KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

****

**Báo Cáo Kết Thúc Môn Công Nghệ Phần Mềm**

**Học Kỳ II Năm Học 2024-2025**

**THIẾT KẾ ỨNG DỤNG WEB QUẢN LÝ CHI TIÊU CÁ NHÂN**

**BẰNG SPRING BOOT**

|  |  |
| --- | --- |
|  | *Sinh viên thực hiện:*  Võ Chí Hải-110122068-DA22TTD  Nguyễn Đỗ Thành Lộc-110122105-DA22TTD  Hoàng Tuấn Kiệt-110122099-DA22TTD |

**Giáo viên hướng dẫn : Nguyễn Bảo Ân**

***Trà Vinh, Tháng 2 Năm 2025***

KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

****

**Báo Cáo Kết Thúc Môn Công Nghệ Phần Mềm**

**Học Kỳ II Năm Học 2024-2025**

**THIẾT KẾ ỨNG DỤNG WEB QUẢN LÝ CHI TIÊU CÁ NHÂN**

**BẰNG SPRING BOOT**

|  |
| --- |
| *Sinh viên thực hiện:*  Võ Chí Hải-110122068-DA22TTD  Nguyễn Đỗ Thành Lộc-110122105-DA22TTD  Hoàng Tuấn Kiệt-110122099-DA22TTD |

**Giáo viên hướng dẫn : Nguyễn Bảo Ân**

***Trà Vinh, Tháng 2 Năm 2025***

*Trà Vinh, ngày … tháng … năm 2025*

**Giáo viên hướng dẫn**

*(Ký và ghi rõ họ tên)*

**NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG**

*Trà Vinh, ngày … tháng … năm 2025*

**Thành viên hội đồng**

*(Ký và ghi rõ họ tên)*

**LỜI CẢM ƠN**

Để hoàn thành bài báo cáo này, em xin gửi lời cảm ơn đến các Quý Thầy cô Khoa kỹ thuật và công nghệ, Trường đại học Trà Vinh đã tạo cơ hội cho em được học tập, rèn luyện và tích lũy kiến thức, kỹ năng để thực hiện bài báo cáo này.

Đặc biệt, em xin gửi lời cảm ơn đến Giảng viên hướng dẫn thầy Nguyễn Bảo Ân đã tận tình chỉ dẫn, theo dõi và đưa ra những lời khuyên bổ ích giúp em giải quyết được các vấn đề gặp phải trong quá trình nghiên cứu và hoàn thành đề tài một cách tốt nhất.

Do kiến thức của bản thân còn hạn chế và thiếu kinh nghiệm thực tiễn nên nội dung bài báo cáo khó tránh những thiếu sót. Em rất mong nhận được sự góp ý, chỉ dạy thêm từ Quý Thầy cô.

Cuối cùng, em xin chúc Quý Thầy Cô luôn thật nhiều sức khỏe và đạt được nhiều thành công trong công việc.

**Mục lục**

[CHƯƠNG 1. GIỚI THIỆU 1](#_Toc12800)

[1.1. Giới thiệu đề tài 1](#_Toc8771)

[1.2. Mục tiêu của ứng dụng 1](#_Toc5954)

[1.3. Lý do chọn đề tài 1](#_Toc29155)

[CHƯƠNG 2. PHÂN TÍCH YÊU CẦU 3](#_Toc21589)

[2.1. Các chức năng chính của hệ thống 3](#_Toc14447)

[2.1.1Quản lý người dùng 3](#_Toc644)

[2.1.2.Ghi chép thu – chi 3](#_Toc14086)

[2.1.3.Quản lý danh mục chi tiêu 3](#_Toc16235)

[2.1.4.Xem thống kê và báo cáo tài chính 3](#_Toc23020)

[2.1.5.Đặt hạn mức chi tiêu 4](#_Toc16136)

[2.2. Các yêu cầu phi chức năng 4](#_Toc17095)

[2.2.1.Hiệu năng 4](#_Toc27526)

[2.2.2.Tính khả dụng 4](#_Toc11718)

[2.2.3.Bảo mật 4](#_Toc26684)

[2.2.4.Tính mở rộng 4](#_Toc20775)

