

**TRƯỜNG ĐẠI HỌC TRÀ VINH**  
**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**  
**KHOA CÔNG NGHỆ THÔNG TIN**

---



**BÁO CÁO KẾT THÚC HỌC PHẦN**  
**CÔNG NGHỆ PHẦN MỀM**  
**(MSHP: 220055)**

**TÊN ĐỀ TÀI**  
**XÂY DỰNG ỨNG DỤNG GIỚI THIỆU**  
**CÂY TRỒNG**

**Sinh viên thực hiện:**

110122070	Đỗ Gia Hào	DA22TTD
110122090	La Thuần Khang	DA22TTD
110122103	Hà Gia Lộc	DA22TTD

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

**Trà Vinh, tháng 7 năm 2025**

**TRƯỜNG ĐẠI HỌC TRÀ VINH**  
**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**  
**KHOA CÔNG NGHỆ THÔNG TIN**

---



**BÁO CÁO KẾT THÚC HỌC PHẦN**  
**CÔNG NGHỆ PHẦN MỀM**  
**(MSHP: 220055)**

**TÊN ĐỀ TÀI**  
**XÂY DỰNG ỨNG DỤNG GIỚI THIỆU**  
**CÂY TRỒNG**

**Sinh viên thực hiện:**

110122070	Đỗ Gia Hào	DA22TTD
110122090	La Thuần Khang	DA22TTD
110122103	Hà Gia Lộc	DA22TTD

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

**Trà Vinh, tháng 7 năm 2025**

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Trà Vinh, ngày ... tháng ... năm 2025  
**Giáo viên hướng dẫn**  
(Ký và ghi rõ họ tên)

**NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN**

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Trà Vinh, ngày ... tháng ... năm 2025  
**Giáo viên phản biện**  
(Ký và ghi rõ họ tên)

## LỜI CẢM ƠN

Nhóm chúng em xin được gửi lời tri ân sâu sắc và chân thành nhất đến Thầy **Nguyễn Bảo Ân** – người đã không quản ngại khó khăn, luôn tận tâm, tận lực truyền đạt cho chúng em những kiến thức quý giá trong suốt quá trình học tập học phần **Công nghệ phần mềm**.

Chặng đường học tập vừa qua, với sự hướng dẫn tận tình của Thầy, chúng em không chỉ được tiếp cận với những khái niệm và kiến thức chuyên môn sâu rộng mà còn học hỏi được phương pháp tư duy logic, làm việc nhóm hiệu quả, và đặc biệt là tinh thần trách nhiệm trong từng sản phẩm mình tạo ra. Những bài giảng của Thầy luôn chứa đựng không chỉ tri thức mà còn là nguồn cảm hứng to lớn, giúp chúng em thêm yêu thích môn học này và nhận ra được vai trò quan trọng của công nghệ phần mềm trong cuộc sống cũng như trong sự nghiệp sau này.

Chúng em càng thêm biết ơn khi Thầy đã trực tiếp hướng dẫn, hỗ trợ và góp ý chân thành trong quá trình chúng em thực hiện bài báo cáo cuối kỳ. Sự chỉ bảo tận tình, sự khắt khe đúng mực và lòng nhiệt huyết của Thầy đã giúp nhóm em hiểu rõ hơn về quy trình phát triển phần mềm, cách áp dụng kiến thức vào thực tế, cũng như rèn luyện cho chúng em tinh thần cầu thị và không ngừng hoàn thiện bản thân.

Tuy đã cố gắng hết sức trong khả năng của mình, nhưng do còn hạn chế về kiến thức cũng như kinh nghiệm thực tế, nhóm em khó tránh khỏi những thiếu sót trong quá trình tìm hiểu, phân tích và trình bày nội dung bài báo cáo. Chúng em rất mong nhận được sự góp ý quý báu từ Thầy để bài báo cáo được hoàn thiện hơn, và để chúng em có thể tích lũy thêm nhiều bài học giá trị cho chặng đường học tập sắp tới.

Cuối cùng, bằng tất cả sự kính trọng và biết ơn, nhóm chúng em xin kính chúc Thầy **Nguyễn Bảo Ân** luôn dồi dào sức khỏe, giữ vững ngọn lửa đam mê với nghề giáo và tiếp tục truyền cảm hứng, lan tỏa tri thức đến nhiều thế hệ sinh viên hơn nữa.

Nhóm chúng em xin chân thành cảm ơn!

# MỤC LỤC

CHƯƠNG 1: TỔNG QUAN.....	1
1.1. Tên dự án và chủ đề.....	1
1.2. Mục tiêu của đề tài.....	1
1.3. Lý do chọn đề tài .....	2
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT .....	3
2.1. Agile.....	3
2.1.1. Giới thiệu về Agile .....	3
2.1.2. Tuyên ngôn của Agile.....	3
2.2. SCRUM .....	4
2.2.1. Giới thiệu về SCRUM.....	4
2.2.2. SCRUM và Sprints.....	5
2.2.3. Các thực hành chính của SCRUM .....	6
2.2.4. Trạng thái của Product Backlog Item.....	7
2.3. Jira.....	7
2.3.1. Giới thiệu về Jira .....	7
2.3.2. Các khái niệm cốt lõi trong Jira .....	7
CHƯƠNG 3: PHÂN TÍCH YÊU CẦU VÀ THIẾT KẾ HỆ THỐNG.....	9
3.1. Các yêu cầu chức năng .....	9
3.2. Các yêu cầu phi chức năng .....	10
3.3. Kiến trúc hệ thống.....	10
3.4. Các công nghệ sử dụng.....	11
3.4.1. Frontend.....	11
3.4.2. Backend .....	12
3.4.3. DataBase.....	14
3.4.4. Docker .....	14
3.4.5. GitHub Action .....	15
3.5. Giao diện hệ thống.....	19
CHƯƠNG 4: QUẢN LÝ DỰ ÁN.....	27
4.1. Jira để quản lý kế hoạch.....	27

4.2. Phân công từng thành viên trong nhóm.....	27
4.2.1. Đối với Frontend .....	27
4.2.2. Đối với Backend.....	28
4.2.3. Đối với Database và Docker .....	29
CHƯƠNG 5: TRIỂN KHAI VÀ KIỂM THỬ .....	30
5.1. Postman.....	30
5.1.1. Gửi yêu cầu GET để lấy danh sách cây trồng.....	30
5.1.2. Gửi yêu cầu POST để thêm cây trồng mới .....	31
5.1.3. Gửi yêu cầu GET để lấy chi tiết cây trồng.....	31
5.1.4. Gửi yêu cầu PUT để cập nhật thông tin cây.....	32
5.1.5. Gửi yêu cầu DELETE để xóa cây khỏi hệ thống.....	33
5.2. Swagger.....	33
5.3. Kiểm thử API .....	34
CHƯƠNG 6: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	35
6.1. Kết luận.....	35
6.2. Hạn chế .....	35
6.3. Hướng phát triển .....	36
CHƯƠNG 7: TÀI LIỆU THAM KHẢO .....	38

# DANH MỤC HÌNH

Hình 1 SCRUM.....	5
Hình 2 Sprints.....	6
Hình 3 Sơ đồ kiến trúc hệ thống Client/Server .....	11
Hình 4 Giao diện trang chủ hệ thống .....	20
Hình 5 Giao diện trang danh mục hệ thống .....	21
Hình 6 Giao diện trang chi tiết cây trồng.....	22
Hình 7 Giao diện trang kết quả tìm kiếm.....	23
Hình 8 Giao diện trang đăng nhập dành cho Quản trị viên .....	24
Hình 9 Giao diện trang Tổng quan của Quản trị viên .....	24
Hình 10 Giao diện trang Quản lý cây trồng của Quản trị viên .....	25
Hình 11 Giao diện trang Quản lý danh mục của Quản trị viên.....	25
Hình 12 Giao diện trang Quản lý ảnh của Quản trị viên .....	26
Hình 13 Backlog của thành viên thứ nhất.....	27
Hình 14 Backlog của thành viên thứ hai.....	28
Hình 15 Backlog của thành viên thứ ba .....	29
Hình 16 Gửi yêu cầu GET để lấy danh sách cây .....	30
Hình 17 Kết quả kiểm thử API GET để lấy danh sách cây trồng .....	30
Hình 18 Gửi yêu cầu POST để thêm cây mới.....	31
Hình 20 Kết quả kiểm thử API POST .....	31
Hình 20 Gửi yêu cầu GET để lấy chi tiết cây .....	31
Hình 21 Kết quả kiểm thử API GET .....	32
Hình 22 Gửi yêu cầu PUT để cập nhật cây .....	32
Hình 23 Kết quả kiểm thử API PUT .....	32
Hình 24 Gửi yêu cầu DELETE để xóa cây .....	33
Hình 25 Giao diện Swagger tổng quan .....	33
Hình 26 Kiểm thử API thực tế trên giao diện người dùng .....	34



# CHƯƠNG 1: TỔNG QUAN

## 1.1. Tên dự án và chủ đề

Dự án website được xây dựng và phát triển với tên gọi chính thức là “Green Garden”. Tên gọi mang ý nghĩa là “Khu vườn xanh” không chỉ gợi lên hình ảnh một không gian trong lành, tràn đầy sức sống của thế giới thực vật mà còn định vị website như một điểm đến thanh bình, một “khu vườn tri thức” trên không gian mạng. Đây là nơi người đọc có thể thanh thoi dạo bước, khám phá và thu nhận những kiến thức bổ ích để tự tay kiến tạo nên khu vườn xanh của riêng mình.

Với định hướng đó, dự án xác định chủ đề cốt lõi là: “Một thư viện số toàn diện, chuyên cung cấp kiến thức về thế giới cây trồng”. Trọng tâm của “Green Garden” là hoạt động cung cấp thông tin một chiều, đóng vai trò như một cuốn bách khoa toàn thư kỹ thuật số đáng tin cậy. Nền tảng này được thiết kế để trở thành một nguồn tham khảo tri thức, nơi mọi thông tin đều được nghiên cứu, biên soạn và trình bày một cách khoa học, trực quan và dễ tiếp cận. Thay vì tạo ra một diễn đàn thảo luận, “Green Garden” tập trung vào việc xây dựng một kho tàng kiến thức vững chắc, giúp người đọc tự tìm thấy câu trả lời và cảm hứng để bắt đầu hoặc tiếp tục hành trình chăm sóc cây.

## 1.2. Mục tiêu của đề tài

Với định hướng là một trang thông tin chuyên biệt, website “Green Garden” hướng đến các mục tiêu cụ thể sau:

**Xây dựng một cơ sở dữ liệu cây trồng phong phú và đáng tin cậy:** Mục tiêu hàng đầu của dự án là biên soạn và cung cấp một danh mục chi tiết về các loại cây trồng phổ biến, từ cây cảnh nội thất, cây văn phòng, cây thủy sinh đến các loại cây có giá trị phong thủy. Mỗi hồ sơ về cây sẽ bao gồm thông tin đầy đủ về đặc điểm hình thái, nguồn gốc, yêu cầu về ánh sáng, nước, độ ẩm, loại đất và các lưu ý đặc biệt, giúp người đọc có một cái nhìn tổng quan và chính xác nhất.

**Cung cấp các bài viết chuyên sâu:** Website sẽ xuất bản các bài viết, cẩm nang hướng dẫn chi tiết về kỹ thuật trồng và chăm sóc cây. Các chủ đề bao gồm

cách chọn chậu, cách trộn giá thể, phương pháp tưới nước hiệu quả, cách nhận biết và phòng trừ sâu bệnh, kỹ thuật nhân giống... Tất cả nhằm trang bị cho người đọc kiến thức cần thiết để cây trồng của họ luôn khỏe mạnh và phát triển tốt.

**Tối ưu hóa trải nghiệm tìm kiếm và tiếp nhận thông tin:** Giao diện website được thiết kế để trở nên thân thiện, trực quan, với hệ thống phân loại và bộ lọc thông minh. Người dùng có thể dễ dàng tìm kiếm cây theo tên, đặc tính (ưa bóng, ưa sáng), hoặc theo nhu cầu sử dụng (cây để bàn, cây lọc không khí). Cách trình bày nội dung kết hợp giữa văn bản và hình ảnh chất lượng cao sẽ giúp việc tiếp thu kiến thức trở nên sinh động và thú vị hơn.

**Truyền cảm hứng về lối sống xanh:** Thông qua việc cung cấp thông tin và những hình ảnh đẹp mắt, website gián tiếp khuyến khích và truyền cảm hứng cho người đọc về một lối sống gần gũi với thiên nhiên. Việc mang cây xanh vào không gian sống không chỉ giúp cải thiện thẩm mỹ mà còn mang lại nhiều lợi ích cho sức khỏe tinh thần, và dự án mong muốn lan tỏa thông điệp tích cực này.

### 1.3. Lý do chọn đề tài

Việc lựa chọn đề tài xuất phát từ những lý do sau:

**Xu hướng sống xanh ngày càng phổ biến:** Ngày nay, đặc biệt là giới trẻ, có xu hướng decor nhà cửa bằng cây xanh. Việc trồng cây không chỉ để trang trí mà còn giúp cải thiện chất lượng không khí và mang lại cảm giác thư thái.

