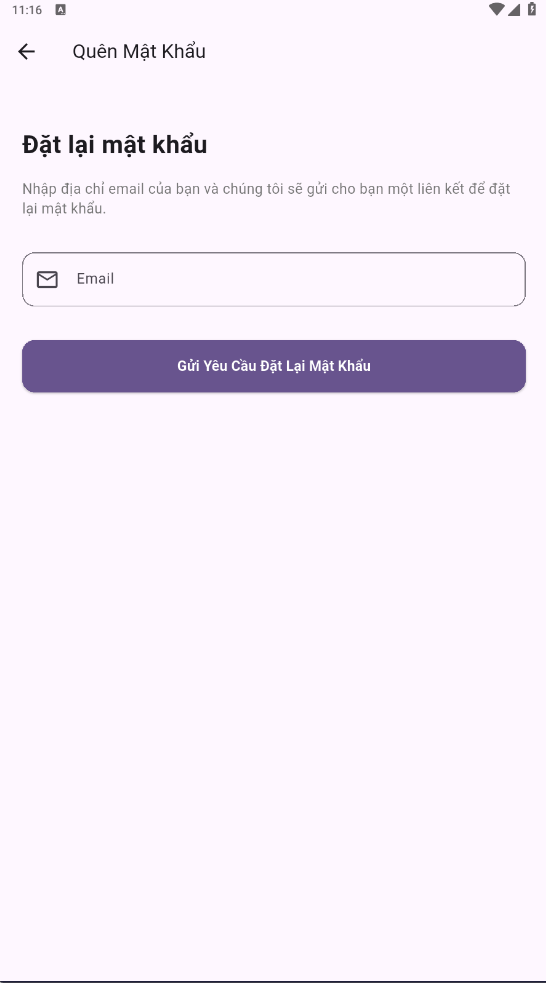
2.4.4.1. Giao diện Đăng nhập và Đăng ký



Hình 2.18: Giao diện màn hình Đăng nhập

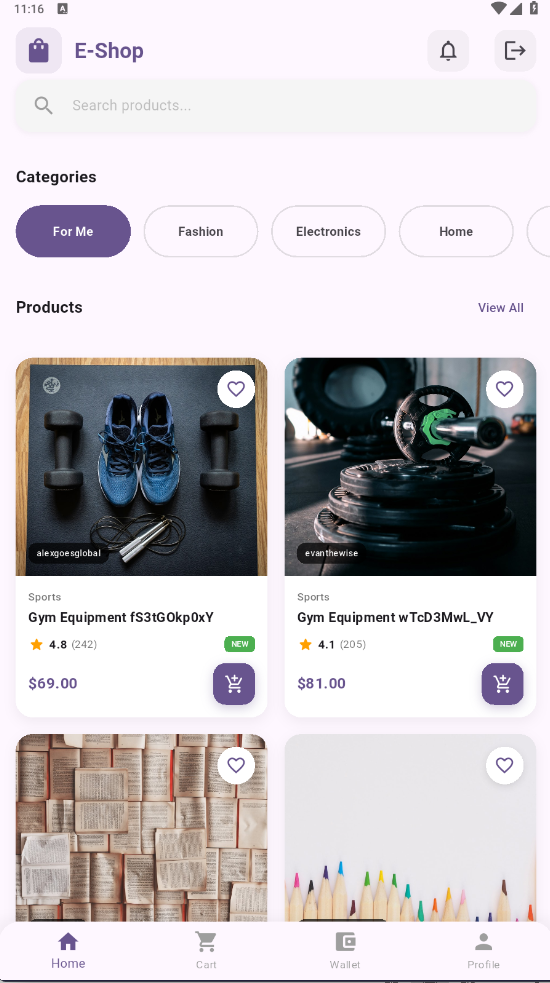


Hình 2.19: Giao diện màn hình Đăng ký

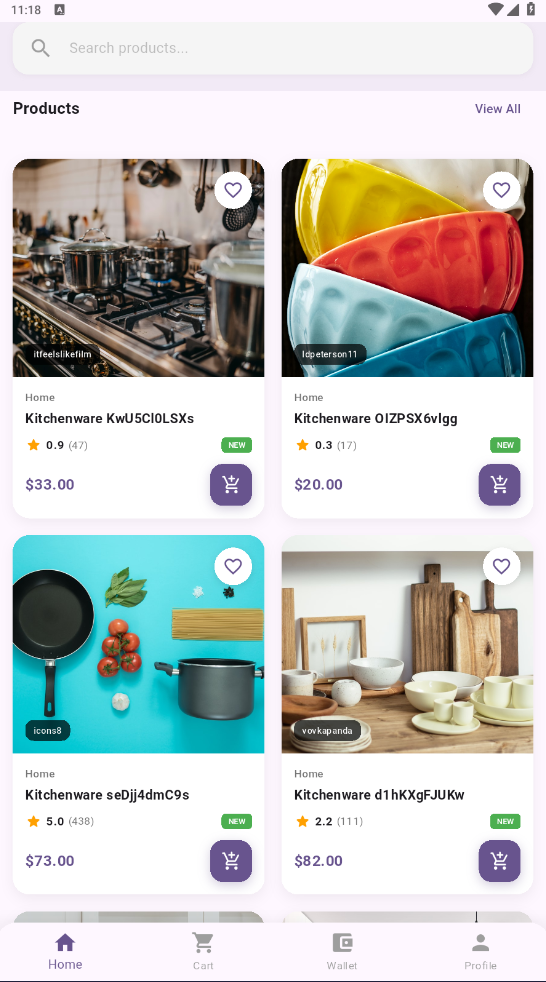


Hình 2.20: Giao diện màn hình Quên mật khẩu

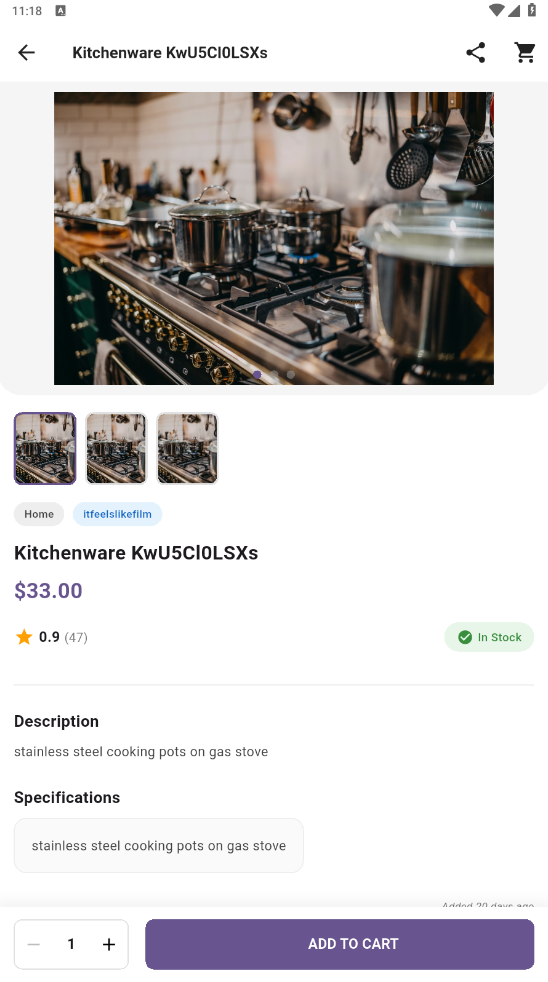
