|  |  |
| --- | --- |
|  |  |
|  | **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **---------------------------------------** |
|  |
| **BÁO CÁO THỰC TẬP CƠ SỞ NGÀNH** |
| **ĐỀ TÀI : XÂY DỰNG WEBSITE QUẢN LÝ**  **THU CHI CÁ NHÂN** |
| GVHD: **TS. VŨ VIỆT THẮNG**  LỚP:2022IT6055001 - Nhóm01 |
| Thành viên:  NGUYỄN THỊ THÚY DÂN - 2021601846 |
| NGUYỄN THỊ ĐIỆP - 2021601120 |
|  | ĐINH HỒNG LIỄU - 2021601817 |
| LÊ THỊ NGỌC - 2021601311 |
| THÂN NGỌC THIỆN - 2021602775 |
| Hà Nội – Năm 2023 |

# MỤC LỤC

[MỤC LỤC 1](#_Toc137621217)

[Mục lục ảnh 3](#_Toc137621218)

[Mục lục bảng 5](#_Toc137621219)

[LỜI MỞ ĐẦU 6](#_Toc137621220)

[CHƯƠNG 1: TỔNG QUAN VỀ NỘI DUNG NGHIÊN CỨU 7](#_Toc137621221)

[I. Tổng quan về ngôn ngữ lập trình Java, JavaScript, HTML, CSS và phần mềm Visual Studio code 7](#_Toc137621222)

[1. Ngôn ngữ lập trình Java 7](#_Toc137621223)

[2. Ngôn ngữ lập trình JavaScript 9](#_Toc137621224)

[3. Ngôn ngữ đánh dấu siêu văn bản HTML và ngôn ngữ thiết kế CSS 10](#_Toc137621225)

[4. Phần mềm Visual Studio Code 13](#_Toc137621226)

[II. Giới thiệu về Json, Servlet, Netbeans và Glassfish 15](#_Toc137621227)

[1. Json 15](#_Toc137621228)

[2. Servlet 16](#_Toc137621229)

[3. Netbeans 18](#_Toc137621230)

[4. Glassfish 19](#_Toc137621231)

[III. Một số công cụ hỗ trợ thiết kế phần mềm khác 21](#_Toc137621232)

[1. StarUML 21](#_Toc137621233)

[2. Rational Rose 22](#_Toc137621234)

[3. Balsamiq mockup 23](#_Toc137621235)

[4. Case studio 2 26](#_Toc137621236)

[CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 27](#_Toc137621237)

[I. Khảo sát hệ thống 27](#_Toc137621238)

[1. Khảo sát sơ bộ 27](#_Toc137621239)

[2. Tài liệu đặc tả người dùng 33](#_Toc137621240)

[II. Phân tích hệ thống 35](#_Toc137621241)

[1. Mô hình hóa dữ liệu của hệ thống 35](#_Toc137621242)

[2. Thiết kế cơ sở dữ liệu 37](#_Toc137621243)

[3. Biểu đồ use case 39](#_Toc137621244)

[4. Mô tả chi tiết các use case 44](#_Toc137621245)

[5. Biểu đồ trình tự 63](#_Toc137621246)

[III. Thiết kế hệ thống 69](#_Toc137621247)

[1. Trang đăng kí 69](#_Toc137621248)

[2. Trang đăng nhập 70](#_Toc137621249)

[3. Trang Transaction 72](#_Toc137621250)

[4. Nút Add Transaction 73](#_Toc137621251)

[5. Trang report 74](#_Toc137621252)

[6. Trang Categories 75](#_Toc137621253)

[7. Trang Account 76](#_Toc137621254)

[8. Trang Wallet 77](#_Toc137621255)

[Tài liệu tham khảo 78](#_Toc137621256)

# MỤC LỤC HÌNH ẢNH

[Hình 1.1. Java 7](#_Toc137558609)

[Hình 1.2. JavaScrip 9](#_Toc137558610)

[Hình 1.3. Lập trình tương tác cho trang web 11](#_Toc137558611)

[Hình 1.4. Visual Studio Code 14](#_Toc137558612)

[Hình 1.5. Giao diện dòng lệnh trong Visual Studio Code 15](#_Toc137558613)

[Hình 1. 6. Servlet 17](#_Toc137558614)

[Hình 1.7. Netbeans 19](#_Toc137558615)

[Hình 1.8. StarUML - Mô hình phần mềm tiên tiến 21](#_Toc137558616)

[Hình 1.9. Giao diện phần mềm Balsamiq Mockups 24](#_Toc137558617)

[Hình 1.10. Phác thảo ý tưởng nhanh mà hiệu quả 25](#_Toc137558618)

[Hình 1.11*.* Thao tác kéo thả trong Balsamiq Mockups 25](#_Toc137558619)

[Hình 2.1. Biểu đồ mức chi tiêu trung bình/tháng 27](#_Toc137558646)

[Hình 2.2. Biểu đồ khảo sát việc gặp khó khăn 29](#_Toc137558647)

[Hình 2.3. Biểu đồ khảo sát về cách quản lý tài chính 30](#_Toc137558648)

[Hình 2.4. Biểu đồ khảo sát về nhu cầu sử dụng một ứng dụng 31](#_Toc137558649)

[Hình 2.5. Mô hình hóa hệ thống 36](#_Toc137558650)

[Hình 2.6. Thiết kế ánh xạ cơ sở dữ liệu 39](#_Toc137558651)

[Hình 2. 7. Sơ đồ usecase 41](#_Toc137558652)

[Hình 2.8. Use case phía frontend 42](#_Toc137558653)

[Hình 2.9. Use case phía frontend 43](#_Toc137558654)

[Hình 2.10. Use case đăng nhập 63](#_Toc137558655)

[Hình 2.11. Use case tạo tài khoản 63](#_Toc137558656)

[Hình 2.12. Use case tạo ví 64](#_Toc137558657)

[Hình 2.13. Use case xem giao dịch 64](#_Toc137558658)

[Hình 2.14. Use case xem loại hình chi tiêu 65](#_Toc137558659)

[Hình 2.15. Use case điều chỉnh số dư 65](#_Toc137558660)

[Hình 2.16. Use case bảo trì ví 66](#_Toc137558661)

[Hình 2.17. Use case quản lý tài khoản khách hàng 67](#_Toc137558662)

[Hình 2.18. Use case bảo trì giao dịch 68](#_Toc137558663)

[Hình 2.19. Giao diện trang đăng ký 69](#_Toc137558664)

[Hình 2.20. Giao diện trang đăng nhập 70](#_Toc137558665)

[Hình 2.21. Giao diện trang Transaction 72](#_Toc137558666)

[Hình 2.22. Giao diện nút Add Transaction 73](#_Toc137558667)

[Hình 2.23. Giao diện trang report 74](#_Toc137558668)

[Hình 2.24. Giao diện trang Category 75](#_Toc137558669)

[Hình 2.25. Giao diện trang Account 76](#_Toc137558670)

[Hình 2.26. Giao diện trang Wallet 77](#_Toc137558671)

# MỤC LỤC BẢNG

[Bảng 1. Bảng CategoryTransaction 37](#_Toc137558672)

[Bảng 2. Bảng User 37](#_Toc137558673)

[Bảng 3. Bảng Wallet 38](#_Toc137558674)

[Bảng 4. Bảng InOutMoney 38](#_Toc137558675)

[Bảng 5. Bảng Transaction 39](#_Toc137558676)

# LỜI MỞ ĐẦU

Hiện nay trên thế giới nói chung và nước ta nói riêng đang diễn ra quá trình tin học hóa toàn diện trên mọi lĩnh vực hoạt động của xã hội loài người, loài người đang chuyển sang kỷ nguyên của Công nghệ thông tin .Các máy tính đặc biệt là các máy vi tính xuất hiện khắp nơi, hoặc hỗ trợ hoặc thay thế toàn bộ con người thực hiện những công việc do con người giao cho thông qua các chương trình máy tính.Với sự hỗ trợ đắc lực của Công nghệ thông tin, nó đã giúp cho công việc quản lý ngày càng hiệu quả và dễ dàng xử lý các tình huống, các yêu cầu, đưa ra các con số, và các báo cáo một cách chính xác và nhanh chóng nhất và bên cạnh đó, nó còn giảm thiểu thời gian rất lớn và công sức để hoàn thành công việc.

Đứng trước vai trò của công nghệ thông tin, rất nhiều các ứng dụng, website quản lý doanh nghiệp, quản lý cá nhân ra đời đã ra đời như quản lý nhân sự; quản lý hàng hoá, vật tư; quản lý sức khoẻ… và thực sự chiếm được lòng tin của khách hàng. Tuy nhiên để cho ra đời một sản phẩm có chất lượng thì người thiết kế cần đáp ứng được một số điều kiện sau: độ chính xác cao, lưu trữ dữ liệu thuận tiện, dễ dàng tra cứu, đáp ứng tốt các nhu cầu mới phát sinh, cho hiệu quả cao, phù hợp với nhu cầu, điều kiện người dùng. Đặc biệt đối với mỗi cá nhân việc sử dụng một website quản lý giúp cá nhân dễ làm chủ trong mọi tình huống. Việc quản lý thu chi đang là một bài toán khó đối với xã hội đặc biệt đối với cá nhân. Nhiều cá nhân muốn trang bị thêm kiến thức, mong muốn hạn chế số chi tiêu không hợp lý, nhu cầu muốn có một công cụ dễ dàng, thuận tiện quản lý thu chi của mình.

Trong xu thế đó, để tiếp cận và góp phần đẩy mạnh sự phổ biến của website quản lý ở Việt Nam, giải quyết vấn đề thu chi của cá nhân, nhóm em đã quyết định thực hiện đề tài: “Xây dựng website quản lý thu chi cá nhân”. Với mong muốn đáp ứng nhu cầu người dùng: chi tiêu hợp lý, không lãng phí, không bị gò bó bởi đồng tiền giúp nâng cao chất lượng cuộc sống.

Nhóm sinh viên thực hiện đề tài!

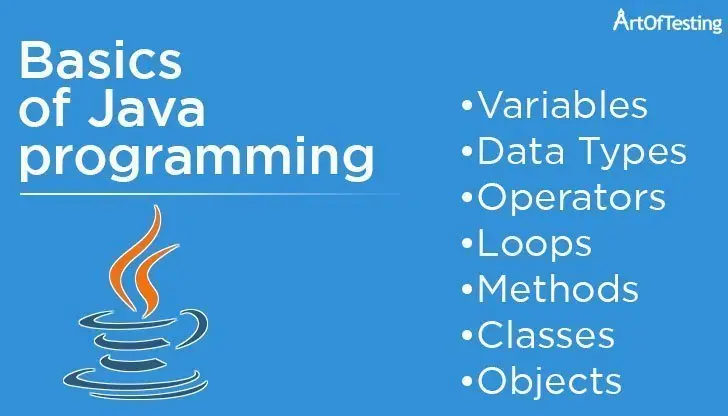
# CHƯƠNG 1: TỔNG QUAN VỀ NỘI DUNG NGHIÊN CỨU

## I. Tổng quan về ngôn ngữ lập trình Java, JavaScript, HTML, CSS và phần mềm Visual Studio code

### 1. Ngôn ngữ lập trình Java

Java là một ngôn ngữ lập trình cao cấp, được sử dụng trong phát triển phần mềm trang web ban đầu được phát triển bởi Sun Microsystems và được phát hành vào năm 1994. Java chạy trên nhiều nền tảng, chẳng hạn như Windows, Mac OS và các phiên bản khác nhau của UNIX.

Phiên bản mới nhất của Java Standard Edition là Java SE 17. Với những tiến bộ và nhiều cấu hình đã được xây dựng để phù hợp với nhiều loại nền tảng khác nhau, công cụ ngày càng phổ biến. Ví dụ: Java EE cho các ứng dụng doanh nghiệp, J2ME cho các ứng dụng di động [1].



Hình 1.1. Java

Đặc trưng cơ bản của Java [2]:

* Hướng đối tượng – Trong Java, mọi thứ đều là một Object. Java có thể dễ dàng mở rộng vì nó được dựa trên mô hình Object.
* Nền tảng độc lập – Không giống nhiều ngôn ngữ lập trình khác như C và C++, khi Java được biên dịch, nó không được biên dịch vào nền tảng máy tính cụ thể, thay vào đó là mã byte nền tảng độc lập. Mã byte này được phân phát trên web và được thông dịch bởi Virtual Machine (JVM) trên nền tảng nào đó mà nó đang chạy.
* Đơn giản – Java được thiết kế rất dễ học. Nếu bạn hiểu khái niệm cơ bản của OOP Java, bạn sẽ rất dễ làm chủ nó.
* Bảo mật - Với tính năng an toàn của Java, nó cho phép phát triển các hệ thống không có virus, giả mạo. Các kỹ thuật xác thực dựa trên mã hóa khóa công khai.
* Kiến trúc trung lập – Trình biên dịch Java tạo ra một định dạng tệp đối tượng kiến ​​trúc trung lập, làm cho mã biên dịch được thực thi trên nhiều bộ vi xử lý, với sự hiện diện của hệ điều hành Java.
* Portable – Là kiến ​​trúc trung lập và không bị phụ thuộc làm cho Java có thể mang đi dễ dàng. Trình biên dịch trong Java được viết bằng ANSI C đó là một tập hợp con POSIX.
* Mạnh mẽ – Java nỗ lực để loại trừ các tình huống dễ bị lỗi bằng cách kiểm tra lỗi thời gian biên dịch và kiểm tra thời gian chạy.
* Đa luồng – Với tính năng đa luồng của Java, có thể viết các chương trình có thể thực hiện nhiều tác vụ đồng thời. Tính năng thiết kế này cho phép các nhà phát triển xây dựng các ứng dụng tương tác có thể chạy trơn tru.
* Thông dịch – Mã byte Java được dịch trực tiếp tới các máy tính gốc và không được lưu trữ ở bất cứ đâu. Quá trình phát triển nhanh hơn và phân tích hơn.
* Hiệu năng cao – Với việc sử dụng trình biên dịch Just-In-Time, Java cho phép thực hiện chương trình với hiệu năng cao.
* Phân tán – Java được thiết kế cho môi trường phân tán của internet.
* Năng động **–**Java được xem là năng động hơn C hoặc C ++ vì nó được thiết kế để thích nghi với môi trường đang phát triển. Các chương trình Java có thể mang một lượng lớn thông tin run-time, có thể được sử dụng để xác minh và giải quyết các truy cập vào các đối tượng trong thời gian chạy.

### 2. Ngôn ngữ lập trình JavaScript

JavaScript là ngôn ngữ lập trình phổ biến dùng để tạo ra các trang web tương tác. Được tích hợp và nhúng vào HTML giúp website trở nên sống động hơn. JavaScript đóng vai trò như một phần của trang web, thực thi cho phép Client-Side Script từ phía người dùng cũng như phía máy chủ (Nodejs) tạo ra các trang web động [3].

JavaScript còn là một ngôn ngữ lập trình thông dịch với khả năng hướng đến đối tượng. Là một trong 3 ngôn ngữ chính trong lập trình web và có mối liên hệ lẫn nhau để xây dựng một website sống động, chuyên nghiệp, bạn có thể nhìn tổng quan như sau:

* HTML: Cung cấp cấu trúc cơ bản, hỗ trợ trong việc xây dựng layout, thêm nội dung dễ dàng trên website.
* CSS: Được sử dụng để kiểm soát và hỗ trợ việc định dạng thiết kế, bố cục, style, màu sắc,…
* JavaScript: Tạo nên những nội dung “động” trên website.

A picture containing text, screenshot, font, graphics

Description automatically generated

Hình 1.2. JavaScrip

Cách thức hoạt động của JavaScript trên trang web

* JavaScript thường sẽ được nhúng trực tiếp vào một trang web hoặc được tham chiếu qua file .js riêng. JavaScript là ngôn ngữ từ phía client nên script sẽ được tải về máy client khi truy cập và được xử lý tại đó. Thay vì tải về máy server và sau khi xử lý xong mới phản hồi kết quả đến client.
* JS sẽ thực hiện từ phía client thay vì phía server.

### 3. Ngôn ngữ đánh dấu siêu văn bản HTML và ngôn ngữ thiết kế CSS

#### 3.1 Định nghĩa về HTML và CSS

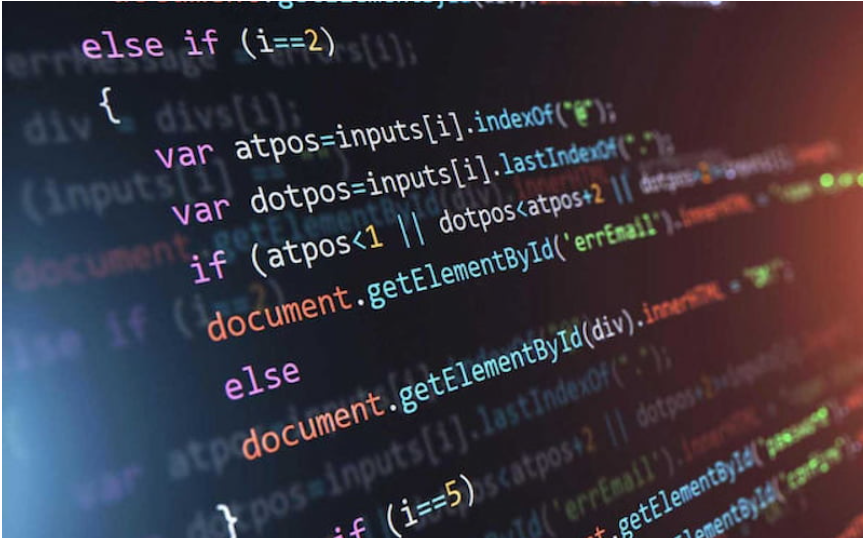
HTML (HyperText Markup Language) : là một ngôn ngữ đánh dấu được thiết kế ra để tạo nên các trang web, nghĩa là các mẩu thông tin được trình bày trên World Wide Web.

CSS (Cascading Style Sheets) : định nghĩa về cách hiển thị của một tài liệu HTML. CSS đặc biệt hữu ích trong việc thiết kế Web. Nó giúp cho người thiết kế dễ dàng áp đặt các phong cách đã được thiết kế lên bất kì page nào của website một cách nhanh chóng, đồng bộ.

#### 3.2 Ngôn ngữ đánh dấu siêu văn bản HTML

HTML được dùng với 3 mục đích chính là tạo nội dung, thiết kế giao diện và lập trình tương tác cho web. Cụ thể như sau [4]:

* Tạo nội dung cho trang web: HTML cho phép trang web của bạn có thể lưu trữ âm thanh, video, văn bản và một số ứng dụng khác.
* Thiết kế giao diện cho trang web: HTML5 thường được sử dụng để thiết kế giao diện website.Tuy nhiên, bạn chỉ có thể dùng HTML để tạo bộ khung sườn cho trang web mà thôi. Để thiết kế hoàn thiện một trang web, bạn cần sử dụng đến CSS để chỉnh sửa màu sắc, kích thước, vị trí của các biểu tượng và một số vấn đề phức tạp khác.
* Lập trình tương tác cho trang web: thông qua HTML, bạn có thể lập trình tương tác giữa người dùng với trang web. Để làm được điều này, bạn cần dùng code JavaScript. JavaScript sẽ tạo ra những hiệu ứng khi người dùng nhấp và di chuyển chuột trên website.