**Nhu cầu tìm kiếm thông tin về cây trồng tăng cao:** Nhiều người gặp khó khăn trong việc tìm kiếm thông tin đáng tin cậy về cách chăm sóc cây. Website sẽ là một nguồn tài nguyên hữu ích, giúp giải đáp những thắc mắc này.

**Mong muốn tạo ra một không gian kết nối:** Hiện nay có nhiều cộng đồng yêu cây trên mạng xã hội, cho thấy nhu cầu kết nối và chia sẻ của những người có cùng sở thích là rất lớn. “Green Garden” sẽ là một nền tảng chuyên biệt, đáp ứng nhu cầu này một cách toàn diện hơn.

**Góp phần bảo vệ môi trường:** Việc khuyến khích trồng cây cũng là một hành động thiết thực góp phần bảo vệ môi trường, chống biến đổi khí hậu và làm cho không gian sống trở nên trong lành hơn.

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

## 2.1. Agile

### 2.1.1. Giới thiệu về Agile

Agile là một phương pháp luận phát triển phần mềm linh hoạt, tập trung vào việc cung cấp các chức năng phần mềm một cách nhanh chóng và hiệu quả. Agile cho phép các nhóm phát triển phản hồi nhanh chóng đối với các thay đổi và phản hồi từ khách hàng, giảm thiểu chi phí phát triển thông qua các vòng lặp phát triển ngắn và tăng trưởng liên tục. Nó ra đời như một giải pháp cho các vấn đề của những mô hình phát triển truyền thống như mô hình Thác nước (Waterfall), vốn cứng nhắc và khó thay đổi khi có yêu cầu mới.

Trọng tâm của Agile là phát triển phần mềm theo từng giai đoạn ngắn, lặp đi lặp lại (gọi là các "iteration" hoặc "sprint"). Sau mỗi giai đoạn, nhóm phát triển sẽ tạo ra một phần của sản phẩm có thể hoạt động được và nhận phản hồi từ khách hàng. Điều này giúp sản phẩm cuối cùng bám sát hơn với mong muốn của khách hàng và cho phép đội ngũ thích ứng nhanh chóng với các thay đổi.

### 2.1.2. Tuyên ngôn của Agile

Tuyên ngôn Agile là tài liệu nền tảng, định hình tư duy Agile, được tạo ra vào năm 2001. Nó bao gồm 4 giá trị cốt lõi và 12 nguyên tắc.

❖ 4 giá trị cốt lõi của Agile:

1. Cá nhân và sự tương tác hơn là quy trình và công cụ.
2. Phần mềm chạy tốt hơn là tài liệu đầy đủ.
3. Hợp tác với khách hàng hơn là đàm phán hợp đồng.
4. Phản hồi với thay đổi hơn là bám sát kế hoạch.

❖ 12 nguyên tắc của Agile:

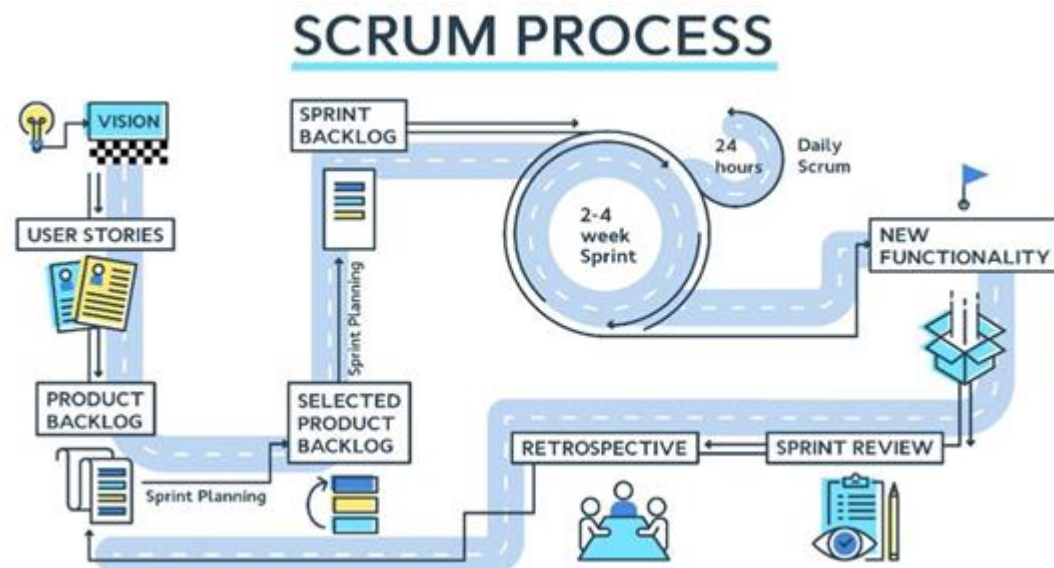
1. Ưu tiên khách hàng: Thỏa mãn khách hàng là ưu tiên số một, bằng cách giao sản phẩm có giá trị một cách sớm và liên tục.
2. Chào đón thay đổi: Luôn sẵn sàng chấp nhận các yêu cầu thay đổi, kể cả ở giai đoạn muộn, để tạo ra lợi thế.

3. Giao hàng thường xuyên: Giao sản phẩm hoạt động tốt một cách thường xuyên trong các khoảng thời gian ngắn (vài tuần hoặc vài tháng).
4. Hợp tác mỗi ngày: Người làm kinh doanh và đội phát triển phải hợp tác chặt chẽ với nhau hàng ngày.
5. Tin tưởng và trao quyền: Xây dựng dự án quanh những cá nhân có động lực, cung cấp cho họ sự hỗ trợ và tin tưởng cần thiết.
6. Giao tiếp trực tiếp: Trao đổi trực tiếp (mặt đối mặt) là cách giao tiếp hiệu quả nhất.
7. Sản phẩm là thước đo: Phần mềm hoạt động tốt là thước đo chính cho sự tiến bộ.
8. Phát triển bền vững: Duy trì một nhịp độ làm việc ổn định và bền vững mà mọi người có thể theo được lâu dài.
9. Chú trọng kỹ thuật: Luôn chú trọng đến chất lượng kỹ thuật và thiết kế tốt để duy trì sự linh hoạt.
10. Giữ sự đơn giản: Sự đơn giản là cốt lõi – hãy tối đa hóa những việc không cần làm.
11. Nhóm tự tổ chức: Những kiến trúc và thiết kế tốt nhất được tạo ra bởi các nhóm có khả năng tự tổ chức.
12. Cải tiến liên tục: Nhóm phải thường xuyên nhìn lại và điều chỉnh cách làm việc để ngày càng hiệu quả hơn.

## **2.2. SCRUM**

### **2.2.1. Giới thiệu về SCRUM**

Scrum là một trong những phương pháp (methodology) phổ biến và cụ thể nhất để triển khai tư duy Agile. Nó cung cấp một bộ quy tắc, vai trò, sự kiện và công cụ rõ ràng để quản lý và hoàn thành công việc. Scrum giúp các nhóm giải quyết các vấn đề phức tạp bằng cách chia nhỏ chúng thành những phần việc nhỏ hơn và giải quyết chúng một cách tuần tự trong các khoảng thời gian cố định.



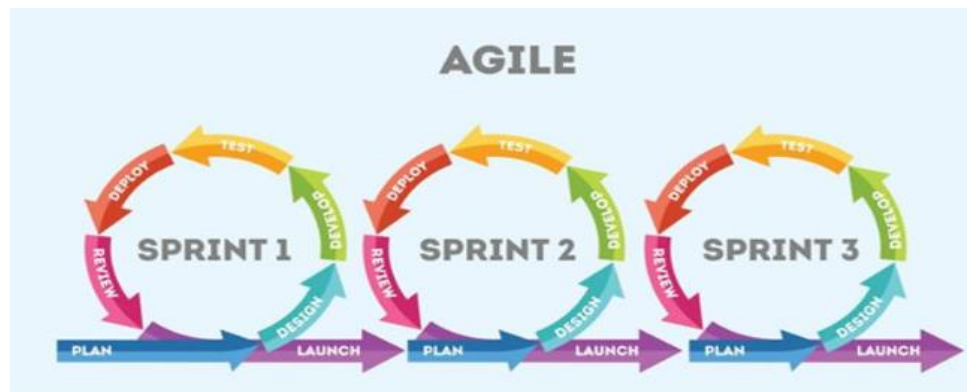
Hình 1 SCRUM

Trong Scrum, công việc cần hoàn thành được duy trì trong product backlog – một danh sách các hạng mục công việc cần hoàn thành. Mỗi phần tăng trưởng của phần mềm thực hiện một số hạng mục công việc từ product backlog.

### 2.2.2. SCRUM và Sprints

Sprint là trái tim của Scrum. Đó là một khoảng thời gian cố định, thường từ 1 đến 4 tuần, trong đó nhóm phát triển tập trung hoàn thành một lượng công việc đã được chọn lọc để tạo ra một phần sản phẩm hoàn chỉnh và có khả năng sử dụng được.

Một Sprint mới sẽ bắt đầu ngay sau khi Sprint trước đó kết thúc, tạo thành một chuỗi phát triển liên tục. Toàn bộ các sự kiện quan trọng của Scrum như Lập kế hoạch Sprint (Sprint Planning), Họp hằng ngày (Daily Scrum), Sơ kết Sprint (Sprint Review), và Cải tiến Sprint (Sprint Retrospective) đều diễn ra trong một Sprint.



Hình 2 Sprints

### 2.2.3. Các thực hành chính của SCRUM

Scrum được vận hành thông qua các vai trò, sự kiện và công cụ (artefact) chính:

#### ❖ Các vai trò (Scrum Team):

Product Owner: Người chịu trách nhiệm về giá trị của sản phẩm, quản lý Product Backlog.

Scrum Master: Người đảm bảo nhóm tuân thủ đúng quy trình Scrum, loại bỏ các trở ngại.

Development Team (Nhóm Phát triển): Nhóm các chuyên gia tự quản lý để hoàn thành công việc.

#### ❖ Các sự kiện (Events):

Sprint Planning: Lập kế hoạch cho Sprint.

Daily Scrum: Cuộc họp ngắn hằng ngày để đồng bộ công việc.

Sprint Review: Sơ kết, trình bày sản phẩm cho các bên liên quan.

Sprint Retrospective: Nhìn lại để cải tiến quy trình làm việc.

#### ❖ Các công cụ (Artifacts):

Product Backlog: Danh sách các yêu cầu, tính năng của sản phẩm.

Sprint Backlog: Danh sách các mục được chọn từ Product Backlog để hoàn thành trong một Sprint.

Increment: Phần sản phẩm "Hoàn thành" được tạo ra sau mỗi Sprint.

#### 2.2.4. Trạng thái của Product Backlog Item

Một **Product Backlog Item (PBI)** là một đơn vị công việc trong Product Backlog, thường ở dạng một User Story (câu chuyện người dùng). Các PBI không có một bộ trạng thái tiêu chuẩn cố định trong Scrum, nhưng trong thực tế, các nhóm thường sử dụng một luồng công việc để theo dõi chúng. Một luồng trạng thái phổ biến có thể bao gồm:

**New/To Do** : Yêu cầu vừa được thêm vào Product Backlog.

**Approved/Refined** : Yêu cầu đã được Product Owner xem xét, làm rõ và sẵn sàng để được đưa vào một Sprint.

**Committed/In Sprint**: Nhóm phát triển đã chọn PBI này để thực hiện trong Sprint hiện tại.

**Done**: PBI đã đáp ứng tất cả các tiêu chí trong "Định nghĩa Hoàn thành" và được bàn giao.

### 2.3. Jira

#### 2.3.1. Giới thiệu về Jira

Jira là một công cụ phần mềm do công ty Atlassian của Úc phát triển. Ban đầu, nó được tạo ra với mục đích chính là theo dõi lỗi cho các nhà phát triển phần mềm. Tuy nhiên, qua nhiều năm, Jira đã phát triển thành một nền tảng quản lý dự án cực kỳ mạnh mẽ và linh hoạt, được sử dụng rộng rãi bởi các đội nhóm ở mọi quy mô và lĩnh vực, đặc biệt là các nhóm phát triển phần mềm theo phương pháp Agile.

#### 2.3.2. Các khái niệm cốt lõi trong Jira

##### ❖ Project (Dự án)

Là không gian làm việc chính, là một "container" chứa tất cả các công việc, thông tin và thành viên liên quan đến một mục tiêu cụ thể. Mỗi dự án trong Jira có thể được cấu hình với một quy trình làm việc (workflow) và một bộ các loại công việc (issue types) riêng.