2.4.4.2. Giao diện Trang chủ và Danh sách sản phẩm



Hình 2.21: Giao diện màn hình Trang chủ

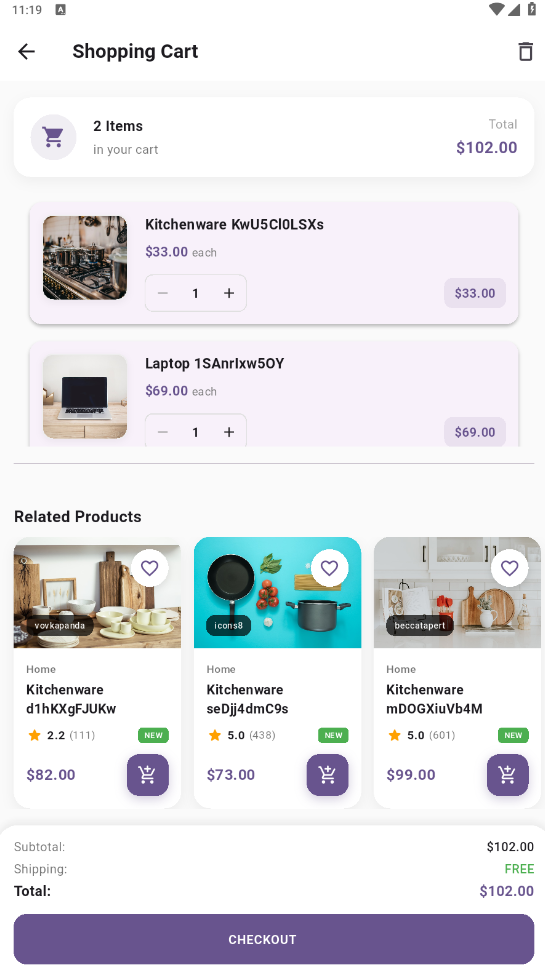


Hình 2.22: Giao diện màn hình Danh sách sản phẩm

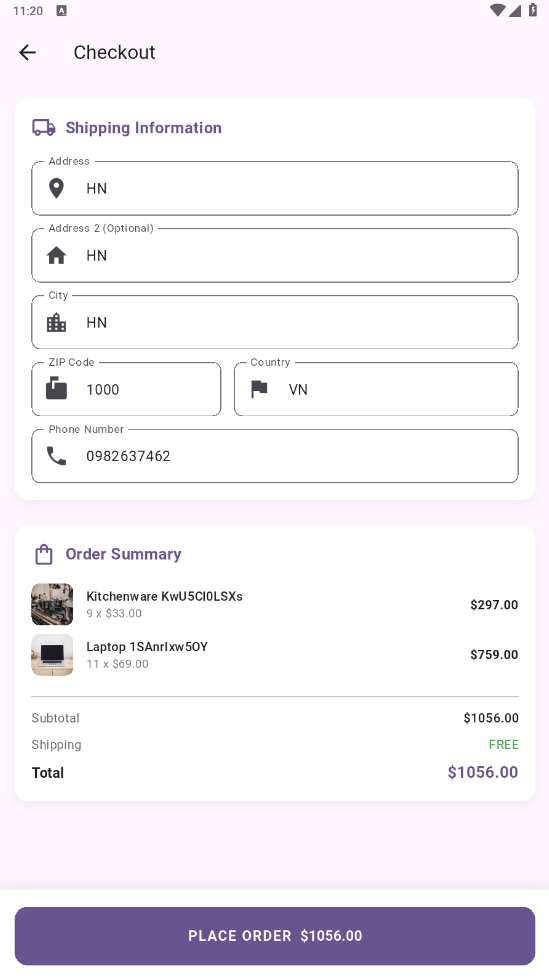


Hình 2.23: Giao diện màn hình Chi tiết sản phẩm

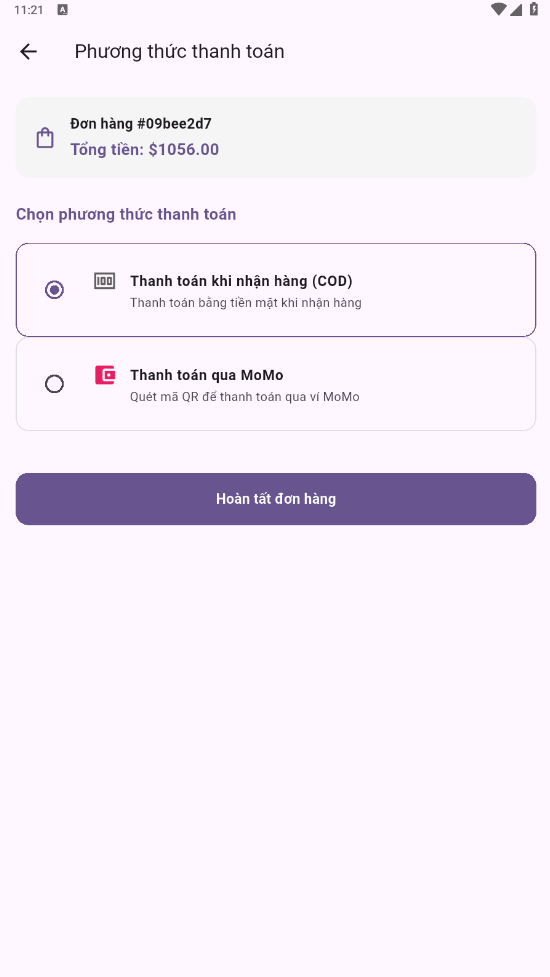
2.4.4.3. Giao diện Giỏ hàng và Thanh toán



