QUẢN LÝ QUY TRÌNH PHẦN MỀM

SOFTWARE ARCHITECTURE

Mạng xã hội chia sẻ công thức nấu ăn

GVHD: TS. Ngô Huy Biên

SVTH: Nhóm 3 (22HCB_LT)

Bộ môn Công nghệ phần mềm

Khoa Công nghệ thông tin

Đại học Khoa Học Tự Nhiên TP HCM

MỤC LỤC

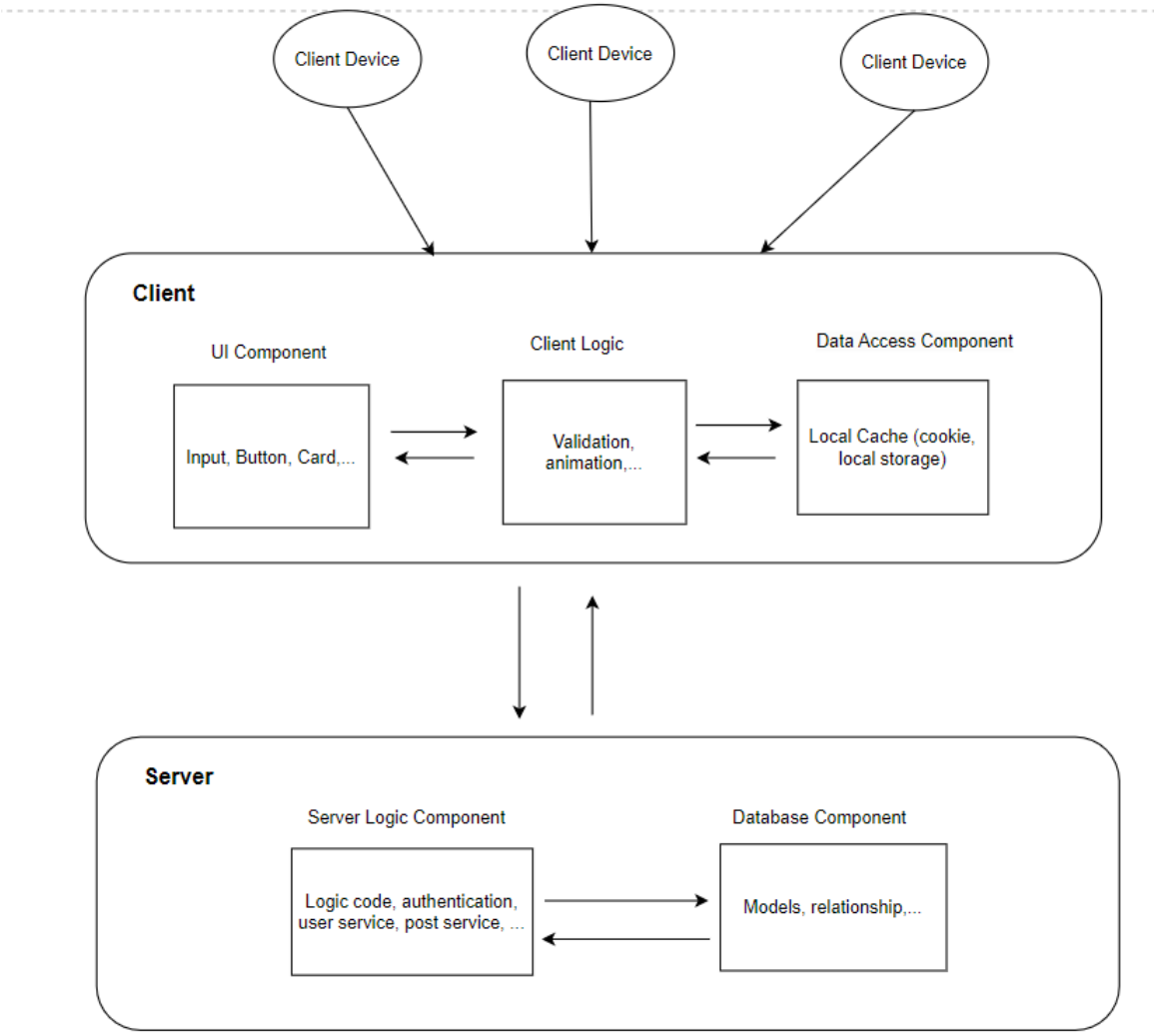
Nội dung

1 Kiến trúc tổng quan

1.1. Kiến trúc Client-Server

Dự án "Mạng xã hội chia sẻ công thức nấu ăn" được phát triển dựa trên kiến trúc Client-Server, bao gồm hai thành phần cơ bản:

- Client (máy khách):
    - Là một ứng dụng hoặc thiết bị yêu cầu dữ liệu hoặc dịch vụ từ máy chủ. Client gửi yêu cầu tới máy chủ, nhận và xử lý dữ liệu từ máy chủ, sau đó hiển thị thông tin cho người dùng. Client có thể là máy tính cá nhân, trình duyệt web, ứng dụng di động hoặc bất kỳ thiết bị nào có khả năng kết nối mạng.
- Server (máy chủ):
    - Là một hệ thống hoặc phần mềm có khả năng lắng nghe yêu cầu từ client, xử lý chúng và trả về kết quả. Server chịu trách nhiệm quản lý dữ liệu, tài nguyên và các chức năng của ứng dụng. Máy chủ có thể là máy chủ web, máy chủ cơ sở dữ liệu, máy chủ thư điện tử hoặc bất kỳ hệ thống nào cung cấp dịch vụ cho client.
- Client và Server giao tiếp với nhau qua giao thức HTTPs (Hypertext Transfer Protocol Secure)