## ❖ Issue (Vấn đề / Công việc)

Đây là khái niệm cơ bản và quan trọng nhất trong Jira. Issue là một đơn vị công việc cần được thực hiện. Mọi thứ trong Jira đều là một "Issue", từ một tính năng lớn đến một lỗi nhỏ. Các loại Issue phổ biến nhất bao gồm:

**Epic:** Một khối công việc lớn, một tính năng chính mà có thể được chia thành nhiều Story nhỏ hơn.

**Story (User Story):** Một yêu cầu hoặc một tính năng được viết dưới góc nhìn của người dùng cuối.

**Task:** Một công việc cụ thể cần được hoàn thành nhưng không nhất thiết phải là một tính năng dành cho người dùng.

**Bug:** Một lỗi hoặc một vấn đề trong sản phẩm cần được sửa.

**Sub-task:** Một công việc con, chi tiết hơn để hoàn thành một Story hoặc một Task.

## ❖ Board (Bảng)

Board là công cụ trực quan hóa công việc của nhóm. Nó hiển thị các "Issue" dưới dạng các thẻ (card) và được sắp xếp vào các cột tương ứng với trạng thái của chúng. Có hai loại Board chính:

**Scrum Board:** Được thiết kế cho các nhóm làm việc theo Scrum. Nó tập trung vào việc hoàn thành công việc trong một khoảng thời gian cố định (Sprint). Board này có một Backlog nơi chứa tất cả công việc và một chế độ xem Active Sprint để theo dõi công việc trong Sprint hiện tại.

**Kanban Board:** Được thiết kế cho các nhóm tập trung vào luồng công việc liên tục (continuous flow). Mục tiêu của Kanban là tối ưu hóa luồng công việc và giới hạn số lượng công việc đang được thực hiện cùng lúc (Work In Progress - WIP).

## ❖ Workflow (Luồng công việc)

Workflow là chu trình sống của một "Issue", mô tả các trạng thái mà một Issue có thể đi qua từ lúc bắt đầu cho đến khi kết thúc. Một workflow đơn giản nhất thường là:

To Do (Cần làm) → In Progress (Đang làm) → Done (Hoàn thành)



# CHƯƠNG 3: PHÂN TÍCH YÊU CẦU VÀ THIẾT KẾ HỆ THỐNG

## 3.1. Các yêu cầu chức năng

Đây là các chức năng cụ thể mà hệ thống phải thực hiện. Các chức năng này được xác định dựa trên hai đối tượng tương tác chính với hệ thống: **Người dùng** (khách truy cập website) và **Quản trị viên** (người quản lý nội dung).

### ❖ Đối với người dùng:

Hệ thống phải hiển thị danh sách các cây trồng nổi bật hoặc mới được cập nhật ngay trên trang chủ.

Hệ thống phải cung cấp chức năng tìm kiếm, cho phép người dùng tìm cây theo tên (tiếng Việt hoặc tên khoa học).

Kết quả tìm kiếm phải được hiển thị một cách rõ ràng và trực quan.

Người dùng có thể xem trang chi tiết của một loại cây khi nhấp vào từ danh sách hoặc kết quả tìm kiếm.

Trang chi tiết phải hiển thị đầy đủ thông tin: hình ảnh, tên, mô tả, hướng dẫn chăm sóc (ánh sáng, nước, đất...).

Hệ thống phải hỗ trợ phân trang (pagination) cho danh sách cây trồng để tránh tải quá nhiều dữ liệu cùng lúc.

### ❖ Đối với quản trị viên:

Hệ thống phải có một trang đăng nhập riêng và an toàn dành cho Quản trị viên.

Sau khi đăng nhập thành công, Quản trị viên được chuyển đến trang Dashboard quản lý.

Tại trang quản lý, Quản trị viên có quyền Thêm một loại cây mới vào cơ sở dữ liệu.

Quản trị viên có quyền Sửa thông tin của một loại cây đã tồn tại.

Quản trị viên có quyền Xóa một loại cây khỏi hệ thống.

### 3.2. Các yêu cầu phi chức năng

Thời gian tải lần đầu của trang chủ không vượt quá 3 giây trên kết nối mạng trung bình.

Thời gian phản hồi của API cho các truy vấn lấy dữ liệu (lấy danh sách, lấy chi tiết) phải dưới 500ms.

Website phải tương thích và hiển thị tốt (responsive) trên các trình duyệt phổ biến (Chrome, Firefox, Safari) và trên các kích thước màn hình khác nhau (Desktop, Tablet, Mobile).

Mã nguồn cần được tổ chức rõ ràng, tuân thủ các quy tắc về coding convention.

Các API phải được tài liệu hóa (documented) bằng công cụ như Swagger/OpenAPI để dễ dàng cho việc kiểm thử và phát triển Frontend.

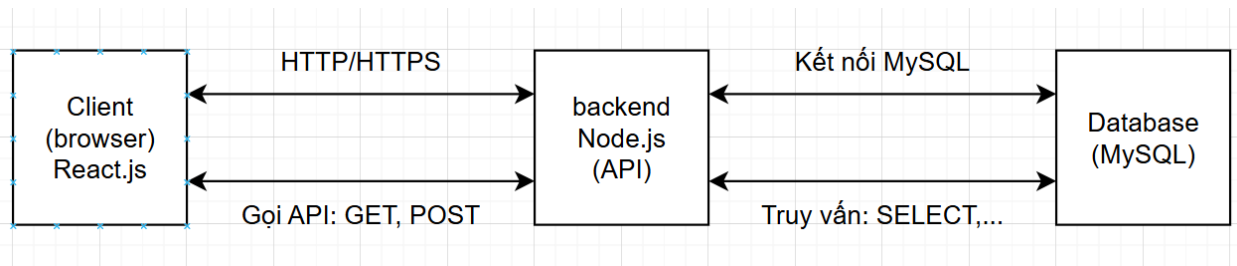
### 3.3. Kiến trúc hệ thống

Dự án áp dụng kiến trúc Client-Server ba lớp (3-tier) hiện đại, với sự phân tách rõ ràng giữa lớp trình bày (Presentation Layer), lớp ứng dụng (Application Layer) và lớp dữ liệu (Data Layer).

**Lớp Trình Bày (Client - Frontend):** Được xây dựng bằng React.js. Lớp này hoàn toàn chịu trách nhiệm về giao diện người dùng (UI) và trải nghiệm người dùng (UX). Nó giao tiếp với Backend thông qua các lời gọi HTTP API để hiển thị hoặc gửi dữ liệu.

**Lớp Ứng Dụng (Server - Backend):** Được xây dựng bằng Node.js. Lớp này đóng vai trò trung gian, chứa toàn bộ logic nghiệp vụ của ứng dụng (xử lý yêu cầu, xác thực,...) và là nơi duy nhất được phép tương tác trực tiếp với cơ sở dữ liệu.

**Lớp Dữ Liệu (Database):** Sử dụng hệ quản trị cơ sở dữ liệu quan hệ MySQL. Lớp này chỉ chịu trách nhiệm lưu trữ và truy xuất dữ liệu một cách hiệu quả và an toàn.



Hình 3 Sơ đồ kiến trúc hệ thống Client/Server

Luồng hoạt động của một yêu cầu tiêu biểu (Ví dụ: Người dùng xem chi tiết cây):

**Người dùng** nhấp vào một cây trên giao diện Frontend.

**React (Client)** bắt sự kiện, sử dụng React Router để cập nhật URL và gọi một hàm để lấy dữ liệu. Hàm này sử dụng **Axios** để gửi một request GET đến Backend, ví dụ: GET /api/plants/KimTien.

**Node.js/Express (Backend)** nhận được request, phân tích và chuyển đến controller xử lý /plants/:name. Controller sử dụng một thư viện kết nối (ví dụ: mysql2 hoặc một ORM như Sequelize) để thực thi câu lệnh SQL tới cơ sở dữ liệu: SELECT \* FROM plants WHERE name = KimTien;

**MySQL (Database)** nhận câu lệnh, tìm kiếm và trả về hàng (row) dữ liệu tương ứng.

**Backend** nhận dữ liệu từ MySQL, định dạng nó thành một đối tượng JSON và gửi lại cho Client trong một response HTTP với status 200.

**React (Client)** nhận được response JSON, cập nhật state của component và tự động render lại giao diện để hiển thị thông tin chi tiết của cây.

### 3.4. Các công nghệ sử dụng

#### 3.4.1. Frontend

**React.js:** Là một thư viện JavaScript mã nguồn mở chuyên dùng để xây dựng giao diện người dùng (UI). Nó cho phép chia nhỏ giao diện phức tạp thành các thành phần (components) độc lập và có thể tái sử dụng nhiều lần. Điều này giúp mã nguồn dễ quản lý, bảo trì và mở rộng.

ví dụ: Header.jsx, Footer.jsx, PlantCard.jsx

**React Router:** Là thư viện định tuyến (routing) tiêu chuẩn cho các ứng dụng React được dùng để quản lý việc điều hướng giữa các trang mà không cần tải lại toàn bộ trang web (Single Page Application - SPA).

Ví dụ: cho phép tạo các đường dẫn URL rõ ràng như `./plants/:name`, giúp người dùng có trải nghiệm liền mạch khi chuyển từ trang chủ sang trang chi tiết của một loại cây.

### 3.4.2. Backend

**Node.js:** Là một môi trường thực thi JavaScript phía máy chủ, có khả năng xử lý nhiều yêu cầu cùng lúc mà không bị chặn, rất phù hợp cho các ứng dụng web có nhiều truy vấn I/O (Input/Output) như việc truy vấn cơ sở dữ liệu.

**Express.js:** Là một framework ứng dụng web nhỏ gọn và linh hoạt cho Node.js. Cung cấp một bộ khung sườn vững chắc để xây dựng các RESTful API, giúp đơn giản hóa các tác vụ như định nghĩa các endpoint (ví dụ: `app.get('/api/plants', ...)`), quản lý request/response, và xử lý lỗi.

**RESTful API:** là một kiểu kiến trúc API phổ biến, dùng các phương thức HTTP như GET, POST, PUT, DELETE để thao tác với dữ liệu, và truyền dữ liệu qua định dạng như JSON.

Các phương thức:

- GET: truy xuất dữ liệu từ server.
- POST: Gửi dữ liệu mới lên server để tạo bản ghi
- PUT: Cập nhật toàn bộ thông tin của bản ghi
- DELETE: Xóa bản ghi khỏi cơ sở dữ liệu.

Ví dụ: một cây có “Name”: “Kim tiền” và “id”: “2”, sử dụng phương thức GET:

Yêu cầu: **GET /api/plants/2**

Phản hồi:

```
{  
  "id": 2,  
  "Name": "Cây Kim Tiền",
```

...  
}

Phương thức POST:

Yêu cầu (Admin): **POST /api/plants**

**Body:** {"vietnameseName": "Cây Lưỡi Hổ", ...}

Phản hồi:

{  
  "\_\_id": 2,  
  "vietnameseName": "Cây Lưỡi Hổ",  
  ...  
}

Phương thức PUT:

Yêu cầu (Admin): **PUT /api/plants/2**

**Body:** {"vietnameseName": "Cây Kim Tiền (Tài Lộc)", ...}

Phản hồi:

{  
  "\_\_id": 2,  
  "vietnameseName": "Cây Kim Tiền (Tài Lộc)",  
  ...  
}

Phương thức DELETE:

Yêu cầu (Admin): **DELETE /api/plants/2**

Phản hồi:

{  
  "success": true,  
  "message": "Đã xóa thành công."  
}

### 3.4.3. DataBase

**MySQL:** là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) mã nguồn mở phổ biến và mạnh mẽ nhất thế giới, cung cấp các cơ chế mạnh mẽ để đảm bảo tính toàn vẹn của dữ liệu thông qua các ràng buộc (khóa chính, khóa ngoại, not null,...), đảm bảo dữ liệu luôn chính xác và nhất quán.

### 3.4.4. Docker

Docker là một nền tảng giúp bạn đóng gói ứng dụng và các thành phần liên quan vào trong các container để chạy ở bất kỳ đâu, đảm bảo rằng môi trường chạy ứng dụng (Frontend, Backend, Database) trên máy của lập trình viên, trên server kiểm thử và server sản phẩm là hoàn toàn giống nhau, loại bỏ triệt để lỗi "Sao trên máy kia chạy được mà ta".

Docker Compose cho phép định nghĩa và khởi chạy toàn bộ hệ thống (gồm 3 container: frontend, backend, mysql\_db) chỉ bằng một lệnh duy nhất (docker-compose up), giúp quá trình triển khai trở nên cực kỳ đơn giản và nhanh chóng.