[2.2.5.Khả năng bảo trì 5](#_Toc14767)

[2.2.6.Tính thân thiện với người dùng 5](#_Toc4526)

[CHƯƠNG 3. THIẾT KẾ HỆ THỐNG 6](#_Toc8859)

[3.1. Kiến trúc tổng thể hệ thống 6](#_Toc31754)

[3.2. Sơ Đồ Kiến Trúc Hệ Thống 6](#_Toc1613)

[3.3. Sơ đồ ERD 7](#_Toc12866)

[3.4. Thiết kế api swagger mô tả các endpoint cấu trúc request respone 8](#_Toc25307)

[3.5. Thiết kế giao diện (UI/UX) 11](#_Toc13720)

[3.5.1. Giao diện trang đăng nhập 11](#_Toc18933)

[3.5.2. Giao diện trang đăng ký 12](#_Toc30862)

[3.5.3. Giao diện trang chủ 13](#_Toc13671)

[3.5.4. Giao diện menu 14](#_Toc16868)

[3.5.6. Giao diện trang quản lý thu nhập 16](#_Toc15491)

[CHƯƠNG 4. TRIỂN KHAI CÁC CÔNG CỤ SỬ DỤNG 17](#_Toc28882)

[4.1. Danh sách các công cụ sử dụng trong dự án 17](#_Toc6671)

[Spring Boot 17](#_Toc29643)

[Tailwind CSS 17](#_Toc11882)

[MySQL 17](#_Toc3717)

[Postman 17](#_Toc18430)

[Docker 17](#_Toc12576)

[Swagger 19](#_Toc3144)

[GitHub 19](#_Toc29083)

[JIRA 19](#_Toc28141)

[GitHub Actions 19](#_Toc518)

[4.2. Quy trình CI/CD với GitHub Actions 19](#_Toc16574)

[CI (Continuous Integration): 19](#_Toc4356)

[CD (Continuous Deployment): 20](#_Toc17047)

[4.3. Cấu hình Docker và quy trình triển khai ứng dụng 20](#_Toc12188)

[Cấu hình Docker: 20](#_Toc26750)

[Quy trình triển khai ứng dụng: 20](#_Toc28277)

[CHƯƠNG 5. QUẢN LÝ DỰ ÁN 21](#_Toc31085)

**Danh mục hình ảnh và bảng biểu**

*[Hình 1 Sở đồ kiến trúc của hệ thống 1](#_Toc31480)* [6](#_Toc31480)

*[HÌnh 2 Sở đồ ERD](#_Toc7404)* [7](#_Toc7404)

*[Bảng 1 Mô tả các API và Endpoint](#_Toc8200)* [10](#_Toc8200)

[Hình 1 Trang đăng nhập 11](#_Toc16637)

[Hình 2 trang đăng ký 12](#_Toc28593)

[Hình 3 Trang chủ 13](#_Toc15755)

[Hình 4 Menu 3.5.5 . Giao diện trang quản lý chi tiêu 14](#_Toc23054)

[Hình 5 Trang quản lý chi tiêu 15](#_Toc30333)

[Hình 6 Trang quản lý thu nhập 16](#_Toc32446)

# GIỚI THIỆU

## Giới thiệu đề tài

**Tên đề tài**: *Xây dựng ứng dụng web quản lý chi tiêu cá nhân sử dụng Spring Boot*

**Chủ đề**: Phát triển ứng dụng hỗ trợ người dùng ghi chép, phân loại và phân tích các khoản thu – chi cá nhân thông qua giao diện web trực quan, thân thiện, áp dụng công nghệ Spring Boot.

## Mục tiêu của ứng dụng

Ứng dụng hướng đến việc hỗ trợ người dùng:

Quản lý thu nhập và chi tiêu hàng ngày.

Phân loại các khoản chi theo danh mục (ăn uống, đi lại, học tập…).

Hiển thị báo cáo thống kê theo tuần/tháng/năm bằng biểu đồ.

Đặt giới hạn chi tiêu theo tháng và cảnh báo khi vượt hạn mức.