Hình 2.24: Giao diện màn hình Giỏ hàng

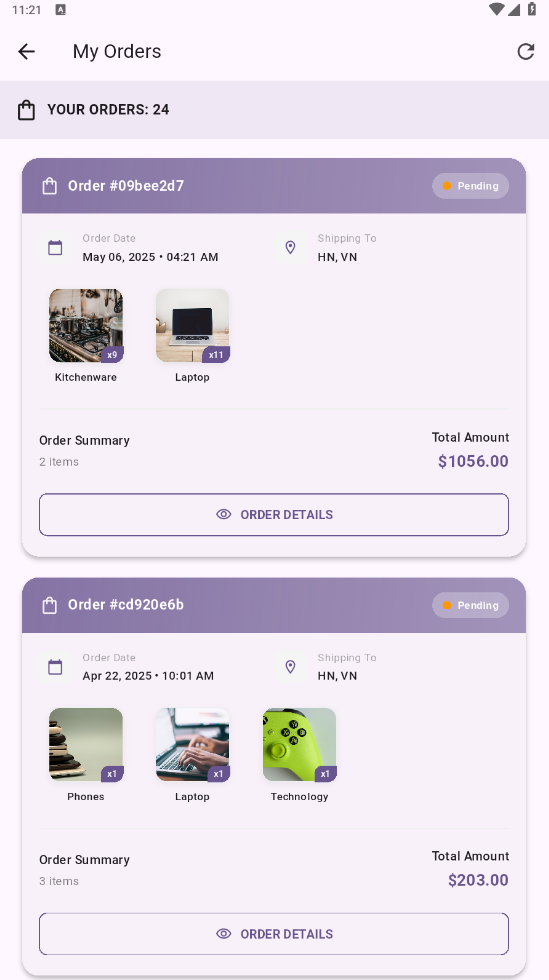


Hình 2.25: Giao diện màn hình Thanh toán

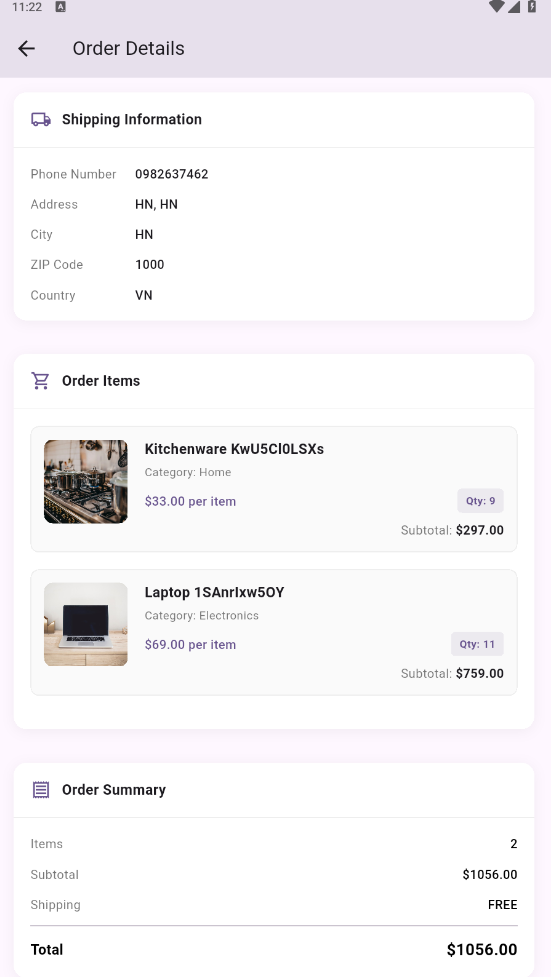


Hình 2.26: Giao diện màn hình chọn phương thức thanh toán