Hình 1.3. Lập trình tương tác cho trang web

#### 3.4 Bố cục và cấu trúc của một đoạn CSS

Bố cục của một đoạn CSS: chủ yếu dựa vào hình hộp với mỗi hộp chiếm những khoảng trống trên trang web với các thuộc tính chính như [5]:

* Padding: Là các không gian xung quanh nội dung (ví dụ: không gian xung quanh đoạn văn bản).
* Border: Là các đường nằm ngoài phần đệm.
* Margin: Là khoảng cách bao quanh phía ngoài của phần tử.

Cấu trúc của một đoạn CSS

* Thông thường, một đoạn CSS sẽ bao gồm các phần:vùng chọn {thuộc tính: giá trị; thuộc tính: giá trị;….. }
* Đoạn CSS sẽ được khai báo bằng vùng chọn, các thuộc tính, giá trị nằm trong dấu ngoặc nhọn. Mỗi thuộc tính là một giá trị riêng ở dạng số, hoặc chính là tên của các giá trị đã có trong danh sách của CSS.
* Quy tắc khai báo đó chính là: thuộc tính và giá trị cần cách nhau bằng dấu hai chấm, mỗi dòng khai báo thuộc tính cần có dấu chấm phẩy cuối cùng. Các thuộc tính không bị giới hạn ở một vùng chọn.
* Trong đó:
* Bộ chọn (Selector): mẫu để chọn phần tử HTML mà bạn muốn định nghĩa phong cách. Bạn có thể áp dụng các selector cho các trường hợp sau:
* Tất cả những phần tử được định dạng theo một dạng cụ thể nào đó, ví dụ phần tử tiêu đề h2.
* Thuộc tính id, class của phần tử.
* Các phần tử có mối liên quan với các phần tử khác trong hệ thống cây phân cấp tài liệu.
* Khai báo (Declaration): Khối khai báo có thể chứa một hoặc nhiều khai báo và chúng được phân tách với nhau bằng dấu chấm phẩy. Quy tắc khai báo CSS là chúng luôn phải kết thúc bằng dấu chấm phẩy, và khối khai báo phải nằm trong các dấu ngoặc móc.
* Thuộc tính (Properties): Thuộc tính là các cách thức mà bạn có thể tạo kiểu cho một phần tử HTML.
* Giá trị thuộc tính: Được nằm ở bên phải của thuộc tính. Việc lựa chọn một thuộc tính trong số đó phụ thuộc vào số lần xuất hiện của thuộc tính.

#### 3.5 Cách nhúng CSS vào website

Có ba cách nhúng CSS vào website:

* Nhúng CSS trực tiếp CSS vào tài liệu HTML (Inline CSS)
* Với cách này, chúng ta đặt mã CSS vào thẳng thuộc tính style của phần tử. Và với cách nhúng trực tiếp, mã CSS sẽ chỉ tác động lên chính phần tử đó.
* Để chỉ định nhiều quy tắc CSS, chúng ta có thể sử dụng dấu chấm phẩy để ngăn cách giữa các rules. Các rules này sẽ được đặt bằng cách sử dụng thuộc tính “style” kèm theo tên thuộc tính, giá trị thuộc tính.
* Nội tuyến (Internal CSS)
* Với cách nhúng nội tuyến, bạn cần dùng thẻ <style> để tạo ra khu vực viết CSS. Bạn có thể để CSS ở bất kỳ đâu trong HTML, nhưng nên đặt <style> trong thẻ <head>.
* Ngoại tuyến (External CSS)
* Trong External CSS, chúng ta sẽ sử dụng phần tử “link” để thêm các style sheet ở bên ngoài vào trong tài liệu HTML.
* Trước hết chúng ta cần tạo các rules (quy tắc) trong một file riêng có phần đuôi mở rộng là .css. Tiếp theo, bạn cần thêm file CSS này vào phần tử head trong tài liệu HTML.
* Đây là phương pháp phổ biến nhất để nhúng CSS vào tài liệu HTML. Với cách chèn này, các lập trình viên có thể viết CSS cho nhiều trang web khác nhau và bổ sung cùng một file CSS cho tất cả các trang tương tự.

### 4. Phần mềm Visual Studio Code

#### 4.1 Visual Studio Code là gì?

Visual Studio Code chính là ứng dụng cho phép biên tập, soạn thảo các đoạn code để hỗ trợ trong quá trình thực hiện xây dựng, thiết kế website một cách nhanh chóng. Visual Studio Code hay còn được viết tắt là VS Code. Trình soạn thảo này vận hành mượt mà trên các nền tảng như Windows, macOS, Linux. Hơn thế nữa, VS Code còn cho khả năng tương thích với những thiết bị máy tính có cấu hình tầm trung vẫn có thể sử dụng dễ dàng.



Hình 1.4. Visual Studio Code

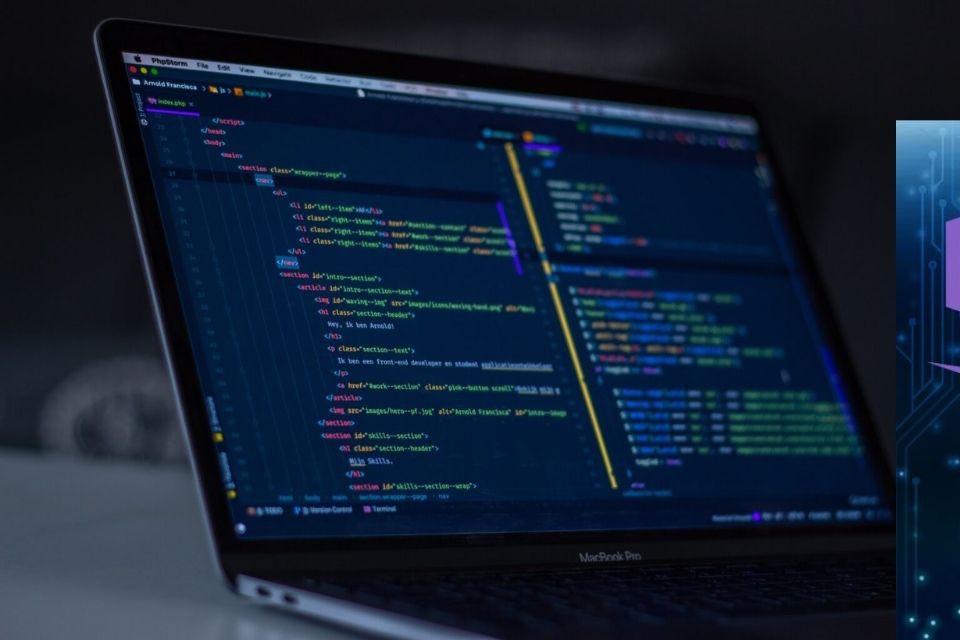
Visual Studio Code hỗ trợ đa dạng các chức năng Debug, đi kèm với Git, có Syntax Highlighting. Đặc biệt là tự hoàn thành mã thông minh, Snippets, và khả năng cải tiến mã nguồn. Nhờ tính năng tùy chỉnh, Visual Studio Code cũng cho phép các lập trình viên thay đổi Theme, phím tắt, và đa dạng các tùy chọn khác. Mặc dù trình soạn thảo Code này tương đối nhẹ, nhưng lại bao gồm các tính năng mạnh mẽ.

Dù mới được phát hành nhưng VSCode là một trong những Code Editor mạnh mẽ và phổ biến nhất dành cho lập trình viên. Nhờ hỗ trợ nhiều ngôn ngữ lập trình phổ biến, tích hợp đầy đủ các tính năng và khả năng mở rộng, nên VSCode trở nên cực kì thân thuộc với bất kì lập trình viên nào.

#### 4.2 Những ưu điểm nổi bật của Visual Studio Code

Visual Studio Code là gì được rất nhiều người tìm hiểu. Đây cũng là một trong các ứng dụng được dân IT “săn đón” và tải về và sử dụng rất nhiều. Visual Studio Code cũng luôn có những cải tiến và tạo ra đa dạng các tiện ích đi kèm từ đó giúp cho các lập trình viên sử dụng dễ dàng hơn. Trong đó có thể kể đến những ưu điểm sau:

* Đa dạng ngôn ngữ lập trình giúp người dùng thỏa sức sáng tạo và sử dụng như HTML, CSS, JavaScript, C++,…
* Ngôn ngữ, giao diện tối giản, thân thiện, giúp các lập trình viên dễ dàng định hình nội dung.
* Các tiện ích mở rộng rất đa dạng và phong phú.
* Tích hợp các tính năng quan trọng như tính năng bảo mật (Git), khả năng tăng tốc xử lý vòng lặp (Debug),…
* Đơn giản hóa việc tìm quản lý hết tất cả các Code có trên hệ thống.

****

Hình 1.5. Giao diện dòng lệnh trong Visual Studio Code

Visual Studio Code là một trong những trình biên tập Code rất phổ biến nhất hiện nay. Ứng dụng này cũng ngày càng chứng tỏ ưu thế vượt trội của mình khi so sánh với những phần mềm khác. Tuy bản miễn phí không có nhiều các tính năng nâng cao nhưng Visual Studio Code thực sự có thể đáp ứng được hầu hết nhu cầu cơ bản của lập trình viên.

## II. Giới thiệu về Json, Servlet, Netbeans và Glassfish

### 1. Json

JSON là một định dạng dữ liệu dùng cho việc chuyển đổi dữ liệu giữa các hệ thống phần mềm khác nhau (data interchange format). [JSON](https://en.wikipedia.org/wiki/JSON) đưa ra các quy tắc về các kiểu dữ liệu dạng đối tượng (JavaScript Object Notation) dựa trên một tập con của ngôn ngữ lập trình [Javascript](https://vi.wikipedia.org/wiki/JavaScript) (JavaScript Programming Language Standard ECMA-262) năm 1999.

Mục đích [6]:

* Với mục đích thiết kế cho việc chuyển đổi dữ liệu, [JSON](https://www.json.org/json-en.html) được cung cấp dưới định dạng chuỗi ký tự (text format) có cấu trúc giúp nó hoàn toàn độc lập với bất cứ các ngôn ngữ lập trình nào mà bạn quen thuộc cũng có thể dễ dàng tạo ra như: C, C++, C#, Java, JavaScript, Perl, Python…
* Nhờ sự mạnh mẽ của ngôn ngữ lập trình [Javascript](https://vi.wikipedia.org/wiki/JavaScript) đối với việc phát triển web phía client (client side), đặc biệt là kỹ thuật lập trình Ajax với hỗ trợ việc gửi và nhận dữ liệu một cách bất đồng bộ giữa client và server mà không ảnh hưởng tới trang, đã khiến cho [JSON](https://developer.mozilla.org/vi/docs/Web/JavaScript/Reference/Global_Objects/JSON) ngày càng trở lên phổ biến và thay thế dần [XML](https://vi.wikipedia.org/wiki/XML) trong việc chuyển đổi dữ liệu giữa các nền tảng.
* Những năm gần đây, cùng với sự hỗ trợ [JSON](https://www.json.org/json-en.html) một cách mạnh mẽ của các framework [Javascript](https://hoctoantap.com/category/lap-trinh/ngon-ngu-lap-trinh/javascript) như: [JQuery](https://hoctoantap.com/category/lap-trinh/framework/jquery), [Bootstrap](https://hoctoantap.com/category/lap-trinh/framework/bootstrap), [Angular](https://hoctoantap.com/category/lap-trinh/framework/angular-js), [React JS](https://hoctoantap.com/category/lap-trinh/framework/react-js), Vue JS,.. đã khiến cho việc phát triển view truyền thống (kết hợp giữa ngôn ngữ lập trình phía server với html, css, [Javascript](https://hoctoantap.com/category/lap-trinh/ngon-ngu-lap-trinh/javascript) như [jsp](https://vi.wikipedia.org/wiki/JSP), php, asp, [asp.net Web Forms](https://hoctoantap.com/category/lap-trinh/framework-net/asp-net-web-forms), [asp.net mvc](https://hoctoantap.com/category/lap-trinh/framework-net/asp-net)) ngày càng mất dần đi vị trí của nó. Chúng làm cho việc phát triển web phía Front End trở lên hoàn toàn độc lập với việc phát triển Back End. Ngày nay các lập trình viên [Backend](https://en.wikipedia.org/wiki/Front_end_and_back_end) sẽ chủ yếu tập trung xử lý nghiệp vụ và cung cấp các API trả về dữ liệu dạng JSON, các lập trình viên phía [Front End](https://en.wikipedia.org/wiki/Front_end_and_back_end) sẽ chủ yếu dựa vào dữ liệu JSON nhận được để xử lý giao diện.

### 2. Servlet

Servlet có thể được mô tả bằng nhiều cách, tùy thuộc vào ngữ cảnh:

* Một công nghệ được sử dụng để tạo ra ứng dụng web.
* Một API cung cấp các interface và lớp bao gồm các tài liệu.
* Một thành phần web được triển khai trên máy chủ để tạo ra trang web động. Có nhiều interface và các lớp trong API servlet như Servlet, GenericServlet, HttpServlet, ServletRequest, ServletResponse, …
* Java Servlet là chương trình chạy trên một Web hoặc ứng dụng máy chủ (Application Server). Nó hoạt động như một lớp trung gian giữa một yêu cầu đến từ một trình duyệt Web hoặc HTTP khách (Client) khác và cơ sở dữ liệu hoặc các ứng dụng trên máy chủ HTTP (HTTP Server) với nhiệm vụ chính là giúp thực thi câu lệnh một cách độc lập giúp kết nối các lớp với nhau.

A diagram of a computer

Description automatically generated with medium confidence

Hình 1. 6. Servlet

Công dụng:

* Nhận client request và lấy thông tin từ request: Đọc dữ liệu rõ ràng do khách hàng (trình duyệt) gửi.
* Xử lý nghiệp vụ và phát sinh chuyên môn (bằng cách truy cập database): Quá trình xử lý dữ liệu và tạo ra các kết quả này có thể yêu cầu nói chuyện với một cơ sở dữ liệu, thực hiện một cuộc gọi RMI hoặc CORBA, gọi một dịch vụ Web, hoặc tính trực tiếp phản hồi.
* Tạo và gửi request đến client hoặc tại request mới đến Servlet mới hoặc JSP mới: gửi dữ liệu rõ ràng tới khách hàng (trình duyệt) dưới nhiều định dạng, gửi phản hồi HTTP ẩn cho khách hàng (trình duyệt). Điều này bao gồm nói với trình duyệt hoặc các trình khách khác loại tài liệu đang được trả về (ví dụ, HTML), thiết lập cookie và các tham số bộ nhớ đệm, và các tác vụ khác.

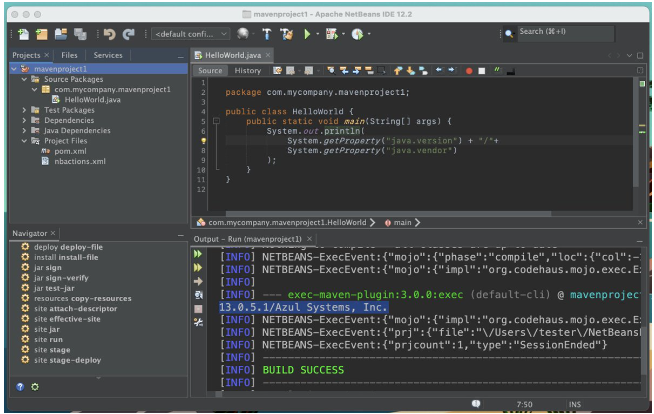
Môi trường làm việc

* Một Servlet chính là một lớp Java nên nó cần được thực thi trên máy ảo Java (gọi là JVM) thông qua một dịch vụ có tên là Servlet engine.
* Servlet engine sẽ thực hiện tải lớp Servlet đầu tiên mà nó được yêu cầu hoặc tại thời điểm khi servlet engine bắt đầu.
* Servlet sẽ ngừng tải để tập trung nguồn lực xử lý các yêu cầu khi Servlet engine bị dừng hoặc tắt.

### 3. Netbeans

Netbeans được biết đến là một công cụ hỗ trợ lập trình cũng như viết mã code hoàn toàn miễn phí, chủ yếu được tin dùng bởi các lập trình viên xây dựng và phát triển phần mềm Java.

Để mang đến sự tiện lợi tối đa trong quá trình sử dụng, Netbeans hiện đã hỗ trợ 3 hệ điều hành phổ biến nhất trên máy tính là Windows, macOS và Linux. Nhưng dung lượng của công cụ này khá nặng nên bạn cần có một thiết bị với cấu hình cao để có thể tận hưởng trải nghiệm tốt nhất.



Hình 1.7. Netbeans

Những tính năng chính

* Netbeans có khả năng hỗ trợ rất nhiều loại ngôn ngữ lập trình khác nhau như Javascript, Python hay C++… Từ đó, mọi lập trình viên đều có thể tin tưởng sử dụng công cụ này.
* IDE này còn sở hữu giao diện đơn giản, giúp tối ưu hóa sự dễ dàng trong khi lập trình. Sẽ không mất quá nhiều thời gian để bạn làm quen với Netbeans.
* Chỉnh sửa mã nguồn thông minh, gỡ lỗi mạng nội bộ từ xa,…

### 4. Glassfish

GlassFish là một máy chủ ứng dụng Java EE (Java Platform, Enterprise Edition) mã nguồn mở, được phát triển bởi Sun Microsystems và sau đó được Oracle mua lại. Nó cung cấp một môi trường chạy ứng dụng Java EE để triển khai và quản lý các ứng dụng doanh nghiệp phức tạp.

Dưới đây là một số điểm nổi bật về GlassFish:

* Hỗ trợ đầy đủ tiêu chuẩn Java EE: GlassFish tuân thủ các tiêu chuẩn của Java EE như Servlet, JSP, EJB, JPA, JMS và nhiều tiêu chuẩn khác. Cho phép phát triển và triển khai các ứng dụng Java Enterprise đa dạng và mạnh mẽ.
* Mã nguồn mở và cộng đồng sôi nổi: GlassFish là một dự án mã nguồn mở, có sự đóng góp và hỗ trợ từ cộng đồng phát triển rộng lớn. Nhằm đảm bảo tính ổn định, bảo mật và khả năng mở rộng của máy chủ ứng dụng.
* Quản lý và triển khai dễ dàng: GlassFish cung cấp giao diện quản lý web (GlassFish Admin Console) giúp quản lý, cấu hình và triển khai ứng dụng một cách dễ dàng. Nó cũng hỗ trợ công cụ dòng lệnh (Command-line Interface) cho việc tự động hóa và kịch bản hóa công việc quản lý.
* Khả năng mở rộng và tích hợp: GlassFish có khả năng mở rộng linh hoạt, cho phép thêm các plugin và mô-đun bổ sung để mở rộng chức năng và tích hợp với các công nghệ và dịch vụ khác nhau.
* Hiệu suất cao và tin cậy: GlassFish được tối ưu hóa để cung cấp hiệu suất cao và khả năng xử lý tốt cho các ứng dụng Java EE. Nó cung cấp cơ chế quản lý tài nguyên thông minh để tối ưu hóa việc sử dụng bộ nhớ và CPU.
* Hỗ trợ công cụ phát triển: GlassFish tương thích với các công cụ phát triển phổ biến như NetBeans và Eclipse IDE. Điều này giúp nhà phát triển tạo, kiểm thử và triển khai ứng dụng một cách thuận tiện.