Cấu hình Docker của dự án:

```
version: '3.8'

services:
  mysql_db:
    image: mysql:8.0
    container_name: mysql_db
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: greengarden
      MYSQL_USER: user
      MYSQL_PASSWORD: password
    ports:
      - "3306:3306"
    volumes:
      - mysql_data:/var/lib/mysql

  backend:
    build:
      context: ./backend
    environment:
      - MYSQL_DATABASE=greengarden
```

```
- MYSQL_USER=user
- MYSQL_PASSWORD=password
- MYSQL_HOST=mysql_db
container_name: backend
ports:
  - "3000:3000"
depends_on:
  - mysql_db

frontend:
  build:
    context: ./frontend
  container_name: frontend
  ports:
    - "5173:5173"
  depends_on:
    - backend

volumes:
  mysql_data:
```

### 3.4.5. GitHub Action

```
name: CI/CD for Frontend and Backend

on:
  push:
    branches: [main] # Kích hoạt khi push lên nhánh main

jobs:
  build-frontend:
    runs-on: ubuntu-latest

    steps:
      - name: Clone code
        uses: actions/checkout@v3
```

```
- name: Setup Node.js
  uses: actions/setup-node@v3
  with:
    node-version: "20.19.3"

- name: Cache node_modules
  uses: actions/cache@v3
  with:
    path: src/frontend/node_modules
    key: ${{ runner.os }}-frontend-node-${{ hashFiles('src/frontend/package-lock.json') }}
    restore-keys: |
      ${{ runner.os }}-frontend-node-

- name: Install dependencies
  working-directory: src/frontend
  run: npm install --legacy-peer-deps

- name: Lint frontend
  working-directory: src/frontend
  run: npm run lint || echo "No lint script"

- name: Build Frontend (React + Vite)
  working-directory: src/frontend
  run: npm run build

- name: Kiểm tra dist có tồn tại không
  working-directory: src/frontend
  run: |
    echo "Kiểm tra thư mục dist"
```



ls -la

ls -la dist

- name: Upload build artifact

uses: actions/upload-artifact@v4

with:

name: frontend-build

path: src/frontend/dist

- name: List dist folder

working-directory: src/frontend

run: ls -la dist

- name: Build Docker image for frontend

working-directory: src/frontend

run: docker build -t your-frontend-image .

build-backend:

runs-on: ubuntu-latest

needs: build-frontend

steps:

- name: Clone code

uses: actions/checkout@v3

- name: Setup Node.js

uses: actions/setup-node@v3

with:

node-version: "20.19.3"

```
- name: Cache node_modules
  uses: actions/cache@v3
  with:
    path: src/backend/node_modules
    key: ${{ runner.os }}-backend-node-${{ hashFiles('src/backend/package-lock.json') }}
    restore-keys: |
      ${{ runner.os }}-backend-node-

- name: Install dependencies
  working-directory: src/backend
  run: npm install

- name: Lint backend
  working-directory: src/backend
  run: npm run lint || echo "No lint script"

- name: Download frontend build artifact
  uses: actions/download-artifact@v4
  with:
    name: frontend-build
    path: src/backend/public

- name: Test backend
  working-directory: src/backend
  run: npm test || echo "Không có test backend"

- name: Build Docker image for backend
  working-directory: src/backend
  run: docker build -t your-backend-image .
```

```
- name: Build Docker Compose
  run: docker compose -f src/docker-compose.yml build

- name: Deploy (nếu cần)
  run: echo "Triển khai lên server hoặc cloud tại đây"
```

### 3.5. Giao diện hệ thống

Thiết kế giao diện hệ thống được thực hiện Figma nhằm 4 mục đích chính:

**Biến ý tưởng thành sản phẩm nhìn thấy và bấm được mà không cần viết code.** Giúp hình dung rõ ràng sản phẩm cuối cùng sẽ trông như thế nào và hoạt động ra sao.

**Là "bản thiết kế chung" cho cả đội.** Quản lý, lập trình viên, và khách hàng đều nhìn vào một thứ duy nhất để trao đổi và thống nhất, tránh mọi hiểu lầm.

**Phát hiện lỗi thiết kế sớm và tiết kiệm chi phí.** Sửa một nút trên Figma chỉ mất vài phút, trong khi sửa một chức năng đã được lập trình xong có thể mất nhiều ngày.

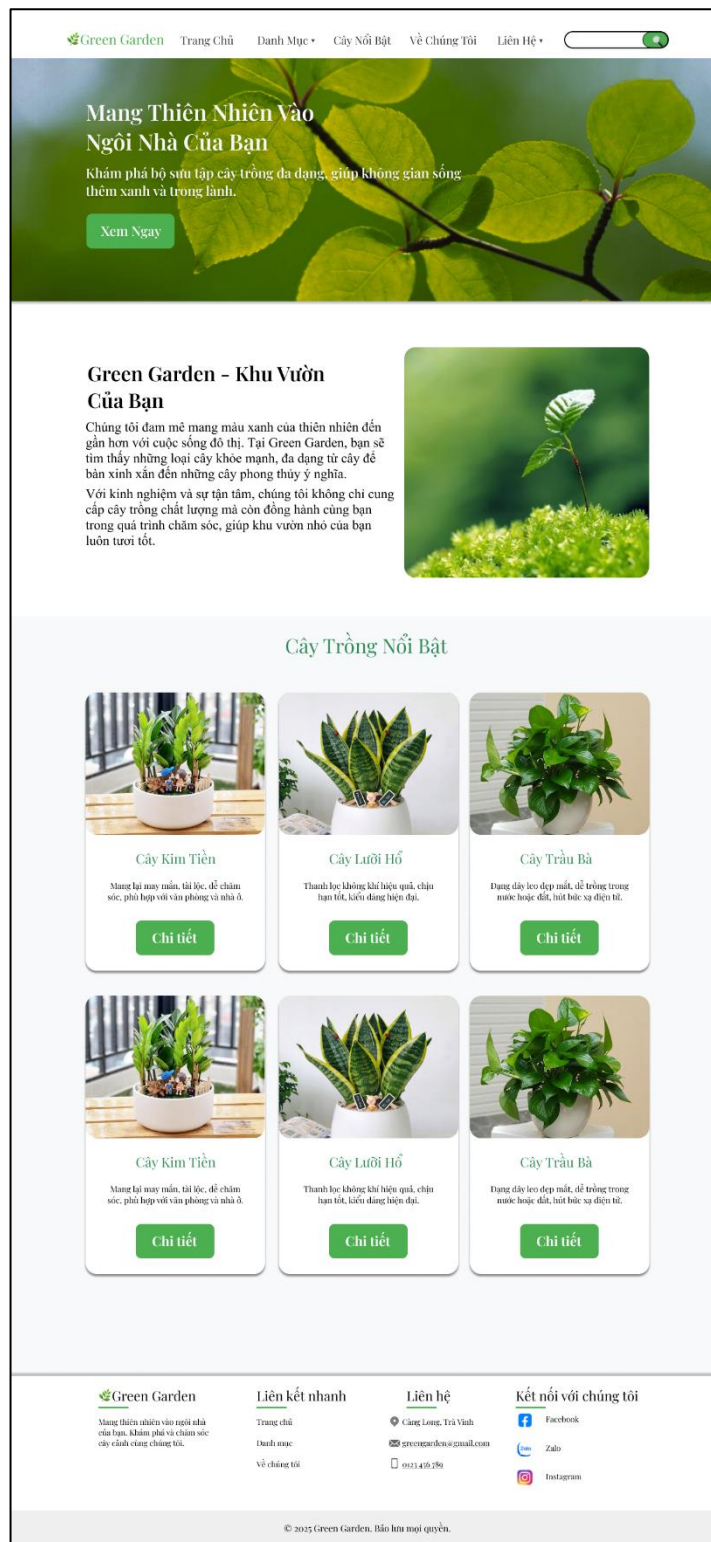
**Tăng tốc cho lập trình viên.** Cung cấp mọi thông số kỹ thuật chính xác (kích thước, màu sắc, font chữ, khoảng cách...). Lập trình viên chỉ việc lấy và code, không cần phải "đoán mò".

Giao diện của hệ thống sẽ gồm 2 phần:

Trang Giao diện bên phía **người dùng (client)**.

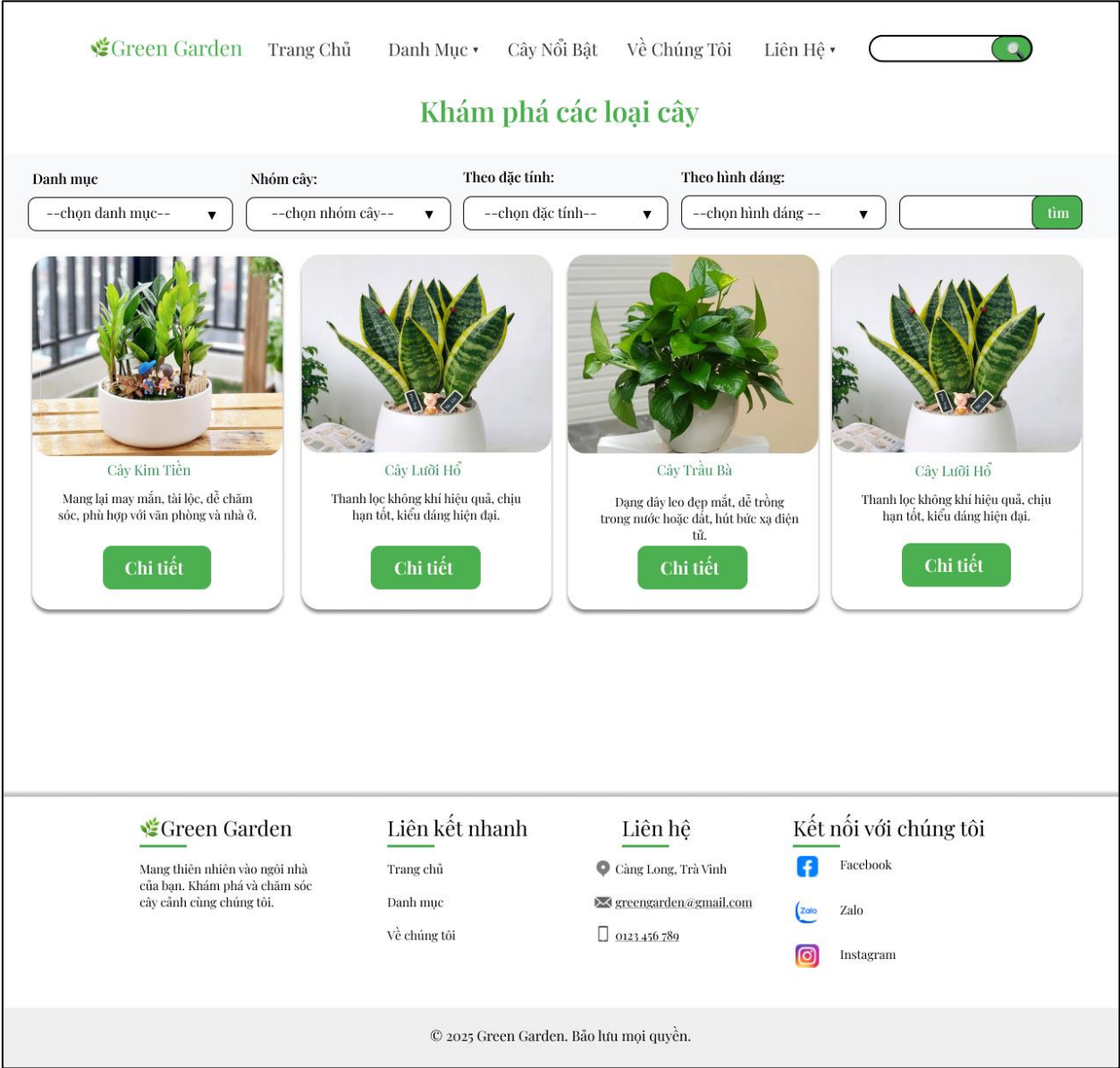
Trang Quản trị dành cho **quản trị viên (server admin)**.

❖ Giao diện Trang Chủ hệ thống của người dùng khi vừa vào web



Hình 4 Giao diện trang chủ hệ thống

❖ Giao diện trang Danh mục hệ thống khi người dùng ấn vào:



Hình 5 Giao diện trang danh mục hệ thống

❖ Giao diện trang Chi tiết khi người dùng xem thông tin của cây

Green Garden


Trang Chủ

Danh Mục ▾

Cây Nổi Bật

Về Chúng Tôi

Liên Hệ ▾



## Cây Kim Tiền

Cây Kim Tiền là loại cây cảnh phong thủy được ưa chuộng với những tán lá xanh bóng, mọc đối xứng và dày dặn, biểu tượng cho sự giàu sang và thịnh vượng.

### Đặc điểm nổi bật:

- Lá xanh bóng, mọc đều hai bên thân cây
- Thân cây mọng nước, có khả năng trữ nước tốt
- Sống tốt trong môi trường ánh sáng yếu hoặc phòng điều hòa
- Là loại cây phong thủy hút tài lộc, may mắn
- Dễ trồng và chăm sóc, phù hợp cho văn phòng và nhà ở

### Hướng dẫn chăm sóc:

- Ánh sáng: Ưa sáng nhẹ, tránh ánh nắng gắt trực tiếp, có thể đặt trong nhà hoặc phòng làm việc
- Nước: Tưới 1-2 lần mỗi tuần, tránh để đất quá ẩm dễ gây úng
- Đất: Đất tơi xốp, thoát nước tốt, có thể trộn với xơ dừa hoặc sỏi nhẹ
- Nhiệt độ: Thích hợp với nhiệt độ từ 18-30°C, tránh để cây nơi quá lạnh hoặc gió mạnh

Green Garden

Mang thiên nhiên vào ngôi nhà của bạn. Khám phá và chăm sóc cây cảnh cùng chúng tôi.

