

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



ĐỒ ÁN MÔN HỌC
CÔNG NGHỆ PHẦN MỀM

TÊN ĐỀ TÀI
XÂY DỰNG ỨNG DỤNG CHIA SẺ TÀI LIỆU
TRỰC TUYẾN (DOCSHARE)

Sinh viên thực hiện:

110122066	Trương Hoàng Giang	DA22TTD
110122028	Liều Kiện An	DA22TTD
110122064	Trương Mỹ Duyên	DA22TTD

Giảng viên hướng dẫn: TS. Nguyễn Bảo Ân

Vĩnh Long, tháng 7 năm 2025

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



ĐỒ ÁN MÔN HỌC
CÔNG NGHỆ PHẦN MỀM

TÊN ĐỀ TÀI
XÂY DỰNG ỨNG DỤNG CHIA SẺ TÀI LIỆU
TRỰC TUYẾN (DOCSHARE)

Sinh viên thực hiện:

110122066	Trương Hoàng Giang	DA22TTD
110122028	Liều Kiện An	DA22TTD
110122064	Trương Mỹ Duyên	DA22TTD

Giảng viên hướng dẫn: TS. Nguyễn Bảo Ân

Vĩnh Long, tháng 7 năm 2025

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Vĩnh Long, ngày tháng năm

Giáo viên hướng dẫn

(Ký tên và ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN HỘI ĐỒNG

Vĩnh Long, ngày tháng năm

Giáo viên hội đồng

(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Trước tiên, nhóm chúng em xin bày tỏ lòng biết ơn chân thành và sự tri ân sâu sắc đến Thầy Nguyễn Bảo Ân, người đã tận tình giảng dạy và truyền đạt những kiến thức quý báu trong suốt quá trình học môn *Công nghệ phần mềm*. Những kiến thức thầy cung cấp không chỉ là nền tảng vững chắc giúp chúng em thực hiện tốt bài báo cáo này, mà còn là hành trang giá trị để áp dụng vào thực tiễn công việc sau này.

Chúng em cũng xin gửi lời cảm ơn đặc biệt đến Thầy/Cô trong hội đồng chấm báo cáo, những người đã dành thời gian quý báu để lắng nghe, đánh giá và đóng góp ý kiến cho bài làm của nhóm. Sự chỉ dẫn, góp ý của quý Thầy/Cô là động lực để chúng em hoàn thiện hơn về cả kiến thức chuyên môn lẫn kỹ năng trình bày, nghiên cứu.

Mặc dù đã cố gắng hết mình, nhưng do còn hạn chế về kinh nghiệm thực tế và kiến thức chuyên môn, nhóm không tránh khỏi những thiếu sót trong quá trình thực hiện và trình bày báo cáo. Chúng em rất mong nhận được những nhận xét và góp ý quý báu từ quý Thầy/Cô để bài làm của nhóm được hoàn thiện hơn trong tương lai.

Cuối cùng, nhóm xin kính chúc Thầy Nguyễn Bảo Ân cùng toàn thể quý Thầy/Cô luôn dồi dào sức khỏe, hạnh phúc và tiếp tục giữ vững nhiệt huyết trong sự nghiệp trồng người cao cả.

Nhóm chúng em xin chân thành cảm ơn!

MỤC LỤC

CHƯƠNG 1 GIỚI THIỆU.....	1
1.1 Mô tả bài toán.....	1
1.2 Mô tả ứng dụng.....	1
1.3 Đặc tả các chức năng hệ thống.....	2
1.3.1 Chức năng đăng nhập.....	2
1.3.2 Chức năng tìm kiếm tài liệu.....	2
1.3.3 Chức năng xem tài liệu.....	3
1.3.4 Chức năng tải xuống tài liệu.....	3
1.4 Thiết kế dữ liệu.....	3
1.4.1 Lược đồ cơ sở dữ liệu.....	3
1.4.2 Mô hình dữ liệu.....	4
CHƯƠNG 2 CƠ SỞ LÝ THUYẾT.....	5
2.1 Tổng quan về Agile.....	5
2.1.1 Giới thiệu về Agile.....	5
2.1.2 Tuyên ngôn Agile.....	5
2.1.2.1 Giá trị của Tuyên ngôn Agile.....	5
2.1.2.2 Nguyên tắc của Tuyên ngôn Agile.....	5
2.2 Tổng quan về SCRUM.....	6
2.2.1 SCRUM là gì ?.....	6
2.2.2 Phương pháp SCRUM.....	6
2.2.2.1 Thuật ngữ SCRUM.....	7
2.2.2.2 Các vai trò quan trọng trong Scrum.....	7

2.2.3 Scrum và Sprints.....	7
2.2.4 Các thực hành chính của Scrum.....	8
2.2.5 Trạng thái của Product Backlog Item.....	8
2.3 Công nghệ sử dụng.....	9
2.3.1 RESTful API.....	9
2.3.1.1 Tổng quan về RESTful API.....	9
2.3.1.2 RESTful hoạt động như thế nào?.....	9
2.3.2 Next.js & React.js.....	10
2.3.3 Node.js & framework Express.js.....	10
2.3.3.1 Node.js.....	10
2.3.3.2 Express.js.....	11
2.3.4 MongoDB.....	12
CHƯƠNG 3 XÁC ĐỊNH NHU CẦU.....	13
3.1 Product Backlog.....	13
3.2 Sprint backlog.....	13
3.3 Technical Backlog.....	14
3.4 Support Backlog.....	15
CHƯƠNG 4 LẬP KẾ HOẠCH SCRUM.....	17
4.1 Sprint 1:.....	17
4.2 Sprint 2:.....	18
4.3 Sprint 3:.....	18
4.4 Sprint 4:.....	19
CHƯƠNG 5 THIẾT KẾ HỆ THỐNG.....	20

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

5.1. Sprint 1 - Giao diện cơ bản và xác thực người dùng.....	20
5.1.1. Giao diện ứng dụng vẽ bằng figma.....	20
5.1.2. Mã nguồn.....	22
5.2. Sprint 2 - Chia sẻ, phân loại.....	26
5.2.1. Giao diện ứng dụng vẽ bằng figma.....	26
5.2.2. Mã nguồn.....	27
5.3. Sprint 3 - Thống kê và phân quyền tài liệu.....	29
5.3.1. Giao diện ứng dụng vẽ bằng figma.....	29
5.3.2. Mã nguồn.....	30
5.4. Sprint 4 - Testing, Fix bugs và Triển khai.....	34
CHƯƠNG 6 KẾT QUẢ THỰC NGHIỆM.....	36
6.1. Kết quả đạt được.....	36
6.2. Kết quả thực nghiệm.....	36
6.2.1. Chức năng đăng nhập.....	36
6.2.2. Chức năng đăng ký.....	37
6.2.3. Chức năng tải tài liệu.....	37
6.2.4. Chức năng xem tài liệu.....	39
6.2.5. Chức năng tải xuống tài liệu.....	40
6.2.6. Chức năng tìm kiếm tài liệu.....	40
TÀI LIỆU THAM KHẢO.....	41

DANH MỤC HÌNH ẢNH

Hình 1: Mô hình dữ liệu.....	4
Hình 2: Giao diện đăng nhập.....	20
Hình 3: Giao diện đăng ký.....	20
Hình 4: Giao diện trang chủ.....	21
Hình 5: Giao diện upload tài liệu.....	21
Hình 6: Giao diện danh sách tài liệu người dùng.....	22
Hình 7: Giao diện chia sẻ tài liệu.....	26
Hình 8: Giao diện gắn thư mục cho tài liệu.....	26
Hình 9: Giao diện chia sẻ tài liệu.....	29
Hình 10: Giao diện trang chủ.....	36
Hình 11: Giao diện trang đăng nhập.....	37
Hình 12: Giao diện trang đăng ký.....	37
Hình 13: Giao diện tải tài liệu.....	38
Hình 14: Giao diện tải tài liệu.....	38
Hình 15: Giao diện tải tài liệu thành công.....	39
Hình 16: Giao diện xem tài liệu.....	39
Hình 17: Giao diện xem tài liệu.....	40
Hình 18: Giao diện chức năng tìm kiếm.....	40

CHƯƠNG 1 GIỚI THIỆU

1.1 Mô tả bài toán

Trong môi trường làm việc và học tập hiện đại, việc phân phối và truy cập các tài liệu dùng chung như quy định, biểu mẫu, và tài liệu hướng dẫn là một nhu cầu cơ bản. Tuy nhiên, trên thực tế, quy trình này thường thiếu tổ chức. Tài liệu bị phân tán qua nhiều kênh như email, nhóm chat, khiến việc tìm kiếm phiên bản mới nhất trở nên khó khăn và tốn thời gian. Người dùng mới gặp khó khăn trong việc tiếp cận nguồn tài nguyên chính thống, trong khi người quản lý liên tục bị làm phiền bởi các yêu cầu gửi lại tài liệu.

Sự thiếu vắng một "nguồn sự thật duy nhất" (single source of truth) không chỉ làm giảm hiệu suất làm việc mà còn ảnh hưởng đến tính chuyên nghiệp của tổ chức. Vì vậy, bài toán đặt ra là xây dựng một cổng thông tin tài liệu nội bộ (Internal Document Portal). Nền tảng này sẽ đóng vai trò là điểm đến duy nhất, đáng tin cậy, nơi các tài liệu được đăng tải và quản lý tập trung, giúp mọi thành viên có thể dễ dàng truy cập và xem các tài nguyên đã được phê duyệt sau khi xác thực danh tính.

1.2 Mô tả ứng dụng

DocShare là một Cổng thông tin tài liệu nội bộ (Internal Document Portal), được xây dựng như một giải pháp tập trung và an toàn nhằm giải quyết tình trạng tài liệu phân mảnh trong môi trường làm việc.

Về mặt kiến trúc, ứng dụng được phát triển trên nền tảng công nghệ hiện đại, với sự tách biệt rõ ràng giữa hệ thống phụ trợ (Backend) và giao diện người dùng (Frontend). Cách tiếp cận này không chỉ đảm bảo hiệu suất hoạt động ổn định, bảo mật cao mà còn mang lại khả năng bảo trì và mở rộng dễ dàng. Đặc biệt, dự án còn cho thấy sự chú trọng vào quy trình vận hành và triển khai tự động (DevOps), thể hiện sự đầu tư vào việc đảm bảo tính ổn định của hệ thống trong dài hạn.

Về giao diện, DocShare mang đến một trải nghiệm người dùng mượt mà và trực quan. Thiết kế theo phong cách hiện đại, tối giản trên nền tối tạo cảm giác chuyên nghiệp, hướng người dùng tập trung vào chức năng cốt lõi là truy cập tài liệu. Ngay sau khi đăng nhập, người dùng sẽ được tiếp cận một thư viện số được tổ chức khoa học, tích hợp công

cụ tìm kiếm thông minh và một trình xem tài liệu mạnh mẽ, cho phép đọc trực tiếp trên trình duyệt.

Điểm đặc biệt và giá trị nhất của DocShare nằm ở mô hình quản lý nội dung. Thay vì cho phép người dùng tự do tải lên, toàn bộ tài liệu trên cổng thông tin đều được đưa lên hệ thống thông qua một quy trình quản trị nội bộ ở cấp độ backend, sau đó được lưu trữ trong một không gian được quản lý chặt chẽ trên máy chủ. Cách tiếp cận này biến DocShare thành một "nguồn sự thật duy nhất" (single source of truth) đáng tin cậy, đảm bảo mọi thành viên chỉ tiếp cận được những phiên bản tài liệu chính thức và đã được phê duyệt, từ đó nâng cao hiệu suất và tính nhất quán cho toàn bộ tổ chức.