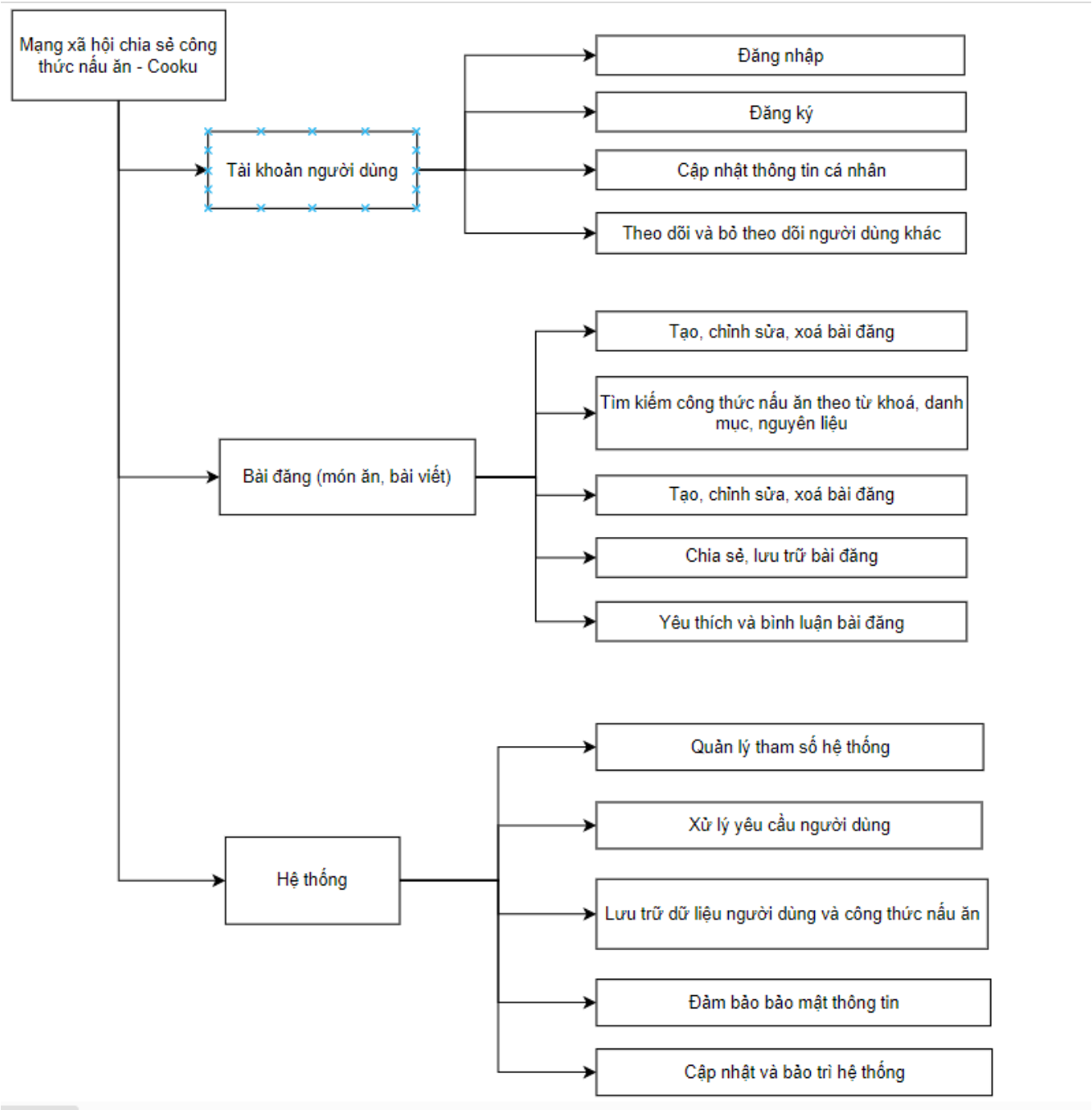
Hình trên biểu diễn kiến trúc Client-Server, bao gồm các thành phần nhỏ khác:

- Client:
  - UI Component: tương tác trực tiếp với người dùng để hiển thị giao diện và lấy đầu vào từ người dùng.
  - Client Logic Component: xử lý các yêu cầu từ UI Component và gửi yêu cầu tương ứng đến Server.
  - Data Access Component: quản lý dữ liệu tạm trên Client, bao gồm lưu trữ và truy xuất dữ liệu.

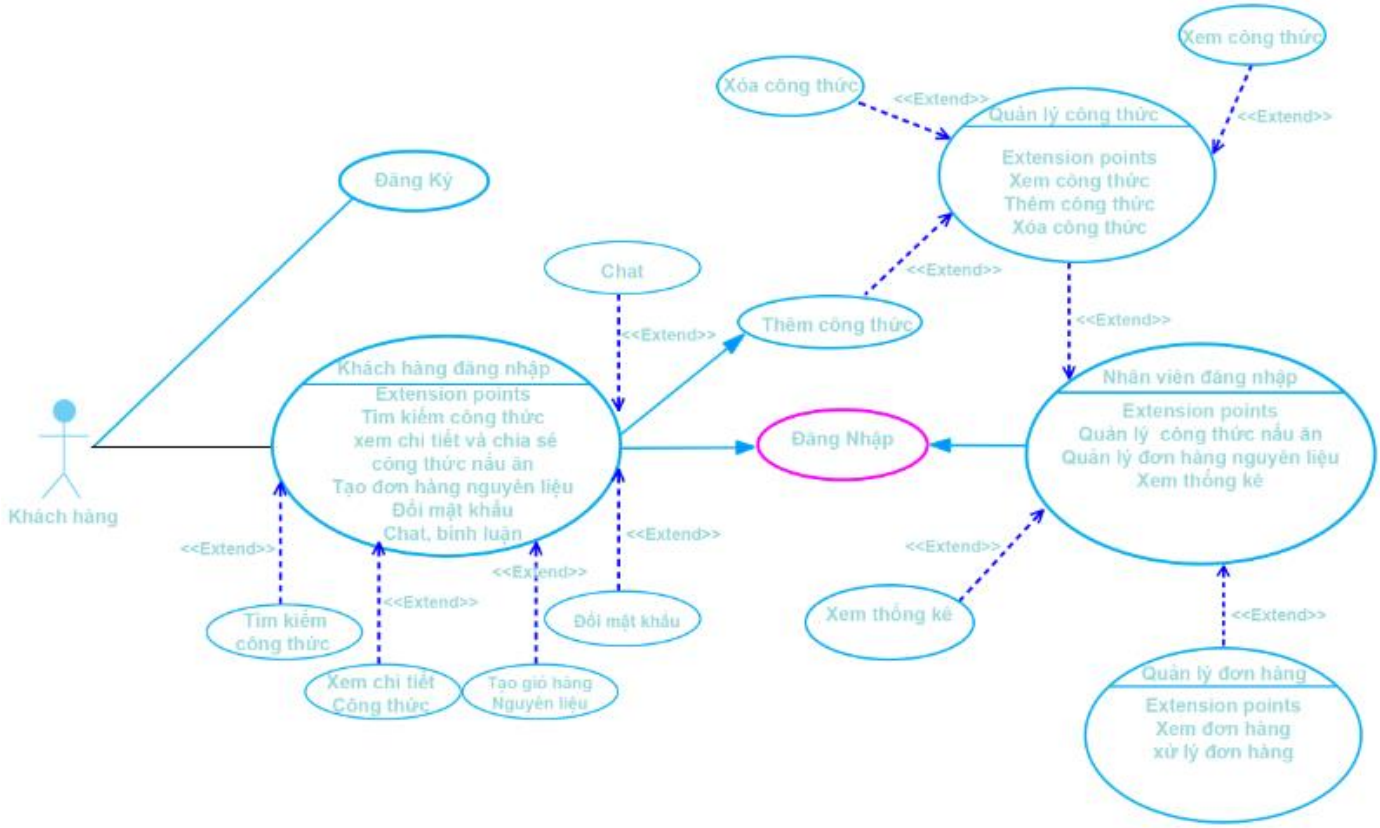
- Server:
    - Server Logic Component: xử lý các yêu cầu từ Client, truy xuất cơ sở dữ liệu và gửi phản hồi tương ứng.
    - Database Component: quản lý cơ sở dữ liệu, bao gồm lưu trữ và truy xuất dữ liệu.