Liên kết nhanh

Trang chủ

Danh mục

Về chúng tôi

Liên hệ

Cảng Long, Trà Vinh

greengarden@gmail.com

0123.456.789

Kết nối với chúng tôi

Facebook

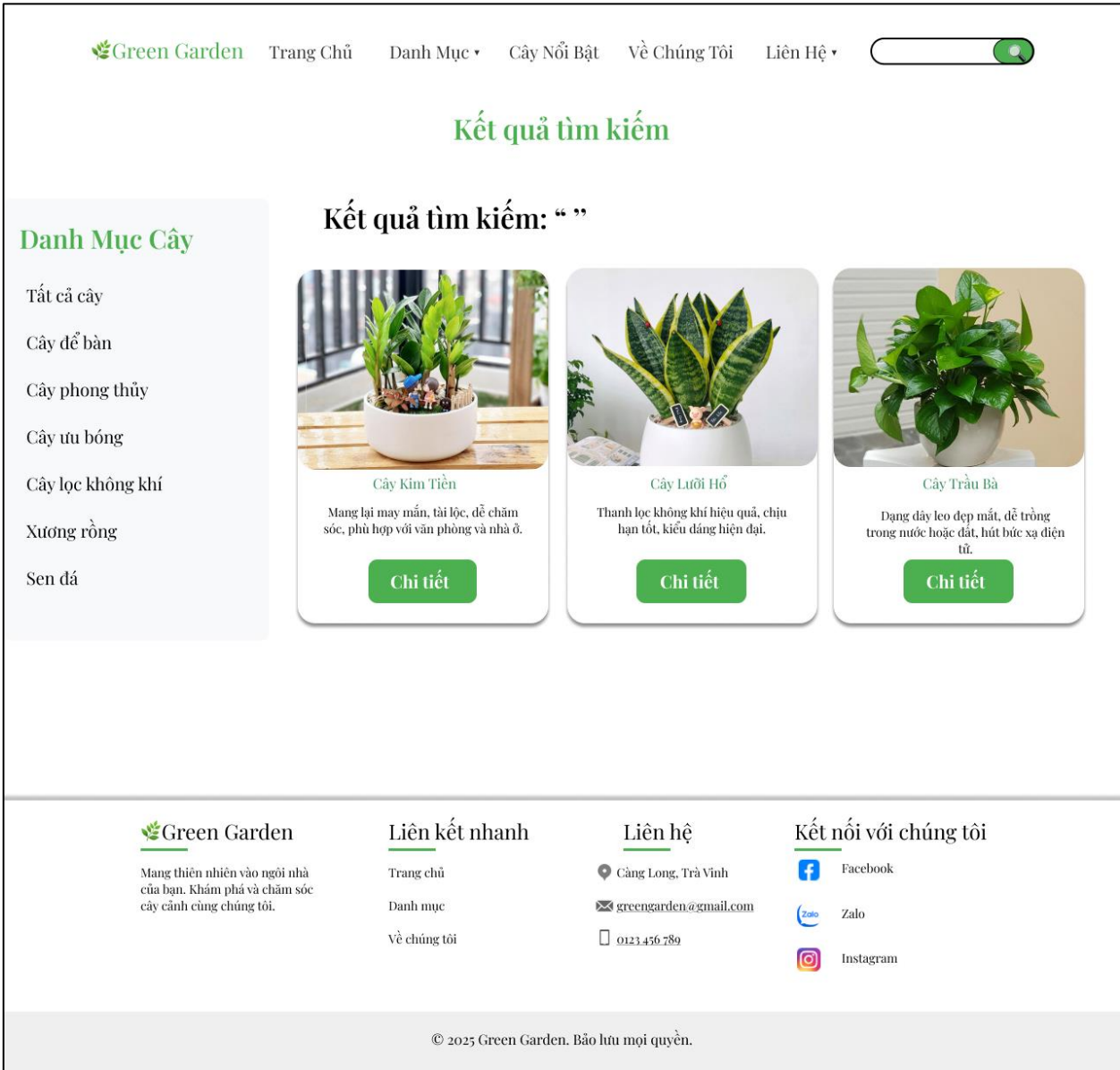
Zalo

Instagram

© 2025 Green Garden. Bảo lưu mọi quyền.

Hình 6 Giao diện trang chi tiết cây trồng

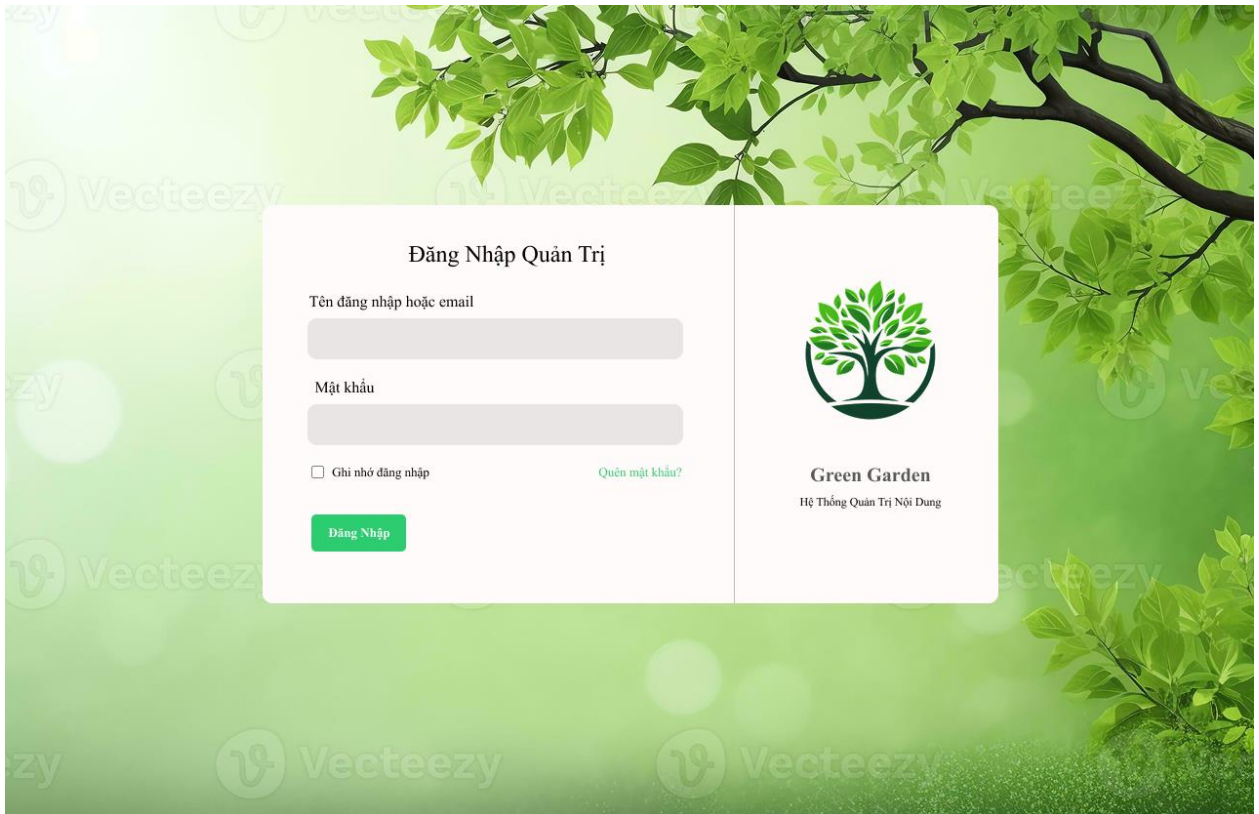
❖ Giao diện trang Kết quả tìm kiếm khi người dùng tìm kiếm trên hệ thống



Hình 7 Giao diện trang kết quả tìm kiếm

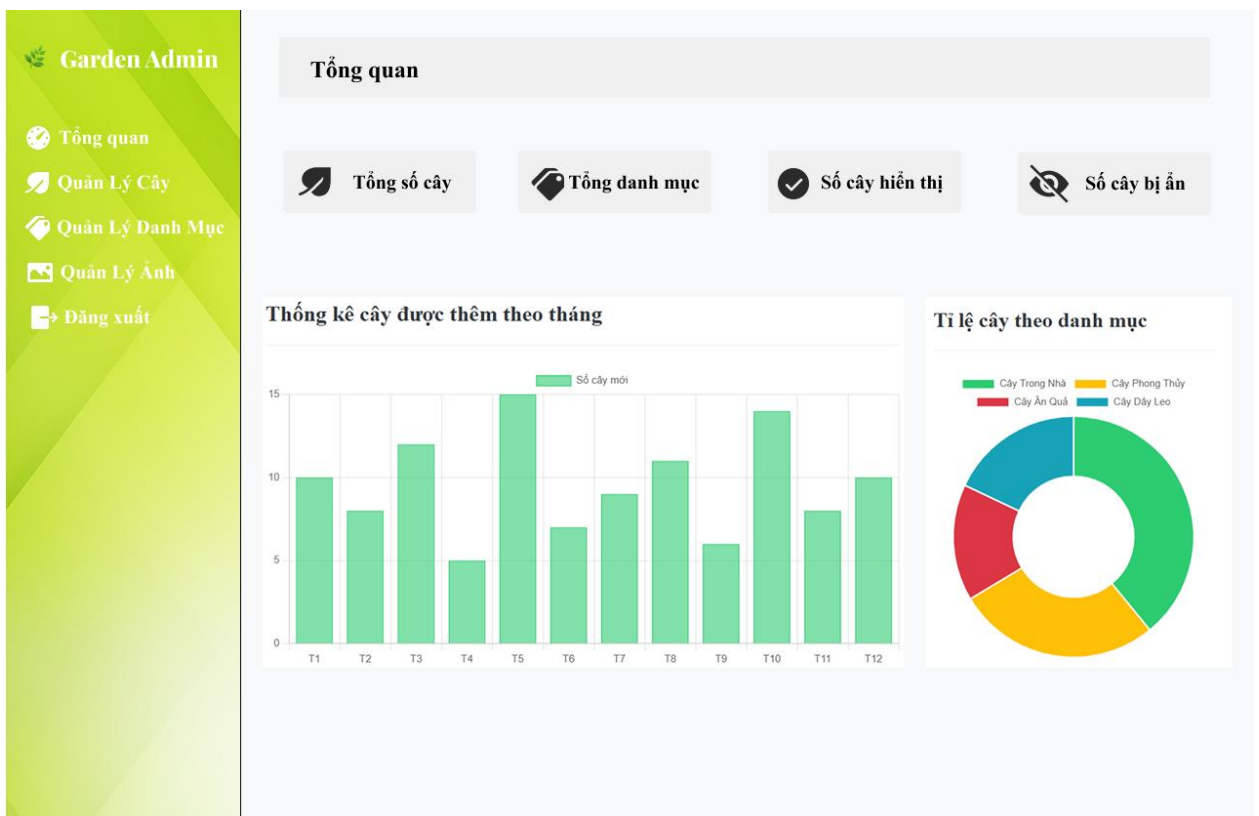


❖ Giao diện trang Đăng nhập quản trị khi Quản trị viên vào



Hình 8 Giao diện trang đăng nhập dành cho Quản trị viên

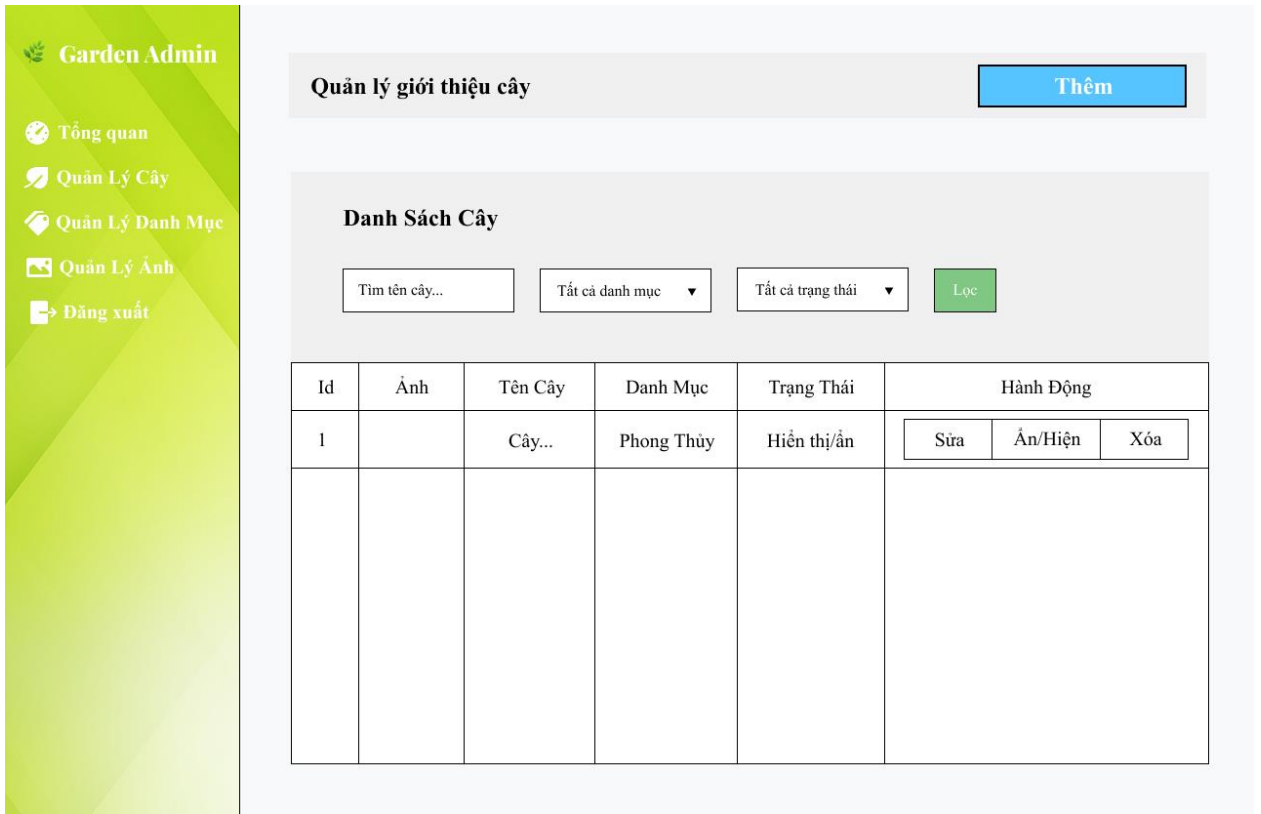
❖ Giao diện trang Tổng quan khi vừa vào trang quản trị