2.4.4.4. Giao diện Quản lý đơn hàng



Hình 2.27: Giao diện màn hình Lịch sử đơn hàng

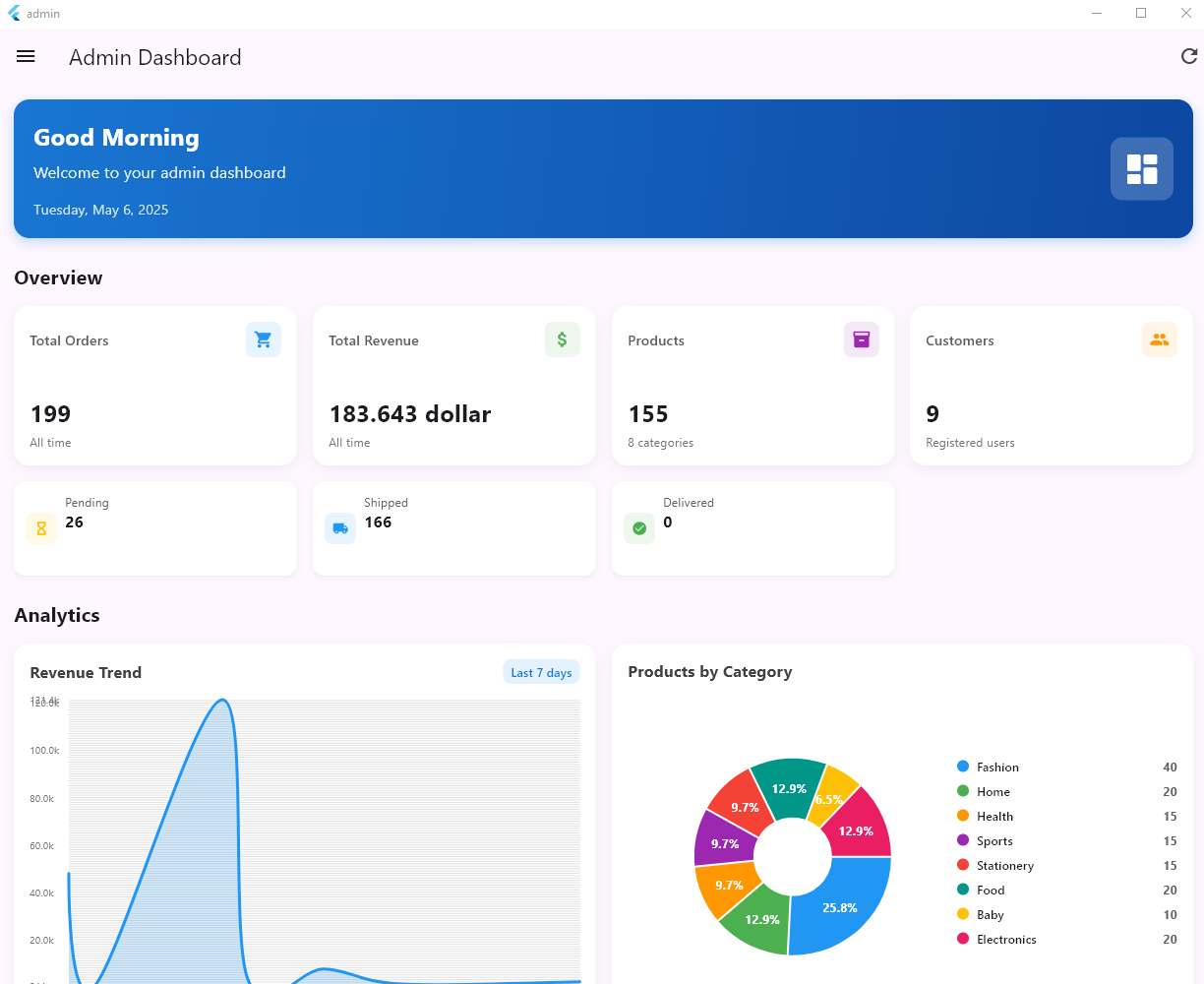


Hình 2.28: Giao diện màn hình Chi tiết đơn hàng

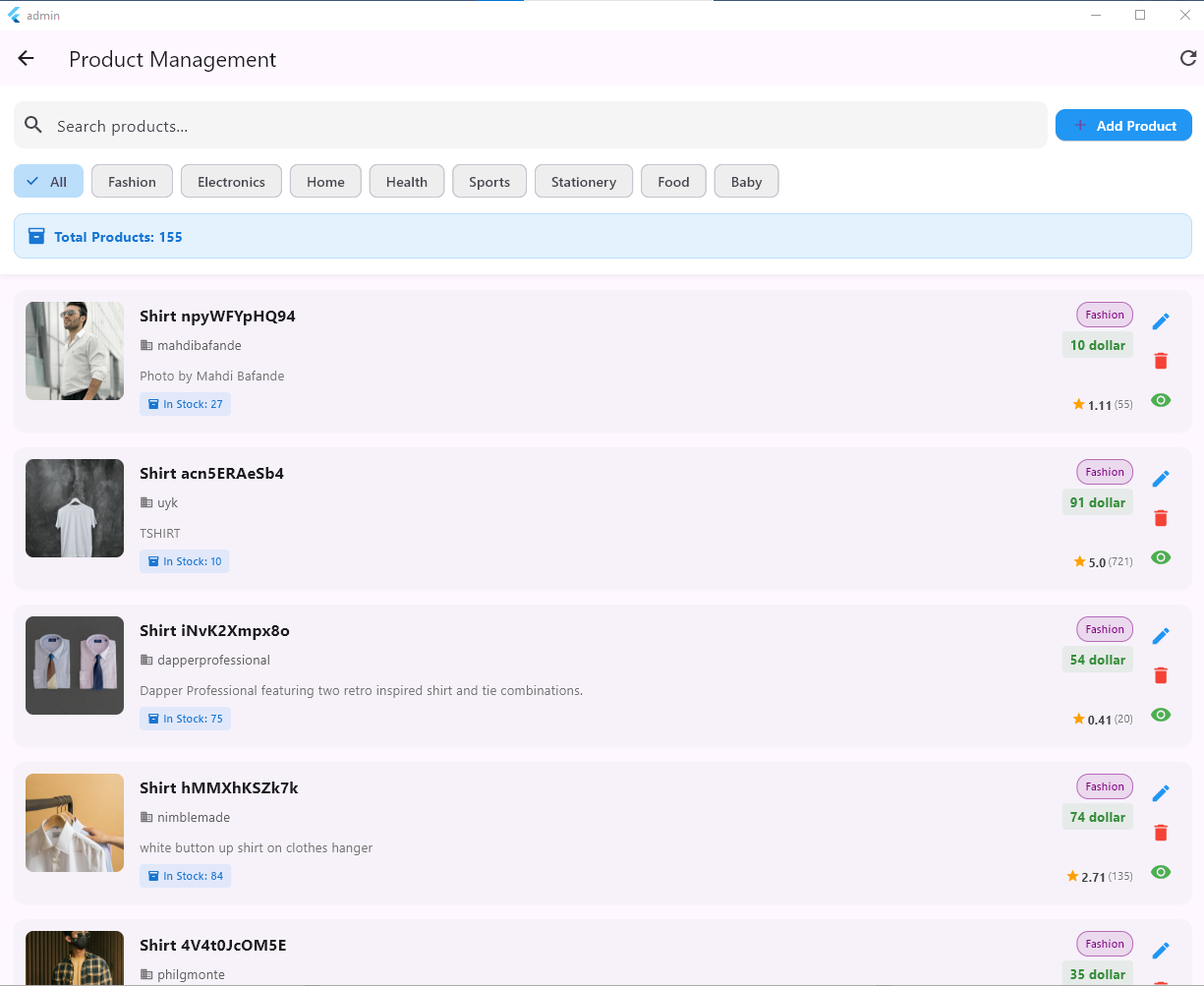
2.4.4.5. Giao diện Đánh giá sản phẩm