## Lý do chọn đề tài

Trong đời sống hiện đại, việc quản lý tài chính cá nhân ngày càng trở thành một kỹ năng quan trọng, đặc biệt đối với sinh viên, người đi làm và những người trẻ tuổi mới bắt đầu xây dựng cuộc sống độc lập. Với sự phát triển của công nghệ, nhu cầu số hóa trong quản lý chi tiêu không chỉ là một xu hướng mà còn là một giải pháp thiết yếu giúp người dùng kiểm soát tốt hơn thói quen tiêu dùng, từ đó hướng tới một lối sống tiết kiệm và ổn định về tài chính.

Tuy nhiên, trên thực tế, không phải ai cũng có thể duy trì việc ghi chép thu – chi một cách đều đặn bằng các phương pháp thủ công như sổ tay hay bảng tính Excel. Những cách này tuy đơn giản nhưng thiếu tính linh hoạt, khó đồng bộ và dễ gây nhàm chán. Mặt khác, nhiều ứng dụng quản lý chi tiêu hiện có trên thị trường thường tích hợp quá nhiều tính năng phức tạp, yêu cầu người dùng phải làm quen trong thời gian dài hoặc gặp rào cản ngôn ngữ (đa phần là tiếng Anh). Điều đó khiến người dùng phổ thông dễ nản và từ bỏ việc theo dõi tài chính cá nhân.

Chính vì vậy, nhóm (hoặc cá nhân) đã quyết định lựa chọn đề tài “Ứng dụng quản lý chi tiêu cá nhân” với mục tiêu xây dựng một hệ thống đơn giản, dễ sử dụng, giao diện thân thiện, tập trung vào những chức năng thiết yếu nhất như: ghi chép thu chi, phân loại khoản mục, xem báo cáo thống kê và cảnh báo khi vượt ngân sách. Ứng dụng được xây dựng dưới dạng website để người dùng có thể dễ dàng truy cập từ mọi thiết bị có trình duyệt, đồng thời sử dụng Spring Boot – một framework mạnh mẽ, hiện đại và phù hợp với mô hình phát triển web chuyên nghiệp, giúp đảm bảo hiệu suất và tính mở rộng của hệ thống.

Bên cạnh đó, việc chọn đề tài này còn xuất phát từ mong muốn củng cố và nâng cao kỹ năng lập trình web backend bằng Java – một trong những ngôn ngữ phổ biến nhất hiện nay trong lĩnh vực phát triển phần mềm doanh nghiệp. Việc áp dụng Spring Boot giúp sinh viên không chỉ hiểu rõ hơn về cấu trúc và luồng xử lý trong một ứng dụng web, mà còn rèn luyện được các kỹ năng như thiết kế API RESTful, quản lý cơ sở dữ liệu quan hệ, triển khai ứng dụng thực tế, xử lý bảo mật và xác thực người dùng.

# PHÂN TÍCH YÊU CẦU

## Các chức năng chính của hệ thống

## 2.1.1Quản lý người dùng

Đăng ký tài khoản: người dùng có thể tạo tài khoản mới với tên người dùng và mật khẩu.

Đăng nhập/đăng xuất: xác thực người dùng và cho phép đăng nhập vào hệ thống.

Bảo mật thông tin người dùng bằng cách mã hóa mật khẩu (dùng BCrypt).

### 2.1.2.Ghi chép thu – chi

Giao diện đơn giản để nhập thông tin thu nhập hoặc khoản chi tiêu.

Các trường thông tin bao gồm: loại giao dịch (thu/chi), số tiền, danh mục, ghi chú, ngày thực hiện.

Hỗ trợ phân loại khoản mục theo danh mục như: Ăn uống, Giao thông, Giải trí, Học tập, Khác...

Cho phép chỉnh sửa hoặc xóa giao dịch đã tạo

### 2.1.3.Quản lý danh mục chi tiêu

Cho phép người dùng tùy chỉnh danh mục chi tiêu theo nhu cầu (thêm, sửa, xóa danh mục).

Hệ thống cung cấp sẵn một số danh mục mặc định.

Các danh mục được sử dụng để thống kê và hiển thị biểu đồ.