Hình 9 Giao diện trang Tổng quan của Quản trị viên

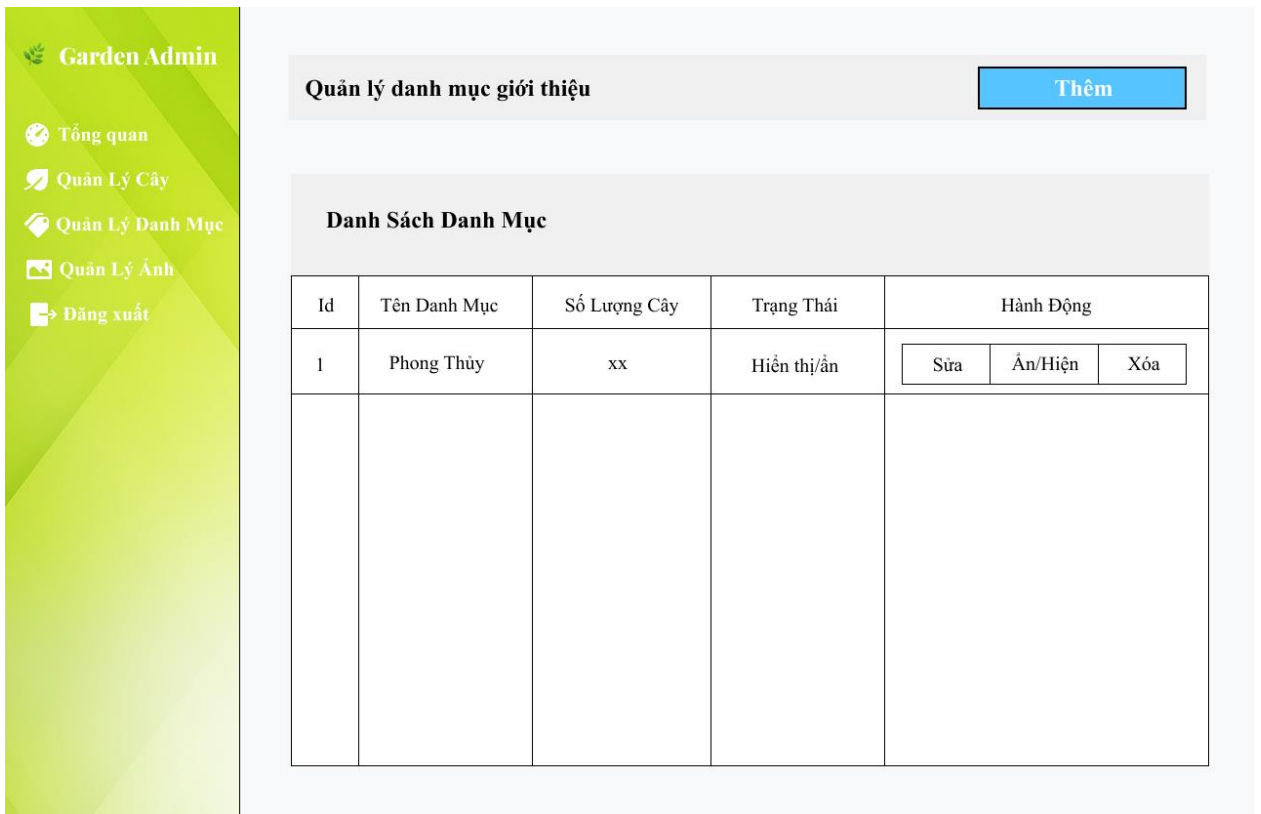


❖ Giao diện trang Quản lý cây trồng của Quản trị viên




Hình 10 Giao diện trang Quản lý cây trồng của Quản trị viên


❖ Giao diện trang Quản lý danh mục của Quản trị viên





Hình 11 Giao diện trang Quản lý danh mục của Quản trị viên


Giao diện trang Quản lý ảnh của Quản trị viên


 Garden Admin

 Tổng quan

 Quản Lý Cây

 Quản Lý Danh Mục

 Quản Lý Ảnh

 Đăng xuất

Quản lý danh mục giới thiệu

Thêm

Danh Sách Danh Mục

Id	Tên Danh Mục	Số Lượng Ảnh	Hành Động
1	Phong Thủy	xx	<div>SửaẤn/HiệnXóa</div>

Danh Sách Ảnh

Tìm tên file ảnh...

tất cả danh mục ảnh▼

Lọc ảnh

Ảnh 1

Ảnh 2

Hình 12 Giao diện trang Quản lý ảnh của Quản trị viên

26

## CHƯƠNG 4: QUẢN LÝ DỰ ÁN

### 4.1. Jira để quản lý kế hoạch

Tên Project trên Jira: Giới Thiệu Cây Trồng

Sử dụng SCRUM để lập ra kế hoạch cho từng thành viên

Có phân công cụ thể cho từng thành viên gồm các task và user story

### 4.2. Phân công từng thành viên trong nhóm

#### 4.2.1. Đối với Frontend

Thành viên được phân công: *La Thuần Khang*.

Công nghệ sử dụng: React.js.

Backlog của Sprint 1:

<input type="checkbox"/> GTCT-10	Là người dùng, tôi muốn có trang chủ đẹp và dễ nhìn	DONE	-	TL
<input checked="" type="checkbox"/> GTCT-21	Thiết kế trang chủ bằng Figma	DONE	-	TL
<input checked="" type="checkbox"/> GTCT-22	Viết giao diện cho trang chủ	DONE	-	TL
<input type="checkbox"/> GTCT-11	Là người dùng, tôi muốn phân loại sự kiện theo danh mục	DONE	-	TL
<input checked="" type="checkbox"/> GTCT-23	Thiết kế trang danh mục bằng Figma	DONE	-	TL
<input checked="" type="checkbox"/> GTCT-24	Viết giao diện cho trang danh mục	DONE	-	TL
<input type="checkbox"/> GTCT-12	Là người dùng, tôi muốn có chức năng tìm kiếm thân thiện với người dùng...	DONE	-	TL
<input checked="" type="checkbox"/> GTCT-25	Thiết kế giao diện trang tìm kiếm bằng Figma	DONE	-	TL
<input checked="" type="checkbox"/> GTCT-26	Viết giao diện cho trang tìm kiếm	DONE	-	TL
<input checked="" type="checkbox"/> GTCT-41	Viết giao diện các trang quản trị	DONE	-	TL
<input checked="" type="checkbox"/> GTCT-34	Viết báo cáo	IN PROGRESS	-	TL

Hình 13 Backlog của thành viên thứ nhất

Tổng cộng có 3 Stories:

**GTCT-10:** Là người dùng, tôi muốn có trang chủ đẹp và dễ nhìn.

**GTCT-11:** Là người dùng, tôi muốn phân loại sự kiện theo danh mục.

**GTCT-12:** Là người dùng, tôi muốn có chức năng tìm kiếm thân thiện với người dùng.

Tổng cộng có 8 Tasks:

**GTCT-21:** Thiết kế trang chủ bằng Figma.

**GTCT-22:** Viết giao diện cho trang chủ.

**GTCT-23:** Thiết kế trang danh mục bằng Figma.

**GTCT-24:** Viết giao diện cho trang danh mục.

**GTCT-25:** Thiết kế giao diện tìm kiếm bằng Figma.

**GTCT-26:** Viết giao diện cho trang tìm kiếm.

**GTCT-41:** Viết giao diện các trang quản trị.

**GTCT-34:** Viết báo cáo.

#### 4.2.2. Đối với Backend

Thành viên được phân công: *Hà Gia Lộc*.

Công nghệ sử dụng: Node.js, RESTful API.

Backlog của Sprint 1:

<input type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-8 Là lập trình viên, tôi muốn hệ thống có cấu trúc backend/frontend rõ ràng	DONE ▾	-	HL
<input type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-15 Tạo cấu trúc thư mục backend/frontend	DONE ▾	-	HL
<input type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-36 Là lập trình viên, tôi muốn trang web có một backend được chia theo module để d...	DONE ▾	-	HL
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-37 Viết backend	DONE ▾	-	HL
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-38 Tích hợp RESTful API	DONE ▾	-	HL
<input type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-40 Là người dùng, tôi muốn có trang quản trị đẹp và dễ nhìn	IN PROGRESS ▾	-	HL
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-42 Cập nhật backend cho các trang quản trị	IN PROGRESS ▾	-	HL
<input type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-13 Là lập trình viên, tôi muốn trang web được tích hợp đầy đủ các tính năng được yêu...	IN PROGRESS ▾	-	HL
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-28 Tích hợp Swagger	DONE ▾	-	HL
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-29 Tích hợp Git	IN PROGRESS ▾	-	HL
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-30 Mô tả chức năng từng phần trong kiến trúc	IN PROGRESS ▾	-	HL
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-31 Thêm GitHub Actions	IN PROGRESS ▾	-	HL
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-32 Viết README.md	IN PROGRESS ▾	-	HL
<input type="checkbox"/>	<input checked="" type="checkbox"/>	GTCT-33 Kết thúc dự án	IN PROGRESS ▾	-	HL

Hình 14 Backlog của thành viên thứ hai

Tổng cộng có 5 Stories:

**GTCT-8:** Là lập trình viên, tôi muốn hệ thống có cấu trúc backend/frontend rõ ràng.

**GTCT-36:** Là lập trình viên, tôi muốn trang web có một backend được chia theo module để dễ bảo trì và mở rộng.

**GTCT-40:** Là người dùng, tôi muốn có trang quản trị đẹp và dễ nhìn.

**GTCT-13:** Là lập trình viên, tôi muốn trang web được tích hợp đầy đủ các tính năng được yêu cầu.

**GTCT-33:** Kết thúc dự án.

Tổng cộng có 9 Tasks:

**GTCT-15:** Tạo cấu trúc thư mục backend/frontend.

**GTCT-37:** Viết backend.

**GTCT-38:** Tích hợp RESTful API.

**GTCT-42:** Cập nhật backend cho các trang quản trị.

**GTCT-28:** Tích hợp Swagger.

**GTCT-29:** Tích hợp Git.

**GTCT-30:** Mô tả chức năng từng phần trong kiến trúc.

**GTCT-31:** Thêm GitHub Actions.

**GTCT-32:** Viết README.md.

### 4.2.3. Đối với Database và Docker

Thành viên được phân công: *Đỗ Gia Hào*.

Công nghệ sử dụng: Docker Desktop và MySQL.