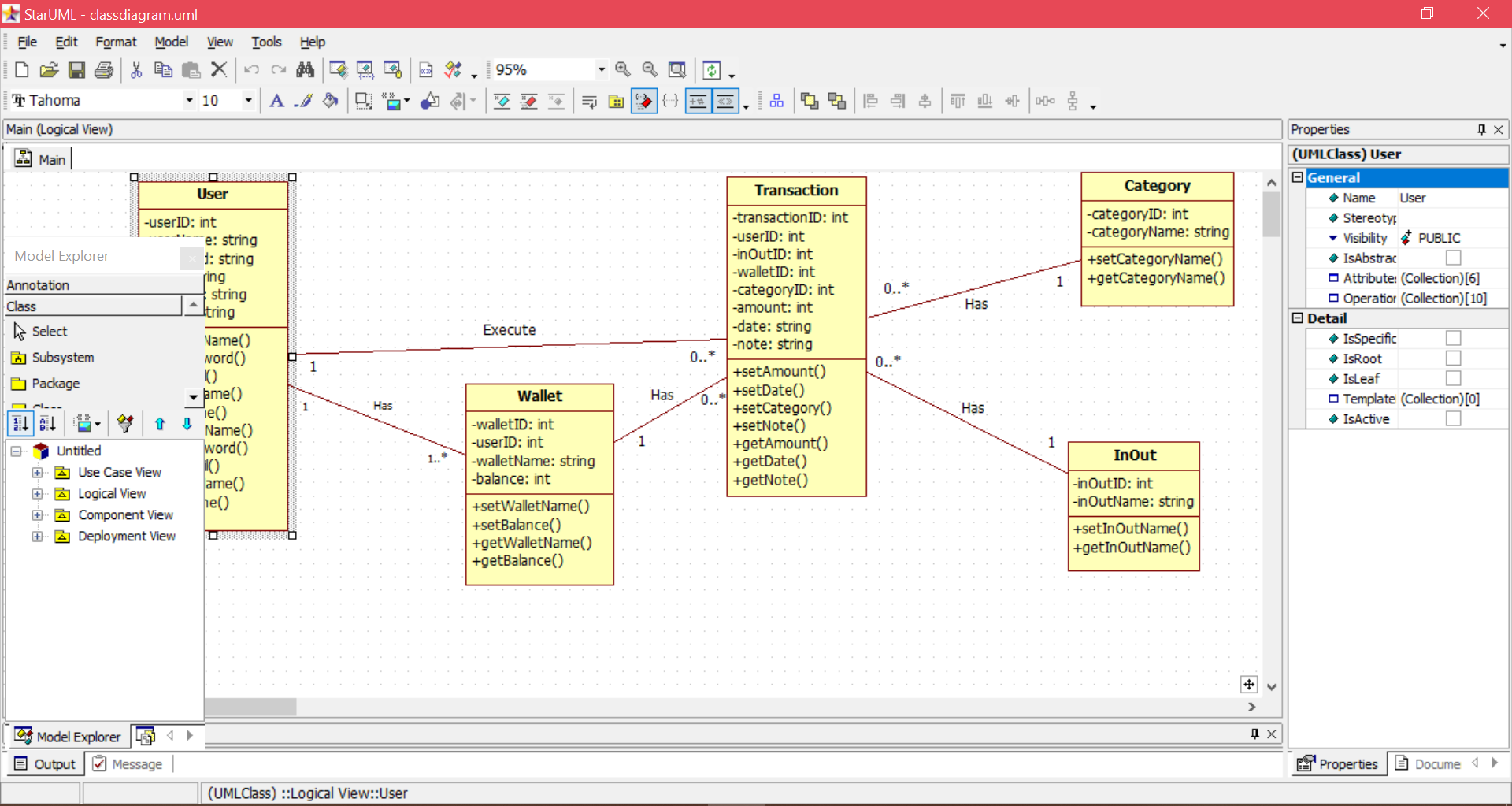
Tóm lại, GlassFish là một máy chủ ứng dụng Java EE mã nguồn mở, mạnh mẽ và linh hoạt, được phát triển và hỗ trợ bởi một cộng đồng lớn. Nó cung cấp một môi trường phát triển và triển khai ứng dụng Java Enterprise đáng tin cậy và hiệu quả.

## III. Một số công cụ hỗ trợ thiết kế phần mềm khác

### 1. StarUML

StarUML là một trình tạo mô hình phần mềm mạnh mẽ, được tích hợp nhiều tính năng và chức năng hữu ích. Ứng dụng hỗ trợ tiêu chuẩn UML 2.x và cho phép bạn tạo và sử dụng 11 loại sơ đồ UML khác nhau, bao gồm Ca sử dụng, Thành phần, Trình tự, Giao tiếp, Sơ đồ hồ sơ, Lớp, Đối tượng, Triển khai, Cấu trúc hỗn hợp, Statechart và Hoạt động.

StarUML cho phép bạn dễ dàng tạo Sơ đồ mối quan hệ thực thể. Đây là những sơ đồ được sử dụng rộng rãi cho các hoạt động mô hình hóa cơ sở dữ liệu. Hơn nữa, ứng dụng cũng đơn giản hóa việc tạo và kết nối các phần tử, chẳng hạn như giao diện hỗ trợ, các lớp con, v.v. StarUML hỗ trợ Retina Display và cung cấp cho bạn khả năng vẽ sơ đồ dưới dạng hình ảnh có độ phân giải cao (JPEG và PNG). Hơn nữa, chương trình cho phép bạn tìm và cài đặt nhiều tiện ích mở rộng mạnh mẽ thông qua Trình quản lý tiện ích mở rộng tích hợp.

****

Hình 1.8. StarUML - Mô hình phần mềm tiên tiến

StarUML cho phép bạn dễ dàng tạo mã tùy chỉnh bằng cách sử dụng các mẫu khác nhau. Tất cả các ngôn ngữ lập trình phổ biến đều được hỗ trợ, bao gồm C ++, Java và C #. Chương trình cho phép bạn tạo mã nguồn trực tiếp từ các mô hình của mình hoặc sử dụng kỹ thuật đảo ngược để xây dựng mô hình tạo thành mã nguồn của bạn. Bạn có thể cài đặt tiện ích mở rộng để xử lý mã nguồn từ Trình quản lý tiện ích mở rộng. Ngoài ra, bạn có thể viết các tiện ích mở rộng của riêng mình bằng CSS3, Node.js. HTML5 hoặc JavaScript.

StarUML cho phép bạn xuất bản Tài liệu HTML trên Web để chia sẻ các mô hình của bạn với các nhà phát triển, nhà phân tích và kiến ​​trúc sư phần mềm từ khắp nơi trên thế giới. Các tài liệu HTML đã tạo được hỗ trợ bởi phần lớn các trình duyệt web.

StarUML cho phép bạn xuất sơ đồ của mình dưới dạng tệp PDF để In sạch. Chương trình cho phép bạn chọn bố cục trang (ngang hoặc dọc) và kích thước trang (Letter, A4, A3, B4, B5, v.v.). Tất cả các sơ đồ có thể được xuất cùng một lúc, hoặc từng cái một.

Các ưu điểm của StarUML

* Cung cấp hỗ trợ cho các phần mở rộng markdown
* Cho phép bạn xuất sơ đồ của mình dưới dạng tệp PDF để In sạch.

### 2. Rational Rose

Rose hỗ trợ để xây dựng các biểu đồ UML mô hình hoá các lớp, các thành phần và mối quan hệ của chúng trong hệ thống một cách trực quan và thống nhất.

Nó cho phép mô tả chi tiết hệ thống bao gồm những cái gì, trao đổi tương tác với nhau và hoạt động như thế nào để người phát triển hệ thống có thể sử dụng mô hình như kế hoạch chi tiết cho việc xây dựng hệ thống.

Rose còn hỗ trợ rất tốt trong giao tiếp với khách hàng và làm hồ sơ, tài liệu cho từng phần tử trong mô hình.

Rose hỗ trợ cho việc kiểm tra tính đúng đắn của mô hình, thực hiện việc chuyển bản thiết kế chi tiết sang mã chương trình trong một ngôn ngữ lập trình lựa chọn và ngược lại, mã chương trình có thể chuyển trở lại yêu cầu hệ thống. Rose hỗ trợ phát sinh mã khung chương trình trong nhiều ngôn ngữ lập trình khác nhau như: C++, Java, Visual Basic, Oracle 8, v.v.

Tổ chức mô hình hệ thống thành một hay nhiều tệp, được gọi là đơn vị điều khiển được. Cho phép phát triển song song các đơn thể điều khiển được của mô hình

Hỗ trợ mô hình dịch vụ nhiều tầng (ba tầng) và mô hình phân tán, cơ chế khách/chủ (Client/Server).

Cho phép sao chép hay chuyển dịch các tệp mô hình, các đơn vị điều khiển được giữa các không gian làm việc khác nhau theo cơ chế “ánh xạ đường dẫn ảo” (Virtual Path Map),

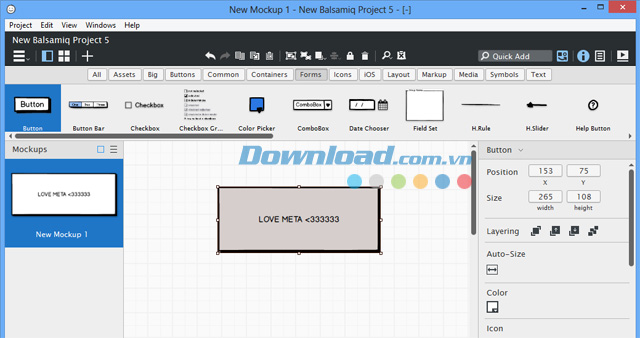
Cho phép quản lý mô hình và tích hợp với những hệ thống điều khiển chuẩn, Rose cung cấp khả năng tích hợp với ClearCase và Microsoft Visual SourceSafe, v.v.

Sử dụng các bộ tích hợp mô hình (Model Integrator) để so sánh và kết hợp các mô hình, các đơn vị điều khiển được với nhau.

Bản thân UML không định nghĩa quá trình phát triển phần mềm, nhưng UML và Rose hỗ trợ rất hiệu quả trong cả quá trình xây dựng phần mềm.

### 3. Balsamiq mockup

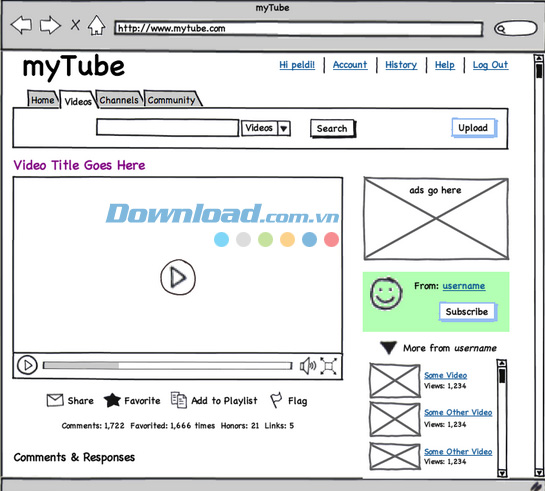
Balsamiq Mockups là một ứng dụng giúp phác họa ý tưởng thiết kế website thật nhanh chóng với giao diện đơn giản, dễ sử dụng.



Hình 1.9. Giao diện phần mềm Balsamiq Mockups

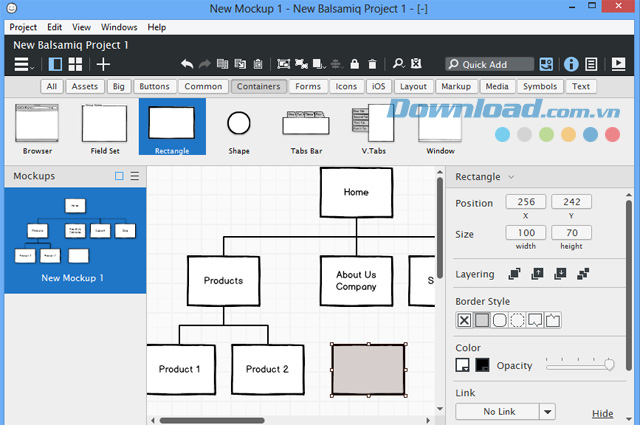
Tính năng của phần mềm thiết kế bố cục website Balsamiq Mockups

* Mockup Balsamiq hỗ trợ để phát thảo ý tưởng về giao diện khi thiết kế web và file xuất ra dưới dạng định dạng PNG, Bmml,... nên chất lượng hình ảnh cực tốt. Các Mockup sẽ giống như đang vẽ bằng tay nên người khác có thể hiểu ngay đó là khung bố cục chứ không phải bản thiết kế cuối.
* Việc sử dụng Balsamiq Mockups với đồ họa đơn giản sẽ giúp cải thiện vấn đề làm việc nhóm. Bằng việc phác thảo nhanh các ý tưởng thông qua các đường vẽ tuy thô nhưng chính điều ấy lại giúp mọi người đoán ra ngay được ý tưởng của bạn mà không phải ngồi mất công suy nghĩ như những bản vẽ phức tạp và nó đặc biệt hữu ích khi sử dụng làm việc với khách hàng.



Hình 1.10. Phác thảo ý tưởng nhanh mà hiệu quả

Sử dụng phần mềm rất đơn giản, các công cụ tổng hợp có sẵn như thanh tiêu đề, hình ảnh, video, banner hay các nút điều khiển... đều được tích hợp sẵn, bạn chỉ việc lựa chọn bằng cách kích đúp chuột trái vào biểu tượng hoặc kéo thả chúng xuống vùng làm việc rồi sắp xếp theo ý muốn.



Hình 1.11*.* Thao tác kéo thả trong Balsamiq Mockups

Với công cụ này, hoàn toàn có thể xây dựng một website từ đơn giản đến phức tạp, từ trang chung cho đến các trang chi tiết một cách dễ dàng và nhanh chóng.

### 4. Case studio 2

CASE Studio 2 là công cụ vẽ mô hình dữ liệu chuyên nghiệp và có thể tùy chỉnh hỗ trợ những lập trình viên hay nhân viên thiết kế dữ liệu trong việc vẽ mô hình Entity Relationship Diagrams (ERD) và Data Flow Diagrams (DFD) cũng như tạo script SQL cho nhiều cơ sở dữ liệu một cách tự động. Phần mềm hỗ trợ đầy đủ cho 20 cơ sở dữ liệu như Oracle, DB2, MS SQL, Sybase, MySQL, Firebird, PostgreSQL vvv.

A blue and orange box with white text

Description automatically generated with low confidence

Hình 1. 12. Case Studio 2

Chức năng chính của CASE Studio 2 : tạo mô hình ERD từ mã script SQL (DDL), Đảo mã từ file HTML cụ thể hay file dữ liệu RTF, xuất mô hình Data Flow Diagrams thành file quản lý định dạng XML, Templates editor và còn nữa.

# CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## I. Khảo sát hệ thống

### 1. Khảo sát sơ bộ

#### 1.1 Mục tiêu

(1) Hiểu rõ hơn về trang web, giao diện và các tính năng của nó để cung

cấp hỗ trợ và giải đáp cho người dùng dễ dàng.

(2) Kiểm tra tính năng tìm kiếm để đảm bảo cho người dùng có thể tìm

kiếm sản phẩm và dịch vụ theo ý muốn.

(3) Dựa vào các đánh giá khảo sát nhu cầu sử dụng để sửa đổi, cải thiện, nâng cấp website, để đáp ứng được phần lớn yêu cầu của người sử dụng.

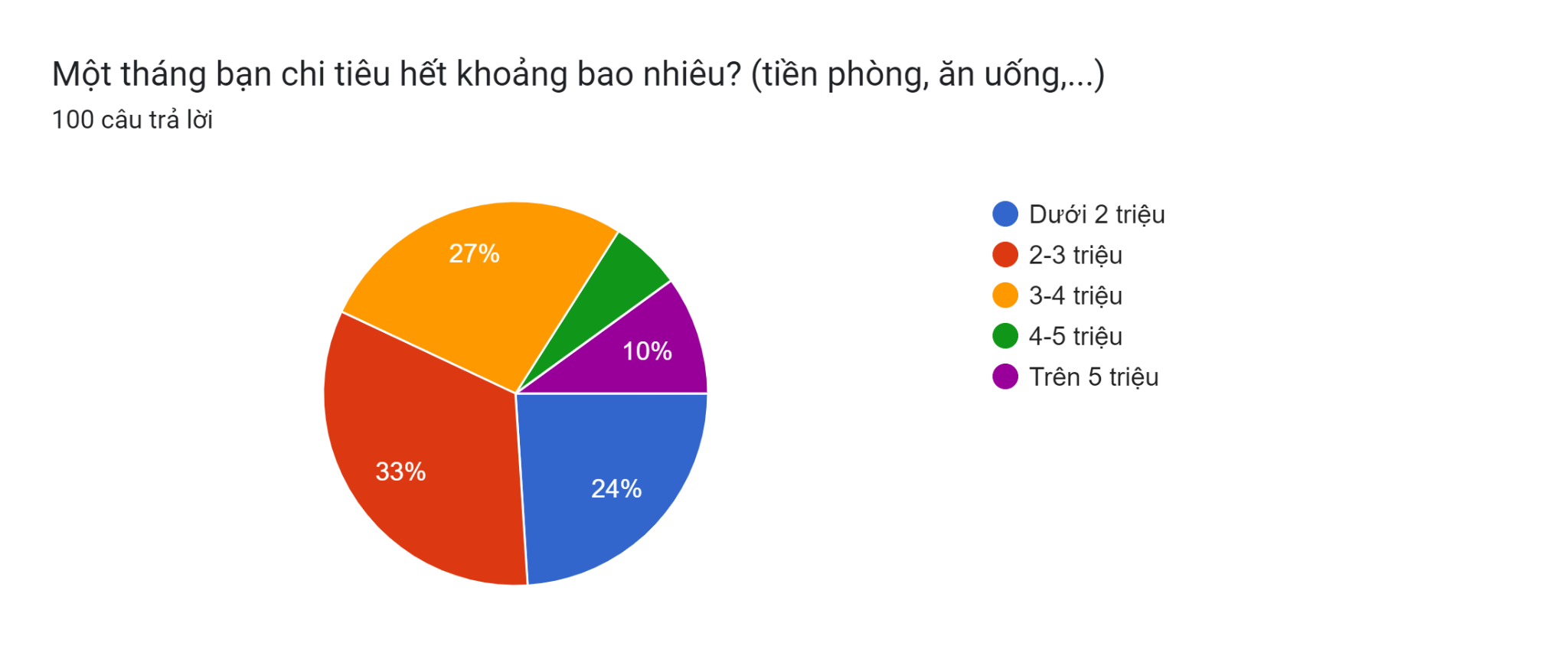
#### 1.2 Phương pháp

Khảo sát người dùng

* Liên kết đến google form: [**https://forms.gle/3xVQJ4svXhiS4k5Q9**](https://forms.gle/3xVQJ4svXhiS4k5Q9)
* Quan sát:

Thông qua việc khảo sát 100 người, chúng em có bản phân tích nhu cầu chi tiêu của người dùng như sau:

* Về mức chi tiêu trung bình / tháng có biểu đồ như sau:

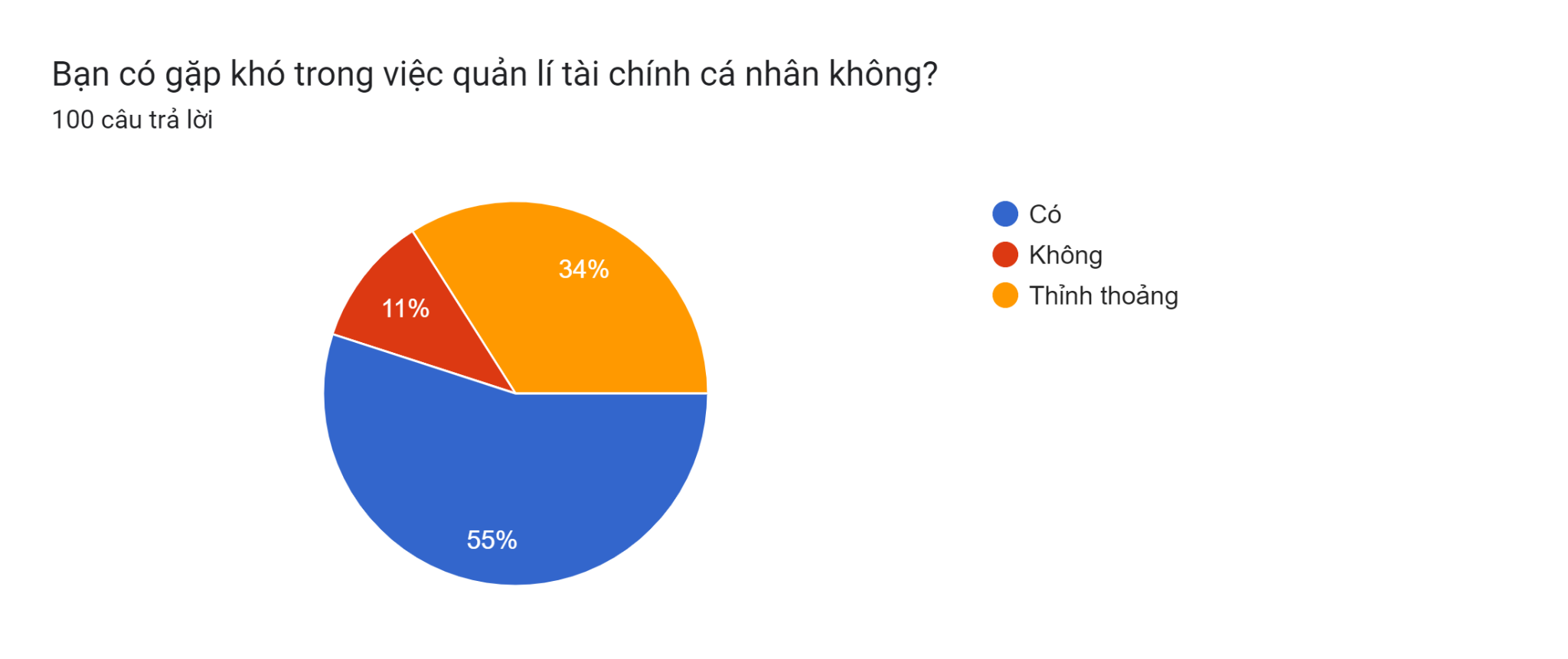


Hình 2.1. Biểu đồ mức chi tiêu trung bình/tháng

Thông tin trên cho thấy phân phối số tiền chi tiêu hàng tháng của những người trả lời theo các khoảng giá trị.

Dưới đây là phân tích chi tiết:

* Dưới 2 triệu (24%): 24% người trong nhóm đã tiêu hết dưới 2 triệu trong một tháng. Điều này cho thấy rằng một phần khá lớn người tiêu dùng có chi tiêu tương đối thấp và có thể đang sống trong điều kiện tài chính hạn chế.
* 2-3 triệu (33%): 33% người trong nhóm đã tiêu tiền từ 2 đến 3 triệu trong một tháng. Nhóm này chiếm tỷ lệ lớn nhất trong số các phạm vi tiêu tiền được cung cấp. Tỉ lệ này cho chúng ta thấy rằng mức tiêu tiền trung bình của nhóm người này nằm trong khoảng từ 2 đến 3 triệu chiếm tỉ lệ cao nhất.
* 3-4 triệu (27%): 27% người trong nhóm đã tiêu tiền từ 3 đến 4 triệu trong một tháng. Nhóm này có một tỷ lệ khá lớn, gần bằng tỷ lệ của nhóm 2-3 triệu. Điều này cho thấy rằng một phần đáng kể người tiêu dùng có khả năng tiêu tiền hơn so với mức trung bình.
* 4-5 triệu (6%): Chỉ có 6% người trong nhóm đã tiêu tiền từ 4 đến 5 triệu trong một tháng. Tỷ lệ này rất thấp so với các phạm vi khác, cho thấy rằng số người có khả năng tiêu tiền trong mức này là ít.
* Trên 5 triệu (10%): 10% người trong nhóm đã tiêu tiền trên 5 triệu trong một tháng. Đây là một tỷ lệ khá cao so với mức tiêu tiền cao nhất trong số các phạm vi được cung cấp. Điều này cho thấy rằng một số người tiêu dùng trong nhóm có khả năng chi tiêu cao hơn và có thể đang sống trong điều kiện tài chính ổn định hoặc dư dả hơn so với các nhóm được cung cấp trên.
* Tóm lại, phân tích trên dựa trên 100 câu trả lời cho thấy có sự đa dạng về mức tiêu tiền trong nhóm người này. Tuy nhiên, tỷ lệ lớn nhất nằm trong khoảng từ 2 đến 3 triệu (chiếm đến 33%) và có một số người có khả năng chi tiêu cao hơn.
* Khảo sát về việc có gặp khó khăn trong quản lý tài chính cá nhân không có biểu đồ như sau:

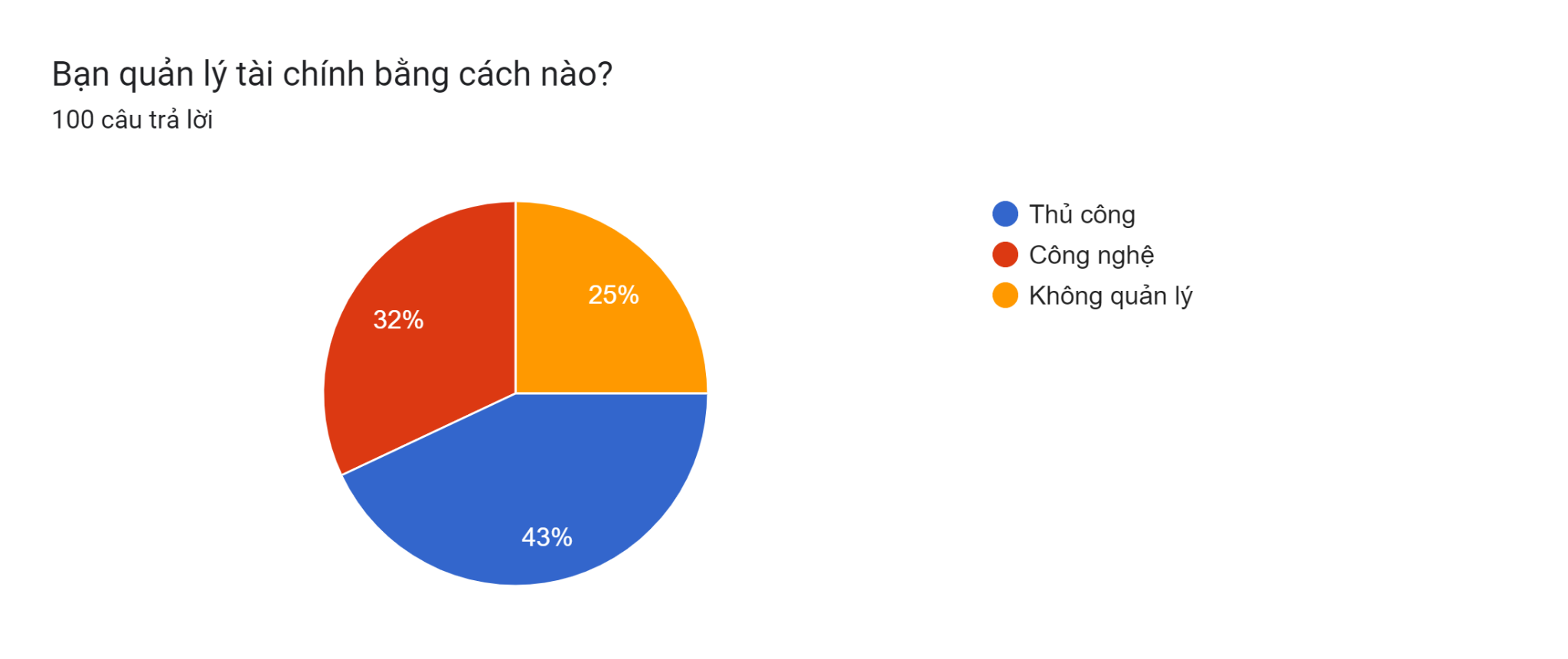


Hình 2.2. Biểu đồ khảo sát việc gặp khó khăn

Dựa trên thông tin đã thu thập, dưới đây là phân tích về khả năng quản lý tài chính cá nhân dựa trên 100 câu trả lời.

Dưới đây là phân tích chi tiết:

* Có khó khăn trong việc quản lý tài chính cá nhân (55%): 55% người trong nhóm đã gặp khó khăn trong việc quản lý tài chính cá nhân. Điều này cho thấy rằng một tỷ lệ lớn người tiêu dùng trong nhóm này đang gặp vấn đề về quản lý tài chính cá nhân và có thể đang gặp khó khăn trong việc duy trì ngân sách và tiết kiệm tiền.
* Không gặp khó khăn trong việc quản lý tài chính cá nhân (11%): Chỉ có 11% người trong nhóm không gặp khó khăn trong việc quản lý tài chính cá nhân. Đây là một tỷ lệ rất thấp, cho thấy rằng số người có khả năng quản lý tài chính cá nhân hiệu quả trong nhóm này là ít.
* Thỉnh thoảng gặp khó khăn trong việc quản lý tài chính cá nhân (34%): 34% người trong nhóm thỉnh thoảng gặp khó khăn trong việc quản lý tài chính cá nhân. Điều này cho thấy rằng một phần đáng kể người tiêu dùng trong nhóm này có thể gặp khó khăn đôi khi và có thể không ổn định trong việc duy trì một ngân sách và tiết kiệm tiền.
* Tóm lại, phân tích trên dựa trên 100 câu trả lời cho thấy tỷ lệ lớn người tiêu dùng trong nhóm này gặp khó khăn trong việc quản lý tài chính cá nhân. Điều này có thể chỉ ra rằng cần cung cấp thêm kiến thức và kỹ năng hoặc các công cụ về quản lý tài chính để giúp nhóm này đạt được sự ổn định và tiết kiệm trong việc quản lý tiền bạc.
* Khảo sát về cách quản lý tài chính có biểu đồ như sau:



Hình 2.3. Biểu đồ khảo sát về cách quản lý tài chính

Dựa trên thông tin đã thu thập, dưới đây là phân tích về cách mà một nhóm người quản lý tài chính cá nhân của họ dựa trên 100 câu trả lời.

* Quản lý tài chính thủ công (43%): 43% người trong nhóm quản lý tài chính cá nhân bằng cách thủ công. Điều này cho thấy rằng một tỷ lệ lớn người tiêu dùng trong nhóm này sử dụng các phương pháp truyền thống như ghi chép tay, sổ sách hoặc bảng tính để theo dõi và quản lý tài chính của mình.
* Quản lý tài chính bằng công nghệ (32%): 32% người trong nhóm quản lý tài chính cá nhân bằng cách sử dụng công nghệ. Điều này có thể bao gồm việc sử dụng ứng dụng di động, phần mềm quản lý tài chính hoặc các công cụ trực tuyến để theo dõi, phân loại và quản lý tài chính cá nhân.
* Không quản lý tài chính (25%): 25% người trong nhóm không quản lý tài chính cá nhân của mình. Điều này cho thấy rằng một tỷ lệ khá lớn người tiêu dùng trong nhóm này không áp dụng bất kỳ phương pháp nào để theo dõi và quản lý tài chính cá nhân của mình.

Tóm lại, phân tích dựa trên 100 câu trả lời cho thấy có sự đa dạng về cách mà nhóm người này quản lý tài chính cá nhân. Mặc dù một số người sử dụng các công nghệ hiện đại để giúp quản lý tài chính, nhưng phần lớn người dùng vẫn dựa vào phương pháp thủ công hoặc không quản lý tài chính. Điều này cho thấy cần cung cấp thêm sự giúp đỡ và tư vấn về quản lý tài chính để tăng cường khả năng quản lý tiền bạc của nhóm này.

* Khảo sát về nhu cầu sử dụng một ứng dụng quản lý tài chính cá nhân có biểu đồ như sau:



Hình 2.4. Biểu đồ khảo sát về nhu cầu sử dụng một ứng dụng

quản lý tài chính cá nhân

Dựa trên thông tin thu thập được, dưới đây là phân tích về sự sẵn sàng sử dụng một ứng dụng quản lý tài chính cá nhân của một nhóm người cụ thể dựa trên 100 câu trả lời mà họ đã cung cấp.

* Sẵn sàng sử dụng ứng dụng quản lý tài chính cá nhân (82%): 82% người trong nhóm sẵn sàng sử dụng một ứng dụng quản lý tài chính cá nhân. Điều này cho thấy rằng một tỷ lệ lớn người tiêu dùng trong nhóm này đã nhận thức và có ý định sử dụng công nghệ để giúp quản lý tài chính cá nhân của họ.
* Không sẵn sàng sử dụng ứng dụng quản lý tài chính cá nhân (18%): Chỉ có 18% người trong nhóm không sẵn sàng sử dụng một ứng dụng quản lý tài chính cá nhân. Đây là một tỷ lệ rất thấp, cho thấy rằng số người không quan tâm hoặc không muốn sử dụng công nghệ để quản lý tài chính cá nhân.

Tóm lại, phân tích trên dựa trên 100 câu trả lời cho thấy tỷ lệ lớn người tiêu dùng trong nhóm này sẵn sàng sử dụng một ứng dụng quản lý tài chính cá nhân. Điều này cho thấy xu hướng tích cực trong việc chấp nhận công nghệ và mong muốn sử dụng các công cụ đơn giản và tiện lợi để quản lý tài chính cá nhân.

Tính tiện lợi và truy cập dễ dàng: Sự sẵn sàng sử dụng ứng dụng quản lý tài chính cá nhân có thể được giải thích bởi tính tiện lợi và khả năng truy cập dễ dàng của công nghệ di động. Với một ứng dụng quản lý tài chính, người dùng có thể theo dõi, phân loại và quản lý tài chính của mình một cách thuận tiện từ điện thoại di động hoặc máy tính bảng.

Tính linh hoạt và cá nhân hóa: Các ứng dụng quản lý tài chính cá nhân thường cung cấp tính linh hoạt và khả năng cá nhân hóa, cho phép người dùng điều chỉnh theo nhu cầu và mục tiêu tài chính của riêng họ. Điều này giúp người dùng tạo ra kế hoạch tài chính, theo dõi ngân sách và đặt mục tiêu tiết kiệm một cách hiệu quả.

Tính an toàn và bảo mật: Một ứng dụng quản lý tài chính cá nhân tốt thường có các biện pháp bảo mật và mã hóa mạnh mẽ để bảo vệ thông tin tài chính của người dùng. Điều này giúp tăng cường sự tin tưởng và sẽ làm giảm các lo ngại về việc chia sẻ thông tin cá nhân và tài chính với một ứng dụng.

Nhu cầu kiểm soát tài chính: Sự sẵn sàng sử dụng ứng dụng quản lý thu chi cá nhân có thể cho thấy nhu cầu kiểm soát và quản lý tài chính cá nhân trong nhóm người này. Người dùng có thể theo dõi chi tiêu, tạo ngân sách, tiết kiệm, câm nhắc để giảm chi phí các khoản không cần thiết và đầu tư một cách thông minh để đạt được mục tiêu tài chính của mình.

#### 1.3 Kết quả sơ bộ

Mô tả hệ thống:

* Tên dự án: Website quản lí thu chi cá nhân (Money smart).
* Loại website: Multi-page.
* Tổ chức:
* Admin:
* Quản lý hoạt động của trang web, quản lý 1 số thông tin cơ bản của trang web.
* Đăng nhập.
* Quản lý thông tin, tài khoản người dùng tương ứng có thể tìm kiếm, thêm, sửa, xóa tài khoản người dùng.
* Người dùng:
* Có thể truy cập vào trang web,tạo tài khoản, đăng nhập, xem thông tin trang web, theo dõi các khoản chi tiêu, tạo ví, điều chỉnh số dư, xem báo cáo chi tiêu.

### 2. Tài liệu đặc tả người dùng

#### 2.1 Khảo sát chi tiết

Các hoạt động của hệ thống

* Tạo ví chi tiêu:
* Mở trang web, đăng nhập vào hệ thống
* Chọn mục My wallet.
* Lựa chọn loại ví muốn tạo, lựa chọn ngân hàng.
* Lựa chọn mức phí sử dụng.
* Chọn Đăng ký là xong.
* Xem thống kê chi tiêu:
* Mở trang web, đăng nhập vào hệ thống.
* Chọn vào mục Report.
* Chọn thời gian cần xem thống kê.
* Nhấn gửi để xem chi tiết.
* Điều chỉnh số dư:
* Mở trang web, đăng nhập vào hệ thống
* Chọn biểu tượng 3 gạch trên cùng góc bên trái.
* Chọn mục My Wallet
* Nhập số dư mới
* Nhấn save để lưu lại thông tin
* Xem báo cáo chi tiêu:
* Chọn vào biểu tượng 3 gạch ở trên cùng góc bên trái.
* Chọn Categories.
* Muốn xem loại hình chi tiêu nào lựa chọn loại hình chi tiêu đó.

#### 2.2 Yêu cầu chức năng

Yêu cầu chức năng: Khách hàng có thể vào xem, tìm kiếm, đăng nhập,xem lịch sử chi tiêu, số dư. Admin có thể đăng nhập, quản lý tài khoản khách hàng, quản lý một số thông tin trang web.

#### 2.3 Yêu cầu phi chức năng

Yêu cầu phi chức năng:

* Trang web hỗ trợ cứu hộ hay trang quản trị hoạt động 24/24h, giao diện(UI) dễ nhìn có thiện cảm với người dùng ngoài ra các vấn đề thao tác của người dùng cần đơn giản dễ thao tác tránh cảm giác gây ức chế khi người dùng thao tác.
* Ổn định, xử lý nhanh, Tốc độ tải trang hay thực hiện công việc không được quá lâu, hiện các thông báo khi kết thúc thao tác công việc nào đó.
* Trang web có thể tương tích với các trình duyệt khác nhau các thiết bị truy cập khác nhau.
* An toàn, bảo mật:Đảm bảo an toàn thông tin người dùng. Các thao tác nghiệp vụ của người quản trị chỉ thực hiện được khi có quyền.