1.3 Đặc tả các chức năng hệ thống

Hệ thống DocShare được thiết kế với một triết lý rõ ràng: đơn giản hóa trải nghiệm cho người dùng cuối và đảm bảo tính toàn vẹn của dữ liệu. Do đó, các chức năng được phân chia thành hai nhóm: chức năng dành cho thành viên và quy trình quản lý nội dung ở phía sau.

Người dùng cuối (thành viên) sau khi được cấp tài khoản sẽ có thể sử dụng các chức năng chính sau để tiếp cận và khai thác kho tài liệu:

1.3.1 Chức năng đăng nhập

Mục đích: Xác thực danh tính của thành viên để cấp quyền truy cập vào cổng thông tin. Đây là lớp bảo mật bắt buộc để đảm bảo chỉ những người dùng hợp lệ mới có thể xem tài liệu.

Quy trình: Thành viên sử dụng Email và Mật khẩu đã được cung cấp để điền vào biểu mẫu đăng nhập. Hệ thống sẽ kiểm tra thông tin. Nếu hợp lệ, người dùng sẽ được chuyển đến trang thư viện tài liệu; nếu không, một thông báo lỗi sẽ xuất hiện.

1.3.2 Chức năng tìm kiếm tài liệu

Mục đích: Giúp thành viên nhanh chóng định vị tài liệu cần thiết trong một kho tài nguyên lớn.

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

Quy trình: Tại trang thư viện, thành viên nhập từ khóa liên quan đến tiêu đề tài liệu vào thanh tìm kiếm. Hệ thống sẽ ngay lập tức lọc và hiển thị danh sách các kết quả phù hợp.

1.3.3 Chức năng xem tài liệu

Mục đích: Cho phép thành viên đọc nội dung tài liệu một cách trực quan ngay trên trình duyệt mà không cần tải về hoặc sử dụng phần mềm của bên thứ ba.

Quy trình: Thành viên nhấp vào tiêu đề của một tài liệu trong danh sách. Hệ thống sẽ mở tài liệu đó trong một trình xem chuyên dụng, tích hợp sẵn trên trang web, hỗ trợ các thao tác cơ bản như lật trang, phóng to và thu nhỏ.

1.3.4 Chức năng tải xuống tài liệu

Mục đích: Cung cấp khả năng lưu một bản sao của tài liệu về thiết bị cá nhân để sử dụng khi không có kết nối mạng hoặc cho mục đích lưu trữ.

Quy trình: Khi đang ở trong giao diện xem tài liệu, thành viên có thể nhấp vào nút "Tải xuống". Hệ thống sẽ gửi yêu cầu để trình duyệt bắt đầu quá trình tải tập tin.

1.4 Thiết kế dữ liệu

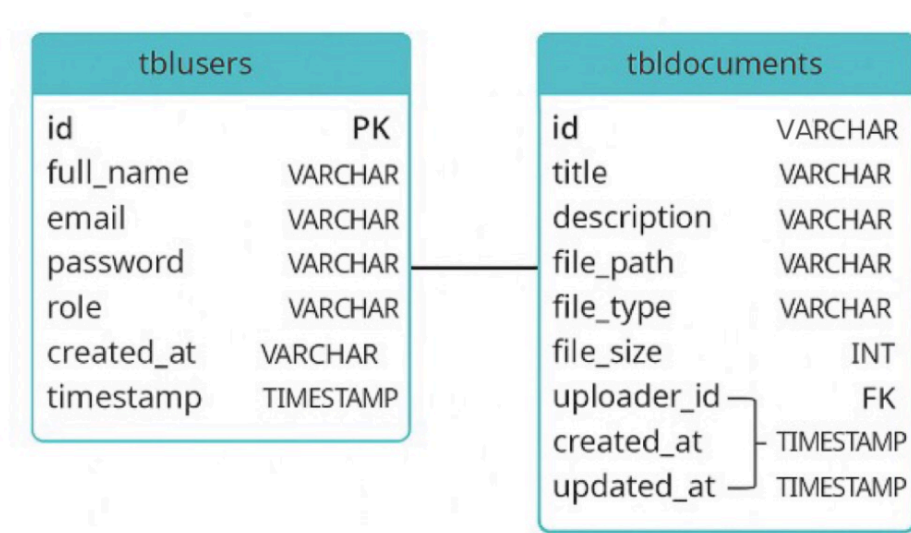
1.4.1 Lược đồ cơ sở dữ liệu

tblusers(id, full_name, email, password, role, created_at, updated_at)

tbldocuments(id, title, description, file_path, file_type, file_size, uploader_id, created_at, updated_at)

STT	Tên thực thể	Diễn giải
1	tblusers	Lưu trữ thông tin tài khoản của các thành viên được phép truy cập vào hệ thống
2	tbldocuments	Lưu trữ thông tin mô tả và đường dẫn của tất cả tài liệu được đăng tải lên cổng thông tin.

1.4.2 Mô hình dữ liệu



Hình 1: Mô hình dữ liệu

CHƯƠNG 2 CƠ SỞ LÝ THUYẾT

2.1 Tổng quan về 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.

2.1.2 Tuyên ngôn Agile

Tuyên ngôn Agile, công bố năm 2001, đề ra 4 giá trị cốt lõi và 12 nguyên tắc hướng dẫn cho phương pháp phát triển phần mềm linh hoạt.

2.1.2.1 Giá trị của Tuyên ngôn Agile

Cá nhân và tương tác hơn là quy trình và công cụ: Tôn trọng các cá nhân và mối quan hệ tương tác giữa các thành viên.

Phần mềm hoạt động tốt hơn là tài liệu đầy đủ: Ưu tiên việc tạo ra phần mềm hoạt động hơn là viết tài liệu chi tiết.

Hợp tác với khách hàng hơn là đàm phán hợp đồng: Tăng cường hợp tác liên tục với khách hàng để đảm bảo sản phẩm phát triển đúng hướng.

Ứng phó với các thay đổi hơn là làm theo kế hoạch: Linh hoạt và sẵn sàng thay đổi để đáp ứng các yêu cầu mới.

2.1.2.2 Nguyên tắc của Tuyên ngôn Agile

Làm hài lòng khách hàng thông qua việc cung cấp phần mềm liên tục và sớm.

Chào đón sự thay đổi trong suốt quá trình phát triển.

Cung cấp phần mềm hoạt động định kỳ.

Tạo điều kiện cho các cuộc hợp tác hàng ngày giữa đội phát triển và khách hàng.

Xây dựng các dự án xoay quanh các cá nhân có động lực.

Thúc đẩy các cuộc trò chuyện trực tiếp.

Phần mềm hoạt động là thước đo chính của tiến độ.

Các quy trình Agile thúc đẩy sự phát triển bền vững.

Liên tục quan tâm đến kỹ thuật xuất sắc và thiết kế tốt.

Tính đơn giản.

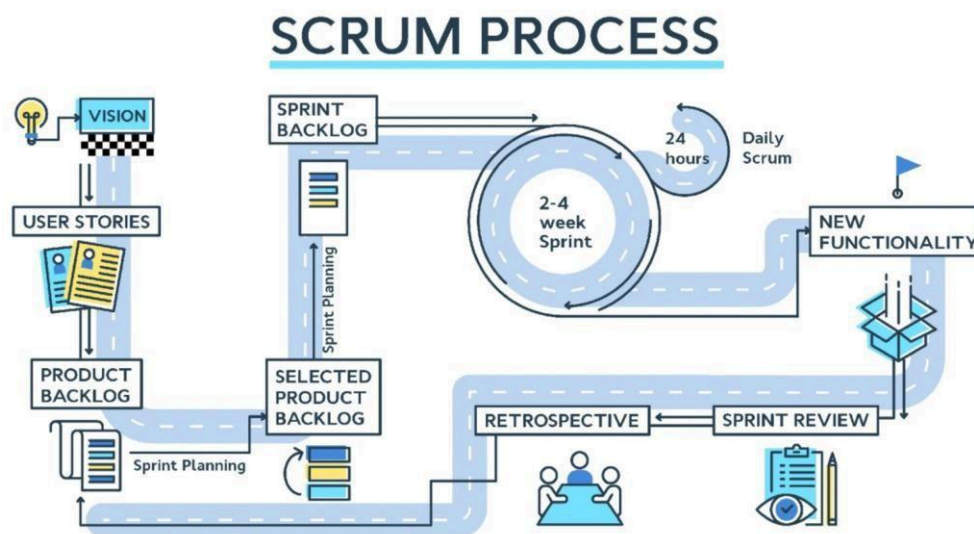
Các nhóm tự tổ chức.

Phản ánh và điều chỉnh thường xuyên.

2.2 Tổng quan về SCRUM

2.2.1 SCRUM là gì ?

Scrum là một phương pháp linh hoạt tập trung vào lập kế hoạch và quản lý linh hoạt. Không giống như XP, nó không xác định các thực hành kỹ thuật sẽ được sử dụng. Nhóm phát triển có thể sử dụng bất kỳ phương pháp kỹ thuật nào mà họ cho là phù hợp với sản phẩm đang được phát triển.



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 Phương pháp SCRUM

SCRUM là một phương pháp quản lý dự án thuộc Agile, cung cấp một khuôn khổ linh hoạt cho việc tổ chức và lập kế hoạch dự án. Không bắt buộc bất kỳ thực hành kỹ thuật cụ thể nào, SCRUM giúp quản lý các công việc cần hoàn thành, thời gian và chi phí phát triển phần mềm, cũng như thời điểm sản phẩm có thể chuyển giao/bán ra thị trường.

2.2.2.1 Thuật ngữ SCRUM

Daily Scrum: Cuộc họp nhóm hàng ngày để xem xét tiến độ và công việc phải hoàn thành.

Sprint: Khoảng thời gian ngắn (thường 2-4 tuần) khi phát triển một phần gia tăng của sản phẩm.

ScrumMaster: Người hướng dẫn nhóm sử dụng hiệu quả phương pháp Scrum.

Product: Sản phẩm phần mềm đang được phát triển bởi nhóm Scrum.

ProductOwner: Người chịu trách nhiệm xác định các tính năng và thuộc tính của sản phẩm, xem xét công việc đã hoàn thành và giúp kiểm tra sản phẩm.

Product backlog: Danh sách việc cần làm bao gồm các lỗi, tính năng và cải tiến sản phẩm chưa hoàn thành.

Development team: Nhóm nhỏ tự tổ chức từ 5-8 người chịu trách nhiệm phát triển sản phẩm.

Potentially shippable product increment: Đầu ra của một Sprint có chất lượng đủ cao để triển khai cho khách hàng sử dụng.

Velocity: Ước tính khối lượng công việc mà một nhóm có thể thực hiện trong một Sprint

2.2.2.2 Các vai trò quan trọng trong Scrum

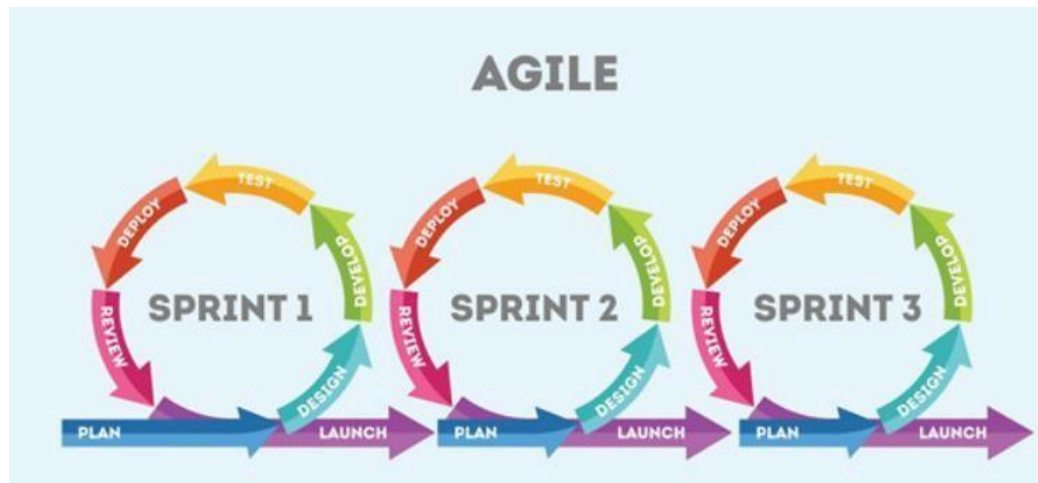
Product Owner: Đảm bảo nhóm phát triển tập trung vào sản phẩm, thường là người quản lý sản phẩm trong các công ty sử dụng Scrum.