Backlog của Sprint 1:

<input type="checkbox"/> GTCT-18 Là lập trình viên, tôi muốn tích hợp Docker	DONE ▾	-	
<input checked="" type="checkbox"/> GTCT-16 Tạo file docker-compose.yml, dockerfile cho frontend, dockerfile cho backend	DONE ▾	-	
<input type="checkbox"/> GTCT-9 Là lập trình viên, tôi muốn có database với thiết kế người dùng	DONE ▾	-	
<input checked="" type="checkbox"/> GTCT-17 Tạo database	DONE ▾	-	
<input checked="" type="checkbox"/> GTCT-20 Thiết kế bảng trong MySQL	DONE ▾	-	
<input checked="" type="checkbox"/> GTCT-43 Cập nhật cơ sở dữ liệu cho các trang quản trị	IN PROGRESS ▾	-	
<input checked="" type="checkbox"/> GTCT-35 Thiết kế slide thuyết trình	IN PROGRESS ▾	-	

*Hình 15 Backlog của thành viên thứ ba*

Tổng cộng có 2 Stories:

**GTCT-18:** Là lập trình viên, tôi muốn tích hợp Docker.

**GTCT-9:** Là lập trình viên, tôi muốn có database với thiết kế người dùng.

Tổng cộng có 5 Tasks :

**GTCT-16:** Tạo file docker-compose.yml, dockerfile cho frontend, dockerfile cho backend.

**GTCT-17:** Tạo database.

**GTCT-20:** Thiết kế bảng trong MySQL.

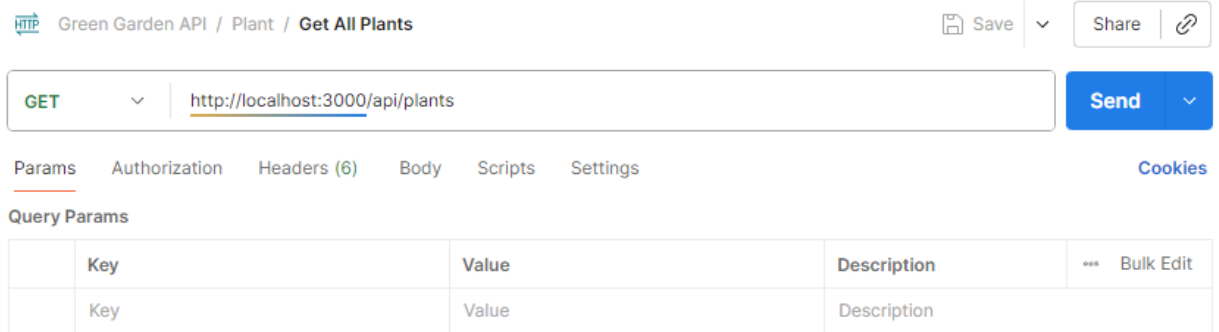
**GTCT-43:** Cập nhật cơ sở dữ liệu cho các trang quản trị.

**GTCT-35:** Thiết kế slide thuyết trình.

# CHƯƠNG 5: TRIỂN KHAI VÀ KIỂM THỬ

## 5.1. Postman

### 5.1.1. Gửi yêu cầu GET để lấy danh sách cây trồng

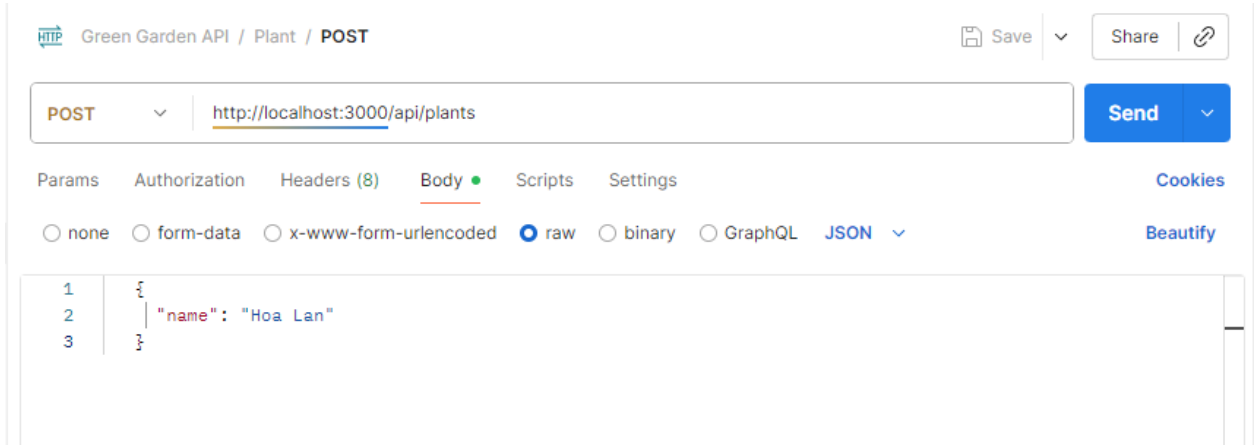


Hình 16 Gửi yêu cầu GET để lấy danh sách cây

```
1  [
2    {
3      "id": 1,
4      "name": "Cây Test",
5      "description": "Cây phong thủy mang lại tài lộc, may mắn và thịnh vượng, được ưa chuộng trong văn phòng và gia đình.",
6      "image": "/plantsImage/Kt.jpg",
7      "category": "Cây Phong Thủy",
8      "status": "Ẩn",
9      "featured": false,
10     "features": [
11       "Cực kỳ dễ chăm sóc, chịu hạn tốt",
12       "Lọc không khí, loại bỏ độc tố",
13       "Ý nghĩa phong thủy tốt lành"
14     ],
15     "care": {
16       "soil": "Đất tơi xốp, thoát nước cực tốt (có thể trộn thêm đá perlite).",
17       "temp": "22-35°C",
18       "light": "Ánh sáng gián tiếp hoặc trong bóng râm. Tránh nắng gắt trực tiếp.",
19       "water": "Tưới nước khi đất đã khô hoàn toàn, khoảng 1-2 tuần/lần."
20     },
21     "createdAt": "2025-07-22T17:39:28.000Z",
22     "updatedAt": "2025-07-23T03:41:39.000Z"
23   },
24   {
25     "id": 2,
26     "name": "Cây Kim Tiền",
27     "description": "Cây phong thủy mang lại tài lộc, may mắn và thịnh vượng, được ưa chuộng trong văn phòng và gia đình.",
28     "image": "/plantsImage/Kt.jpg",
29     "category": "Cây Phong Thủy",
30     "status": "Hiển thị",
31     "featured": true,
32     "features": [
33       "Cực kỳ dễ chăm sóc, chịu hạn tốt",
34       "Lọc không khí, loại bỏ độc tố",
35       "Ý nghĩa phong thủy tốt lành"
36     ],
37     "care": {
38       "soil": "Đất tơi xốp, thoát nước cực tốt (có thể trộn thêm đá perlite).",
39       "temp": "22-35°C",
40       "light": "Ánh sáng gián tiếp hoặc trong bóng râm. Tránh nắng gắt trực tiếp.",
41       "water": "Tưới nước khi đất đã khô hoàn toàn, khoảng 1-2 tuần/lần."
42     },
43     "createdAt": "2025-07-22T17:40:25.000Z",
44     "updatedAt": "2025-07-23T03:14:46.000Z"
45   },
46 ]
```

Hình 17 Kết quả kiểm thử API GET để lấy danh sách cây trồng

### 5.1.2. Gửi yêu cầu POST để thêm cây trồng mới

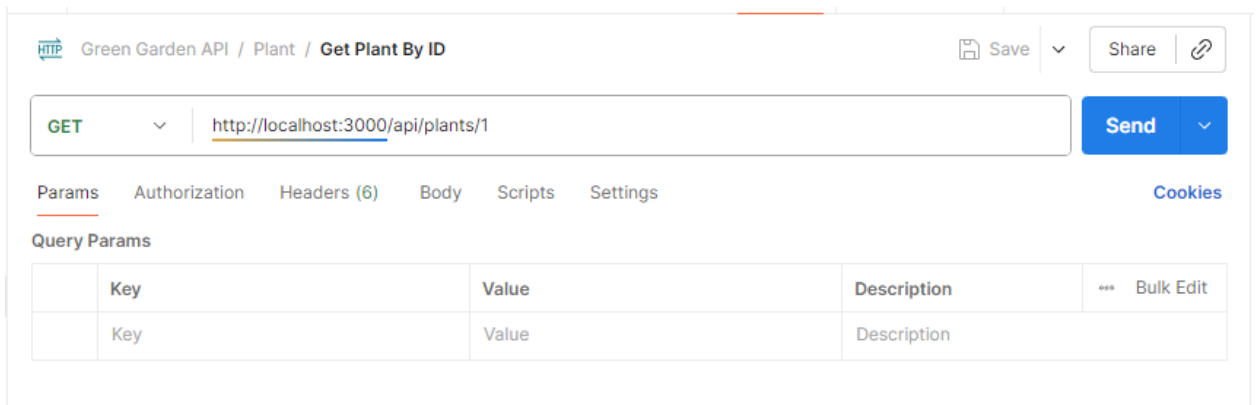


Hình 18 Gửi yêu cầu POST để thêm cây mới

```
1  {
2      "status": "Hiển thị",
3      "featured": false,
4      "features": [],
5      "care": {},
6      "id": 6,
7      "name": "Hoa Lan",
8      "updatedAt": "2025-07-23T04:09:48.252Z",
9      "createdAt": "2025-07-23T04:09:48.252Z"
10 }
```

Hình 19 Kết quả kiểm thử API POST

### 5.1.3. Gửi yêu cầu GET để lấy chi tiết cây trồng



Hình 20 Gửi yêu cầu GET để lấy chi tiết cây

```

1  {
2      "id": 1,
3      "name": "Cây Test",
4      "description": "Cây phong thủy mang lại tài lộc, may mắn và thịnh vượng, được ưa chuộng trong văn phòng và gia đình.",
5      "image": "/plantsImage/Kt.jpg",
6      "category": "Cây Phong Thủy",
7      "status": "Ẩn",
8      "featured": false,
9      "features": [
10         "Cực kỳ dễ chăm sóc, chịu hạn tốt",
11         "Lọc không khí, loại bỏ độc tố",
12         "Ý nghĩa phong thủy tốt lành"
13     ],
14     "care": {
15         "soil": "Đất tơi xốp, thoát nước cực tốt (có thể trộn thêm đá perlite).",
16         "temp": "22-35°C",
17         "light": "Ánh sáng gián tiếp hoặc trong bóng râm. Tránh nắng gắt trực tiếp.",
18         "water": "Tưới nước khi đất đã khô hoàn toàn, khoảng 1-2 tuần/lần."
19     },
20     "createdAt": "2025-07-22T17:39:28.000Z",
21     "updatedAt": "2025-07-23T03:41:39.000Z"
22 }

```

Hình 21 Kết quả kiểm thử API GET

#### 5.1.4. Gửi yêu cầu PUT để cập nhật thông tin cây

The screenshot shows a REST client interface for a PUT request. The URL is `http://localhost:3000/api/plants/1`. The request body is a JSON object with the following fields: `"name": "Hoa Hồng"`, `"status": "Ẩn"`, and `"id": 1` (implied by the first line of the JSON). The interface includes tabs for Params, Authorization, Headers (8), Body (selected), Scripts, and Settings. The Body tab shows the JSON payload. There are also buttons for Save, Share, Send, and Beautify.

```

1  {
2      "name": "Hoa Hồng",
3      "status": "Ẩn"
4  }

```

Hình 22 Gửi yêu cầu PUT để cập nhật cây

```

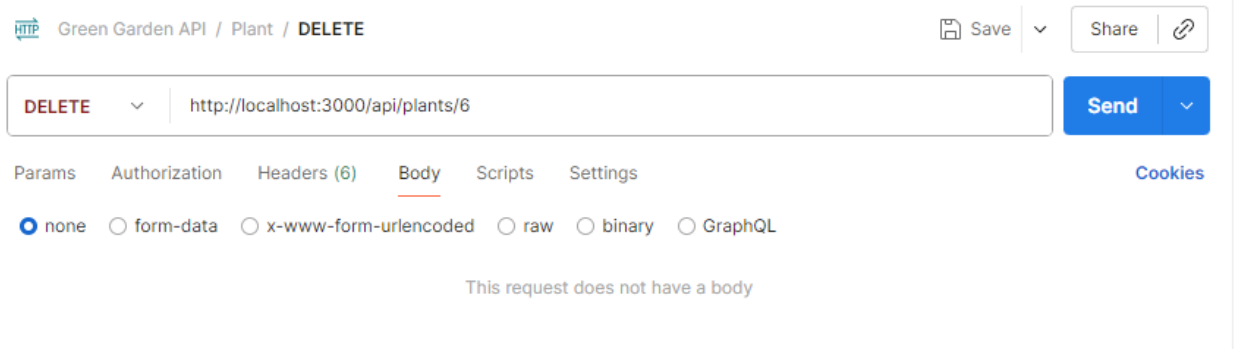
1  {
2      "id": 1,
3      "name": "Hoa Hồng",
4      "description": "Cây phong thủy mang lại tài lộc, may mắn và thịnh vượng, được ưa chuộng trong văn phòng và gia đình.",
5      "image": "/plantsImage/Kt.jpg",
6      "category": "Cây Phong Thủy",
7      "status": "Ẩn",
8      "featured": false,
9      "features": [
10         "Cực kỳ dễ chăm sóc, chịu hạn tốt",
11         "Lọc không khí, loại bỏ độc tố",
12         "Ý nghĩa phong thủy tốt lành"
13     ],
14     "care": {
15         "soil": "Đất tơi xốp, thoát nước cực tốt (có thể trộn thêm đá perlite).",
16         "temp": "22-35°C",
17         "light": "Ánh sáng gián tiếp hoặc trong bóng râm. Tránh nắng gắt trực tiếp.",
18         "water": "Tưới nước khi đất đã khô hoàn toàn, khoảng 1-2 tuần/lần."
19     },
20     "createdAt": "2025-07-22T17:39:28.000Z",
21     "updatedAt": "2025-07-23T04:24:10.978Z"
22 }