## II. Phân tích hệ thống

### 1. Mô hình hóa dữ liệu của hệ thống

Cách tìm lớp: Từ nhu cầu thực tế của người dùng để chuyển hóa thành dữ liệu

* Người dùng: Để phục vụ tốt nhất cho khách hàng chúng ta cần khách hàng cung cấp một số thông tin để tiện cho việc quản lý cũng như cung cấp dịch vụ phù hợp. Ta có bảng “User” để lưu trữ thông tin mà khách hàng cung cấp...
* Bảng “Wallet” cho phép người dùng tạo ví của họ cho mọi loại chi phí. Khi sử dụng tính năng này, người dùng có thể điều chỉnh số dư, theo dõi số dư để có một chi tiêu hợp lý.
* Bảng “Transaction” để lưu trữ thông tin các giao dịch gồm loại giao dịch, ngày giao dịch, số tiền và các ghi chú. Đây là một chức năng ghi nhật ký, đó là ghi lại bất kì hoạt động nào được thực hiện bởi người dùng. Điều này được sử dụng để cung cấp dữ liệu lịch sử có thể được sử dụng để đánh giá hoạt động trong quá khứ hoặc để lập kế hoạch cho mọi hoạt động trong tương lai.
* Bảng “CategoryTransaction” là các loại hình chi tiêu giúp người dùng có thể theo dõi các khoản chi tiêu của mình vào những loại hình gì: thực phẩm, các hóa đơn, bảo trì nhà, mua sắm, trang điểm, tiền lương, tiền lãi, đầu tư,...
* Bảng “InOutMoney” kiểm soát dòng tiền ra vào. Dòng tiền ra là các khoản chi phí, dòng tiền vào: lương, tiền lãi,...
* Kết luận: vậy ta có thể tổng kết các lớp có trong chương trình.

Các yêu cầu về dữ liệu:

1. User: userID, userName, password, email, fullname, phone

Mỗi tài khoản có 1 hoặc nhiều ví, mỗi ví thuộc về 1 tài khoản.

1. Wallet: walletID, walletName, balance

Mỗi ví tham gia 0, 1 hoặc nhiều giao dịch. Mỗi giao dịch chỉ thực hiện trên 1 ví

1. Transaction: transactionID, amount

Mỗi người dùng thực hiện 0, 1 hoặc nhiều giao dịch. Mỗi giao dịch được thực hiện bởi 1 người dùng.

1. CategoryTransaction: categoryID, categoryName

Mỗi loại hình chi tiêu có trong 0, 1 hoặc nhiều giao dịch. Mỗi giao dịch chỉ có 1 loại hình chi tiêu

1. InOutMoney: inOutID, inOutName

Mỗi trạng thái giao dịch có trong 0, 1 hoặc nhiều giao dịch. Mỗi giao dịch chỉ có 1 trạng thái giao dịch.

A picture containing text, screenshot, font, line

Description automatically generated

Hình 2.5. Mô hình hóa hệ thống

### 2. Thiết kế cơ sở dữ liệu

#### 2.1 Bảng CategoryTransaction

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Data Type | Not Null | Auto Increment |
| categoryID | int | YES | YES |
| categoryName | nvarchar(50) | YES | NO |

Bảng 1. Bảng CategoryTransaction

#### 2.2 Bảng User

|  |  |  |  |
| --- | --- | --- | --- |
| Column | data type | Not Null | Auto Increment |
| userID | int | YES | YES |
| userName | nvarchar(50) | YES | NO |
| password | nvarchar(50) | YES | NO |
| email | nvarchar(50) | YES | NO |
| fullName | nvarchar(50) | YES | NO |
| phone | nvarchar(12) | YES | NO |

Bảng 2. Bảng User

#### 2.3 Bảng Wallet

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Data Type | Not Null | Auto Increment |
| walletID | int | YES | YES |
| walletName | nvarchar(50) | YES | NO |
| balance | int | YES | NO |

Bảng 3. Bảng Wallet

#### 2.4 Bảng InOutMoney

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Data Type | Not Null | Auto Increment |
| inOutlID | int | YES | YES |
| inOutName | nvarchar(50) | YES | NO |

Bảng 4. Bảng InOutMoney

#### 2.5 Bảng Transaction

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Data Type | Not Null | Auto Increment |
| transactionID | int | YES | YES |
| amount | int | YES | NO |
| date | nvarchar(50) | YES | NO |
| note | nvarchar(50) | YES | NO |

Bảng 5. Bảng Transaction

#### 2.6 Thiết kế dữ liệu- Ánh xạ lớp sang bảng

**A picture containing text, diagram, screenshot, line

Description automatically generated**

Hình 2.6. Thiết kế ánh xạ cơ sở dữ liệu

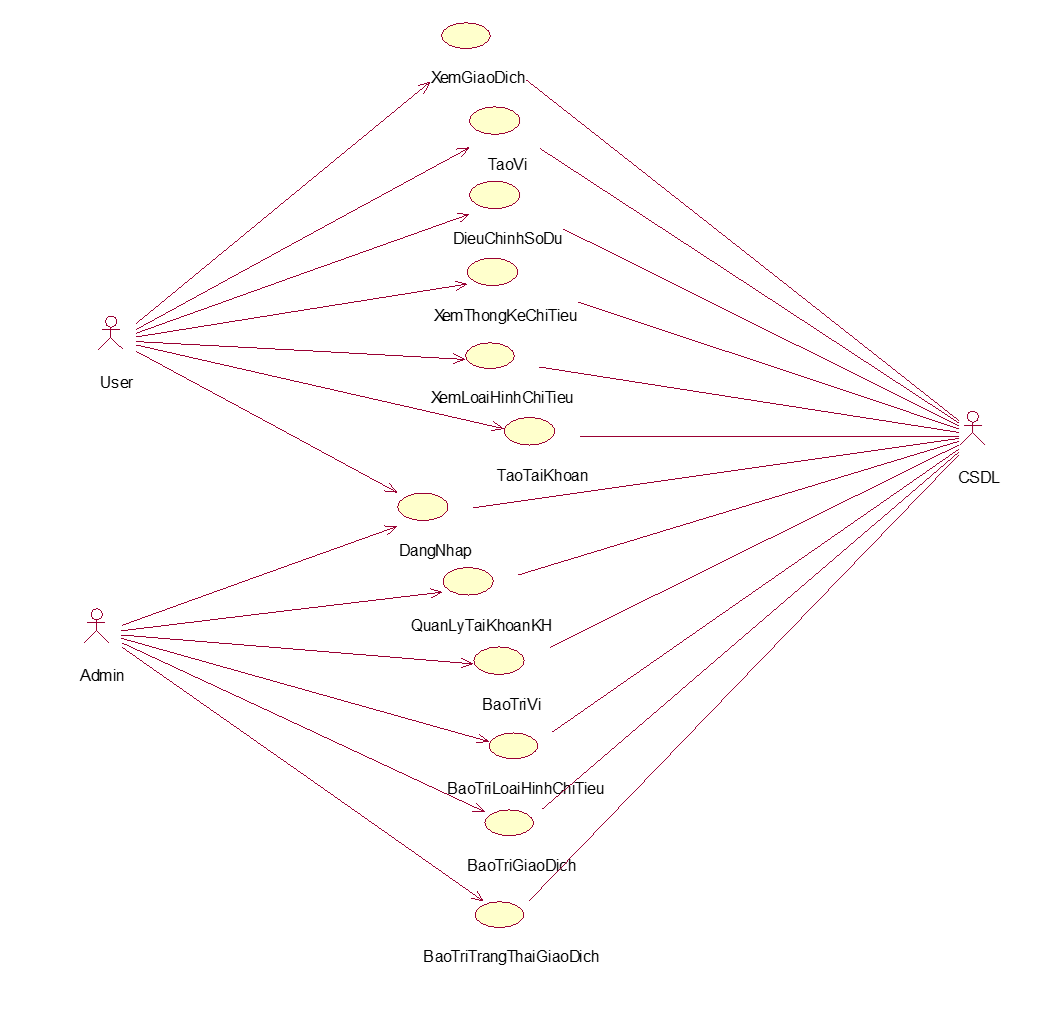
### 3. Biểu đồ use case

Các actor

* User
* Admin
* CSDL

Các use case

* Đăng nhập
* Tạo tài khoản
* Xem giao dịch
* Tạo ví
* Điều chỉnh số dư
* Xem thống kê chi tiêu
* Xem loại hình chi tiêu
* Quản lý tài khoản khách hàng
* Bảo trì ví
* Bảo trì loại hình chi tiêu
* Bảo trì giao dịch
* Bảo trì trạng thái giao dịch



Hình 2. 7. Sơ đồ usecase

Phân rã một số use case

* Use case phía frontend

A picture containing christmas tree, star

Description automatically generated

Hình 2.8. Use case phía frontend

* Xem giao dịch: use case này cho phép người dùng xem các thông tin của các giao dịch đã thực hiện.
* Tạo ví: use case này cho phép người dùng tạo ví của mình.
* Điều chỉnh số dư: use case này cho phép người dùng điều chỉnh số dư ví của mình.
* Xem thống kê chi tiêu: use case này cho phép người dùng xem thống kê chi tiêu của mình theo tháng và năm.
* Xem loại hình chi tiêu: use case này cho phép người dùng xem các loại hình chi tiêu.
* Tạo tài khoản: use case này cho phép người dùng tạo tài khoản của mình.
* Đăng nhập: use case này cho phép người dùng đăng nhập vào tài khoản của mình.
* Use case phía backend



Hình 2.9. Use case phía frontend

* Đăng nhập: use case này cho phép người quản trị đăng nhập vào tài khoản của mình.
* Quản lý tài khoản khách hàng: use case này cho phép người quản trị xem, sửa trạng thái, xóa các thông tin trong bảng User.
* Bảo trì ví: use case này cho phép người quản trị xem, sửa trạng thái, xóa các thông tin trong bảng Wallet.
* Bảo trì loại hình chi tiêu: use case này cho phép người quản trị xem, sửa trạng thái, xóa các thông tin trong bảng CategoryTransaction.
* Bảo trì giao dịch: use case này cho phép người quản trị xem, sửa trạng thái, xóa các thông tin trong bảng Transaction.
* Bảo trì trạng thái giao dịch: use case này cho phép người quản trị xem, sửa trạng thái, xóa các thông tin giao dịch trong bảng InOutMoney.

### 4. Mô tả chi tiết các use case

#### 4.1 Use case Tạo tài khoản

1. Tên use case: tạo tài khoản.
2. Mô tả vắn tắt: Use case này cho phép người dùng tạo tài khoản của mình.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case này bắt đầu khi người dùng kích vào nút register. Hệ thống hiển thị form đăng ký tài khoản ra màn hình.
2. Người dùng nhập thông tin và nhấn nút register. Hệ thống lưu thông tin người dùng vào bảng User. Thông báo đăng ký thành công. Use case kết thúc.

ii) Luồng rẽ nhánh:

1. Nếu để trống dữ liệu hoặc điền dữ liệu không đúng format và nhấn nút register. Hệ thống báo lỗi và quay lại bước 2 trong luồng cơ bản. Use case kết thúc.
2. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với CSDL thì hệ thống sẽ hiển thị một thông báo lỗi. Use case kết thúc.
3. Các yêu cầu đặc biệt:

Không có.

1. Tiền điều kiện:

Không có.

1. Hậu điều kiện:

Nếu use case này thực hiện thành công, hệ thống sẽ cập nhật bảng User trong cơ sở dữ liệu.

1. Điểm mở rộng:

Đăng nhập.

#### 4.2 Use case Đăng nhập

1. Tên use case: Đăng nhập.
2. Mô tả vắn tắt: use case này cho phép người dùng đăng nhập vào website để sử dụng các chức năng của trang web hoặc cho phép người quản trị đăng nhập để quản lý và bảo trì hệ thống.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case bắt đầu khi người dùng hoặc người quản trị kích chuột vào nút “Login” trên thanh menu. Hệ thống sẽ hiển thị form đăng nhập lên màn hình.
2. Người dùng nhập hoặc người quản trị tiến hành nhập tên tài khoản và mật khẩu sau đó kích chuột vào nút “Login”. Hệ thống lấy thông tin từ bảng User và kiểm tra tài khoản mà người dùng hoặc người quản trị vừa nhập có tồn tại trong hệ thống không. Nếu có, hiển thị giao diện trang chủ của hệ thống vời tài khoản mà khách hàng hoặc người quản trị vừa nhập.

Use case kết thúc.

ii) Luồng rẽ nhánh:

1. Tại bước 2 trong luồng cơ bản, khi người dùng nhập không đầy đủ thông tin thiếu tên đăng nhập hoặc mật khẩu hệ thống sẽ thông báo “Vui lòng điền vào trường này”.
2. Tại bước 2 trong luồng cơ bản, khi người dùng nhập thông tin mà hệ thống không tìm thấy, hay nhập không đúng tên tài khoản (hoặc mật khẩu). Hệ thống sẽ hiển thị thông báo “Tên đăng nhập hoặc mật khẩu không đúng”. Người dùng có thể tiếp tục nhập lại để đăng nhập hoặc hủy bỏ thao tác.
3. Tại bất kỳ thời điểm nào hệ thống không kết nối được với CSDL. Hệ thống sẽ hiển thị một thông báo lỗi. Use case kết thúc.
4. Các yêu cầu đặc biệt:

Không có.

1. Tiền điều kiện:

Không có.

1. Hậu điều kiện:

Nếu use case thành công thì người dùng sẽ đăng nhập được vào hệ thống, trạng thái người dùng thay đổi.

1. Điểm mở rộng:

Không có.

#### 4.3 Use case Xem giao dịch

1. Tên use case: Xem giao dịch.
2. Mô tả vắn tắt: Use case này cho phép người dùng xem các thông tin của các giao dịch đã thực hiện.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case này bắt đầu khi người dùng click vào biểu tượng Transactions của trang web. Hệ thống sẽ lấy thông tin từ bảng Transaction và hiển thị ra màn hình.
2. Người dùng chọn tháng(tháng hiện tại, các tháng trước đó) mà muốn xem các giao dịch của mình. Hệ thống sẽ lấy ra các thông tin giao dịch bao gồm: khoản thu, khoản chi, tổng tiền, thông tin các khoản: ngày tháng năm, loại giao dịch, ghi chú.. và hiển thị ra màn hình.

ii) Luồng rẽ nhánh

1. Tại bước 1 trong luồng cơ bản, nếu trong bảng Transaction không có dữ liệu thì hệ thống sẽ hiển thị thông báo “No transactions” và use kết thúc.
2. Tại bước 2 trong luồng cơ bản, nếu tháng nào không có giao dịch, hệ thống sẽ hiển thị thông báo “No transactions”, ngược lại hệ thống sẽ hiển thị các thông tin về các giao dịch mà người dùng đã giao dịch trong tháng.
3. Trong bất kỳ thời điểm nào của quá trình thực hiện use case, nếu không kết nối được với CSDL thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
4. Các yêu cầu đặc biệt:

Không có.

1. Tiền điều kiện:

Người dùng đăng nhập tài khoản vào hệ thống để thực hiện use case.

1. Hậu điều kiện:

Không có.

1. Điểm mở rộng:

Không có.

#### 4.4 Use case Tạo ví

1. Tên use case: Tạo ví.
2. Mô tả vắn tắt: Use case này cho phép người dùng tạo ví của mình.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case này bắt đầu khi người dùng kích chọn mục My Wallet. Hệ thống lấy thông tin từ bảng Wallet và hiển thị ra màn hình.
2. Người dùng kích chọn loại ví muốn tạo. Tại đây, hệ thống sẽ hiển thị thông tin về các loại ví muốn tạo bao gồm thông tin về ngân hàng, mức phí sử dụng của loại ví.
3. Người dùng chọn ngân hàng liên kết, và lựa chọn mức phí sử dụng, người dùng nhấn nút “đăng ký”. Hệ thống sẽ tự động lưu thông tin của người dùng theo các bước và thông báo đăng ký thành công.

Use case kết thúc.

ii) Luồng rẽ nhánh

1. Tại bước 1, 2, 3 trong luồng cơ bản, nếu người dùng không làm đúng trình tự các bước thì hệ thống sẽ không hiển thị các bước tiếp theo và hiển thị thông báo “Bạn cần làm theo các bước hiển thị” hoặc người dùng có thể bấm “Cancel” để dừng thao tác tạo ví. Use kết thúc.
2. Trong bất kỳ thời điểm nào của quá trình thực hiện use case, nếu không kết nối được với CSDL thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
3. Các yêu cầu đặc biệt:

Không có.

1. Tiền điều kiện:

Người dùng cần đăng nhập tài khoản vào hệ thống để thực hiện các thao tác.

1. Hậu điều kiện:

Không có.

1. Điểm mở rộng:

Không có.

#### 4.5 Use case Điều chỉnh số dư

1. Tên use case: Điều chỉnh số dư.
2. Mô tả vắn tắt: Use case này cho phép người dùng điều chỉnh số dư ví của mình.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case này bắt đầu khi người dùng đăng nhập vào trang web, chọn biểu tượng ba gạch ở góc trái màn hình. Sau đó kích vào mục My Wallet. Hệ thống lấy thông tin walletName và balance từ bảng Wallet hiển thị lên màn hình.
2. Người dùng kích chuột vào một loại ví bất kì. nhập số dư mới và nhấn nút “Save”. Hệ thống sẽ nhận các thông tin trong bảng Wallet.

Use case kết thúc.

ii) Luồng rẽ nhánh:

1. Tại bước 2 trong luồng cơ bản, nếu người dùng nhấn vào nút “Cancel”, thì hệ thống sẽ huỷ bỏ thao tác. Use case kết thúc.
2. Trong bất kỳ thời điểm nào của quá trình thực hiện use case, nếu không kết nối được với CSDL thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
3. Các yêu cầu đặc biệt:

Không có.

1. Tiền điều kiện:

Người dùng cần đăng nhập tài khoản vào hệ thống để thực hiện các thao tác.

1. Hậu điều kiện:

Use case được thực hiện thì số dư trong bảng Wallet được cập nhật.

1. Điểm mở rộng:

Không có.

#### 4.6 Use case Xem thống kê chi tiêu

1. Tên use case: Xem thống kê chi tiêu.
2. Mô tả vắn tắt: Use case này cho phép người dùng xem thống kê chi tiêu của mình theo tháng và năm.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case này bắt đầu khi khách hàng kích vào mục Report. Hệ thống sẽ hiển thị biểu đồ thống kê biểu hiện mức thu chi của tháng được lấy dữ liệu từ bảng Transaction.
2. Người dùng có thể kích chọn khoảng thời gian mà mình muốn xem. Hệ thống sẽ hiển thị thông tin của thời gian đó ra màn hình.
3. Người dùng kích vào một mục trong khung biểu đồ. Hệ thống sẽ hiển thị biểu đồ thống kê chi tiết, các khoản thu chi trong tháng ra màn hình.

Use case kết thúc.

ii) Luồng rẽ nhánh:

1. Tại bước 2, 3 của luồng cơ bản, nếu mục được chọn không có biến động, hệ thống sẽ hiển thị thông báo “Không có biến động”. Use case kết thúc.
2. Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu, hệ thống sẽ hiển thị một thông báo lỗi ra màn hình. Use case kết thúc.
3. Các yêu cầu đặc biệt:

Không có.

1. Tiền điều kiện:

Người dùng cần đăng nhập tài khoản vào hệ thống.

1. Hậu điều kiện:

Không có.

1. Điểm mở rộng:

Không có.