1.2 Mô hình phân rã hệ thống

Thể hiện một số thành phần chính sau:



1.3 Mô hình use-case tổng quan



Use-case đăng nhập

Tên Use Case	Đăng Nhập
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng đã có khách hàng trong hệ thống
Đảm bảo tối thiểu	Hệ thống cho phép khách hàng đăng nhập lại
Đảm bảo thành công	Khách hàng đăng nhập được vào hệ thống
Chuỗi sự kiện chính: 1. Khách hàng chọn chức năng đăng nhập trên giao diện chính của hệ thống	

2. Hệ thống hiển thị form đăng nhập gồm ô tên đăng nhập, mật khẩu và nút đăng nhập 3. Khách hàng nhập tên đăng nhập và mật khẩu của mình 4. Hệ thống kiểm tra tính hợp lệ của tài khoản và mật khẩu 5. Hệ thống hiển thị giao diện chính tương ứng với các chức năng của tác nhân
Ngoại lệ: 4.1. Khách hàng nhập tài khoản hay mật khẩu không chính xác

Use-case đăng ký

Tên Use Case	Đăng ký
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng được truy cập vào trang web
Đảm bảo tối thiểu	Thông tin khách hàng đăng ký hợp lệ
Đảm bảo thành công	Đăng ký thành công và tạo tài khoản cho khách hàng thành công
Chuỗi sự kiện chính: 1. Khách hàng truy cập vào trang web 2. Khách hàng chọn chức năng đăng kí tài khoản 3. Hệ thống hiển thị form đăng kí cho khách hàng gồm các trường nhập:Tên Tài Khoản, Mật Khẩu, Sđt…. 4. Khách hàng nhập thông tin và click đăng kí 5. Hệ thống kiểm tra thông tin, lưu vào cơ sở dữ liệu và thông báo đăng kí thành công	
Ngoại lệ:	

5.1. Hệ thống thông báo Tên Tài Khoản bị trùng 5.1.1. Hệ thống yêu cầu nhập lại Tên Tài Khoản 2. khách hàng nhập lại và tiếp tục các bước sau

Use-case tạo công thức nấu ăn

Tên Use Case	Tạo công thức
Tác nhân chính	Người dùng
Tiền điều kiện	Người dùng được truy cập vào trang web
Đảm bảo tối thiểu	Cho phép thực hiện thêm 1 công thức
Đảm bảo thành công	Thêm công thức thành công
Chuỗi sự kiện chính: 1. Nhân viên đăng nhập vào hệ thống 2. Giao diện quản trị công thức hiển thị 3. Nhân viên click vào quản lý công thức 4. Giao diện quản lý công thức hiện lên danh sách công thức với các chức năng tìm kiếm, thêm,xem chi tiết ,khoá công thức 5. Nhân viên chọn thêm mới công thức 6. Giao diện hiển thị thông tin công thức cần thêm 7. Nhân viên nhập thông tin công thức và click nút tạo công thức 8. Hệ thống hiển thị công thức trong danh sách công thức	
Ngoại lệ: 7.1 Hệ thống thông báo nhập sai thông tin 7.2 Nhân viên nhập lại công thức và click nút tạo công thức 7.3 Hệ thông hiển thị công thức trong danh sách công thức	

Use-case bình luận

Tên Use Case	Bình luận
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng đã truy cập vào trang web thành công
Đảm bảo tối thiểu	Khách hàng bình luận thành công
Đảm bảo thành công	

Chuỗi sự kiện chính:

Khách hàng có thể bình luận với sản phẩm mình đã đặt hoặc chưa đặt:

1. Với sản phẩm chưa đặt

1. Khách hàng xem chi tiết 1 công thức món ăn bất kì

2. Khách hàng nhập bình luận ở form bình luận tương ứng với sản phẩm và xác nhận bình luận

3. Hệ thống kiểm tra thông tin và lưu vào cơ sở dữ liệu, đồng thời thông báo khách hàng đã bình luận thành công

4. Hệ thống hiển thị bình luận của khách hàng ở phần sản phẩm tương ứng

2. Với sản phẩm đã đặt theo đơn

1. Khách hàng chọn đơn hàng đã đặt

2. Hệ Thống hiển thị danh sách đơn hàng đã đặt của khách hàng

3. Khách hàng chọn đơn hàng muốn bình luận

4. Hệ thống hiển thị chi tiết đơn hàng tương ứng

5. Khách hàng phản hồi ý kiến bằng cách bình luận ở form bình luận tương ứng và nhấn xác nhận

6. Hệ thống kiểm tra thông tin và lưu thông tin vào cơ sở dữ liệu, trả về thông báo bình luận thành công

7. Hệ thống hiển thị bình luận của khách hàng ở phần sản phẩm tương ứng

thành công

Use-case đánh giá công thức nấu ăn

Tên Use Case	Đánh giá
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng đã đăng nhập vào hệ thống
Đảm bảo thành công	Khách hàng đánh giá công thức món ăn thành công
Chuỗi sự kiện chính: 1. Khách hàng chọn đơn hàng đã thanh toán thành công 2. Hệ thống hiển thị danh sách các đơn hàng đã được thanh toán thành công 3. Khách hàng chọn 1 đơn hàng từ danh sách 4. Hệ thống hiển thị chi tiết đơn hàng đó 5. Khách hàng chọn đánh giá sản phẩm 6. Hệ thống hiển thị các mức điểm đánh giá 7. Khách hàng chọn điểm và xác nhận lưu đánh giá của mình 8. Hệ thống kiểm tra thông tin và lưu vào cơ sở dữ liệu, trả về thông báo đánh giá thành công cho khách hàng	
Ngoại lệ: 3.1 Không tìm thấy sản phẩm 3.1.1 Hệ thống thông báo không tìm thấy sản phẩm 3.1.2 Khách hàng tìm kiếm mặt hàng khác	