Hình 2.29: Giao diện màn hình Đánh giá sản phẩm

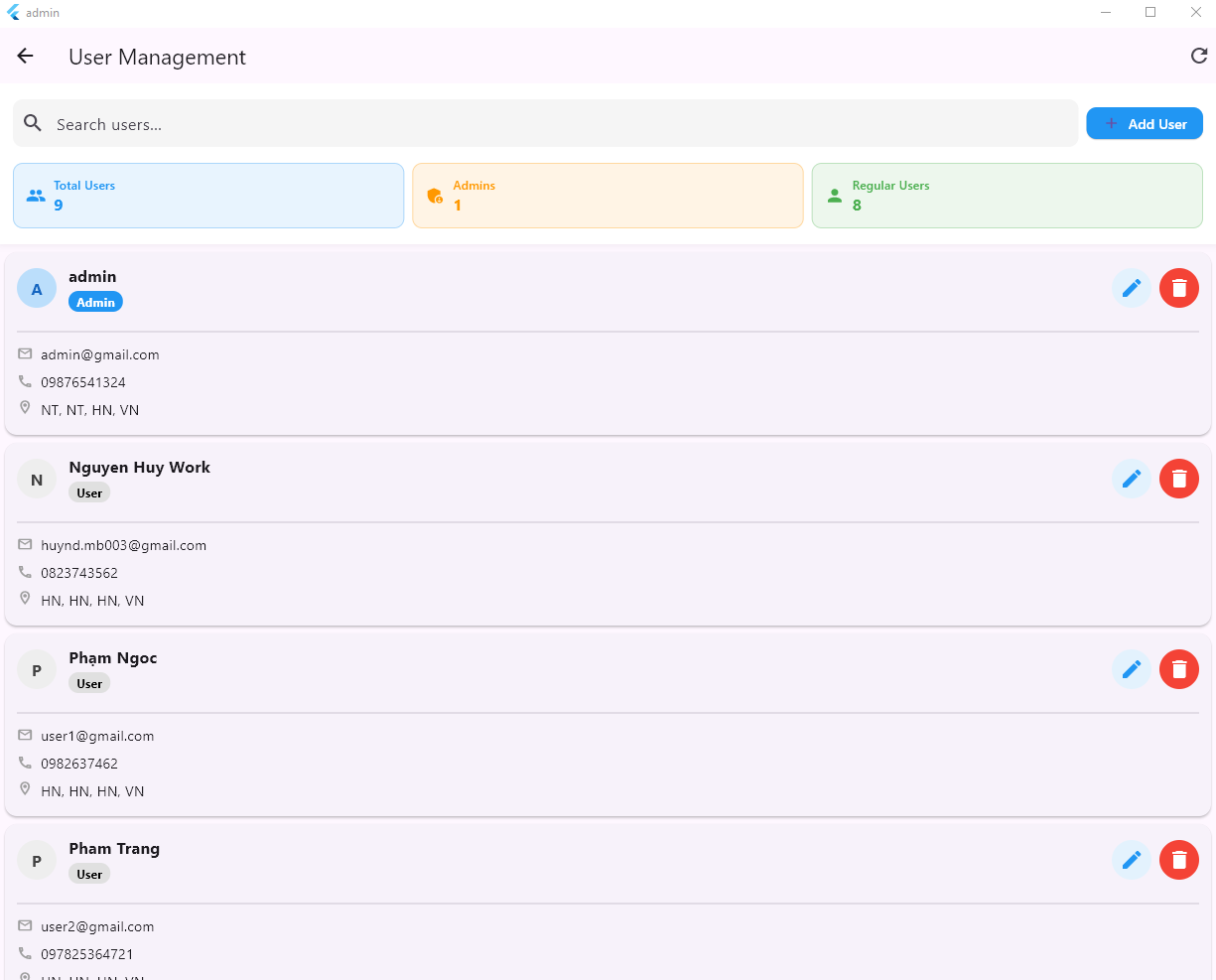
2.4.4.6. Giao diện Quản trị viên



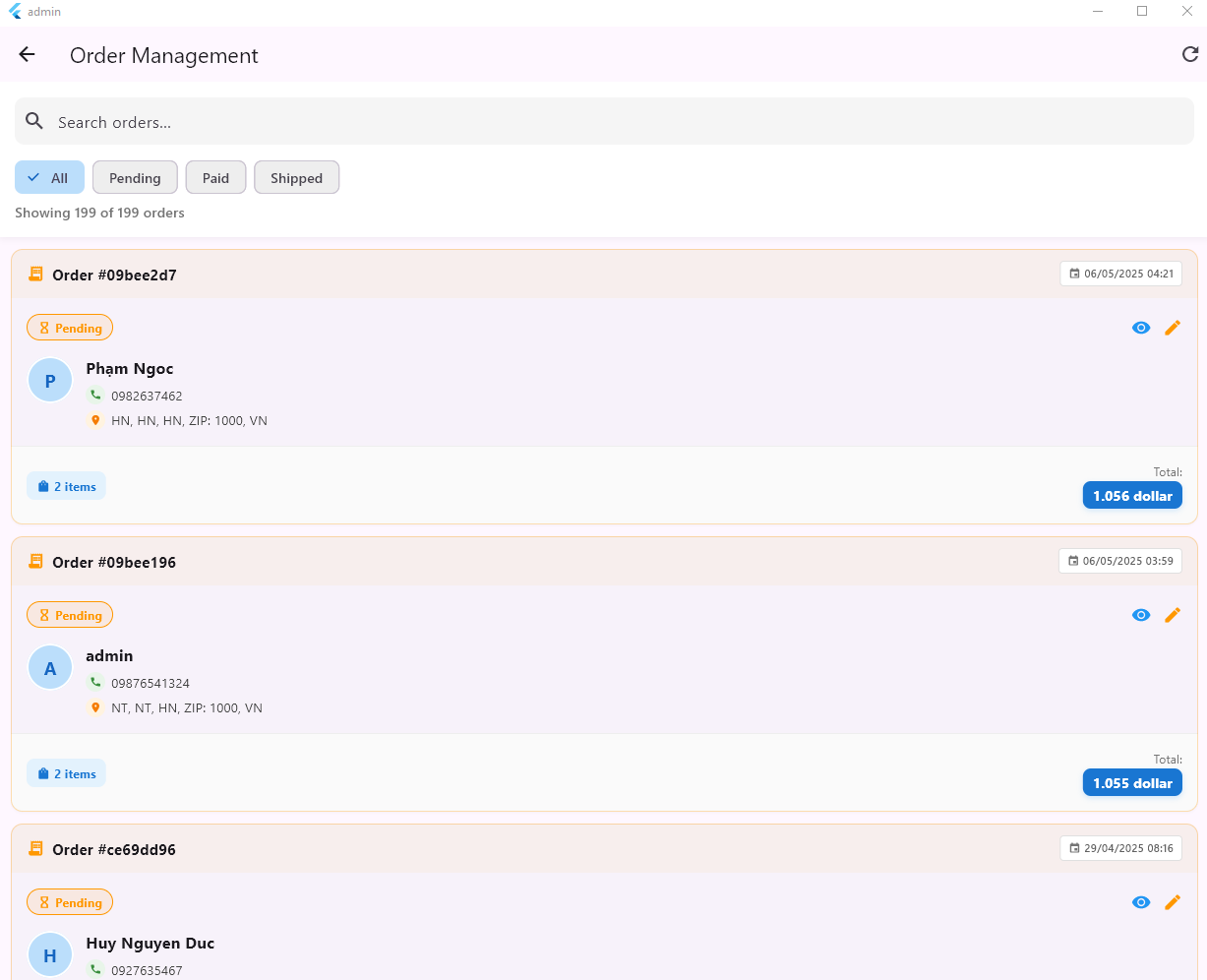
Hình 2.30: Giao diện màn hình Tổng quan quản trị