#### 4.7 Use case Xem loại hình chi tiêu

1. Tên use case: Xem loại hình chi tiêu.
2. Mô tả vắn tắt: Use case này cho phép người dùng xem các loại hình chi tiêu.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case này bắt đầu khi người dùng kích chuột vào biểu tượng 3 gạch ở góc trái màn hình sau đó kích vào nút Categories. Hệ thống sẽ hiển thị ra màn hình danh sách các thông tin từ bảng CategoryTransaction bao gồm: biểu tượng, thể loại, tên các loại hình.
2. Khi người dùng kích chọn vào một loại hình. Hệ thống sẽ lấy thông tin: biểu tượng, tên loại hình, chi phí, thông tin chi tiết và hiển thị ra màn hình.

Use case kết thúc.

ii) Luồng rẽ nhánh:

1. Tại bước 1 và 2 trong luồng cơ bản, nếu bảng CategoryTransaction chưa có thông tin nào, hệ thống sẽ thông báo “chưa có dữ liệu”. Use case kết thúc.
2. Tại bất kỳ thời điểm nào trong quá trình thực hiện ca sử dụng nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi. Use case kết thúc.
3. Các yêu cầu đặc biệt:

Không có.

1. Tiền điều kiện:

Người dùng đăng nhập tài khoản vào hệ thống.

1. Hậu điều kiện:

Không có.

1. Điểm mở rộng:

Không có.

#### 4.8 Use case Bảo trì ví

1. Tên use case: Bảo trì ví.
2. Mô tả vắn tắt: Use case này cho phép người quản trị thực hiện các hoạt động xem, thêm, sửa, xóa các thông tin trong bảng Wallet.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case này bắt đầu khi người quản trị kích vào “Ví” trên menu quản trị. Hệ thống lấy thông tin chi tiết của các loại ví từ bảng Wallet hiển thị các lên màn hình.
2. Thêm:
3. Người quản trị kích vào nút “Thêm mới” hoặc icon “+” trên cửa sổ danh sách các ví. Hệ thống hiển thị màn hình yêu cầu nhập thông tin chi tiết cho loại ví được thêm gồm: Mã ví, tên ví, mô tả, hạn mức, loại ví.
4. Người quản trị nhập thông tin của tên ví và kích vào nút “Tạo”. Hệ thống sẽ tạo một loại ví mới trong bảng Wallet và hiển thị danh sách các loại ví ra màn hình.
5. Sửa thông tin:
   1. Người quản trị kích vào nút “Sửa” trên một dòng chứa thông tin các loại ví. Hệ thống sẽ lấy thông tin cũ của loại ví được chọn từ bảng Wallet và hiển thị lên màn hình.
   2. Người quản trị nhập thông tin mới cho loại ví được chọn và kích vào nút “Cập nhật”. Hệ thống sẽ sửa thông tin của loại được chọn trong bảng Wallet và hiển thị danh sách các ví sau khi cập nhật.
6. Xóa:
   1. Người quản trị kích vào nút “Xóa” trên một dòng chứa loại ví cần xoá. Hệ thống sẽ đưa ra thông báo “Bạn có chắc chắn muốn xóa loại ví này?” lên màn hình.
   2. Người quản trị kích vào nút “Đồng ý” trên thông báo hệ thống đã đưa ra. Hệ thống sẽ xóa ví được chọn khỏi bảng Wallet và hiển thị danh sách các loại ví sau khi cập nhật.

Use case kết thúc.

ii) Luồng rẽ nhánh:

1. Tại bước 2b hoặc 3b trong luồng cơ bản nếu người quản trị nhập thông tin ví không hợp lệ thì hệ thống sẽ hiển thị thông báo “Bạn nhập chưa đúng, vui lòng nhập lại thông tin.”. Người quản trị có thể nhập lại để tiếp tục hoặc kích vào nút “Hủy bỏ” để kết thúc.
2. Tại bước 2b hoặc 3b trong luồng cơ bản nếu người quản trị kích vào nút “Hủy bỏ” hệ thống sẽ bỏ qua thao tác thêm mới hoặc sửa chữa tương ứng và hiển thị danh sách các loại trong bảng Wallet.
3. Tại bước 4b trong luồng cơ bản nếu người quản trị kích vào nút “Không đồng ý” hệ thống sẽ bỏ qua thao tác xóa và hiển thị danh sách các loại trong bảng Wallet.
4. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sử dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
5. Các yêu cầu đặc biệt:

Use case này chỉ cho phép một số vai trò như người quản trị thực hiện.

1. Tiền điều kiện:

Người quản trị cần đăng nhập với vai trò quản trị hệ thống trước khi thực hiện use case này.

1. Hậu điều kiện:

Nếu use case này thành công thì thông tin về ví sẽ được cập nhật trong cơ sở dữ liệu.

1. Điểm mở rộng:

Không có.

#### 4.9 Use case Quản lý tài khoản khách hàng

1. Tên use case: Quản lý các tài khoản khách hàng.
2. Mô tả vắn tắt: Use case này cho phép người quản trị xem, yêu cầu cập nhật thông tin và xóa thông tin người dùng trong bảng User.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case này bắt đầu khi người quản trị kích chuột vào nút “Quản lý tài khoản khách hàng” trên menu quản trị. Hệ thống lấy thông tin từ bảng User và hiển thị ra màn hình danh sách các tài khoản khách hàng.
2. Cập nhật tài khoản:
3. Người quản trị kích vào nút cập nhật trên một tài khoản bất kỳ trên giao diện danh sách các tài khoản. Hệ thống sẽ hiển thị giao diện mẫu chứa các thông tin tài khoản được chọn. Sau khi người quản trị điển các thông tin yêu cầu cần cập nhật của tài khoản và kích chuột vào nút “Thông báo cho người dùng”. Hệ thống sẽ hiển thị một màn hình yêu cầu xác nhận cập nhật thông báo.
4. Người quản trị kích vào nút “Xác nhận”. Hệ thống sẽ lưu thông tin vừa được cập nhật và hiển thị các thông tin của tài khoản sau khi cập nhật.
5. Use case xoá tài khoản:
6. Người quản trị kích vào nút “Xoá” trên một tài khoản cần xoá. Hệ thống sẽ hiển thị một yêu cầu xác nhận xoá.
7. Người quản trị kích vào nút “Đồng ý”. Hệ thống sẽ xoá tài khoản được chọn khỏi danh sách và hiển thị danh sách tài khoản sau khi được cập nhật.

Use case kết thúc.

ii) Luồng rẽ nhánh:

1. Tại bước 2 trong luồng cơ bản, nếu người quản trị kích vào nút huỷ bỏ hệ thống sẽ huỷ bỏ các thao tác cập nhật và trả lại danh sách các đơn hàng.
2. Tại bước 3 trong luồng cơ bản, nếu người quản trị kích vào nút huỷ bỏ hệ thống sẽ huỷ bỏ các thao tác xóa và trả lại danh sách các đơn hàng.
3. Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi. Use case kết thúc.
4. Các yêu cầu đặc biệt:

Use case này chỉ cho phép người quản trị đăng nhập với vai trò là người quản trị hệ thống.

1. Tiền điều kiện:

Người quản trị cần đăng nhập với vai trò quản trị hệ thống trước khi thực hiện use case.

1. Hậu điều kiện:

Nếu use case kết thúc thành công thì các thông tin vừa được thao tác sẽ được cập nhật.

1. Điểm mở rộng:

Không có.

#### 4.10 Use case Bảo trì giao dịch

1. Tên use case: Bảo trì giao dịch.
2. Mô tả vắn tắt: Use case này cho phép người quản trị xem, sửa trạng thái, xóa các thông tin trong bảng Transaction.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case này bắt đầu khi người quản trị kích vào “Giao dịch” trên menu quản trị. Hệ thống lấy thông tin chi tiết của các giao dịch từ bảng Transaction hiển thị các lên màn hình.
2. Sửa trạng thái:

a. Người quản trị kích vào nút “Sửa” trên một dòng chứa giao dịch cần thay đổi. Hệ thống sẽ lấy ra thông tin đã giao dịch của người dùng và hiển thị ra màn hình.

b. Người quản trị nhập thông tin trạng thái mới cho giao dịch được chọn và kích vào nút “Cập nhật”. Hệ thống cập nhật thông tin trạng thái giao dịch được chọn trong bảng Transaction và hiển thị các giao dịch sau khi cập nhật.

1. Xóa thông tin:

a. Người quản trị kích vào nút “Xóa” trên một dòng chứa thông tin giao dịch cần xoá. Hệ thống sẽ đưa ra thông báo “Bạn có chắc chắn muốn xóa thông tin này?” lên màn hình.

b. Người quản trị kích vào nút “Đồng ý” trên thông báo hệ thống đã đưa ra. Hệ thống sẽ xóa thông tin được chọn khỏi bảng Transaction và hiển thị danh sách các giao dịch sau khi xoá.

Use case kết thúc.

ii) Luồng rẽ nhánh:

1. Tại bước 1 trong luồng cơ bản, nếu trong bảng Transaction không có thông tin giao dịch nào, hệ thống sẽ đưa ra thông báo “Không có giao dịch” lên màn hình và use case kết thúc.
2. Tại bước 2b trong luồng cơ bản nếu người quản trị nhập thông tin giao dịch không hợp lệ thì hệ thống sẽ hiển thị thông báo “Bạn nhập chưa đúng, vui lòng nhập lại thông tin.”. Người quản trị có thể nhập lại để tiếp tục hoặc kích vào nút “Hủy bỏ” để kết thúc.
3. Tại bước 2 và 3 trong luồng cơ bản nếu người quản trị đồng ý thực hiện thao tác hệ thống sẽ thực hiện thao tác cập nhật lại các thông tin trong bảng Transaction. Ngược lại, không đồng ý hệ thống sẽ huỷ bỏ các thao tác của người quản trị và use case kết thúc.
4. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sử dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
5. Các yêu cầu đặc biệt:

Use case này chỉ cho phép người quản trị hệ thống.

1. Tiền điều kiện:

Người quản trị cần đăng nhập với vai trò quản trị hệ thống trước khi thực hiện use case.

1. Hậu điều kiện:

Use case được thực hiện thì thông tin các giao dịch được cập nhật trong cơ sở dữ liệu.

1. Điểm mở rộng: Không có.

#### 4.11 Use case Bảo trì loại hình chi tiêu

1. Tên use case: Bảo trì loại hình chi tiêu.
2. Mô tả vắn tắt: Use case này cho phép người quản trị xem, sửa trạng thái, xóa các thông tin trong bảng CategoryTransaction.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case này bắt đầu khi người quản trị kích vào “Loại hình chi tiêu” trên menu quản trị. Hệ thống lấy thông tin chi tiết của các giao dịch từ bảng CategoryTransaction hiển thị các lên màn hình.
2. Sửa trạng thái:

a. Người quản trị kích vào nút “Sửa” trên một dòng chứa giao dịch cần thay đổi. Hệ thống sẽ lấy ra thông tin loại hình chi tiêu và hiển thị ra màn hình.

b. Người quản trị nhập thông tin trạng thái mới cho loại hình được chọn và kích vào nút “Cập nhật”. Hệ thống cập nhật thông tin loại hình chi tiêu được chọn trong bảng CategoryTransaction và hiển thị các loại hình chi tiêu sau khi cập nhật.

1. Xóa thông tin:

a. Người quản trị kích vào nút “Xóa” trên một dòng chứa thông tin loại hình chi tiêu cần cần xoá. Hệ thống sẽ đưa ra thông báo “Bạn có chắc chắn muốn xóa loại hình này?” lên màn hình.

b. Người quản trị kích vào nút “Đồng ý” trên thông báo hệ thống đã đưa ra. Hệ thống sẽ xóa thông tin được chọn khỏi bảng CategoryTransaction và hiển thị danh sách các loại hình chi tiêu sau khi xoá.

Use case kết thúc.

ii) Luồng rẽ nhánh:

1. Tại bước 1 trong luồng cơ bản, nếu trong bảng CategoryTransaction không có thông tin loại hình chi tiêu nào, hệ thống sẽ đưa ra thông báo “Không có loại hình chi tiêu nào” lên màn hình và use case kết thúc.
2. Tại bước 2b trong luồng cơ bản nếu người quản trị nhập thông tin không hợp lệ thì hệ thống sẽ hiển thị thông báo “Bạn nhập chưa đúng, vui lòng nhập lại thông tin.”. Người quản trị có thể nhập lại để tiếp tục hoặc kích vào nút “Hủy bỏ” để kết thúc.
3. Tại bước 2 và 3 trong luồng cơ bản nếu người quản trị đồng ý thực hiện thao tác hệ thống sẽ thực hiện thao tác cập nhật lại các thông tin trong bảng CategoryTransaction. Ngược lại, không đồng ý hệ thống sẽ huỷ bỏ các thao tác của người quản trị và use case kết thúc.
4. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sử dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
5. Các yêu cầu đặc biệt:

Use case này chỉ cho phép người quản trị hệ thống.

1. Tiền điều kiện:

Người quản trị cần đăng nhập với vai trò quản trị hệ thống trước khi thực hiện use case.

1. Hậu điều kiện:

Use case này được thực hiện thì các loại hình chi tiêu sẽ được thay đổi và cập nhật trong cơ sở dữ liệu.

1. Điểm mở rộng:

Không có.

#### 4.12 Use case Bảo trì trạng thái giao dịch

1. Tên use case: Bảo trì trạng thái giao dịch.
2. Mô tả vắn tắt: Use case này cho phép người quản trị xem, sửa trạng thái, xóa các thông tin trong bảng InOutMoney.
3. Luồng sự kiện:

i) Luồng cơ bản:

1. Use case này bắt đầu khi người quản trị kích vào “Trạng thái giao dịch” trên menu quản trị. Hệ thống lấy thông tin chi tiết của các giao dịch từ bảng InOutMoney hiển thị các lên màn hình.
2. Sửa trạng thái:

a. Người quản trị kích vào nút “Sửa” trên một dòng chứa giao dịch cần thay đổi. Hệ thống sẽ lấy ra thông tin trạng thái giao dịch cần sửa và hiển thị ra màn hình.

b. Người quản trị nhập thông tin trạng thái mới cho loại hình được chọn và kích vào nút “Cập nhật”. Hệ thống cập nhật thông tin trạng thái giao dịch được chọn trong bảng InOutMoney và hiển thị trạng thái giao dịch sau khi sửa ra màn hình.

1. Xóa thông tin:

a. Người quản trị kích vào nút “Xóa” trên một dòng chứa thông tin trạng thái giao dịch cần cần xoá. Hệ thống sẽ đưa ra thông báo “Bạn có chắc chắn muốn xóa trạng thái giao dịch này?” lên màn hình.

b. Người quản trị kích vào nút “Đồng ý” trên thông báo hệ thống đã đưa ra. Hệ thống sẽ xóa thông tin được chọn khỏi bảng InOut và hiển thị danh sách các trạng thái giao dịch sau khi xoá.

Use case kết thúc.

ii) Luồng rẽ nhánh:

1. Tại bước 1 trong luồng cơ bản, nếu trong bảng InOutMoney không có thông tin trạng thái giao dịch nào, hệ thống sẽ đưa ra thông báo “Không có trạng thái giao dịch nào” lên màn hình và use case kết thúc.
2. Tại bước 2b trong luồng cơ bản nếu người quản trị nhập thông tin không hợp lệ thì hệ thống sẽ hiển thị thông báo “Bạn nhập chưa đúng, vui lòng nhập lại thông tin.”. Người quản trị có thể nhập lại để tiếp tục hoặc kích vào nút “Hủy bỏ” để kết thúc.
3. Tại bước 2 và 3 trong luồng cơ bản nếu người quản trị đồng ý thực hiện thao tác hệ thống sẽ thực hiện thao tác cập nhật lại các thông tin trong bảng InOutMoney. Ngược lại, không đồng ý hệ thống sẽ huỷ bỏ các thao tác của người quản trị và use case kết thúc.
4. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sử dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
5. Các yêu cầu đặc biệt:

Use case này chỉ cho phép người quản trị hệ thống.

1. Tiền điều kiện:

Người quản trị cần đăng nhập với vai trò quản trị hệ thống trước khi thực hiện use case.

1. Hậu điều kiện:

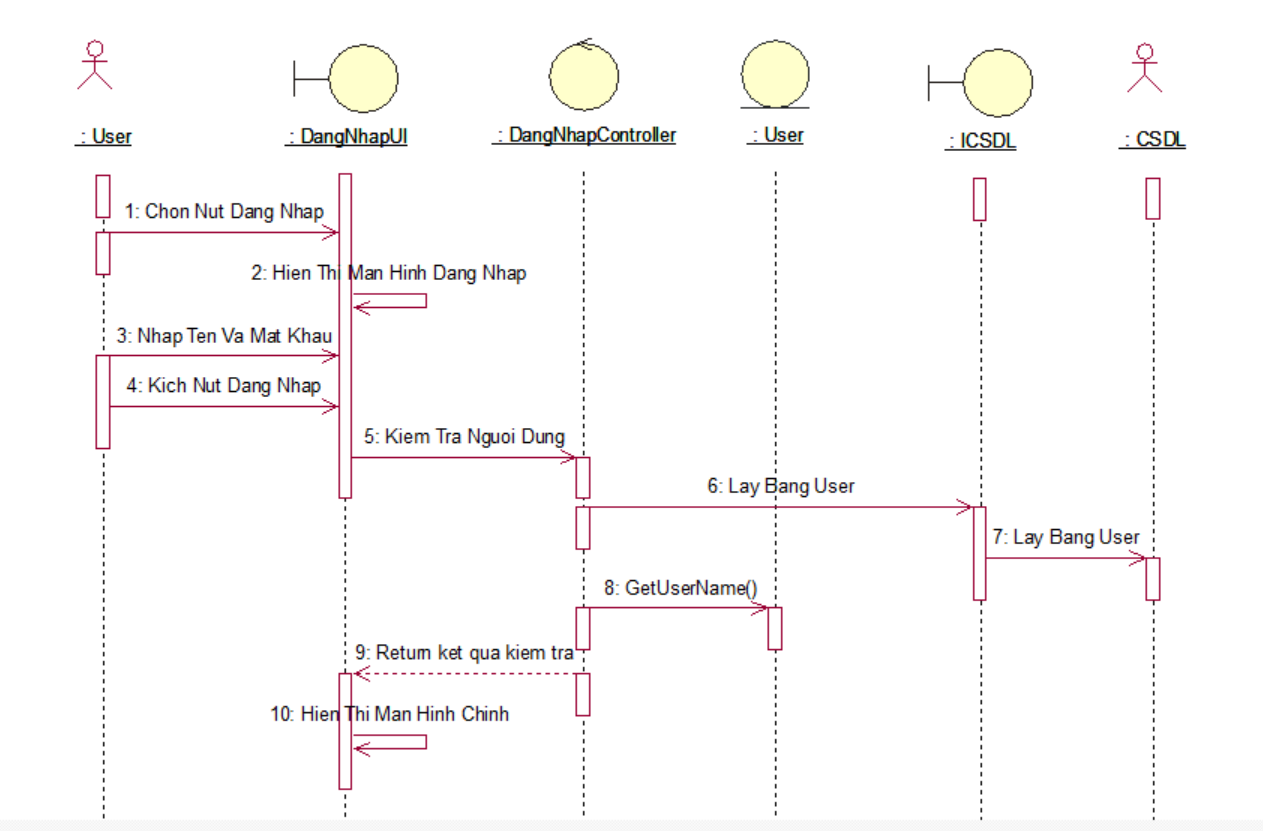
Use case được thực hiện thì thông tin về trạng thái giao dịch sẽ được cập nhật trong cơ sở dữ liệu.