Scrum Master: Hướng dẫn nhóm sử dụng hiệu quả phương pháp Scrum, không phải là người quản lý dự án thông thường mà là huấn luyện viên cho nhóm. Trong nhiều công ty, Scrum Master có thể kiêm nhiệm một số trách nhiệm quản lý dự án.

2.2.3 Scrum và Sprints

Trong Scrum, phần mềm được phát triển qua các Sprint - khoảng thời gian cố định (thường là 2-4 tuần) trong đó các tính năng phần mềm được phát triển và chuyển giao. Mỗi Sprint tạo ra một 'phần gia tăng sản phẩm có thể chuyển giao được'. Nhóm có các

cuộc họp hàng ngày (Scrums) để xem xét tiến độ và cập nhật danh sách các hạng mục công việc chưa hoàn thành.



2.2.4 Các thực hành chính của Scrum

Product backlog: Danh sách việc cần làm bao gồm các mục sẽ được triển khai, được xem xét và cập nhật trước mỗi Sprint.

Timeboxed sprints: Khoảng thời gian cố định (2-4 tuần) trong đó các mục từ product backlog được triển khai.

Self-organizing teams: Các nhóm tự tổ chức đưa ra quyết định của riêng họ và làm việc bằng cách thảo luận các vấn đề và đưa ra quyết định bằng sự đồng thuận.

2.2.5 Trạng thái của Product Backlog Item

Sẵn sàng để xem xét: Ý tưởng cấp cao và mô tả tính năng sẽ được xem xét để đưa vào sản phẩm.

Sẵn sàng để tinh chỉnh: Hạng mục quan trọng cần được triển khai với định nghĩa rõ ràng về yêu cầu, cần thêm công việc để hiểu và tinh chỉnh.

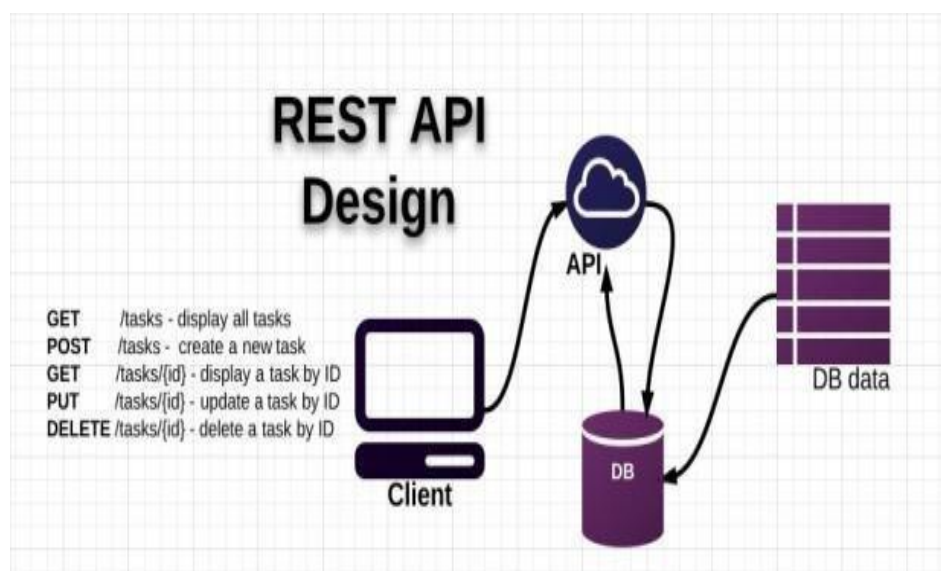
Sẵn sàng triển khai: PBI có đủ thông tin chi tiết để nhóm ước tính nỗ lực liên quan và triển khai, đã xác định các phụ thuộc.

2.3 Công nghệ sử dụng

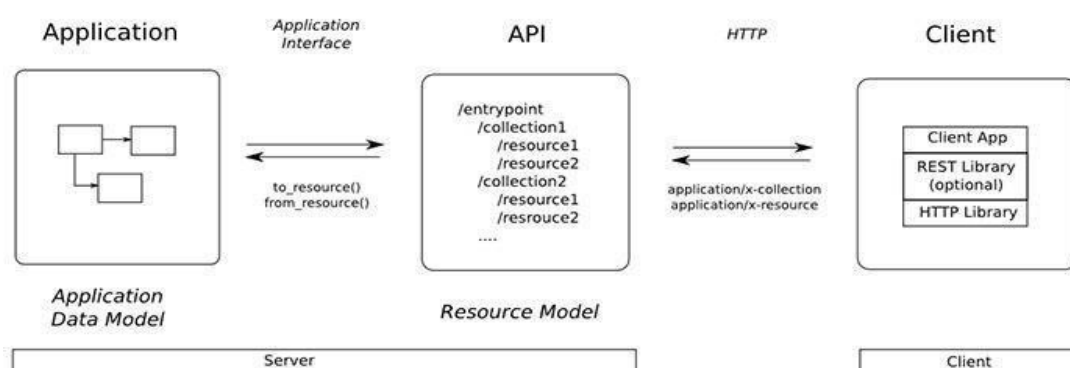
2.3.1 RESTful API

2.3.1.1 Tổng quan về RESTful API

RESTful API là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. Nó chú trọng vào tài nguyên hệ thống (tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu động...), bao gồm các trạng thái tài nguyên được định dạng và được truyền tải qua HTTP.



2.3.1.2 RESTful hoạt động như thế nào?



REST hoạt động chủ yếu dựa vào giao thức HTTP. Các hoạt động cơ bản nêu trên sẽ sử dụng những phương thức HTTP riêng.

GET (SELECT): Trả về một Resource hoặc một danh sách Resource.

POST (CREATE): Tạo mới một Resource.

PUT (UPDATE): Cập nhật thông tin cho Resource.

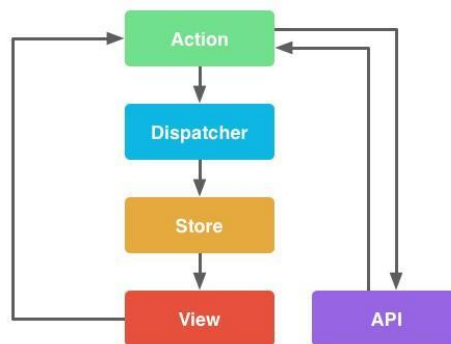
DELETE (DELETE): Xóa một Resource.

Những phương thức hay hoạt động này thường được gọi là CRUD tương ứng với Create, Read, Update, Delete – Tạo, Đọc, Sửa, Xóa.

2.3.2 Next.js & React.js

Next.js là một framework React hiện đại được sử dụng để xây dựng ứng dụng web full-stack. Next.js cung cấp nhiều tính năng mạnh mẽ như Server-Side Rendering (SSR), Static Site Generation (SSG), và API routes tích hợp.

React.js là một framework JavaScript mã nguồn mở được sử dụng để xây dựng giao diện người dùng web. React.js sử dụng mô hình component-based, trong đó giao diện web được xây dựng từ các component nhỏ. Điều này giúp giao diện web trở nên linh hoạt và dễ bảo trì.



2.3.3 Node.js & framework Express.js

2.3.3.1 Node.js

Giới thiệu Node.js

Node.js là một nền tảng JavaScript chạy trên máy chủ. Node.js được sử dụng để phát triển các ứng dụng web, mobile, IoT,...Node.js sử dụng mô hình eventdriven, trong đó các sự kiện được xử lý một cách không đồng bộ. Mô hình này giúp Node.js có thể xử lý nhiều yêu cầu cùng lúc một cách hiệu quả.

Cấu trúc của Node.js

Kernel: Là lớp cơ bản nhất của Node.js, cung cấp các chức năng cơ bản như xử lý sự kiện, quản lý bộ nhớ,...

Modules: Là các thư viện chức năng được sử dụng để xây dựng ứng dụng Node.js.

API: Là các giao diện lập trình ứng dụng cung cấp cho các nhà phát triển khả năng truy cập vào các chức năng của Node.js.

Các tính năng của Node.js

Ứng dụng web: Node.js được sử dụng để xây dựng các ứng dụng web động, hiệu quả và linh hoạt.

Ứng dụng mobile: Node.js được sử dụng để xây dựng các ứng dụng mobile với giao diện web.

Ứng dụng IoT: Node.js được sử dụng để xây dựng các ứng dụng IoT kết nối với các thiết bị vật lý.

2.3.3.2 Express.js

Express là một framework Node.js giúp xây dựng ứng dụng web một cách nhanh chóng và dễ dàng. Express cung cấp nhiều tính năng hữu ích cho việc phát triển ứng dụng web, bao gồm routing, middleware, session management,...

Routing

Routing là tính năng giúp định tuyến các yêu cầu đến các hàm xử lý tương ứng. Express sử dụng routing để định tuyến các yêu cầu đến các hàm xử lý dựa trên URL của yêu cầu.

Middleware

Middleware là các hàm được thực thi trước và sau các hàm xử lý. Middleware thường được sử dụng để thực hiện các tác vụ chung, chẳng hạn như xác thực người dùng, kiểm tra bảo mật,...

Session management

Session management là tính năng giúp lưu trữ thông tin trạng thái của người dùng trong một phiên. Express sử dụng session management để lưu trữ thông tin trạng thái của người dùng, chẳng hạn như tên người dùng, ID người dùng,...

Các tính năng của Express.js

Nhanh chóng: Express giúp xây dựng ứng dụng web một cách nhanh chóng.

Dễ sử dụng: Express có cú pháp đơn giản và dễ học.

Linh hoạt: Express có thể được tùy chỉnh để đáp ứng nhu cầu của các ứng dụng khác nhau.

2.3.4 MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu NoSQL mã nguồn mở, được sử dụng để lưu trữ dữ liệu dạng tài liệu. MongoDB được phát triển bởi MongoDB Inc. và được cấp phép theo Giấy phép Công cộng phía Máy chủ (SSPL).

MongoDB lưu trữ dữ liệu trong các tài liệu JSON. Tài liệu JSON là một tập hợp các cặp khóa-giá trị, trong đó khóa là một chuỗi và giá trị có thể là một chuỗi, số, mảng, hoặc đối tượng JSON.

MongoDB sử dụng một mô hình tài liệu linh hoạt, cho phép bạn lưu trữ dữ liệu theo bất kỳ cách nào phù hợp với nhu cầu của mình. Điều này làm cho MongoDB trở nên linh hoạt và dễ sử dụng cho nhiều loại ứng dụng khác nhau.

Một số tính năng chính của MongoDB bao gồm:

Cơ sở dữ liệu hướng tài liệu: MongoDB lưu trữ dữ liệu trong các tài liệu JSON linh hoạt.

Cấu trúc dữ liệu linh hoạt: MongoDB cho phép bạn lưu trữ dữ liệu theo bất kỳ cách nào phù hợp với nhu cầu của mình.

Tính khả mở: MongoDB có thể được sử dụng trên nhiều nền tảng khác nhau, bao gồm Linux, macOS, và Windows.