Use-case tìm kiếm

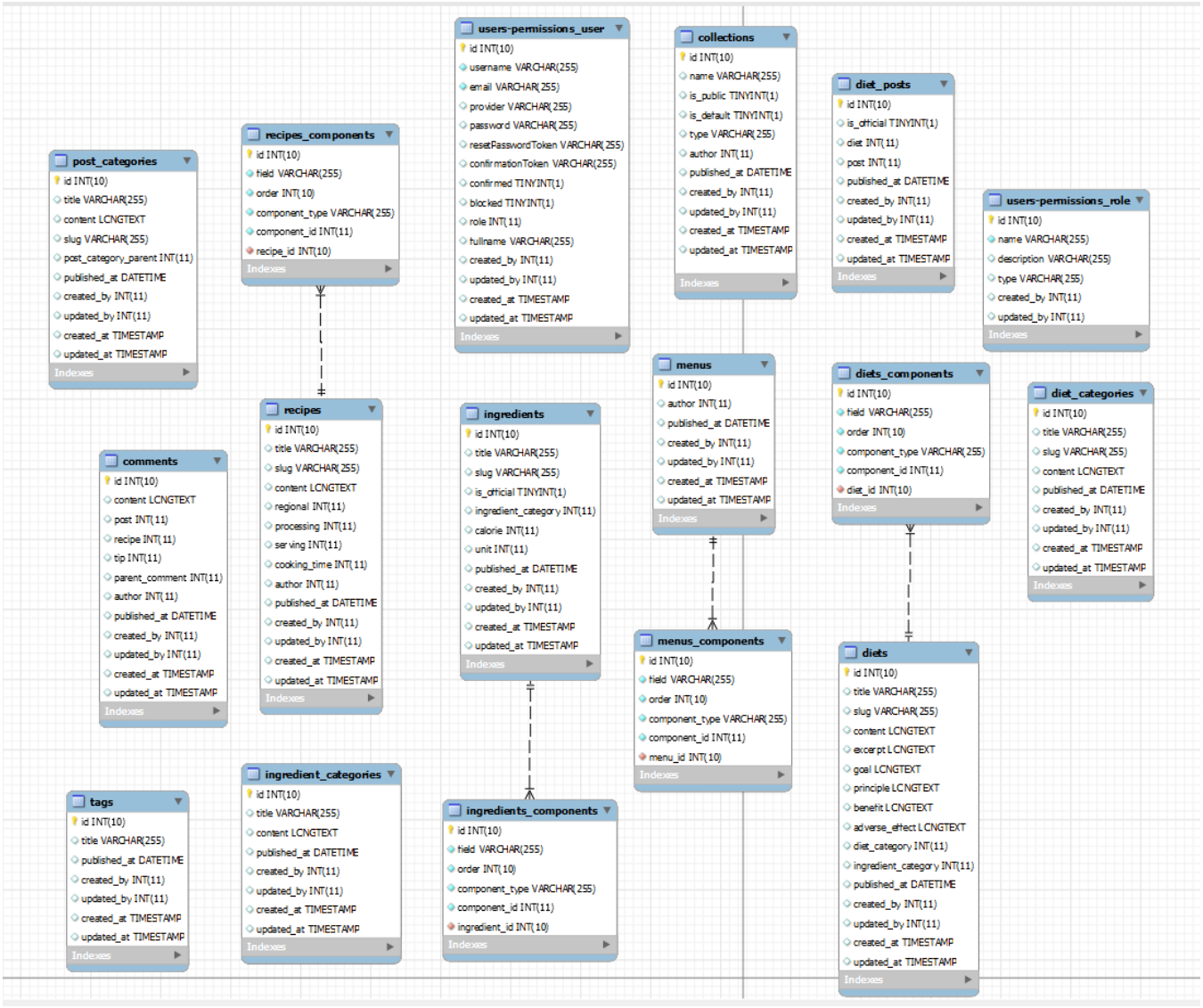
Tên Use Case	Tìm kiếm
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng đã truy cập vào trang web thành công
Đảm bảo tối thiểu	Hủy thao tác trở về trạng thái trước đó
Đảm bảo thành công	Tìm kiếm sản phẩm công thức món ăn thành công tương ứng
Chuỗi sự kiện chính: 1. Khách hàng chọn chức năng "Tìm kiếm" 2. Trên thanh tìm kiếm, khách hàng nhập tên mặt hàng cần tìm kiếm 3. Khách hàng click "Tìm kiếm" 4. Hệ thống hiển thị giao diện trả về mặt hàng mà khách hàng tìm kiếm trước đó 5. Khách hàng tiếp tục xem mặt hàng hoặc thực hiện các chức năng khác thành công	
Ngoại lệ: 3.1 Không tìm thấy sản phẩm 3.1.1 Hệ thống thông báo không tìm thấy sản phẩm 3.1.2 Khách hàng tìm kiếm mặt hàng khác	

Use-case đánh giá bài viết

Tên Use Case	Đánh giá bài viết
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng đã truy cập vào trang web thành công
Đảm bảo tối thiểu	
Đảm bảo thành công	Khách hàng đánh giá điểm bài viết/công thức
Chuỗi sự kiện chính: 1. Khách hàng chọn bài viết/công thức đã tham khảo, đã đọc thành côngthành công 2. Hệ thống hiển thị danh sách các bài viết/công thức đã được tham khảo/đọc thanh toán thành công 3. Khách hàng chọn 1 bài viết/Công thức từ danh sách 4. Hệ thống hiển thị chi tiết bài viết/công thức đó	
5. Khách hàng chọn đánh giá sản phẩm 6. Hệ thống hiển thị các mức điểm đánh giá 7. Khách hàng chọn điểm và xác nhận lưu đánh giá của mình 8. Hệ thống kiểm tra thông tin và lưu vào cơ sở dữ liệu, trả về thông báo đánh giá thành công cho khách hàng	