1. Điểm mở rộng:

Không có.

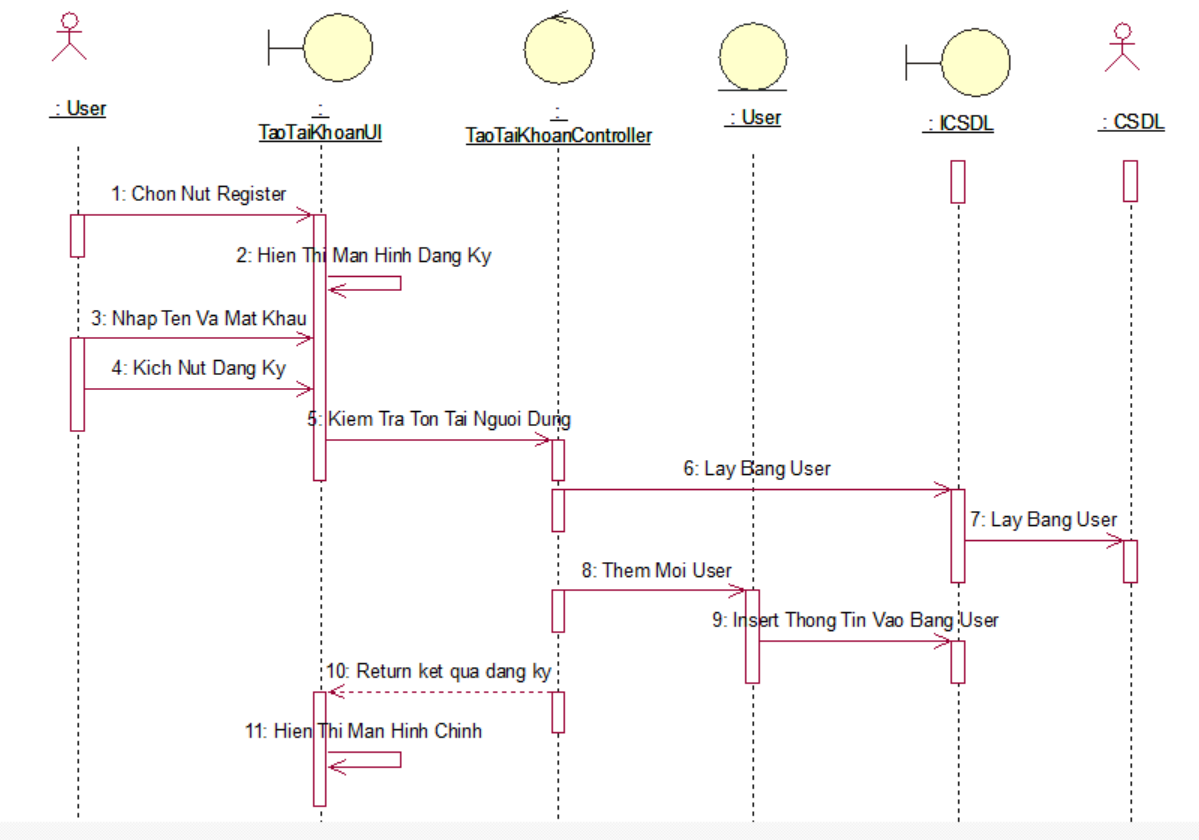
### 5. Biểu đồ trình tự

#### 5.1 Usecase đăng nhập



Hình 2.10. Use case đăng nhập

#### 5.2 Use case tạo tài khoản

****

Hình 2.11. Use case tạo tài khoản

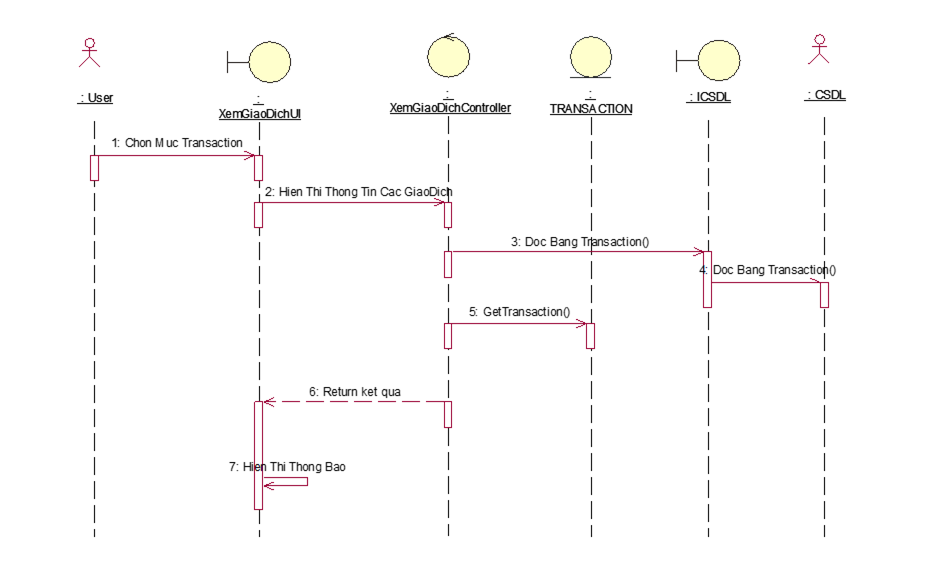
#### 5.3 Use case tạo ví

A screenshot of a video game

Description automatically generated with medium confidence

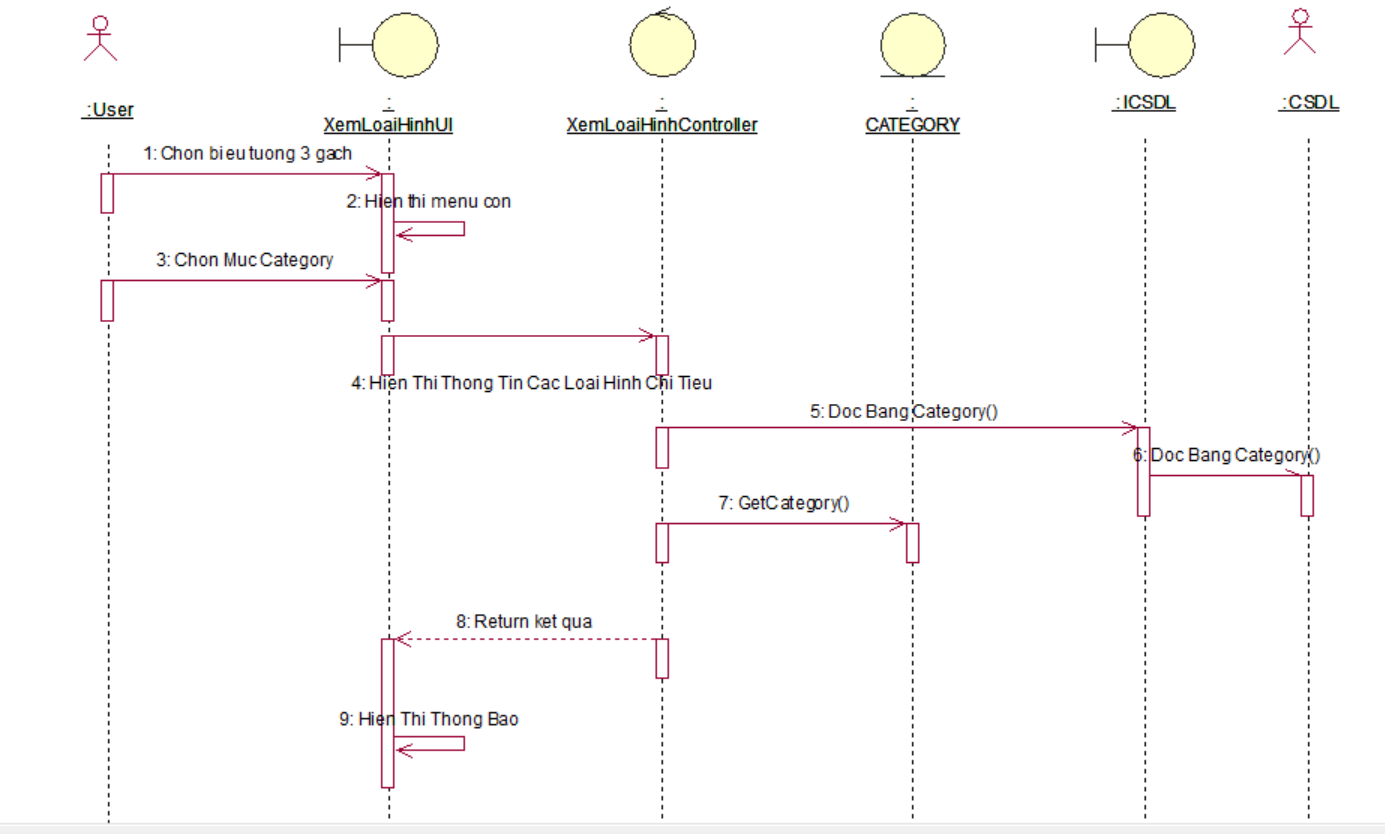
Hình 2.12. Use case tạo ví

#### 5.4 Use case xem giao dịch



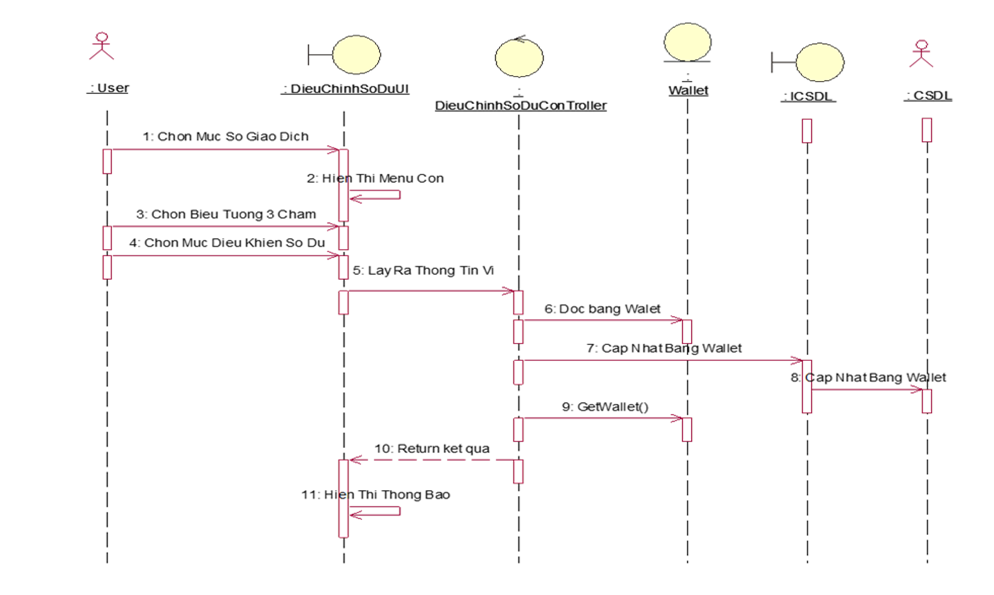
Hình 2.13. Use case xem giao dịch

#### 5.5 Use case xem loại hình chi tiêu



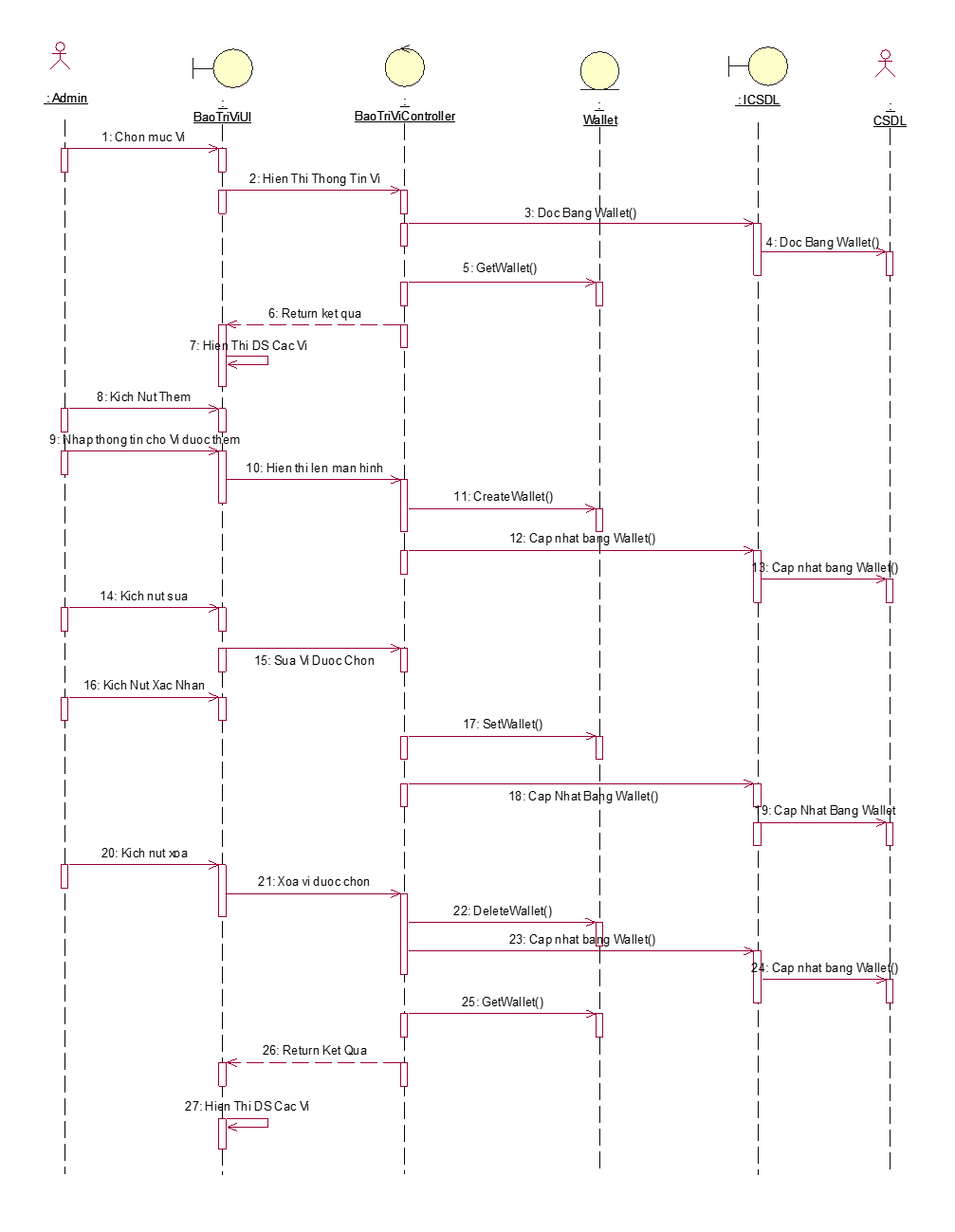
Hình 2.14. Use case xem loại hình chi tiêu