Tính sẵn sàng cao: MongoDB có thể được sử dụng để tạo các ứng dụng có sẵn cao.

Tính bảo mật: MongoDB cung cấp các tính năng bảo mật mạnh mẽ để bảo vệ dữ liệu của bạn.

CHƯƠNG 3 XÁC ĐỊNH NHU CẦU

3.1 Product Backlog

PB1 - Quản lý Truy cập và Định danh Người dùng: Đây là tính năng nền tảng, cho phép người dùng tạo tài khoản cá nhân hoặc đăng nhập, đảm bảo tính bảo mật và tạo ra một không gian làm việc được cá nhân hóa.

PB2 - Tính năng Đóng góp và Lưu trữ Nội dung: Cho phép người dùng dễ dàng tải lên và tập trung hóa tài liệu từ nhiều định dạng phổ biến (PDF, DOCX, PPTX, XLSX,...), qua đó hình thành nên một kho tri thức số phong phú cho nền tảng.

PB3 - Không gian Quản lý Tài liệu Cá nhân: Cung cấp một giao diện trực quan nơi người dùng có thể xem, sắp xếp và quản lý toàn bộ tài liệu họ đã tải lên, đảm bảo quyền kiểm soát tuyệt đối đối với tài sản tri thức của mình.

PB4 - Công cụ Tìm kiếm Thông minh và Khám phá Tri thức: Trang bị khả năng tìm kiếm mạnh mẽ, cho phép truy xuất thông tin nhanh chóng và chính xác thông qua tiêu đề, mô tả hoặc từ khóa, giúp người dùng tiết kiệm thời gian và khai thác tối đa giá trị từ kho tài liệu.

PB5 - Chia sẻ Linh hoạt và Phân quyền Truy cập: Là tính năng hiện thực hóa giá trị cốt lõi của DocShare, cho phép người dùng lan tỏa tri thức bằng cách chia sẻ tài liệu qua liên kết an toàn, đi kèm với khả năng tùy chỉnh quyền hạn (chỉ xem, cho phép tải về) để bảo vệ nội dung.

3.2 Sprint backlog

Sprint 1: Cơ bản & Xác thực

Mục tiêu của Sprint này là hoàn thành bộ khung xương sống của ứng dụng. Về phía frontend, nhóm sẽ thiết kế và hiện thực hóa các giao diện người dùng chính yếu, bao gồm trang đăng ký, đăng nhập và trang quản lý tài liệu. Song song đó, nhóm backend sẽ xây dựng các API trọng yếu: API xác thực người dùng sử dụng JSON Web Tokens (JWT) để đảm bảo an toàn, API cho phép người dùng tải lên tài liệu và API để truy xuất danh sách tài liệu cá nhân. Kết thúc Sprint, hệ thống sẽ có một luồng người dùng cơ bản hoàn chỉnh.

Sprint 2: Chia sẻ & Phân loại

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

Trên nền tảng đã có, Sprint 2 tập trung phát triển các tính năng xã hội và quản lý. Chức năng chia sẻ, trái tim của DocShare, sẽ được phát triển, cho phép tạo liên kết công khai hoặc riêng tư với cơ chế phân quyền truy cập đi kèm. Để người dùng có thể tổ chức kho tri thức của mình, các công cụ như gắn tag và tạo folder sẽ được triển khai. Giao diện chi tiết tài liệu và khả năng đánh dấu (bookmark) cũng sẽ được hoàn thiện trong giai đoạn này.

Sprint 3: Tìm kiếm & Thống kê

Tập trung vào việc gia tăng "trí thông minh" cho sản phẩm. Một công cụ tìm kiếm toàn văn mạnh mẽ sẽ được tích hợp, hỗ trợ lọc kết quả theo nhiều tiêu chí như từ khóa, tag và loại tệp. Để mang lại giá trị phân tích cho người dùng, chức năng thống kê lượt xem và lượt tải sẽ được phát triển. Các cơ chế về phân quyền và bộ sưu tập tài liệu yêu thích sẽ được rà soát và hoàn thiện.

Sprint 4: Kiểm thử & Triển khai

Đây là Sprint cuối cùng, tập trung hoàn toàn vào chất lượng và phát hành sản phẩm. Một quy trình kiểm thử toàn diện (end-to-end testing) sẽ được thực hiện trên cả frontend và backend để phát hiện và khắc phục các lỗi logic, giao diện. Hiệu suất của các API và trải nghiệm người dùng (UX/UI) sẽ được tối ưu hóa. Cuối cùng, sản phẩm sẽ được đóng gói (Docker) và triển khai lên các nền tảng đám mây (Vercel, Railway) để chính thức đi vào hoạt động.

3.3 Technical Backlog

TE1: Phát triển Frontend

Phía client-side, nhóm phát triển chịu trách nhiệm thiết kế và xây dựng giao diện người dùng cho toàn bộ các trang chức năng, bao gồm: Đăng nhập/Đăng ký, Trang chủ, Tải lên tài liệu, Danh sách tài liệu, Xem chi tiết, Chia sẻ và Bookmark. Để tối ưu hóa quá trình phát triển, một thư viện các component tái sử dụng (form, modal, table, card,...) sẽ được xây dựng. Nền tảng giao diện được xây dựng với triết lý "mobile-first" để đảm bảo tính responsive trên mọi thiết bị. Công nghệ styling chính được sử dụng là Tailwind CSS và thư viện component Shadcn/ui.

TE2: Phát triển Backend

Hệ thống backend được xây dựng dưới dạng một bộ các API RESTful dựa trên Express.js, với cơ sở dữ liệu là MongoDB và Mongoose ODM để định hình schema. Các endpoints chính bao gồm: xác thực người dùng (sử dụng JWT), upload tài liệu, chia sẻ, tìm kiếm, quản lý metadata, bookmark và phân quyền chi tiết. Việc upload file được xử lý bởi middleware Multer, và thư viện Sharp được dùng để tự động tạo thumbnail. Các biện pháp bảo mật cốt lõi được tích hợp bao gồm Helmet để bảo vệ các header HTTP, CORS để kiểm soát truy cập liên miền, rate limiting để chống tấn công DoS, và validation để ngăn chặn các lỗ hổng như XSS.

TE3: Tích hợp và Kiểm thử

Quá trình tích hợp giữa frontend và backend được thực hiện thông qua một service layer được định nghĩa rõ ràng. Chiến lược đảm bảo chất lượng được thực thi nghiêm ngặt với việc viết unit test và integration test: React Testing Library được dùng để kiểm thử các component React, trong khi Jest và Supertest được sử dụng để kiểm thử các API endpoint của backend. Các kịch bản kiểm thử bao trùm toàn bộ các khía cạnh: chức năng, hiệu suất, bảo mật và tương tác người dùng.

TE4: Triển khai và Giám sát

Toàn bộ ứng dụng (frontend, backend, database, nginx reverse proxy) được đóng gói bằng Docker theo kiến trúc multi-services. Môi trường production được thiết lập trên các nền tảng đám mây hiện đại: Vercel chịu trách nhiệm triển khai và tối ưu cho frontend, Railway được sử dụng để host backend và database. Một hệ thống giám sát chủ động được thiết lập để theo dõi hiệu suất (Log, Uptime Robot) và lỗi (Sentry), kết hợp với middleware logging để ghi lại các hành vi và lỗi phát sinh trong quá trình người dùng sử dụng.

3.4 Support Backlog

ID: SB1 – Cập nhật tài liệu hướng dẫn sử dụng

Mô tả công việc: Biên soạn và duy trì một bộ tài liệu hướng dẫn sử dụng chi tiết, luôn được cập nhật theo các phiên bản mới nhất của sản phẩm.

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

Đối tượng: Tài liệu sẽ có hai phiên bản riêng biệt dành cho người dùng thông thường và quản trị viên.

Nội dung: Hướng dẫn đầy đủ các bước thực hiện những chức năng quan trọng như: upload, chia sẻ tài liệu, tìm kiếm, phân quyền truy cập, và các tính năng khác.

ID: SB2 – Xây dựng tính năng thống kê & báo cáo

Mô tả công việc: Phát triển một module mới trong trang quản trị dành riêng cho việc thống kê và báo cáo.

Yêu cầu kỹ thuật: Tính năng phải có khả năng truy xuất dữ liệu hoạt động và hiển thị chúng dưới dạng các biểu đồ trực quan, dễ hiểu.

ID: SB3 – Tối ưu tương thích trình duyệt

Mô tả công việc: Thực hiện quy trình kiểm thử chéo trình duyệt (Cross-Browser Testing) một cách định kỳ để phát hiện và khắc phục các lỗi về giao diện hoặc chức năng.

Mục tiêu: Đảm bảo ứng dụng DocShare hoạt động mượt mà và có giao diện hiển thị nhất quán trên các trình duyệt phổ biến, bao gồm Google Chrome, Mozilla Firefox, Microsoft Edge và Safari.

CHƯƠNG 4 LẬP KẾ HOẠCH SCRUM

4.1 Sprint 1:

ID	Issue	Person	Story	Start	End
1	Tạo giao diện Đăng nhập/Đăng ký	Liều Kiện An	5	20/06/2025	27/06/2025
2	Thiết kế giao diện trang chủ/dashboard	Liều Kiện An	3	20/06/2025	27/06/2025
3	Giao diện upload tài liệu	Trương Mỹ Duyên	5	20/06/2025	27/06/2025
4	Giao diện danh sách tài liệu của người dùng	Trương Mỹ Duyên	3	20/06/2025	27/06/2025
5	API xác thực người dùng (JWT)	Trương Hoàng Giang	5	20/06/2025	27/06/2025
6	API lấy danh sách tài liệu cá nhân	Trương Hoàng Giang	3	20/06/2025	27/06/2025
7	API upload tài liệu và sinh thumbnail	Trương Hoàng Giang	5	20/06/2025	27/06/2025
8	API tìm kiếm tài liệu (full-text search)	Trương Hoàng Giang	3	20/06/2025	27/06/2025

4.2 Sprint 2:

ID	Issue	Person	Story	Start	End
9	Giao diện chi tiết tài liệu và tải về	Trương Mỹ Duyên	5	27/06/2025	05/07/2025
10	Giao diện chia sẻ tài liệu	Trương Mỹ Duyên	5	27/06/2025	05/07/2025
11	Giaodiện gắn folder cho tài liệu	Liều Kiện An	5	27/06/2025	05/07/2025

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

12	API chia sẻ tài liệu (public/private + link)	Trương Hoàng Giang	5	27/06/2025	05/07/2025
----	--	--------------------	---	------------	------------

4.3 Sprint 3:

ID	Issue	Person	Story	Start	End
13	Giao diện phân quyền tài liệu (xem / tải)	Liều Kiện An	5	07/07/2025	15/07/2025
14	Giao diện bộ sưu tập bookmark cá nhân	Liều Kiện An	5	07/07/2025	15/07/2025
15	API phân quyền tài liệu (read-only / download)	Trương Hoàng Giang	5	07/07/2025	15/07/2025
16	API theo dõi lượt xem và tải xuống tài liệu	Trương Hoàng Giang	5	07/07/2025	15/07/2025
17	API trích xuất metadata từ file tải lên	Trương Hoàng Giang	5	07/07/2025	15/07/2025
18	Hoàn thiện các chức năng chia sẻ và lọc	Trương Mỹ Duyên	5	07/07/2025	15/07/2025

4.4 Sprint 4:

ID	Issue	Person	Story	Start	End
19	Kiểm thử & sửa lỗi toàn bộ ứng dụng	Trương Mỹ Duyên	5	16/07/2025	22/07/2025
20	Tối ưu giao diện người dùng và responsive	Liều Kiện An	5	16/07/2025	22/07/2025
21	Kiểm tra toàn bộ API & fix lỗi logic	Trương Hoàng Giang	5	16/07/2025	22/07/2025
22	Tối ưu hiệu suất API và database	Liều Kiện An	5	16/07/2025	22/07/2025