2 Logical view

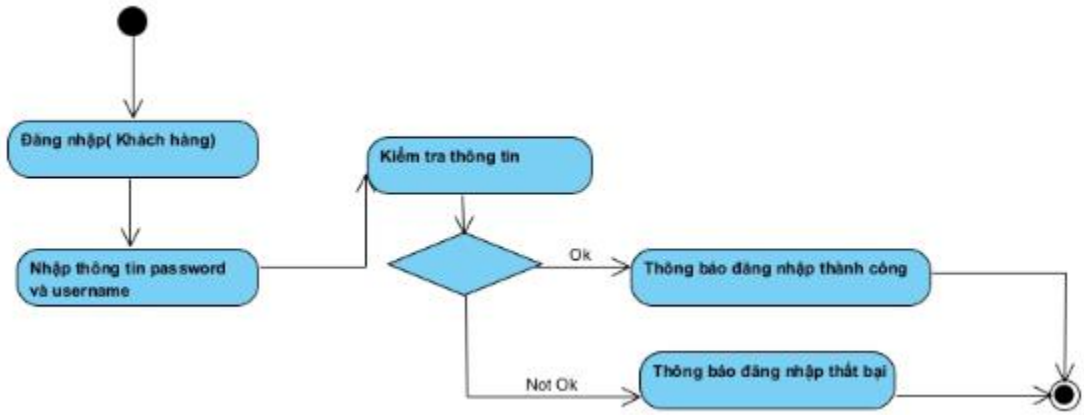
Được minh hoạ bằng Class Diagram sau:



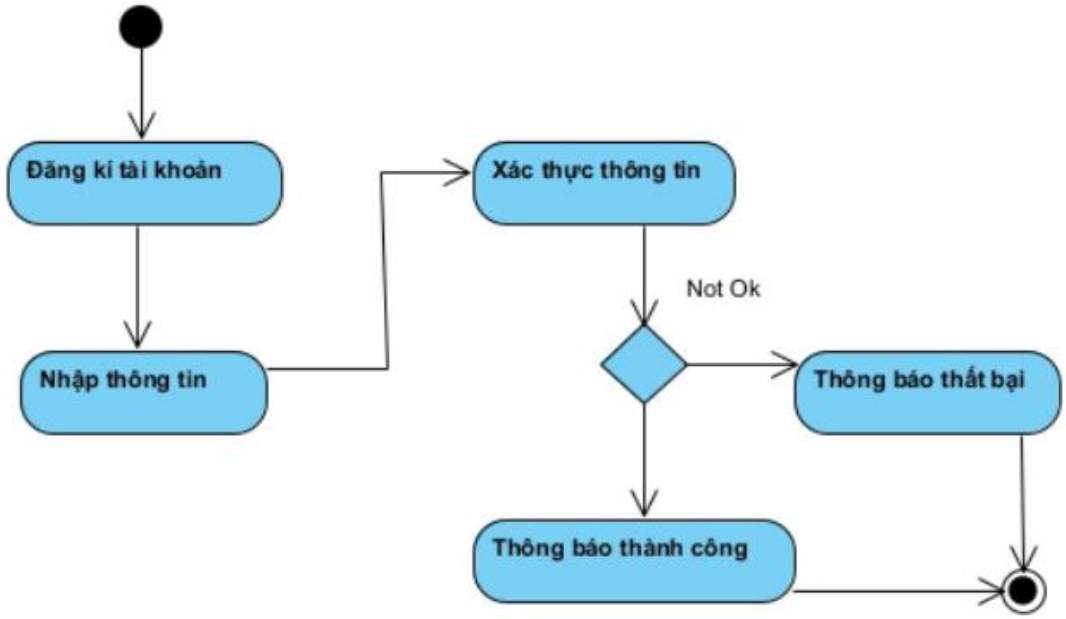
3 Process view

Các luồng thể hiện các use-case chính được mô tả bằng Activity diagram như sau:

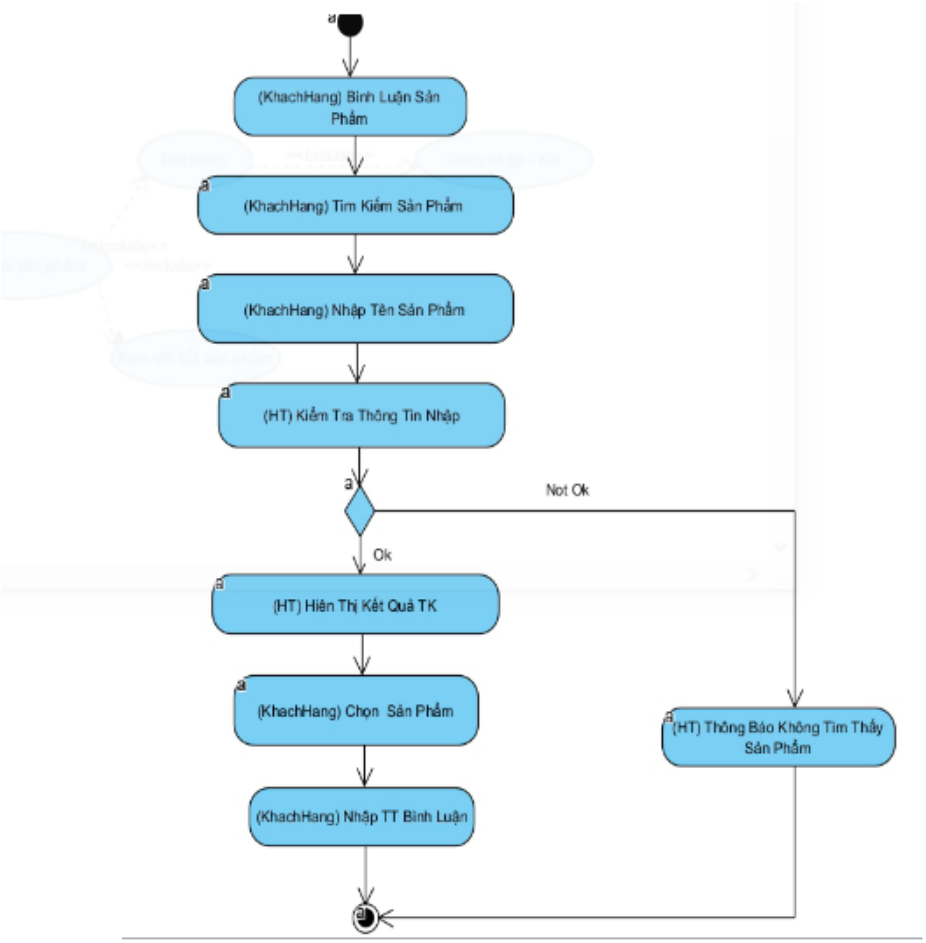
Đăng nhập:



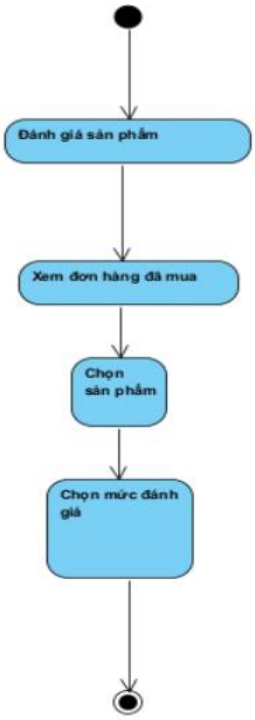
Đăng ký:



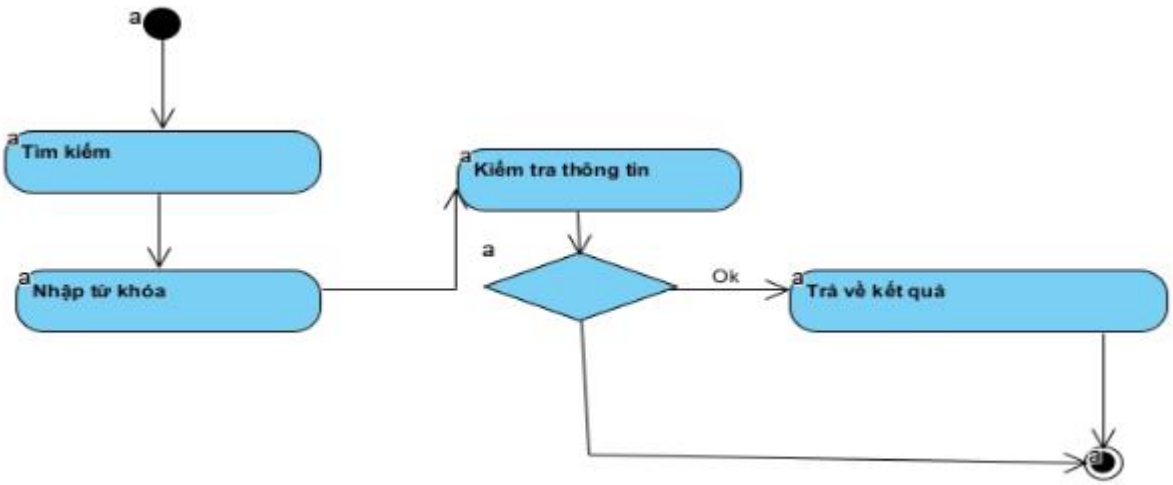
Bình luận:


(KhachHang) Bình Luận Sản Phẩm
(KhachHang) Tìm Kiếm Sản Phẩm
(KhachHang) Nhập Tên Sản Phẩm
(HT) Kiểm Tra Thông Tin Nhập
Not Ok
Ok
(HT) Hiển Thị Kết Quả TK
(KhachHang) Chọn Sản Phẩm
(KhachHang) Nhập TT Bình Luận
(HT) Thông Báo Không Tìm Thấy Sản Phẩm

Đánh giá:



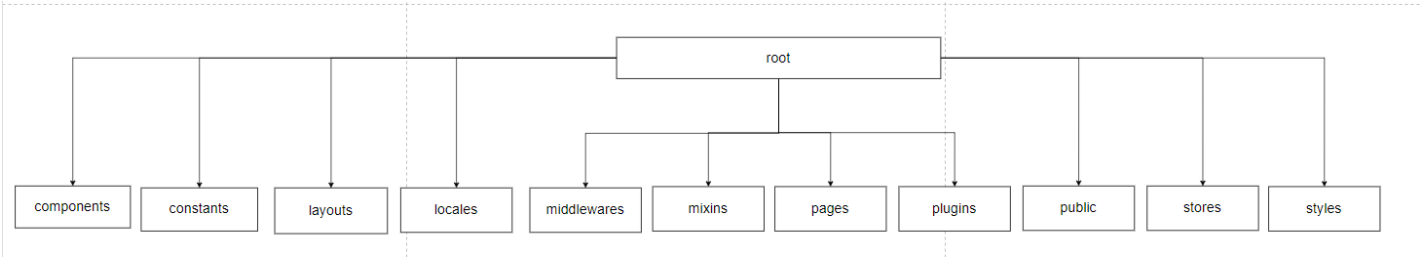
Tìm kiếm:



4 Development view (Implementation view)

Để lập trình hiển thị giao diện, xử lý tương tác phía Client, và xử lý logic và xử liệu phía Server, đội ngũ phát triển cũng tách bạch việc phát triển thành hai phần riêng biệt tương ứng là Frontend và Backend:

- **Phía Front end:**
  - Công nghệ sử dụng: HTML, CSS, Javascript, vue.js, và các thư viện phụ khác.
  - Cấu trúc thư mục mã nguồn:

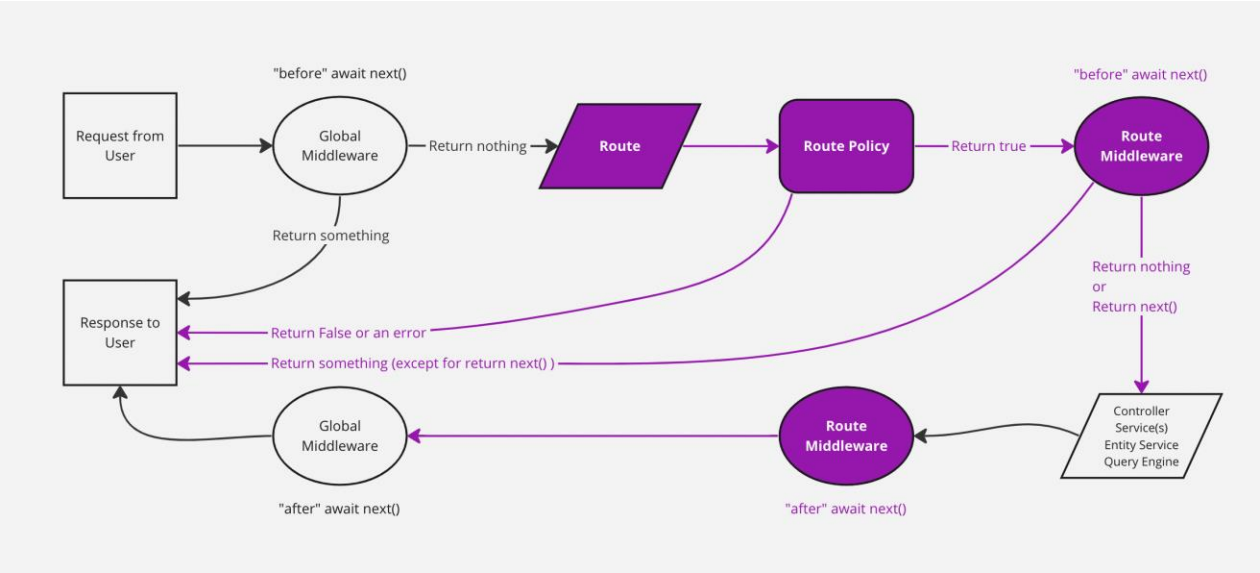


Cấu trúc thư mục mã nguồn Frontend bao gồm:

- *components*: nơi chứa toàn bộ Vue component của dự án, nuxt tự động import toàn bộ component trong thư mục components/ nên có thể sử dụng chúng mà không cần phải import lại
- *constants*: chứa các biến có thể dùng lại nhiều lần, hạn chế sử dụng các biến không xác định trong code
- *layouts*: các component layout được nuxt import tự động và sử dụng (mặc định là default) layout thường được dùng để cài đặt các thuộc tính metadata cho trang web, các thành phần các trang dùng lại nhiều lần như header footer, hoặc để phân quyền người dùng ...

- *locales*: bao gồm các cặp key - value nội dung bên trong trang web bằng các ngôn ngữ khác nhau
- *middleware*: gồm các hàm JavaScript được chạy ngay trước khi một trang được hiển thị.
- *mixins*: chứa các đoạn logic code có thể tái sử dụng nhiều lần trong file vue, tránh việc lặp lại đoạn code và maintain dễ dàng hơn
- *pages*: nuxt tự động tạo các route bất kỳ theo từng file
- *plugins*: đăng ký các biến global hoặc dùng để import global các thư viện bên ngoài
- *public*: chứa các nội dung không cần trong quá trình compile như hình ảnh, icon, robots.txt, favicon.ico
- *stores*: nơi chứa toàn bộ dữ liệu của dự án, các component vue có thể tương tác với toàn bộ dữ liệu chứa trong stores
- *styles*: bao gồm các file css giao diện ứng dụng
- *utils*: nơi chứa các hàm common hay dùng, các hàm trong đây thường không phụ thuộc vào công nghệ dự án

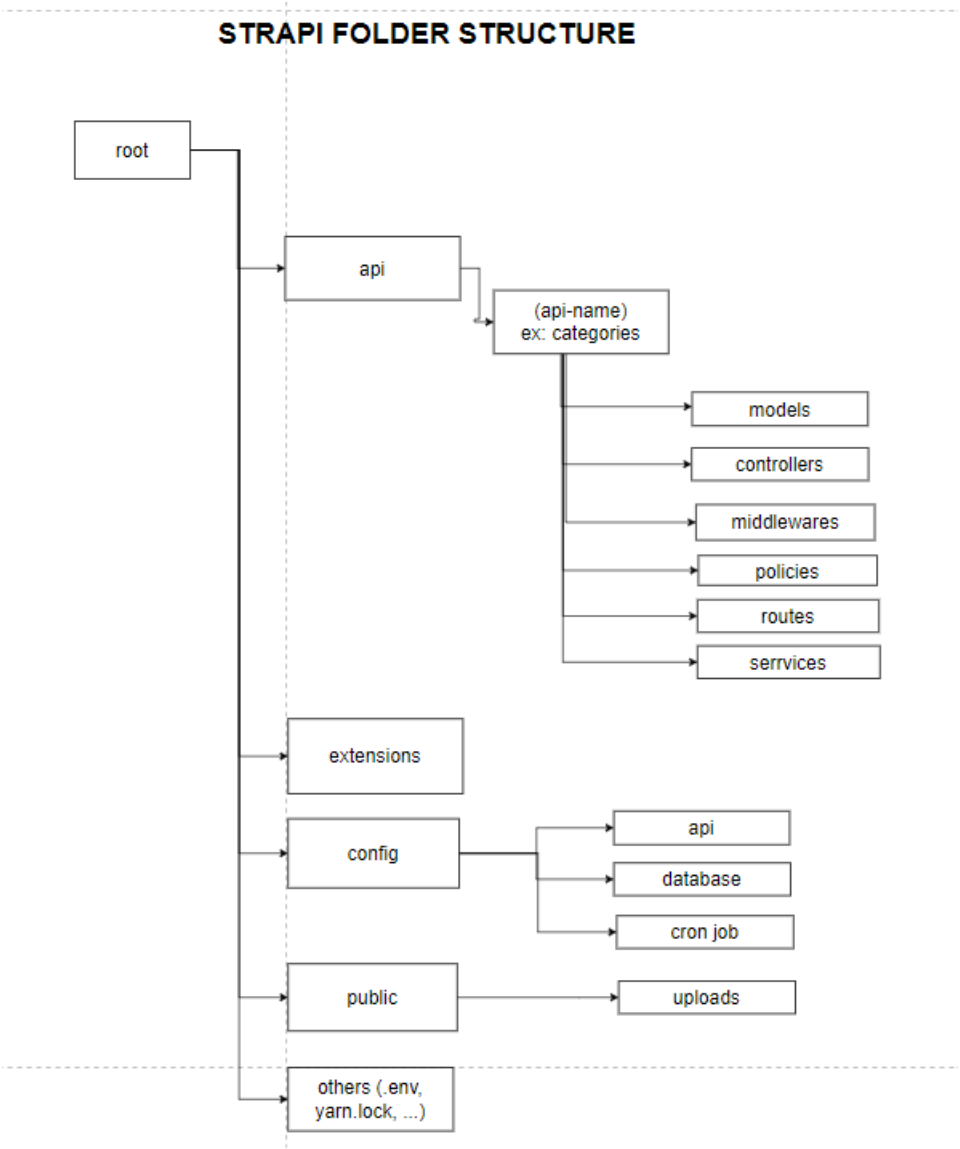
- **Phía Back end:**
    - Công nghệ sử dụng: Strapi – hệ thống quản lý nội dung mã nguồn mở (CMS), được xây dựng trên Node.js. Strapi cho phép bạn dễ dàng tạo, quản lý và triển khai các ứng dụng web và ứng dụng di động có nội dung đa phương tiện. Hai hình thức làm việc chính với Strapi là quản lý Admin Panel thông qua giao diện và cung cấp hàng loạt các APIs với nhiều tính năng như phân quyền, bảo mật, phân trang, … cho phía Frontend sử dụng.