#### 5.9 Use case điều chỉnh số dư



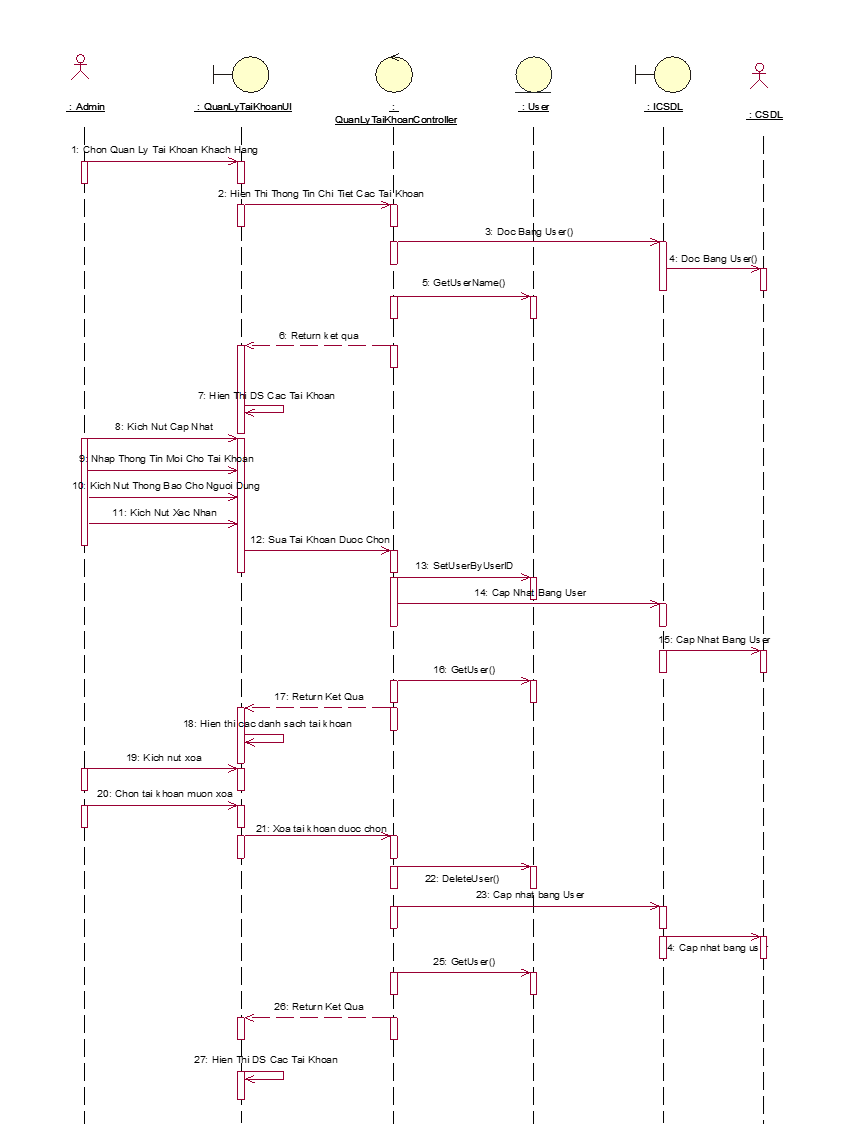
Hình 2.15. Use case điều chỉnh số dư

#### 5.6 Use case bảo trì ví



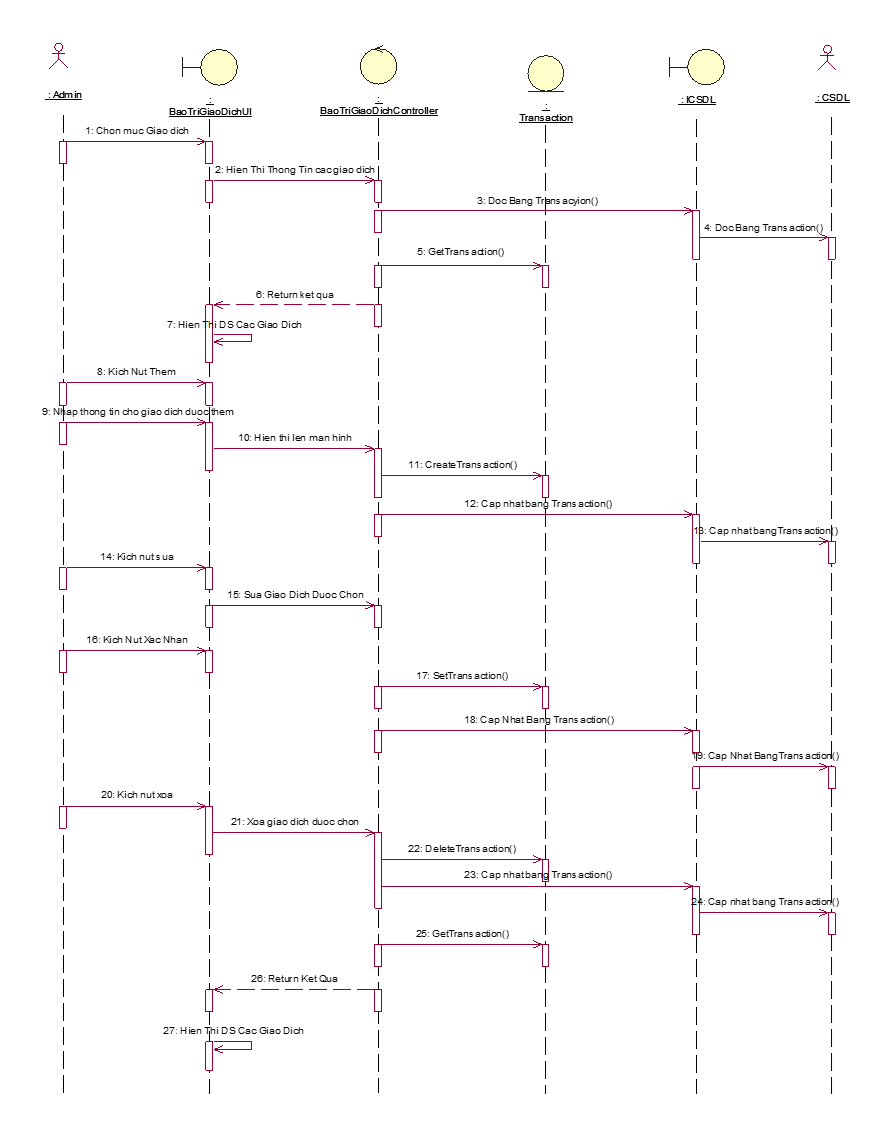
Hình 2.16. Use case bảo trì ví

#### 5.7 Use case quản lý tài khoản khách hàng



Hình 2.17. Use case quản lý tài khoản khách hàng

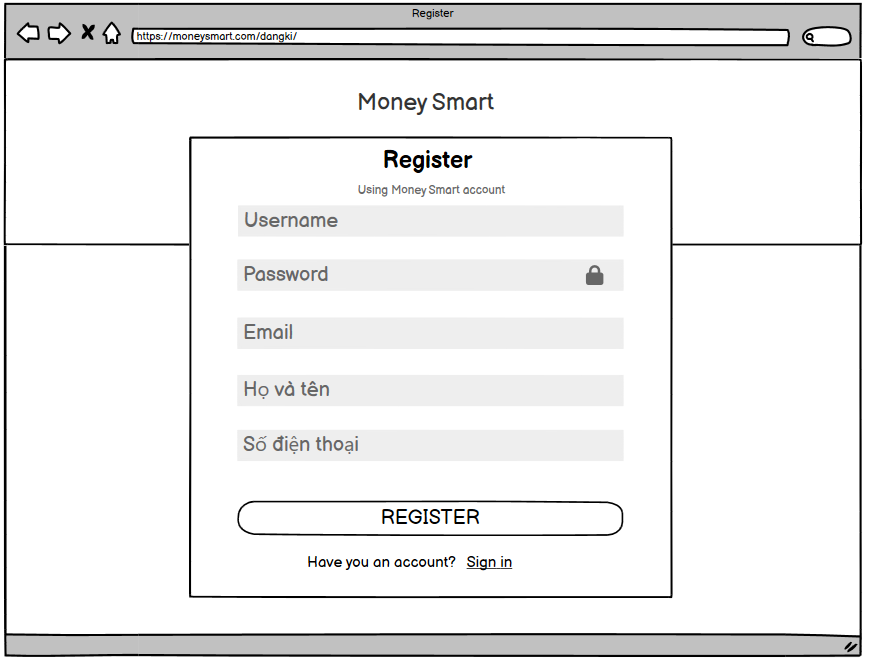
#### 5.8 Use case bảo trì giao dịch



Hình 2.18. Use case bảo trì giao dịch

## III. Thiết kế hệ thống

### 1. Trang đăng kí



Hình 2.19. Giao diện trang đăng ký

Khi muốn đăng ký tài khoản mới, người dùng chọn REGISTER, hệ thống sẽ đưa ra màn hình điền các thông tin người dùng cần thiết để tạo tài khoản mới.

Ở đây sẽ xuất hiện một số trường sau:

* Tên tài khoản: Trường để người dùng nhập tên tài khoản của họ.
* Mật khẩu: Trường để người dùng tạo một mật khẩu bảo mật cho tài khoản. Thông thường, mật khẩu nên được yêu cầu phải có ít nhất 8 ký tự và kết hợp giữa chữ cái hoa, chữ cái thường, số và ký tự đặc biệt.
* Địa chỉ email: Trường để người dùng nhập địa chỉ email duy nhất và hợp lệ.
* Họ và tên: Trường để người dùng nhập đầy đủ họ và tên của họ.
* Số điện thoại: Trường để người dùng nhập số điện thoại liên hệ của họ.

Giao diện xem trang đăng kí giúp người dùng đăng kí tài khoản cho mình, để có tài khoản để đăng nhập và sử dụng weebsite.

### 2. Trang đăng nhập



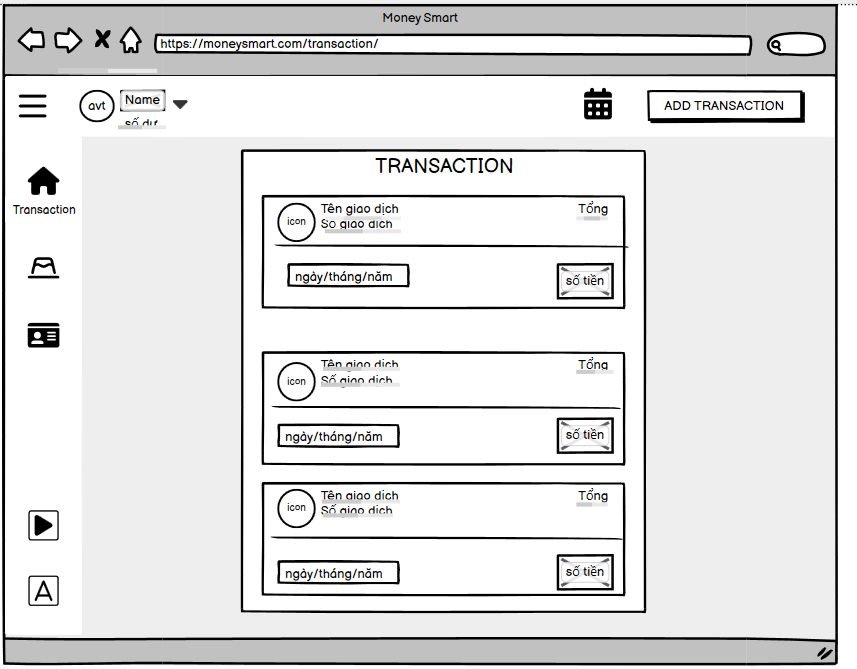
Hình 2.20. Giao diện trang đăng nhập

Khi muốn đăng nhập vào hệ thống, người dùng chọn LOGIN. Sau khi chọn, trang sẽ hiện lên một màn hình đăng nhập.

Ở đây sẽ xuất hiện một số trường sau:

* Trường đăng nhập: Đặt một trường đăng nhập cho tên người dùng hoặc địa chỉ email. Đây là nơi người dùng nhập thông tin cá nhân của mình để đăng nhập vào tài khoản. Trường đăng nhập thường đi kèm với một biểu tượng hình người hoặc biểu tượng thư gửi để chỉ rõ loại thông tin mà người dùng cần nhập.
* Trường mật khẩu: Đặt một trường mật khẩu để người dùng nhập mật khẩu của họ. Trường mật khẩu nên được ẩn và hiển thị dưới dạng dấu chấm hoặc ký tự ẩn khác để bảo mật thông tin người dùng.
* Nút đăng nhập: Đặt một nút "Đăng nhập" hoặc "Login" gần cuối giao diện để người dùng bấm vào sau khi nhập thông tin đăng nhập. Nút này có thể có màu sắc đậm hoặc tương phản với phông nền của trang để thu hút sự chú ý.
* Quên mật khẩu: Thêm một liên kết hoặc nút "Quên mật khẩu?" để người dùng có thể khôi phục mật khẩu nếu cần. Khi người dùng bấm vào liên kết hoặc nút này, họ sẽ được chuyển hướng đến trang khôi phục mật khẩu.
* Đăng ký: Nếu bạn cho phép người dùng tạo tài khoản mới trên cùng giao diện đăng nhập, bạn có thể thêm một liên kết hoặc nút "Đăng ký" để hướng dẫn người dùng đến trang đăng ký.

### 3. Trang Transaction



Hình 2.21. Giao diện trang Transaction

Giao diện xem các giao dịch đã thực hiện gồm một số phần sau:

* Icon: Các hình ảnh, icon tượng trưng cho loại giao dịch.
* Tên giao dịch: Là các tên ngắn gọn đại điện cho các giao dịch đã hoàn thành.
* Số giao dịch: Số các giao dịch thuộc loại giao dịch đó.
* Tổng: Tổng số tiền đã thực hiện của các giao dịch thuộc loại giao dịch đó.
* Ngày/ tháng/ năm: Ngày/ tháng/ năm thực hiện một giao dịch.
* Số tiền: Số tiền đã thực hiện của giao dịch thuộc ngày/ tháng/ năm đó.

Giao diện xem các giao dịch giúp người dùng dễ dàng theo dõi thông tin các giao dịch gồm loại giao dịch, ngày giao dịch, số tiền và các ghi chú. Nó cũng như là một quyển ghi nhật ký, đó là ghi lại bất kì hoạt động nào được thực hiện bởi người dùng. Điều này được sử dụng để cung cấp dữ liệu lịch sử có thể được sử dụng để đánh giá hoạt động trong quá khứ hoặc để lập kế hoạch cho mọi hoạt động trong tương lai.

### 4. Nút Add Transaction

**A screenshot of a computer

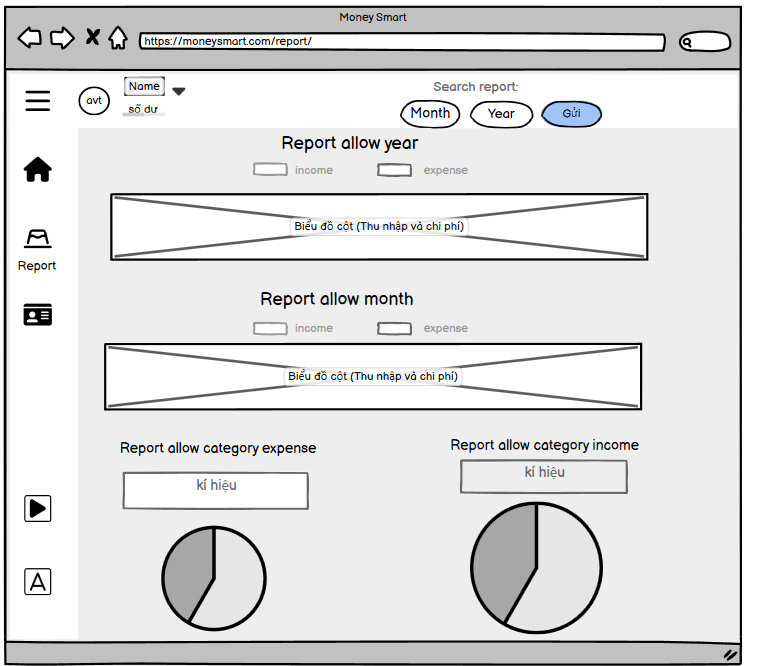
Description automatically generated with medium confidence**

Hình 2.22. Giao diện nút Add Transaction

Giao diện thêm các giao dịch gồm một số phần sau:

* Select in/out: Lựa chọn là loại giao dịch nạp/ tiêu tiền.
* Select Wallet: Lựa chọn loại ví để thực hiện giao dịch là tiền mặt hay thẻ điện tử.
* Select Category: Lựa chọn loại hình chi tiêu cho giao dịch: thực phẩm, các hóa đơn, bảo trì nhà, bảo trì xe, mua sắm, trang điểm, tiền lương, tiền lãi, đầu tư, ...
* Amount: Số tiền thực hiện cho giao dịch.
* Date: Ngày/ tháng/ năm thực hiện giao dịch.
* Note: Các ghi chú cho giao dịch.
* Nút Cancel: Hủy bảo tạo giao dịch.
* Nút Submit: Hoàn thành tạo giao dịch.

### 5. Trang report



Hình 2.23. Giao diện trang report

Giao diện xem thống kê chi tiêu gồm một số phần sau:

* Avt: Hình ảnh đại diện của người dùng.
* Name: Tên người dùng.
* Số dư: Số tiền còn lại trong tài khoản người dùng.
* Search report: Cho phép người dùng xem thống kê chi tiêu của mình theo khoảng thời gian muốn xem (theo tháng và năm).

Giao diện xem thống kê chi tiêu giúp người dùng dễ dàng theo dõi các báo cao về chi tiêu của mình theo tháng và theo năm, trang web hiển thị biểu đồ theo tỉ lệ thu/chi, bên cạnh đó còn hiển thị biểu đồ tỉ lệ các loại giao dịch mà người dùng đã thực hiện trong thời gian đó.

### 6. Trang Categories

A screenshot of a computer

Description automatically generated

Hình 2.24. Giao diện trang Category

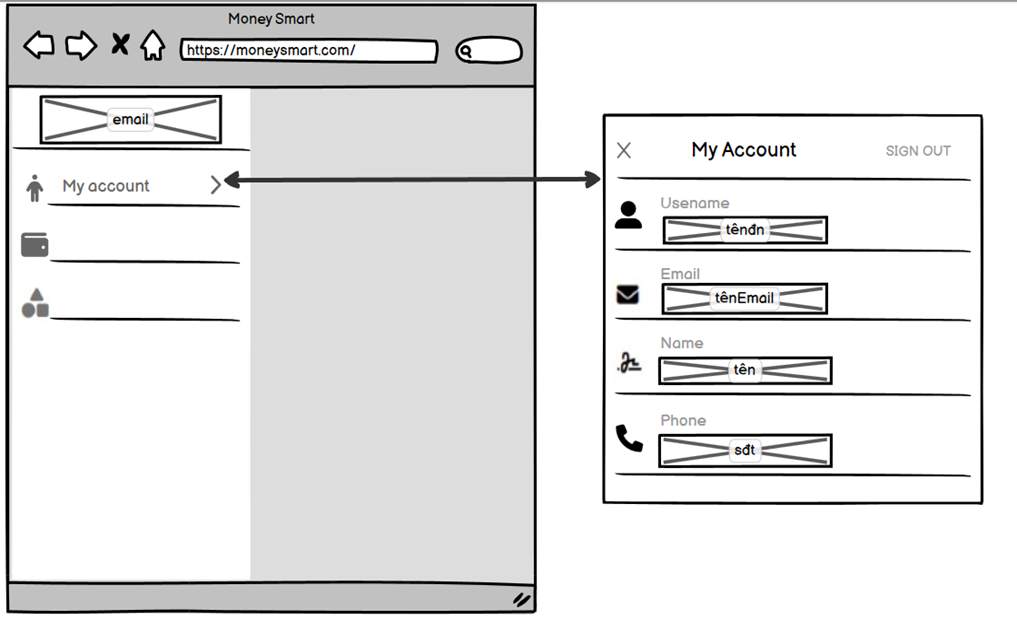
Để truy cập được danh sách các loại hình chi tiêu, người dùng bấm vào biểu tượng ba gạch ở phía trên bên trái màn hình, sau đó chọn mục Category.

Giao diện xem các loại hình chi tiêu gồm một số phần sau:

* Expenses/ Income: Loại hình chi tiêu là nạp vào hoặc tiêu ra .
* Avt: Hình ảnh đại diện cho loại hình chi tiêu đó.
* Tên chi phí: Tên của loại hình chi tiêu đó.
* Thể loại: loại hình chi tiêu cụ thể

Khi người dùng lựa chọn xong loại hình chi tiêu thì chi tiết của loại hình chi tiêu bao gồm các khoản chi tiêu và thu nhập đó sẽ hiện ở bên cạnh bao gồm hình ảnh minh họa, tên giao dịch và thể loại.

### 7. Trang Account



Hình 2.25. Giao diện trang Account

Để truy cập được vào tài khoản của mình, người dùng bấm vào biểu tượng ba gạch ở phía trên bên trái màn hình, sau đó chọn mục Account.

Giao diện xem tài khoản của mình gồm một số phần sau:

* Username: Tên tài khoản của người dùng.
* Email: Địa chỉ email của người dùng.
* Name: Tên đầy đủ của người dùng.
* Phone: Số điện thoại của người dùng.

Giao diện xem tài khoản của mình giúp người dùng kiểm soát được những thông tin của mình, có thể thay đổi email hoặc số điện thoại để phù hợp nhất cho việc liên hệ.

### 8. Trang Wallet

Để truy cập được danh sách các loại hình chi tiêu, người dùng bấm vào biểu tượng ba gạch ở phía trên bên trái màn hình, sau đó chọn mục My Wallet.

Giao diện ‘ví của tôi’ cho phép người dùng tạo ví của họ cho mọi loại chi phí, người dùng có thể điều chỉnh số dư, theo dõi số dư để có một chi tiêu hợp lý

A screenshot of a computer

Description automatically generated with medium confidence

Hình 2.26. Giao diện trang Wallet

Giao diện truy cập vào ‘ví của tôi’ gồm một số phần như sau:

* Loại ví: Tên của loại ví mà người dùng sử dụng.
* Icon: Các hình ảnh, icon tượng trưng cho loại ví.
* Viet Nam Đồng: loại tiền tệ sử dụng ví.
* Số tiền: số dư ban đầu của loại ví đó.
* Cancel: huỷ bỏ tất cả các thay đổi mà người dùng đang thực hiện.
* Save: lưu lại thay đổi.

# Tài liệu tham khảo

[1] <https://www.w3schools.com/java/>

[2] <https://www.youtube.com/>

[3] Lê Ngọc Hồng Quân (31/03/2023). *Visual Studio code là gì? Các tính năng nổi bật của Visual Studio code*.

<https://fptshop.com.vn/tin-tuc/danh-gia/visual-studio-code-la-gi-cac-tinh-nang-noi-bat-cua-visual-studio-code-146213#root-comment>.

[4] <https://www.json.org/>

[5] <https://fptcloud.com/javascript>

[6] Nguyễn Thị Thanh Huyền, Ngô Thị Bích Thúy, Phạm Thị Kim Phượng. *Giáo trình phân tích thiết kế hệ thống*.NXB Giáo dục Việt Nam (2011)