### 2.1.4.Xem thống kê và báo cáo tài chính

Thống kê tổng thu – chi theo tuần, tháng, năm.

Hiển thị số dư hiện tại của người dùng.

Biểu đồ hình tròn và cột thể hiện tỷ lệ các khoản chi theo danh mục.

Biểu đồ đường hiển thị biến động số dư theo thời gian.

### 2.1.5.Đặt hạn mức chi tiêu

Người dùng có thể thiết lập hạn mức chi tiêu cho từng tháng.

Khi tổng chi tiêu vượt quá hạn mức, hệ thống sẽ gửi cảnh báo.

Giao diện hiển thị phần trăm mức tiêu dùng so với hạn mức đã đặt.

## Các yêu cầu phi chức năng

Bên cạnh các chức năng chính, hệ thống cũng cần đáp ứng các yêu cầu phi chức năng nhằm đảm bảo trải nghiệm người dùng và chất lượng phần mềm.

### 2.2.1.Hiệu năng

Hệ thống phải phản hồi nhanh (dưới 2 giây) cho các thao tác như ghi giao dịch, tìm kiếm, thống kê.

Cơ sở dữ liệu được tối ưu bằng chỉ mục (index) cho truy vấn nhanh.

### 2.2.2.Tính khả dụng

Hệ thống cần hoạt động ổn định 24/7 trên môi trường triển khai (Heroku).

Có thể xử lý nhiều người dùng đồng thời.

### 2.2.3.Bảo mật

Mã hóa mật khẩu người dùng khi lưu vào cơ sở dữ liệu.

Sử dụng Spring Security để bảo vệ tài nguyên backend.

Kiểm tra dữ liệu đầu vào để tránh tấn công SQL Injection, XSS.

### 2.2.4.Tính mở rộng

Hệ thống được thiết kế theo mô hình MVC kết hợp với kiến trúc RESTful API, trong đó backend (sử dụng Spring Boot) được xây dựng hoàn toàn độc lập và giao tiếp với frontend thông qua các API chuẩn. Kiến trúc này giúp dễ dàng tách rời giữa phần xử lý dữ liệu và phần hiển thị giao diện, từ đó nâng cao tính linh hoạt, khả năng bảo trì và khả năng phát triển đa nền tảng.

Backend tách biệt với frontend, có thể phát triển phiên bản di động sau này.

### 2.2.5.Khả năng bảo trì

Viết mã sạch, dễ hiểu, có tài liệu chú thích rõ ràng.

Cấu trúc thư mục rõ ràng theo chuẩn Spring Boot: controller, service, repository, model.

Dễ nâng cấp, kiểm thử và chỉnh sửa khi cần thiết.

### 2.2.6.Tính thân thiện với người dùng

Giao diện đơn giản, dễ sử dụng với người không am hiểu công nghệ.

Thiết kế responsive, hiển thị tốt trên cả máy tính và điện thoại.

Sử dụng màu sắc, biểu tượng và biểu đồ hợp lý để minh họa thông tin chi tiêu.

# THIẾT KẾ HỆ THỐNG

## Kiến trúc tổng thể hệ thống

Hệ thống được xây dựng theo mô hình Client - Server với các thành phần chính:

Client: Giao diện người dùng (Web/App) cho phép người dùng tương tác với hệ thống.

Server (API): Là nơi xử lý logic, xác thực người dùng, xử lý dữ liệu và phản hồi các yêu cầu từ client.

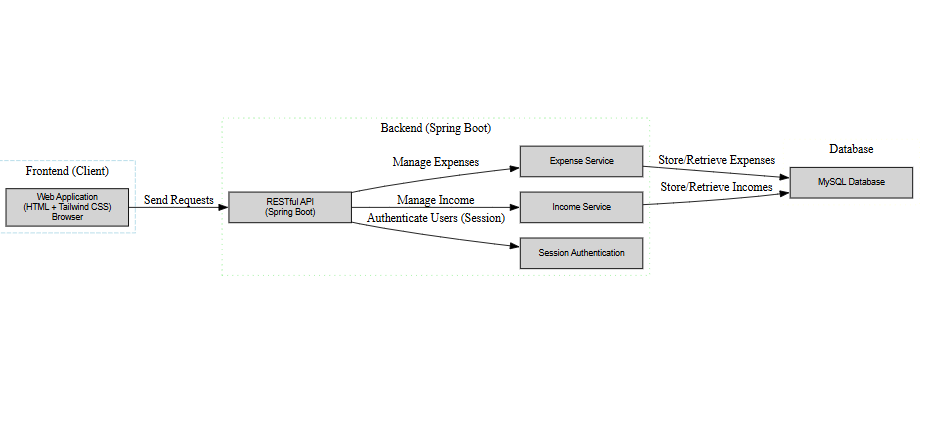
Cơ sở dữ liệu (Database): Lưu trữ toàn bộ dữ liệu liên quan đến người dùng.