Hình 2.31: Giao diện màn hình Quản lý sản phẩm



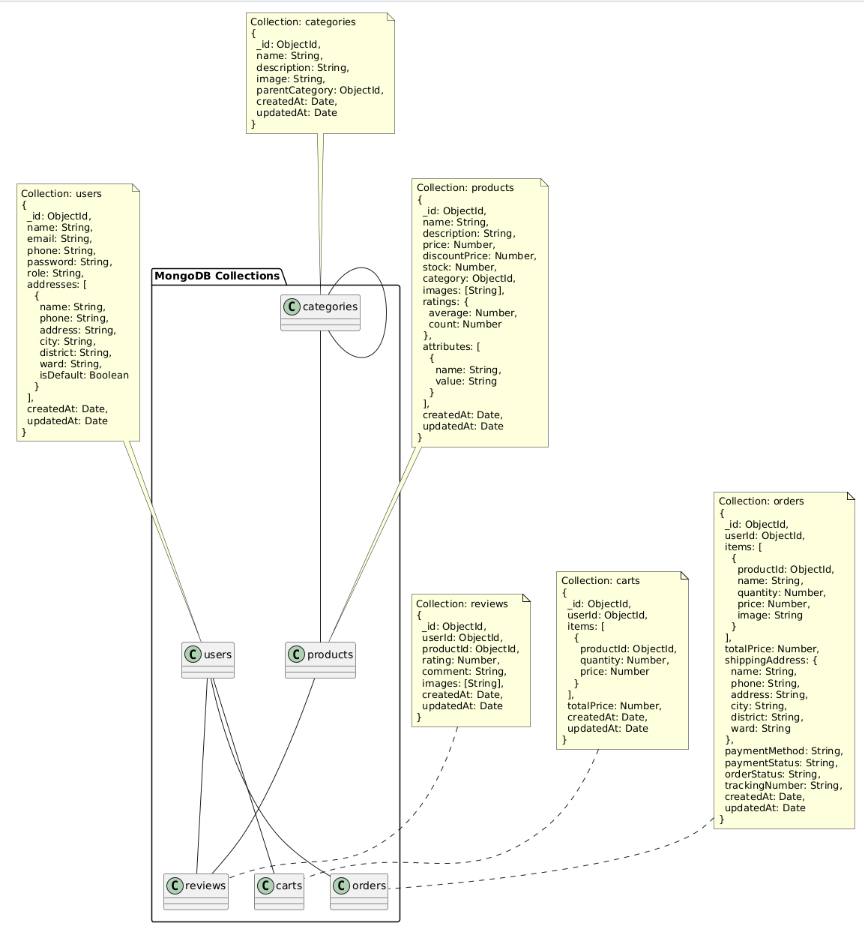
Hình 2.32: Giao diện màn hình Quản lý tài khoản



Hình 2.32: Giao diện màn hình Quản lý đơn hàng

2.4.5. Thiết kế cơ sở dữ liệu

Cơ sở dữ liệu MongoDB được thiết kế với các collection sau:



CHƯƠNG 3. KẾT QUẢ

3.1.Cài đặt môi trường

Để phát triển và triển khai ứng dụng thương mại điện tử ShopEase trên nền tảng di động, em đã tiến hành xây dựng môi trường phát triển với các công cụ và công nghệ hiện đại. Các bước cài đặt môi trường được thực hiện như sau:

Cài đặt Flutter SDK:

Em sử dụng Flutter phiên bản ổn định mới nhất được tải về từ trang chủ flutter.dev, cài đặt và cấu hình biến môi trường để sử dụng các lệnh Flutter trên terminal.

Cài đặt Android Studio:

Android Studio được cài đặt để hỗ trợ biên dịch, chạy ứng dụng trên các thiết bị Android cũng như giả lập (emulator).

Cài đặt Visual Studio Code:

VS Code là công cụ chính để phát triển mã nguồn Flutter nhờ vào sự nhẹ nhàng, hỗ trợ đa dạng extension và tích hợp tốt với Flutter.

Cài đặt Node.js và npm:

Node.js được sử dụng để xây dựng backend cho hệ thống, đồng thời cài đặt các thư viện cần thiết thông qua npm.

Cài đặt MongoDB:

MongoDB Community Server được cài đặt trên máy tính cá nhân, sử dụng để lưu trữ dữ liệu cho hệ thống. Có thể sử dụng MongoDB Atlas để thử nghiệm trên môi trường cloud.