CHƯƠNG 5 THIẾT KẾ HỆ THỐNG

5.1. Sprint 1 - Giao diện cơ bản và xác thực người dùng

5.1.1. Giao diện ứng dụng vẽ bằng figma



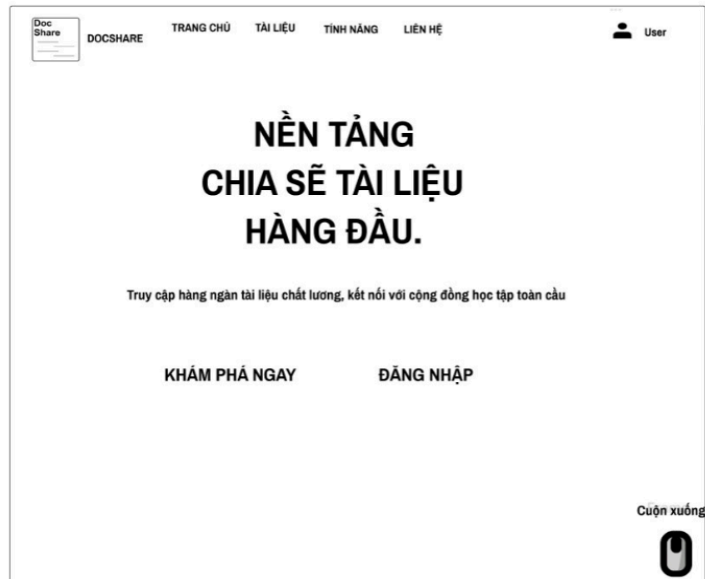
The login form is enclosed in a rounded rectangle. At the top right is a button labeled "ĐĂNG NHẬP". Below it, on the left, is a label "Email" above a text input field with placeholder text "Nhập email của bạn...". Below the email field is a label "Mật khẩu" above another text input field with placeholder text "Nhập mật khẩu của bạn...". Under the password field, there is a radio button followed by the text "Ghi nhớ đăng nhập" and a link "Quên mật khẩu?". At the bottom, there is a blue button with white text "Đăng nhập" and a larger blue button with a Google logo and the text "Đăng nhập với google".

Hình 2: Giao diện đăng nhập

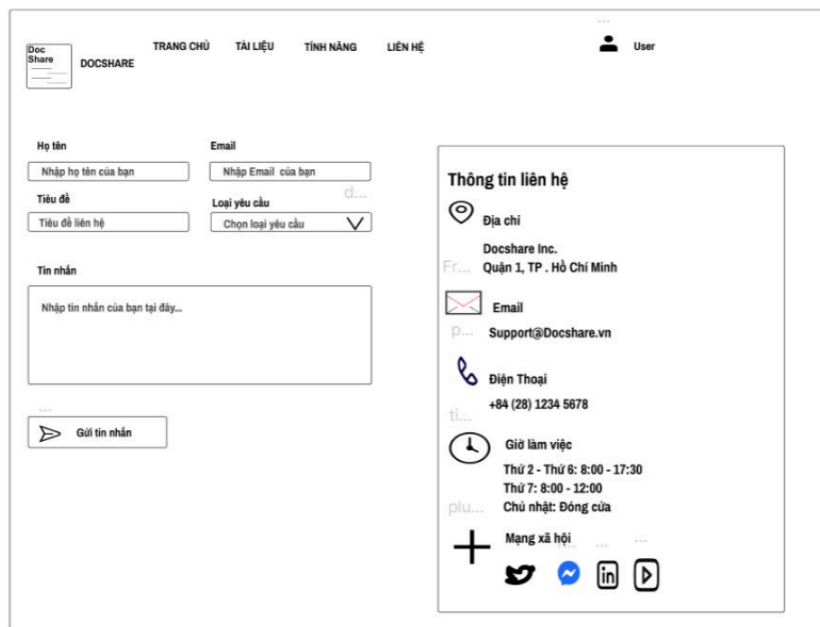


The registration form is enclosed in a rounded rectangle. At the top right is a button labeled "ĐĂNG KÝ". Below it, on the left, is a label "Tên tài khoản" above a text input field with placeholder text "Nhập tên tài khoản của bạn...". Below the name field is a label "Email" above a text input field with placeholder text "Nhập email của bạn...". Below the email field is a label "Mật khẩu" above a text input field with placeholder text "Nhập mật khẩu của bạn...". At the bottom, there is a blue button with white text "Đăng ký" and a larger blue button with a Google logo and the text "Đăng ký với google".

Hình 3: Giao diện đăng ký



Hình 4: Giao diện trang chủ



Hình 5: Giao diện upload tài liệu



Hình 6: Giao diện danh sách tài liệu người dùng

5.1.2. Mã nguồn

Trong thư mục config, file db.js là file cấu hình kết nối đến cơ sở dữ liệu MongoDB. Đầu tiên, cần require thư viện mongoose để tương tác với MongoDB. Một hàm connectDB được tạo ra để kết nối, sử dụng mongoose.connect() để thiết lập chuỗi kết nối lấy từ biến môi trường. Hàm này sau đó được xuất (export) ra để có thể sử dụng tại file khởi tạo của máy chủ, đảm bảo kết nối CSDL được thiết lập ngay khi ứng dụng khởi chạy.

Trong thư mục models chứa các file để tạo ra các model cho những đối tượng cần xử lý trong cơ sở dữ liệu. Tại đây, ta thực hiện require mongoose để sử dụng mongoose.Schema nhằm định nghĩa cấu trúc dữ liệu cho các đối tượng User (người dùng) và Document (tài liệu). Từ các schema này, các Model tương ứng được tạo ra. Model này sẽ làm việc như một interface giữa ứng dụng Node.js và cơ sở dữ liệu MongoDB, cho phép thực hiện các thao tác như tìm kiếm, thêm mới, cập nhật, xóa dữ liệu (CRUD) và nhiều thao tác khác.

Trong mục controllers chứa các file để xử lý các yêu cầu nghiệp vụ từ client. Chẳng hạn như: authController.js xử lý logic xác thực người dùng (đăng ký, đăng nhập), documentController.js chịu trách nhiệm cho các thao tác thêm, sửa, xóa và truy xuất tài liệu.

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

Trong thư mục routes chứa các file quy định các đường dẫn (endpoints) đến các xử lý được viết trong Controllers. Để sử dụng các hàm xử lý này, ta cần require các file controller tương ứng và gọi đến các hàm thông qua các phương thức HTTP (GET, POST, PUT, DELETE) được định nghĩa bởi framework Express.js.

Backend

```
> config
  db.js
  jwt.js

> controllers
  authController.js
  documentController.js

> middleware
  auth.js
  upload.js
  validation.js

> models
  User.js
  Document.js

> routes
  auth.js
  documents.js

> services
  thumbnail.js
  search.js
```

Trong thư mục app chứa các file định tuyến và giao diện của từng trang. Cấu trúc này tuân theo quy tắc "App Router" của Next.js, giúp việc tổ chức trang trở nên trực quan. Cụ thể như sau: app/auth chứa giao diện Đăng nhập/Đăng ký; app/dashboard chứa giao diện trang tổng quan; và app/documents chứa các trang quản lý tài liệu.

Thư mục components chứa các thành phần giao diện (UI Components) có thể tái sử dụng trên toàn bộ ứng dụng. Việc chia nhỏ giao diện thành các component giúp mã nguồn sạch hơn và dễ bảo trì. Ví dụ, components/auth/login-form.tsx là một component form đăng nhập được nhúng vào trang app/auth/login/page.tsx.

Thư mục lib là nơi chứa các logic và cấu hình phía client. File lib/api.ts có thể chứa các cấu hình cho việc gọi API (ví dụ như cấu hình Axios), trong khi thư mục lib/services chứa các hàm xử lý nghiệp vụ phía client. Thư mục types chứa các định nghĩa kiểu dữ liệu của TypeScript, giúp đảm bảo sự nhất quán và an toàn về kiểu cho các đối tượng như User và Document trong toàn bộ mã nguồn frontend.

Sau khi hoàn tất các công việc của Sprint 1, ứng dụng đã có các giao diện và chức năng cốt lõi như: Giao diện đăng ký, giao diện đăng nhập, giao diện dashboard hiển thị thống kê, giao diện tải lên và quản lý danh sách tài liệu.

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

Frontend (src)

```
> app
  > auth
    > login
      page.tsx
    > register
      page.tsx
  > dashboard
    page.tsx
    > stats
      page.tsx
    > activities
      page.tsx
  > documents
    page.tsx
    > upload
      page.tsx
    > [id]
      page.tsx

> components
  > auth
    login-form.tsx
    register-form.tsx
```

```
    > dashboard
      stats-card.tsx
      recent-docs.tsx
      activity-feed.tsx
    > documents
      doc-card.tsx
      upload-form.tsx
      doc-list.tsx

  > lib
    api.ts
    > services
      auth.ts
      documents.ts

  > types
    auth.ts
    documents.ts
```

5.2. Sprint 2 - Chia sẻ, phân loại

5.2.1. Giao diện ứng dụng vẽ bằng figma

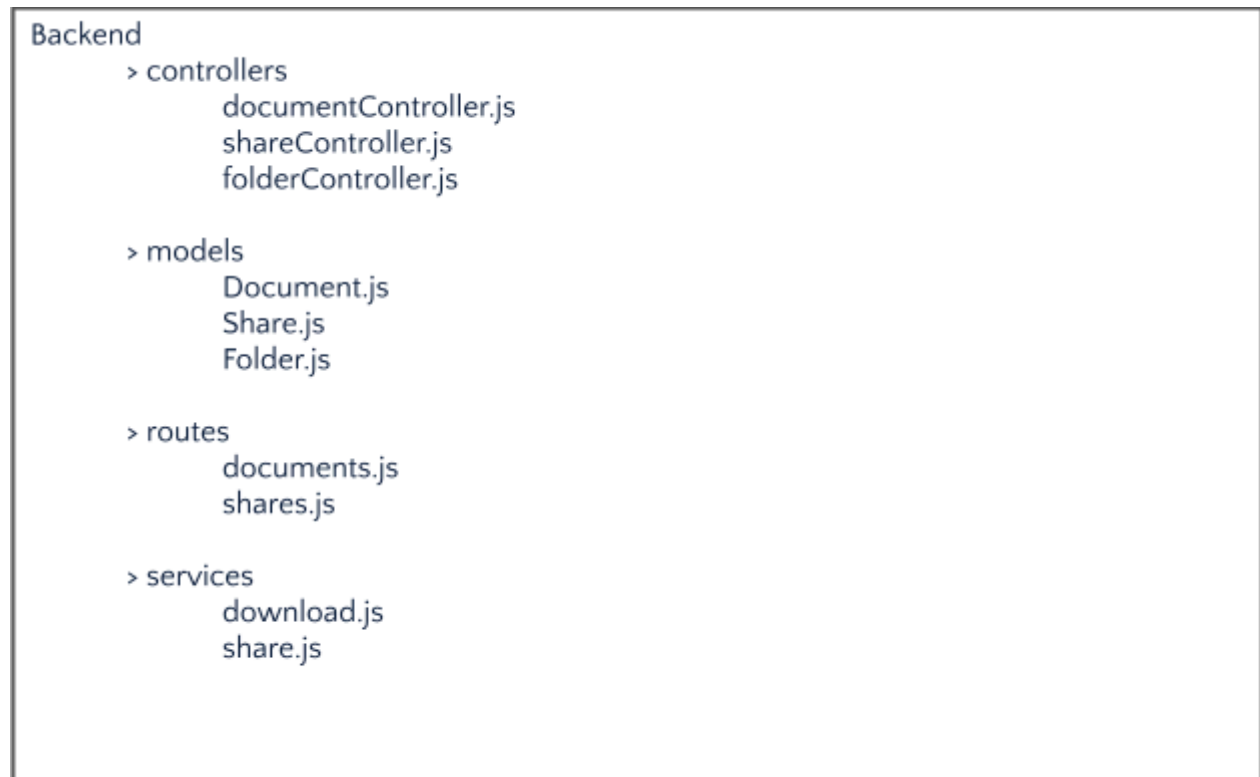


Hình 7: Giao diện chia sẻ tài liệu



Hình 8: Giao diện gắn thư mục cho tài liệu

5.2.2. Mã nguồn



Trong thư mục models, hai model mới đã được tạo ra. File Share.js định nghĩa schema để lưu trữ thông tin về một phiên chia sẻ, bao gồm ID của tài liệu được chia sẻ, một shareId duy nhất cho liên kết công khai, các quyền truy cập và thời gian hết hạn. Tương tự, Folder.js định nghĩa schema cho các thư mục do người dùng tạo ra để phân loại tài liệu.