Frontend: Được xây dựng với HTML, CSS (dùng Tailwind CSS để làm đẹp và responsive).

Backend: Spring Boot xử lý API và logic nghiệp vụ.

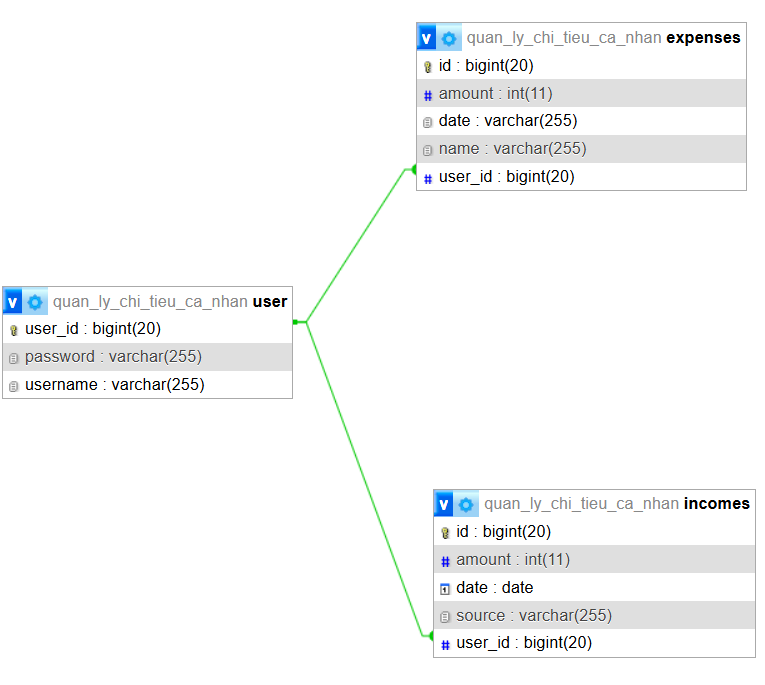
Database: MySQL lưu trữ dữ liệu người dùng, chi tiêu và thu nhập