Nguồn: https://docs.strapi.io/dev-docs/backend-customization/routes

Hình trên là biểu đồ tương tác (Interactive diagram) thể hiện luồng tương tác từ lúc người dùng thực hiện một hành động gửi request từ client lên đến server và nhận phản hồi có trong kiến trúc phía backend được xây dựng bởi Strapi.

- Cấu trúc thư mục mã nguồn:



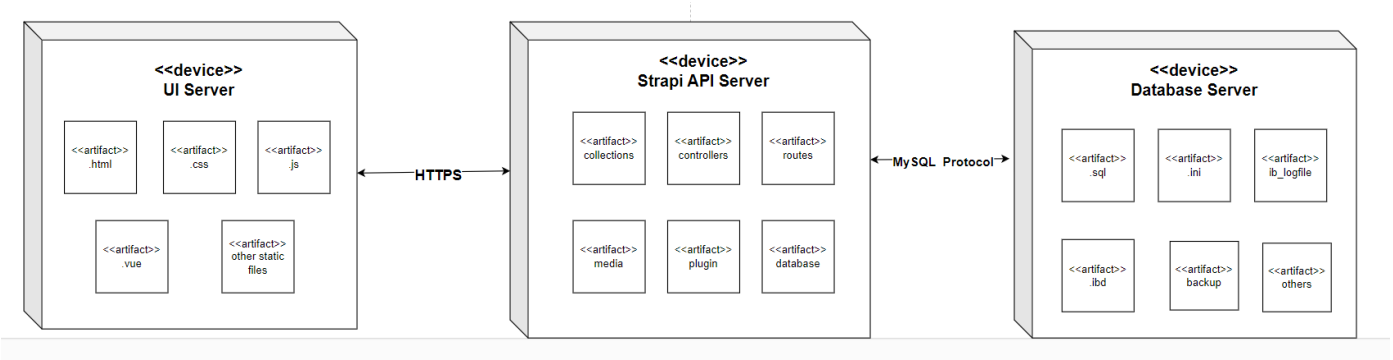
Phân tích cấu trúc thư mục cơ bản khi làm việc với Strapi, bao gồm:

- *Api*: chứa logic nghiệp vụ của dự án, được chia thành các thư mục nhỏ hơn cho mỗi API, bao gồm các thành phần sau:
  - *Models*: Chứa thông tin liên quan đến Schemas như thông tin Schema, các cách cài đặt, và ánh xạ chúng xuống cở sử dữ liệu.

- *Controllers*: chứa các file Javascript (.js) thể hiện tập hợp các phương thức, hành động xử lý logic được kích hoạt mỗi khi client gửi request đến, sau đó phản hồi kết quả lại cho client.
- *Middlewares*: chứa các file xử lý trung gian các request hoặc đính kèm xử lý từ các response, bao gồm nhiều hành động như xác thực dữ liệu, bảo mật, xử lý đầu vào, định dạng dữ liệu trả về, …
- *Policies*: chứa những hàm thực thi các logic cụ thể nào đó trước khi request đó đến được controller, thường thì là các logic liên quan đến nghiệp vụ bảo mật.
- *Routes*: định rõ request nào tương ứng với route nào và chỉ đơn giản là chứa những file xử lý tương tự với từng route
- *Services*: bao gồm nhiều hàm có tính tái sử dụng, được gọi và sử dụng ở nhiều controllers khác nhau

- *Extensions*: Chứa các plugin được cài đặt vào thêm, bao gồm các package từ npm.
- *Config:* chứa cấu hình của API hay các database client khác nhau, cách tương tác và phản hồi của API hay tương tác với cơ sở dữ liệu. Cấu hình làm việc tự động với cron job và nhiều cấu hình khác.
- *Public:* Chứa những nội dung tĩnh bên ngoài có thể truy cập được mà không cần phải xác thực hoặc uỷ quyền.
- *Others:* các file khác như .env, .gitignore, yarn.lock, ….

5 Deployment view (Physical view)



- Hình trên minh hoạ deployment view bằng deployment diagram. Bao gồm 3 nodes, mỗi node là một server, được triển khai trên môi trường cloud, sử dụng dịch vụ của nhà cung cấp Microsoft Azure
    - UI Server
        - Là máy chủ để giao diện có thể chạy khi được xây dựng bởi các file chứa mã nguồn như HTML, CSS, vue.js, ….
        - Giao tiếp với API Server thông qua giao thức HTTPs
        - Cấu hình:
            - Instance: A4v2
            - Cores: 4
            - RAM: 8.00 GB
            - Temporary storage: 40 GB
    - Strapi API Server
        - Là máy chủ để nhận các request từ phía client, xử lý, tính toán, truy vấn dữ liệu (nếu có) từ database và phản hồi về cho client

        - Tương tự, phản hồi các request của client thông qua giao thức HTTPs
        - Truy vấn và tương tác với máy chủ cơ sở dữ liệu thông qua các giao thức MySQL cung cấp
        - Cấu hình:
            - Instance: A4mv2
            - Cores: 4
            - RAM: 32.00 GB
            - Temporary storage: 40 GB
    - Database Server
        - Là máy chủ để lưu trữ cơ sở dữ liệu của hệ thống
        - Cấu hình:
            - Instance: A4mv2
            - Cores: 4
            - RAM: 32.00 GB
            - Temporary storage: 40 GB
- Nhờ sử dụng cloud, ta có thể mở rộng nhanh chóng, chi phí ban đầu thấp và có nhiều sự lựa chọn khác nhau. Tuỳ vào nhu cầu trong quá trình phát triển mà ta sẽ mở rộng thêm các server để xử lý các dịch vụ khác, caching, bảo mật, …