```

Hình 23 Kết quả kiểm thử API PUT

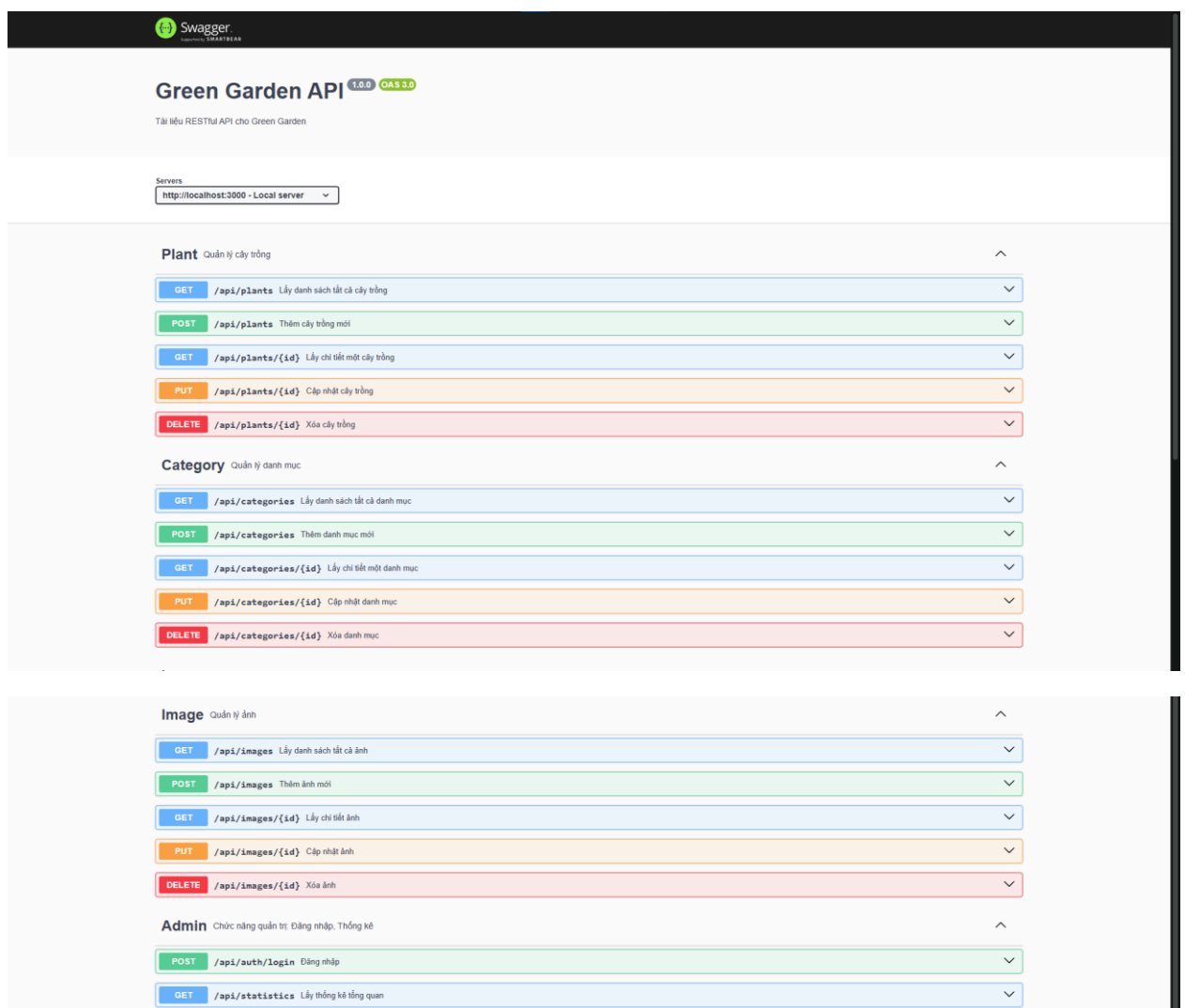


### 5.1.5. Gửi yêu cầu DELETE để xóa cây khỏi hệ thống



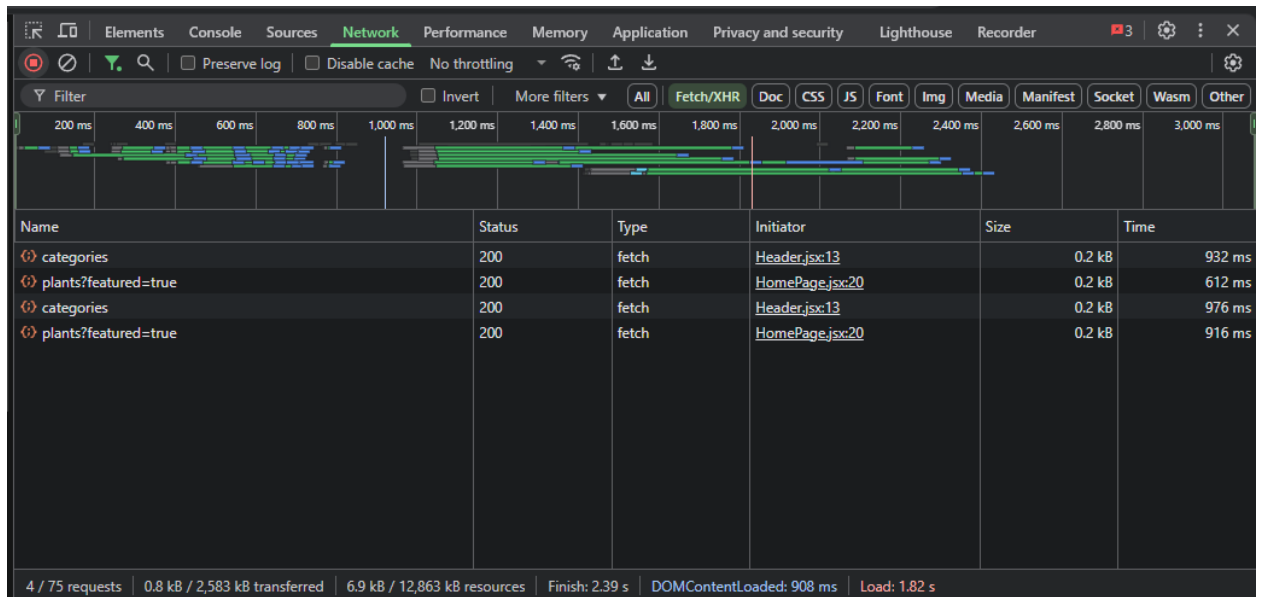
Hình 24 Gửi yêu cầu DELETE để xóa cây

## 5.2. Swagger



Hình 25 Giao diện Swagger tổng quan

## 5.3. Kiểm thử API



Hình 26 Kiểm thử API thực tế trên giao diện người dùng

# CHƯƠNG 6: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 6.1. Kết luận

Dự án "Xây dựng Website Giới thiệu Cây trồng" đã hoàn thành tốt các mục tiêu ban đầu đề ra. Qua quá trình thực hiện, nhóm đã đạt được những kết quả quan trọng sau:

**Về sản phẩm:** Đã xây dựng thành công một ứng dụng web hoàn chỉnh, cho phép người dùng tra cứu, tìm kiếm và xem thông tin chi tiết về các loại cây trồng một cách trực quan và hiệu quả. Sản phẩm đáp ứng đầy đủ các yêu cầu chức năng và các yêu cầu phi chức năng cốt lõi như thiết kế responsive.

**Về công nghệ:** Đã áp dụng thành công ngăn xếp công nghệ hiện đại gồm React.js cho Frontend, Node.js và Express.js cho Backend, và MySQL cho cơ sở dữ liệu. Việc này chứng tỏ khả năng làm chủ công nghệ full-stack của nhóm.

**Về kiến trúc:** Đã thiết kế và triển khai thành công kiến trúc Client-Server với các RESTful API được xây dựng một cách có cấu trúc. Việc sử dụng Swagger để tài liệu hóa API đã đảm bảo sự giao tiếp rõ ràng, minh bạch giữa Frontend và Backend.

**Về quy trình:** Đã thiết lập và vận hành hiệu quả một quy trình làm việc chuyên nghiệp, bao gồm việc sử dụng Docker để container hóa ứng dụng, đảm bảo tính nhất quán của môi trường, và Postman để kiểm thử chất lượng API.

Nhìn chung, dự án không chỉ là một sản phẩm phần mềm có thể hoạt động, mà còn là minh chứng cho khả năng học hỏi và áp dụng các công nghệ, kiến trúc và quy trình phát triển phần mềm tiên tiến vào một bài toán thực tế.

## 6.2. Hạn chế

Bên cạnh những kết quả đạt được, trong khuôn khổ thời gian và nguồn lực có hạn, dự án vẫn còn một số hạn chế nhất định cần được nhìn nhận một cách khách quan:

**Tính năng còn ở mức cơ bản:** Hệ thống hiện tại chưa có các tính năng nâng cao cho người dùng cuối như tạo tài khoản, lưu danh sách cây yêu thích, viết bình luận hay đánh giá. Điều này làm giảm tính tương tác và cá nhân hóa của sản phẩm.

**Phạm vi kiểm thử chưa toàn diện:** Mặc dù đã sử dụng Postman để kiểm thử các endpoint của API, dự án vẫn chưa có hệ thống kiểm thử tự động cho giao diện người dùng (UI Automation Testing) và các bài kiểm thử tích hợp (Integration Test) phức tạp. Việc kiểm thử chủ yếu vẫn dựa trên phương pháp thủ công.

**Chưa tối ưu hóa về hiệu năng ở quy mô lớn:** Hệ thống hoạt động tốt với lượng dữ liệu và lượng truy cập hiện tại. Tuy nhiên, các vấn đề về tối ưu hóa truy vấn cơ sở dữ liệu (query optimization), cơ chế caching, và nén hình ảnh ở quy mô lớn vẫn chưa được đặt ra và giải quyết triệt để.

**Giao diện quản trị còn đơn giản:** Trang quản trị mới chỉ đáp ứng các chức năng Thêm, Sửa, Xóa cơ bản, chưa có các công cụ thống kê, báo cáo trực quan để hỗ trợ người quản trị ra quyết định.

### 6.3. Hướng phát triển

Để tiếp tục hoàn thiện và nâng cao giá trị của sản phẩm, nhóm đề xuất các hướng phát triển và cải tiến trong tương lai như sau:

Xây dựng hệ thống người dùng và cá nhân hóa:

Phát triển chức năng đăng ký, đăng nhập, quản lý thông tin cá nhân.

Cho phép người dùng tạo "khu vườn ảo" của riêng mình, lưu lại các cây yêu thích, và viết ghi chú chăm sóc cá nhân.

Nâng cấp chức năng tìm kiếm và bộ lọc:

Xây dựng một bộ lọc nâng cao, cho phép người dùng lọc cây theo nhiều tiêu chí: nhu cầu ánh sáng (ưa nắng, ưa râm), kích thước (để bàn, cây lớn), đặc tính (lọc không khí, có hoa)...

Phát triển tính năng cộng đồng:

Thêm chức năng bình luận, đánh giá (rating) cho mỗi loại cây để người dùng có thể chia sẻ kinh nghiệm chăm sóc thực tế.

Nghiên cứu và tích hợp một mô hình AI cho phép người dùng tải lên hình ảnh một chiếc lá hoặc cây, và hệ thống sẽ cố gắng nhận diện đó là cây gì, cung cấp thông tin tương ứng.

Hoàn thiện quy trình CI/CD:

Tích hợp GitHub Actions để tự động hóa hoàn toàn quy trình từ lúc đẩy code, chạy test, build Docker image cho đến triển khai lên server, giúp giảm thiểu sai sót và tăng tốc độ ra mắt phiên bản mới.

## TÀI LIỆU THAM KHẢO

- [1] “Quick Start - React,” [Trực tuyến]. Available: <https://react.dev/learn>. [Đã truy cập 20 06 2025].
- [2] “Node.js Document,” [Trực tuyến]. Available: <https://nodejs.org/docs/latest/api/>. [Đã truy cập 20 06 2025].
- [3] “Express.js,” [Trực tuyến]. Available: <https://expressjs.com/>. [Đã truy cập 20 06 2025].
- [4] “Connect Node.js with MySQL (Using Sequelize ORM),” [Trực tuyến]. Available: <https://sequelize.org/docs/v6/getting-started/>. [Đã truy cập 20 06 2025].
- [5] “Introduction to RESTful API,” [Trực tuyến]. Available: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Status>. [Đã truy cập 15 06 2025].
- [6] “Swagger Official Docs:,” [Trực tuyến]. Available: [https://swagger.io/docs/specification/v3\\_0/about/](https://swagger.io/docs/specification/v3_0/about/). [Đã truy cập 22 06 2025].
- [7] “Postman Learning Center,” [Trực tuyến]. Available: <https://learning.postman.com/>. [Đã truy cập 22 06 2025].
- [8] “GitHub Actions Documentation,” [Trực tuyến]. Available: <https://docs.github.com/en/actions>. [Đã truy cập 22 06 2025].
- [9] “Implement CI/CD pipeline with GitLab,” [Trực tuyến]. Available: <https://topdev.vn/blog/trien-khai-ci-cd-voi-gitlab/>. [Đã truy cập 22 06 2025].
- [10] “Github,” [Trực tuyến]. Available: <https://github.com/HaGiaLoc/UngDungGioiThieuCayTrong>. [Đã truy cập 02 06 2025].