## Sơ Đồ Kiến Trúc Hệ Thống



*Hình 1 Sở đồ kiến trúc của hệ thống 1*

## Sơ đồ ERD



*HÌnh 2 Sở đồ ERD*

## Thiết kế api swagger mô tả các endpoint cấu trúc request respone

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Endpoint | Phương thức | Mô tả | Tham số | Yêu cầu  (Request) | Phản hồi (Response) |
| /auth/login | POST | Đăng nhập người dùng bằng username và password | username: Tên người dùng password: Mật khẩu | Json{  "username": "user1",  "password": "password123"} | - **200 OK**:  Đăng nhập thành công. - **401 Unauthorized**:  Thông tin đăng nhập  không hợp lệ. |
| /expenses | GET | Lấy danh sách chi tiêu của người dùng | - Không có tham số (Yêu cầu xác thực người dùng qua đăng nhập) | - Không có request body. | - **200 OK**:  Danh sách chi tiêu  của người dùng.  json[  { "id": 1,  "description":  "Mua cà phê",  "amount": 50.000  "date":  "2023-05-01",  "category": "Ăn uống" } ] |
| /expenses | POST | Thêm mới một giao dịch chi tiêu | - description: Mô tả chi tiêu - amount: Số tiền chi tiêu - category: Danh mục chi tiêu | json<br>{  "description": "Mua cà phê",  "amount": 50.00,  "category": "Ăn uống"} | **201 Created**:  Chi tiêu đã được  tạo thành công. json{ "id": 1 }  **400 Bad Request**:  Thông tin chi tiêu  không hợp lệ. |
| /expenses/  {expenseId} | PUT | Cập nhật một giao dịch chi tiêu | expenseId: ID của chi tiêu | json{"description": "Mua trà sữa",  "amount": 55.00,  "category":  "Ăn uống"} | **200 OK**  : Chi tiêu  đã được  cập nhật  thành  công. **400 Bad Request**:  Dữ liệu  không hợp  lệ.  **404 Not Found**:  Không  tìm thấy  chi tiêu  với ID  này. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| /expenses/  {expenseId} | DELETE | Xóa một giao dịch chi tiêu theo ID | expenseId: ID của chi tiêu cần xóa | Không có request body. | **204 No Content**:  Xóa chi tiêu  thành công. **404 Not Found**:  Không tìm thấy  chi tiêu với ID này. |
| /expenselimit | GET | Lấy giới hạn chi tiêu của người dùng | Không có tham số (Yêu cầu xác thực người dùng qua đăng nhập) | Không có request body. | **200 OK**:  Trả về giới hạn chi  tiêu của người dùng  json{ "limit\_amount"  : 1000.00 } **401 Unauthorized**:  Không có quyền  truy cập. |
| /expenselimit | PUT | Cập nhật giới hạn chi tiêu của người dùng | limit\_  amount: Số tiền giới hạn chi tiêu | json{  "limit\_amout: 1500.00} | - **200 OK**:  Cập nhật giới hạn  chi tiêu thành công. - **400 Bad Request**:  Thông tin giới hạn  chi tiêu không  hợp lệ. |

*Bảng 1 Mô tả các API và Endpoint*

## Thiết kế giao diện (UI/UX)

## 3.5.1. Giao diện trang đăng nhập

## 

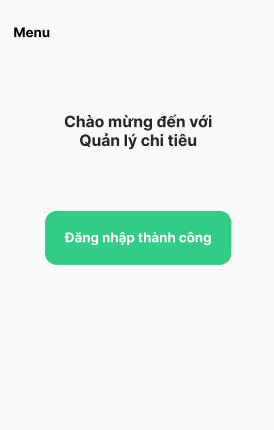
Hình 1 Trang đăng nhập

### 3.5.2. Giao diện trang đăng ký

### 

Hình 2 trang đăng ký

### 3.5.3. Giao diện trang chủ



Hình 3 Trang chủ

## 3.5.4. Giao diện menu



Hình 4 Menu **3.5.5 . Giao diện trang quản lý chi tiêu**

### 

Hình 5 Trang quản lý chi tiêu

### 3.5.6. Giao diện trang quản lý thu nhập

### 

Hình 6 Trang quản lý thu nhập

# TRIỂN KHAI CÁC CÔNG CỤ SỬ DỤNG

## Danh sách các công cụ sử dụng trong dự án

**Spring Boot**

**Mô tả:** Spring Boot là một framework Java mạnh mẽ, giúp xây dựng các ứng dụng web một cách nhanh chóng và dễ dàng. Dự án sử dụng Spring Boot để xây dựng API backend, xử lý logic nghiệp vụ, và tương tác với cơ sở dữ liệu.

**Vai trò trong dự án:** Xử lý các yêu cầu từ phía người dùng, quản lý dữ liệu chi tiêu, và cung cấp API cho các tính năng của ứng dụng.

**Tailwind CSS**

**Mô tả:** Tailwind CSS là một framework CSS tiện lợi cho phép xây dựng giao diện người dùng (UI) theo cách linh hoạt và tùy chỉnh.

**Vai trò trong dự án:** Dùng để thiết kế giao diện người dùng cho ứng dụng, tạo các thành phần UI dễ dàng và đẹp mắt mà không cần viết quá nhiều CSS thủ công.

**MySQL**

**Mô tả:** MySQL là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở.

**Vai trò trong dự án:** Lưu trữ dữ liệu chi tiêu cá nhân, thông tin người dùng, và các thông tin liên quan đến các giao dịch tài chính.