Thiết lập môi trường thử nghiệm:

Tạo các project Flutter mới, thiết lập kết nối backend Node.js với cơ sở dữ liệu MongoDB, cài đặt các package cần thiết như Provider, Dio (cho Flutter), Express, Mongoose, Nodemailer (cho Node.js).

3.2. Cài đặt chương trình

Sau khi hoàn thiện các bước chuẩn bị môi trường, em tiến hành cài đặt, cấu hình và xây dựng ứng dụng với các bước sau:

1. Khởi tạo project Flutter:

Sử dụng lệnh flutter create shopease để tạo project mới, cấu trúc lại thư mục theo mô hình MVVM kết hợp Provider giúp mã nguồn rõ ràng, dễ bảo trì.

1. Xây dựng backend với Node.js:

Khởi tạo project Node.js bằng lệnh npm init, tích hợp các thư viện Express, Mongoose, JWT để xây dựng API RESTful cho ứng dụng.

1. Thiết kế và xây dựng cơ sở dữ liệu MongoDB:

Tạo các collection: users, products, categories, carts, orders, reviews theo thiết kế ở chương 2, đảm bảo dữ liệu được lưu trữ hợp lý, dễ truy xuất và mở rộng.

1. Kết nối frontend (Flutter) với backend (Node.js):

Sử dụng package Dio để thực hiện các request HTTP từ ứng dụng Flutter tới API backend, xử lý dữ liệu trả về và cập nhật UI.

1. Tổ chức mã nguồn:

Chia project Flutter thành các module: Models, ViewModels, Views, Services, Helpers, Resources, tuân thủ nguyên tắc MVVM.

1. Cài đặt quản lý trạng thái bằng Provider:

Provider được tích hợp vào ứng dụng để giúp chia sẻ, cập nhật trạng thái giữa các màn hình một cách hiệu quả.

1. Triển khai chức năng từng phần:

Phát triển lần lượt các chức năng: đăng ký, đăng nhập, quên mật khẩu, tìm kiếm sản phẩm, giỏ hàng, đặt hàng, thanh toán, đánh giá sản phẩm, quản lý sản phẩm (cho admin), thống kê doanh thu, xuất hóa đơn, gửi thông tin đơn hàng qua email...

1. Kiểm thử trên nhiều thiết bị:

Ứng dụng được build và kiểm thử trên các thiết bị Android vật lý và giả lập, đảm bảo hoạt động ổn định, UI nhất quán trên nhiều kích thước màn hình.

3.3. Kết quả đạt được

Sau thời gian nghiên cứu và phát triển, ứng dụng ShopEase đã hoàn thiện với đầy đủ các chức năng cơ bản của một hệ thống thương mại điện tử hiện đại, đáp ứng các tiêu chí đã đề ra trong phạm vi đề tài. Cụ thể:

* Đối với khách hàng:
  + Đăng ký, đăng nhập, quên mật khẩu với xác thực email.
  + Quản lý thông tin cá nhân, địa chỉ giao hàng.
  + Xem, tìm kiếm, lọc, sắp xếp sản phẩm theo danh mục.
  + Xem chi tiết sản phẩm, thêm vào giỏ hàng, quản lý giỏ hàng.
  + Đặt hàng, lựa chọn phương thức thanh toán (COD hoặc online).
  + Nhận email xác nhận đơn hàng, theo dõi trạng thái đơn hàng.
  + Xem lịch sử mua hàng, đánh giá sản phẩm sau khi nhận hàng.
* Đối với quản trị viên:
  + Đăng nhập hệ thống với quyền quản trị.
  + Quản lý sản phẩm: thêm, sửa, xóa, cập nhật tồn kho.
  + Quản lý khách hàng: tìm kiếm, phân loại, xem thông tin chi tiết, lịch sử mua hàng.
  + Quản lý đơn hàng: xem, cập nhật trạng thái, xuất hóa đơn, gửi email.
* Các chức năng bổ trợ:
  + Gửi thông báo đơn hàng qua email.
  + Đảm bảo bảo mật thông tin cá nhân, xác thực và phân quyền hợp lý.
  + Giao diện đẹp, đơn giản tối ưu người dùng.
  + Ứng dụng hoạt động mượt mà, đa nền tảng, dễ dàng mở rộng về sau.

3.4. Kết quả kiểm thử

Trong quá trình phát triển, ứng dụng ShopEase đã được kiểm thử cẩn thận trên nhiều thiết bị, với các trường hợp kiểm thử cụ thể cho từng chức năng chính, nhằm đảm bảo tính đúng đắn, ổn định và thân thiện với người dùng. Dưới đây là bảng tổng hợp các kết quả kiểm thử:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Chức năng** | **Mục đích kiểm thử** | **Đầu vào** | **Đầu ra mong muốn** | **Kết quả** |
| 1 | Đăng ký tài khoản | Đăng ký tài khoản mới | Thông tin hợp lệ/không hợp lệ | Tạo tài khoản/thông báo lỗi | Pass |
| 2 | Đăng nhập | Đăng nhập hệ thống | Email/số điện thoại, mật khẩu | Đăng nhập thành công/thông báo lỗi | Pass |
| 3 | Quên mật khẩu | Khôi phục mật khẩu | Email hợp lệ/không hợp lệ | Nhận mã xác nhận/thông báo lỗi | Pass |
| 4 | Tìm kiếm sản phẩm | Tìm kiếm, lọc sản phẩm | Từ khóa hợp lệ/không hợp lệ | Hiển thị kết quả/thông báo không tìm thấy | Pass |
| 5 | Thêm vào giỏ hàng | Thêm sản phẩm | Số lượng hợp lệ/vượt tồn kho | Thêm/cập nhật số lượng/thông báo lỗi | Pass |
| 6 | Đặt hàng & Thanh toán | Tạo đơn hàng, thanh toán | Đầy đủ thông tin/COD/online | Đơn hàng thành công/thông báo lỗi | Pass |
| 7 | Đánh giá sản phẩm | Đánh giá, nhận xét | Đã mua/Chưa mua | Lưu/hạn chế đánh giá | Pass |
| 8 | Quản lý sản phẩm | Thêm, sửa, xóa sản phẩm (admin) | Thông tin hợp lệ/không hợp lệ | Cập nhật/thông báo lỗi | Pass |
| 9 | Quản lý khách hàng | Tìm kiếm, phân loại | Từ khóa hợp lệ/không có kết quả | Hiển thị/thông báo | Pass |
| 10 | Xuất hóa đơn | Tạo, gửi, tải xuống hóa đơn | Đơn hàng hợp lệ | Email | Pass |
| 11 | Thống kê doanh thu | Thống kê, xuất báo cáo | Khoảng thời gian | Hiển thị, tải file | Pass |
| 12 | Gửi email | Gửi thông tin đơn hàng | Đơn hàng mới/trạng thái thay đổi | Email gửi thành công | Pass |