Để xử lý logic nghiệp vụ cho các tính năng này, các file controller mới được thêm vào thư mục controllers. File shareController.js chịu trách nhiệm xử lý các yêu cầu như tạo link chia sẻ, cập nhật quyền truy cập, và xác thực người dùng khi truy cập link. Trong khi đó, folderController.js quản lý việc tạo, đổi tên, xóa thư mục và thêm/bỏ tài liệu khỏi một thư mục.

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

Tương ứng với các controller mới, thư mục routes được bổ sung file shares.js để định nghĩa các điểm cuối (endpoints) cho API liên quan đến chức năng chia sẻ (ví dụ: POST /shares, PUT /shares/:id).

Thư mục services cũng được mở rộng với download.js để xử lý logic tải file và share.js để chứa các hàm xử lý logic chia sẻ phức tạp có thể được tái sử dụng

Frontend (src)

```
> app
  > documents
    > [id]
      page.tsx
    > share
      page.tsx
    > folders
      page.tsx
  > shared
    > [shareId]
      page.tsx

> components
  > documents
    > detail
      document-view.tsx
      document-info.tsx
      document-actions.tsx
    > share
      share-modal.tsx
      share-link.tsx
      share-settings.tsx
    > folders
      folder-list.tsx
      folder-select.tsx
      folder-create.tsx

> hooks
  use-document.ts
  use-share.ts
  use-folders.ts

> lib
  > services
    document.ts
    share.ts
```

Phía frontend được nâng cấp đáng kể để xây dựng giao diện cho các tính năng chia sẻ và phân loại tài liệu.

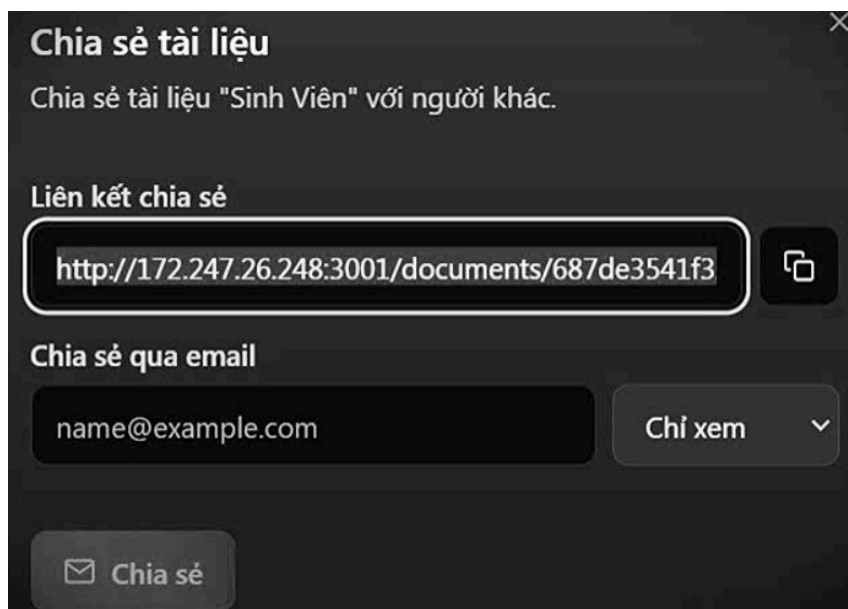
Cấu trúc định tuyến trong thư mục app được mở rộng để tạo ra các trang chuyên biệt cho việc quản lý chia sẻ (share), thư mục (folders), và một trang công khai để xem tài liệu được chia sẻ (shared/[shareId]).

Để xây dựng các trang này, thư mục components được tái cấu trúc, bổ sung các thư mục con như detail, share, và folders để chứa các thành phần giao diện có thể tái sử dụng, ví dụ như share-modal.tsx (cửa sổ chia sẻ) hay folder-list.tsx (danh sách thư mục).

Một điểm nhấn kiến trúc quan trọng là sự ra đời của thư mục hooks. Nơi đây chứa các custom hook (use-share.ts, use-folders.ts) nhằm tách biệt và đóng gói các logic xử lý trạng thái phức tạp, giúp cho các component giao diện trở nên gọn gàng và dễ bảo trì.

5.3. Sprint 3 - Thống kê và phân quyền tài liệu

5.3.1. Giao diện ứng dụng vẽ bằng figma



Hình 9: Giao diện chia sẻ tài liệu

5.3.2. Mã nguồn

Backend

```
> controllers
  permissionController.js
  analyticsController.js
  bookmarkController.js
  metadataController.js

> models
  Permission.js
  Analytics.js
  Bookmark.js
  Metadata.js

> middleware
  permission.js
  analytics.js

> routes
  permissions.js
  analytics.js
  bookmarks.js

> services
  metadata-extractor.js
  analytics-tracker.js
```

Thư mục models được bổ sung các schema mới quan trọng: Permission.js định nghĩa cấu trúc để lưu quyền truy cập chi tiết (xem/tải) cho từng người dùng hoặc vai trò trên một tài liệu; Analytics.js ghi lại các sự kiện như lượt xem, lượt tải; Bookmark.js cho phép người dùng lưu lại các tài liệu quan tâm vào các bộ sưu tập; và Metadata.js lưu trữ các thông tin được trích xuất từ file như tiêu đề, tác giả.

Tương ứng, thư mục controllers có thêm các file mới để xử lý logic nghiệp vụ cho từng tính năng.

Một sự bổ sung quan trọng là thư mục middleware. File permission.js đóng vai trò như một "người gác cổng", kiểm tra quyền truy cập của người dùng trước khi cho phép thực hiện một hành động (ví dụ: tải file). File analytics.js là một middleware ghi nhận, tự động theo dõi các hoạt động của người dùng và lưu vào cơ sở dữ liệu.

Thư mục routes được mở rộng với các file như permissions.js, analytics.js, và bookmarks.js để định nghĩa các API endpoint cho các tính năng mới. Cuối cùng, thư mục services chứa các logic phức tạp như metadata-extractor.js (dịch vụ tự động đọc và trích xuất siêu dữ liệu từ file được tải lên) và analytics-tracker.js (dịch vụ theo dõi hoạt động).

```
> documents
  > [id]
    > permissions
      page.tsx
    > analytics
      page.tsx
  > shared
    > [shareId]
      page.tsx
> bookmarks
  page.tsx
  > [collectionId]
    page.tsx
> components
  > documents
    > permissions
      permission-form.tsx
      user-permissions.tsx
      role-permissions.tsx
    > analytics
      view-stats.tsx
      download-stats.tsx
    > metadata
      doc-metadata.tsx
      metadata-editor.tsx
```



```
> bookmarks
    bookmark-list.tsx
    collection-card.tsx
    bookmark-filters.tsx
> hooks
    use-permissions.ts
    use-analytics.ts
    use-bookmarks.ts
> lib
    > services
        permissions.ts
        analytics.ts
        metadata.ts
```

Trong thư mục app, cấu trúc định tuyến được mở rộng để tích hợp các trang chức năng mới. Giờ đây, trong trang chi tiết của mỗi tài liệu (documents/[id]), người dùng có thể truy cập các trang con permissions để quản lý quyền và analytics để xem thống kê. Một khu vực hoàn toàn mới là bookmarks cũng được tạo ra, cho phép người dùng xem và quản lý các tài liệu đã lưu và các bộ sưu tập của mình.

Để xây dựng các giao diện phức tạp này, thư mục components được mở rộng với các nhóm component chuyên dụng. Nhóm permissions chứa các form và danh sách để cấp và quản lý quyền; nhóm analytics chứa các biểu đồ, bảng biểu (view-stats.tsx, download-stats.tsx) để trực quan hóa dữ liệu; và nhóm bookmarks cung cấp các thẻ, danh sách và bộ lọc cho trang quản lý bộ sưu tập.

Kiến trúc frontend tiếp tục được tối ưu với việc bổ sung các custom hook mới trong thư mục hooks. Các hook như use-permissions.ts, use-analytics.ts, và use-bookmarks.ts giúp đóng gói logic giao tiếp với API và quản lý trạng thái, làm cho mã nguồn tại các component giao diện trở nên sạch sẽ, rõ ràng và dễ bảo trì hơn.

5.4. Sprint 4 - Testing, Fix bugs và Triển khai

Mã nguồn

Sau khi hoàn tất các giai đoạn kiểm thử và tối ưu hóa, bước cuối cùng của Sprint 4 là đóng gói toàn bộ ứng dụng bằng Docker để đảm bảo tính nhất quán và dễ dàng triển khai trên mọi môi trường.

Toàn bộ mã nguồn của ứng dụng từ front-end đến back-end sẽ được "container hóa" bằng Docker.

```
FROM node:20.10.0
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
CMD ["npm", "start"]
```

Về phía Frontend (DocShare), cấu hình Dockerfile như sau:

```
services:
  frontend:
    restart: always
    build:
      context: ./DocShare
      dockerfile: Dockerfile
    ports:
      - "3000:3000"
    networks:
      - docshare-net
  backend:
    restart: always
    build:
      context: ./backend
      dockerfile: Dockerfile
    ports:
      - "3001:3001"
    networks:
      - docshare-net
networks:
  docshare-net:
```

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

Sau khi cấu hình, việc triển khai được thực hiện qua hai lệnh đơn giản trong cửa sổ terminal tại thư mục gốc của dự án:

Build Images: Chạy lệnh `docker compose build` để xây dựng các image cho các dịch vụ đã được định nghĩa trong file `compose.yaml` (cụ thể là frontend và backend).

Start Containers: Chạy lệnh `docker compose up -d` để khởi động các container cho các dịch vụ ở chế độ nền (detached mode).