**Postman**

**Mô tả:** Postman là một công cụ phát triển API phổ biến, dùng để kiểm thử các API và gửi yêu cầu HTTP.

**Vai trò trong dự án:** Sử dụng để kiểm thử API backend của ứng dụng, đảm bảo rằng các yêu cầu được thực thi chính xác và trả về kết quả đúng đắn

**Docker**

**Mô tả:** Docker là một nền tảng để phát triển, vận hành và phân phối các ứng dụng trong các container.

**Vai trò trong dự án:** Sử dụng Docker để đóng gói ứng dụng và các môi trường phát triển thành các container độc lập, giúp triển khai ứng dụng dễ dàng và đồng nhất trên các môi trường khác nhau.

**Swagger**

**Mô tả:** Swagger là một công cụ tạo tài liệu API tự động cho ứng dụng web.

**Vai trò trong dự án:** Tạo và duy trì tài liệu API cho backend, giúp các nhà phát triển khác hiểu rõ hơn về các endpoints và cách sử dụng chúng.

**GitHub**

**Mô tả:** GitHub là một nền tảng quản lý mã nguồn, hỗ trợ công cụ quản lý phiên bản Git.

**Vai trò trong dự án:** Quản lý mã nguồn của dự án, đồng bộ hóa công việc giữa các thành viên trong nhóm và theo dõi lịch sử thay đổi của mã nguồn.

**JIRA**

**Mô tả:** JIRA là công cụ quản lý dự án và theo dõi vấn đề của Atlassian.

**Vai trò trong dự án:** Quản lý và theo dõi tiến độ phát triển, các lỗi phần mềm, và các công việc trong suốt vòng đời của dự án.

**GitHub Actions**

**Mô tả:** GitHub Actions là một công cụ CI/CD tích hợp trong GitHub, cho phép tự động hóa các quy trình phát triển phần mềm.

**Vai trò trong dự án:** Cấu hình quy trình CI/CD để tự động hóa việc build, kiểm thử và triển khai ứng dụng mỗi khi có thay đổi mã nguồn.

## Quy trình CI/CD với GitHub Actions

Quy trình CI/CD của dự án sử dụng GitHub Actions nhằm tự động hóa quá trình xây dựng, kiểm thử và triển khai ứng dụng:

**CI (Continuous Integration):**

Khi có thay đổi mã nguồn được đẩy lên repository trên GitHub, GitHub Actions sẽ tự động thực hiện các bước sau:

Xây dựng lại ứng dụng.

Kiểm thử mã nguồn (unit test, integration test).

Kiểm tra mã nguồn với các công cụ như SonarQube (nếu có).

**CD (Continuous Deployment):**

Sau khi quá trình kiểm thử thành công, GitHub Actions sẽ tự động triển khai ứng dụng vào môi trường sản xuất hoặc staging. Quá trình này được thực hiện thông qua việc chạy các script Docker để xây dựng và triển khai ứng dụng.

## ****Cấu hình Docker và quy trình triển khai ứng dụng****

**Cấu hình Docker:**

**Dockerfile:** Một tệp Dockerfile đã được sử dụng để xây dựng container cho ứng dụng Spring Boot. Tệp này chứa các lệnh để tạo môi trường phù hợp, cài đặt các phụ thuộc và chạy ứng dụng trong container.

**docker-compose.yml:** Docker Compose được sử dụng để định nghĩa và chạy các ứng dụng Docker trong môi trường phát triển, bao gồm cả ứng dụng backend (Spring Boot) và cơ sở dữ liệu MySQL.

**Quy trình triển khai ứng dụng:**

**Bước 1:** Ứng dụng được đóng gói vào một Docker container.

**Bước 2:** Docker container được triển khai lên một môi trường staging hoặc production (có thể sử dụng các dịch vụ như AWS, DigitalOcean hoặc máy chủ riêng).

**Bước 3:** Sau khi triển khai, người dùng có thể truy cập ứng dụng web qua trình duyệt để thực hiện các chức năng quản lý chi tiêu cá nhân.

# QUẢN LÝ DỰ ÁN