KẾT LUẬN

1. Đánh giá kết quả đạt được

Sau quá trình xây dựng và triển khai, ứng dụng thương mại điện tử ShopEase đã đáp ứng tốt các tiêu chí đề ra. Kết quả nổi bật bao gồm:

* Hoàn thiện các chức năng chính của một ứng dụng thương mại điện tử:

Người dùng có thể đăng ký, đăng nhập, đặt lại mật khẩu, duyệt và tìm kiếm sản phẩm, thêm vào giỏ hàng, đặt hàng, thanh toán, theo dõi đơn hàng, đánh giá sản phẩm. Quản trị viên có thể quản lý sản phẩm, khách hàng, đơn hàng và xem thống kê doanh thu.

* Tích hợp thanh toán QR code Momo:

Ứng dụng đã hỗ trợ hình thức thanh toán bằng QR code Momo – một trong những phương thức thanh toán điện tử phổ biến nhất hiện nay tại Việt Nam. Tính năng này giúp thanh toán trở nên nhanh chóng, tiện lợi, phù hợp với xu hướng tiêu dùng số hiện đại.

* Giao diện đẹp, trải nghiệm người dùng tốt:

Ứng dụng xây dựng trên Flutter, giao diện trực quan, dễ sử dụng, phù hợp với nhiều đối tượng người dùng.

* Kiến trúc phần mềm rõ ràng, ổn định:

Ứng dụng tuân thủ mô hình MVVM kết hợp Provider, mã nguồn tổ chức hợp lý, dễ bảo trì, thuận tiện cho việc mở rộng trong tương lai.

* Bảo mật và quản lý dữ liệu hiệu quả:

Dữ liệu người dùng và giao dịch được bảo vệ với các giải pháp xác thực, kiểm tra đầu vào, mã hóa thông tin quan trọng, hạn chế rủi ro về bảo mật.

* Tuy vậy, ứng dụng vẫn còn một số điểm hạn chế như: chưa hỗ trợ đa ngôn ngữ, chưa tích hợp nhiều cổng thanh toán quốc tế và tính năng cá nhân hóa trải nghiệm người dùng còn đơn giản.

2. Hướng phát triển ứng dụng trong tương lai

Để nâng cao chất lượng và khả năng cạnh tranh của ShopEase, các hướng phát triển tiếp theo sẽ tập trung vào:

* Mở rộng phương thức thanh toán:

Bên cạnh QR code Momo đã triển khai, ứng dụng sẽ tích hợp thêm các ví điện tử khác (ZaloPay, VNPay...), thẻ tín dụng quốc tế (Visa, MasterCard), cũng như hỗ trợ thanh toán trả góp.

* Phát triển đa nền tảng, đa ngôn ngữ:

Bổ sung phiên bản cho iOS, website và hỗ trợ nhiều ngôn ngữ (tiếng Việt, tiếng Anh…), giúp ứng dụng tiếp cận đông đảo người dùng hơn.

* Tăng cường bảo mật và tối ưu hiệu năng:

Áp dụng các giải pháp xác thực hai lớp (2FA), tối ưu hóa truy vấn dữ liệu, sử dụng CDN cho hình ảnh sản phẩm, đảm bảo tốc độ và độ an toàn khi số lượng người dùng tăng cao.

* Cá nhân hóa và nâng cao tương tác với người dùng:

Phát triển hệ thống gợi ý sản phẩm thông minh, chatbot tư vấn tự động, thông báo đẩy khuyến mãi, chăm sóc khách hàng sau bán hàng, tích hợp tính năng đánh giá dịch vụ.

* Bổ sung các công cụ quản trị nâng cao:

Cung cấp báo cáo phân tích chuyên sâu về doanh thu, hành vi mua sắm, xu hướng thị trường, hỗ trợ ra quyết định kinh doanh nhanh chóng và chính xác.

* Mở rộng hệ sinh thái dịch vụ:

Kết nối với các dịch vụ vận chuyển, bảo hành điện tử, hỗ trợ đổi trả/hoàn tiền linh hoạt, hướng đến xây dựng một hệ sinh thái thương mại điện tử toàn diện.

TÀI LIỆU THAM KHẢO

[1] Tổ HTTT Đại học Công Nghiệp Hà Nội – Giáo trình hệ quản trị cơ sở dữ

liệu SQL.

[2] Tổ HTTT Đại học Công Nghiệp Hà Nội – Giáo trình phân tích thiết kế hệ

thống.

[3] Flutter, "Flutter Documentation," 2024. [Online]. Available: https://docs.flutter.dev/

[4] Node.js, "Node.js Documentation," 2024. [Online]. Available: https://nodejs.org/en/docs/

[5] MongoDB, "MongoDB Documentation," 2024. [Online]. Available: https://www.mongodb.com/docs/

[6] Dart, "Dart Guides," 2024. [Online]. Available: https://dart.dev/guides/

[7] PlantUML, "PlantUML Documentation," 2024. [Online]. Available: https://plantuml.com/

[8] Provider Package, "Provider State Management for Flutter," 2024. [Online]. Available: https://pub.dev/packages/provider

[9] Express.js, "Express.js Documentation," 2024. [Online]. Available: https://expressjs.com/

[10] Mongoose, "Mongoose Documentation," 2024. [Online]. Available: https://mongoosejs.com/

[11] JSON Web Token (JWT), "JWT Introduction & Guide," 2024. [Online]. Available: https://jwt.io/

[12] Material Design, "Material Design Guidelines," 2024. [Online]. Available: https://material.io/