CHƯƠNG 6 KẾT QUẢ THỰC NGHIỆM

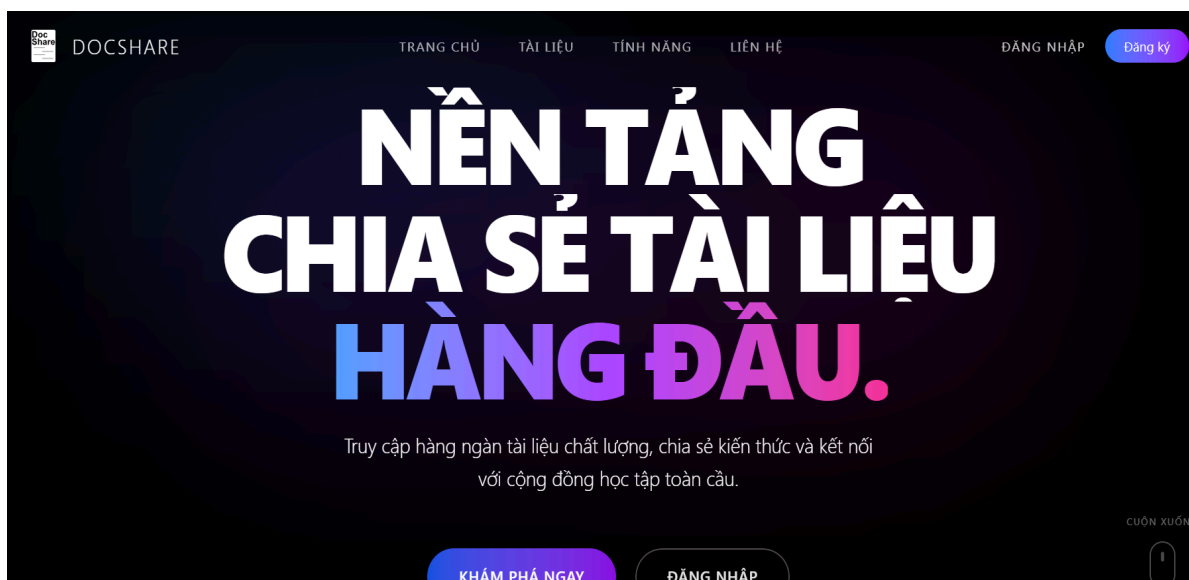
6.1. Kết quả đạt được

Nền tảng chia sẻ tài liệu DOCSHARE đã được xây dựng thành công và đáp ứng đầy đủ các yêu cầu cốt lõi đã đề ra. Nền tảng có khả năng xử lý nhanh chóng, giao diện hiện đại, thân thiện và dễ sử dụng.

Quá trình xây dựng sản phẩm đã tạo ra một hệ thống hoàn chỉnh với các luồng chức năng người dùng liền mạch từ đăng ký, đăng nhập, quản lý tài khoản cho đến xem và tải tài liệu. Sản phẩm cuối cùng không chỉ đáp ứng được các yêu cầu chức năng mà còn thể hiện sự đầu tư kỹ lưỡng vào trải nghiệm người dùng (UX) và giao diện người dùng (UI).

6.2. Kết quả thực nghiệm

Trang chủ hiển thị thông tin gồm: tài liệu, tính năng, liên hệ, đăng nhập, đăng ký, khám phá.



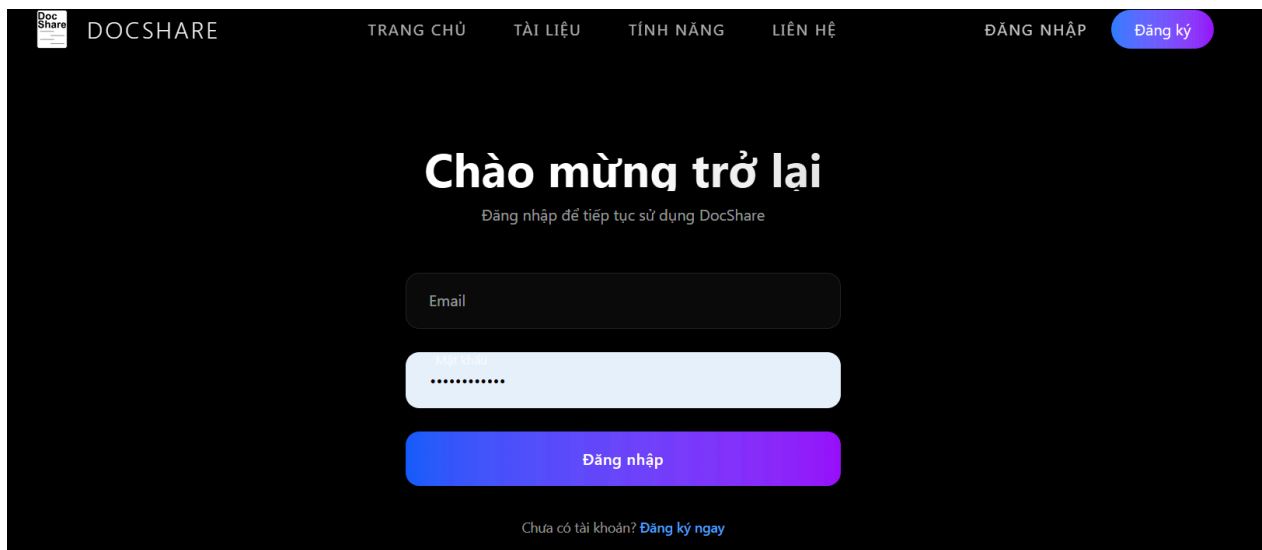
Hình 10: Giao diện trang chủ

6.2.1. Chức năng đăng nhập

Sau khi lựa chọn loại tài khoản đăng nhập, nhập tài khoản, mật khẩu và nhấn vào đăng nhập để đăng nhập. Lưu ý, hệ thống sẽ kiểm tra xem tài khoản bạn vừa đăng nhập

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

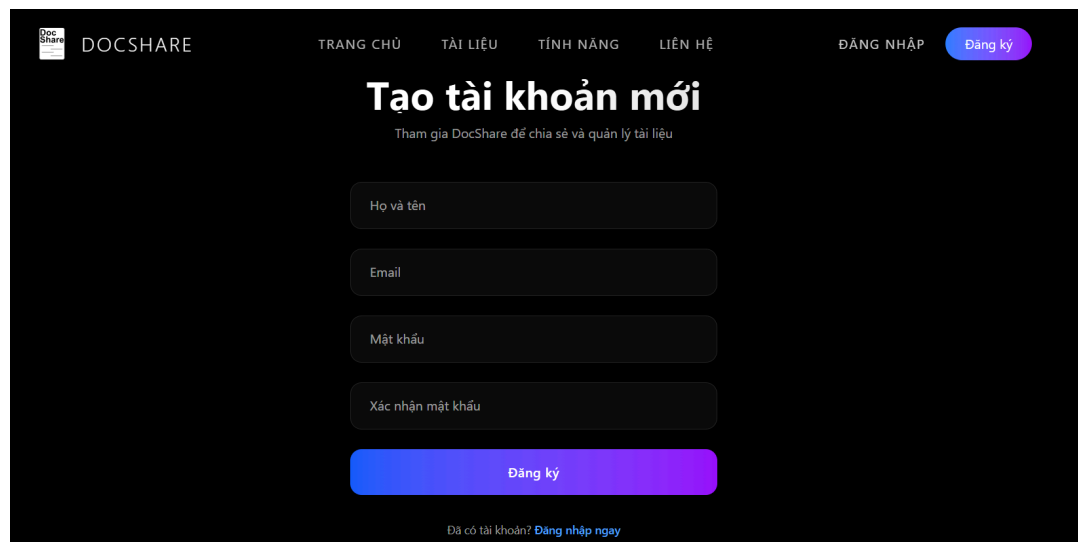
vào có đúng với loại tài khoản bạn lựa chọn, nếu sai hệ thống sẽ hiển thị cảnh báo cho biết loại tài khoản của bạn thuộc tài khoản gì và quay lại trang chủ.

The image shows the login page of the DocShare application. At the top, there is a navigation bar with the DocShare logo and links for 'TRANG CHỦ', 'TÀI LIỆU', 'TÍNH NĂNG', 'LIÊN HỆ', 'ĐĂNG NHẬP', and a 'Đăng ký' button. The main heading is 'Chào mừng trở lại' (Welcome back), followed by the instruction 'Đăng nhập để tiếp tục sử dụng DocShare'. Below this, there are two input fields: 'Email' and 'Mật khẩu' (Password). A large blue 'Đăng nhập' (Login) button is positioned below the password field. At the bottom, there is a link that says 'Chưa có tài khoản? Đăng ký ngay' (Don't have an account? Register now).

Hình 11: Giao diện trang đăng nhập

6.2.2. Chức năng đăng ký

Nhấn vào đăng ký tài khoản để đăng ký tài khoản mới, tài khoản quản trị nội dung không đăng ký được.

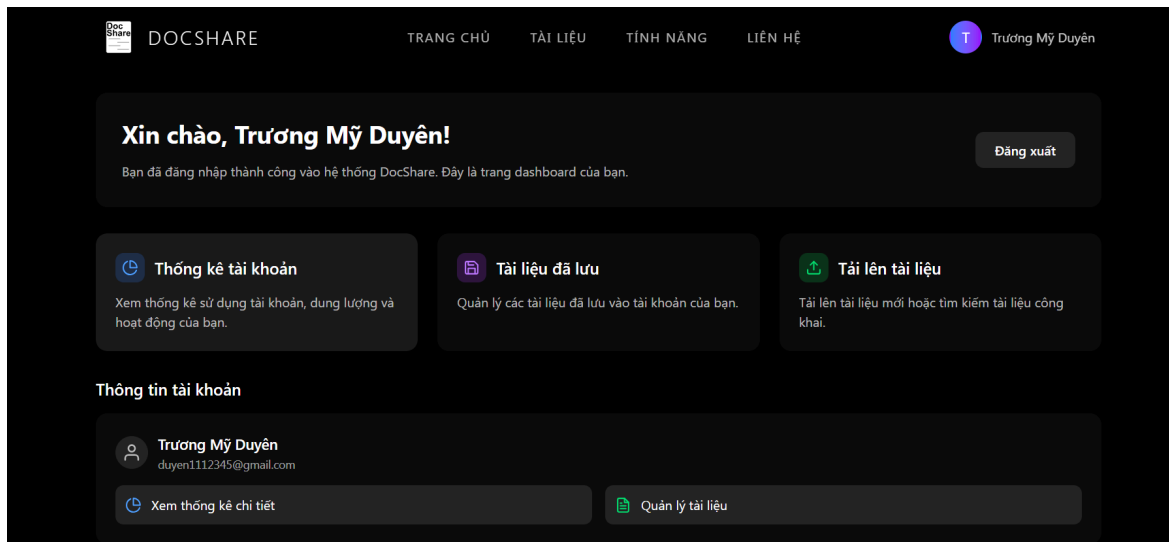
The image shows the registration page of the DocShare application. The navigation bar is identical to the login page. The main heading is 'Tạo tài khoản mới' (Create new account), followed by the instruction 'Tham gia DocShare để chia sẻ và quản lý tài liệu' (Join DocShare to share and manage documents). Below this, there are four input fields: 'Họ và tên' (Full name), 'Email', 'Mật khẩu' (Password), and 'Xác nhận mật khẩu' (Confirm password). A large blue 'Đăng ký' (Register) button is positioned below the confirmation field. At the bottom, there is a link that says 'Đã có tài khoản? Đăng nhập ngay' (Already have an account? Login now).

Hình 12: Giao diện trang đăng ký

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

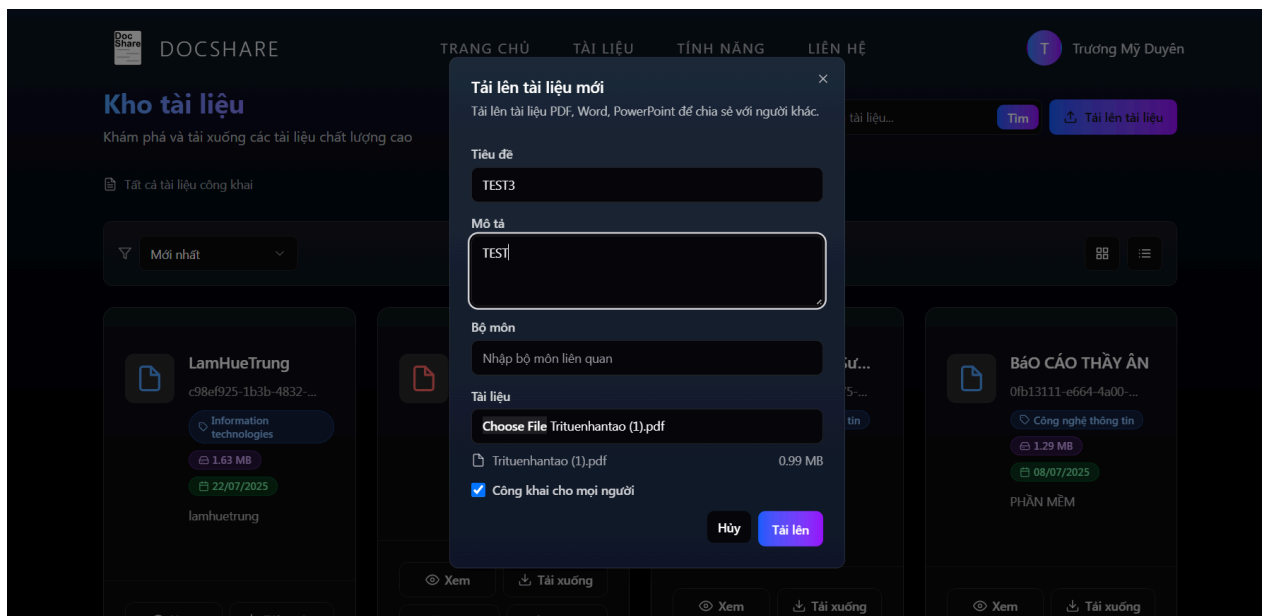
6.2.3. Chức năng tải tài liệu

Khi đăng nhập/đăng ký thành công sẽ hiện giao diện :



Hình 13: Giao diện tải tài liệu

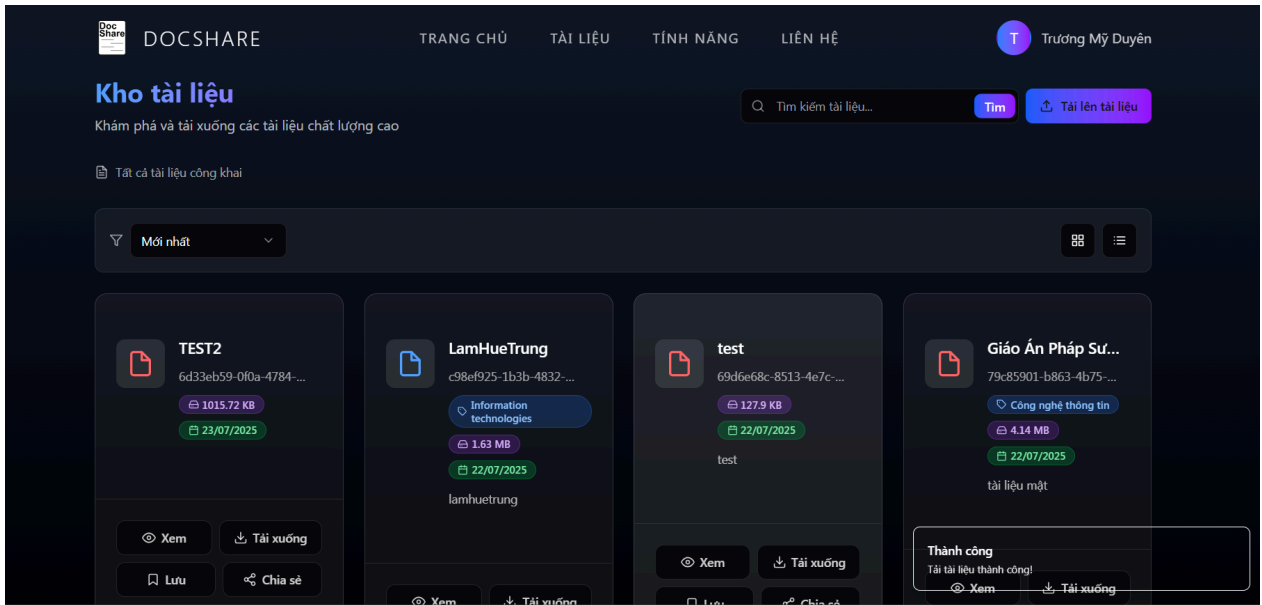
Tiếp đến người dùng bấm vào chữ “tải lên tài liệu” để tải tài liệu:



Hình 14: Giao diện tải tài liệu

Sau khi bấm vào chữ “tải lên” sẽ hiện chữ “tải tài liệu thành công” như hình bên dưới:

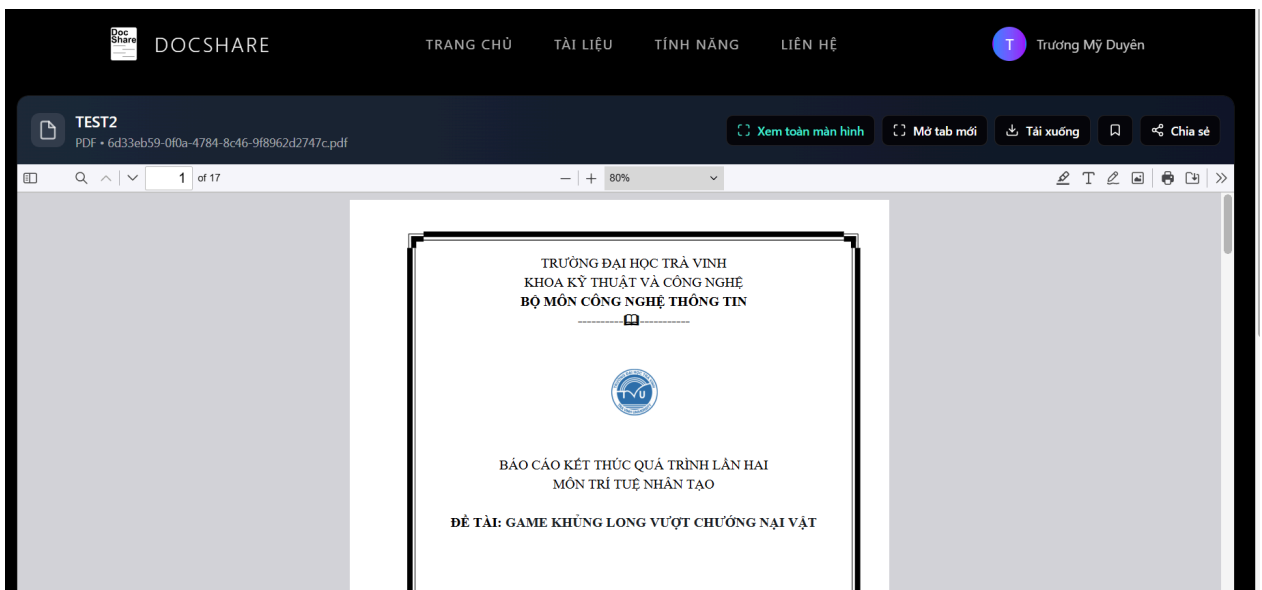
Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)



Hình 15: Giao diện tải tài liệu thành công

6.2.4. Chức năng xem tài liệu

Người dùng muốn xem tài liệu ở file nào trên web thì bấm vào chữ “xem” sau khi bấm vào chữ xem thì sẽ hiện ra giao diện bên dưới:

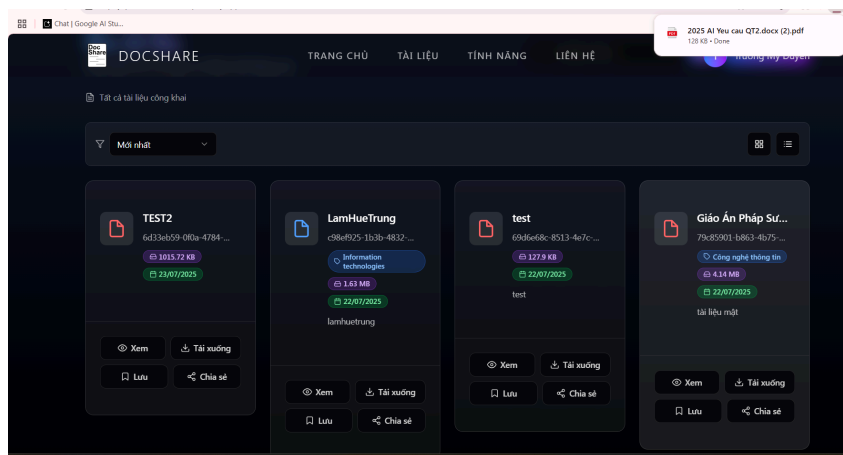


Hình 16: Giao diện xem tài liệu

Xây dựng ứng dụng chia sẻ tài liệu trực tuyến (DOCSHARE)

6.2.5. Chức năng tải xuống tài liệu

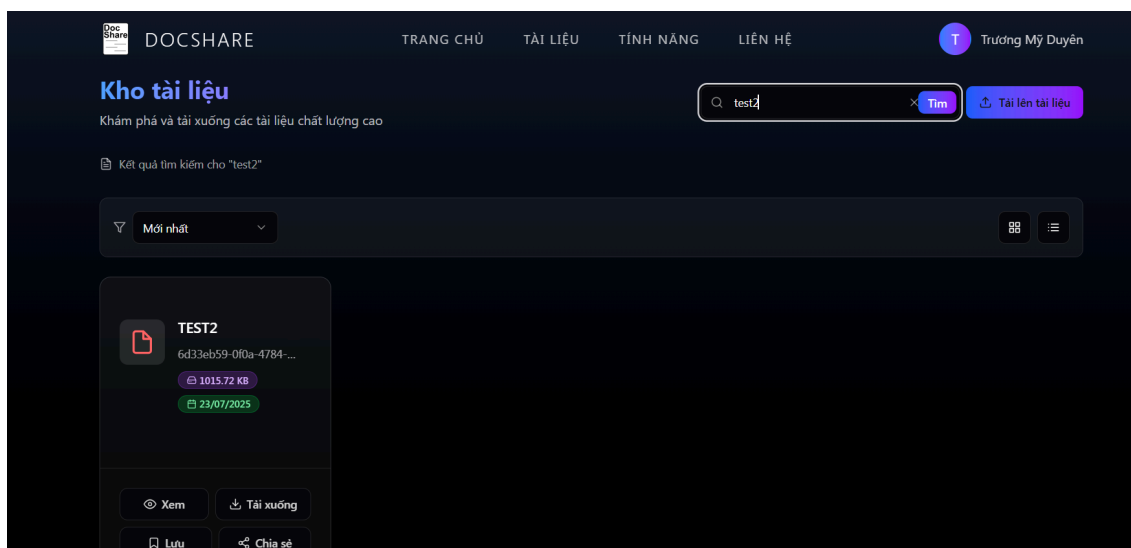
Khi bấm vào chữ “tải xuống” thì thư mục sẽ được tải về máy.



Hình 17: Giao diện xem tài liệu

6.2.6. Chức năng tìm kiếm tài liệu

Người dùng nhấn vào thanh “tìm kiếm” để nhập từ khóa cho thư mục muốn tìm .



Hình 18: Giao diện chức năng tìm kiếm

TÀI LIỆU THAM KHẢO

1. Sommerville, I. (2016). *Software Engineering* (10th ed.). Pearson Education.
2. Pressman, R. S., & Maxim, B. R. (2014). *Software Engineering: A Practitioner's Approach* (8th ed.). McGraw-Hill Education.
3. Beck, K. et al. (2001). *Manifesto for Agile Software Development*. Retrieved from <https://agilemanifesto.org/>
4. Royce, W. W. (1970). *Managing the Development of Large Software Systems*. Proceedings of IEEE WESCON.
5. VersionOne. (2020). *14th Annual State of Agile Report*. Retrieved from <https://stateofagile.com/>
6. IEEE. (2014). *IEEE Standard for Software Life Cycle Processes (IEEE 12207-2008)*. IEEE Computer Society.