

**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

--------------------------



**BÁO CÁO GIỮA KÌ**

**Phân tích, thiết kế và đảm bảo chất lượng phần mềm**

**Đề tài: “XÂY DỰNG TRANG MẠNG XÃ HỘI CHIA SẺ CÔNG THỨC NẤU ĂN”**

**Giảng viên : ThS. Nguyễn Anh Hào**

**Môn : Phân tích, thiết kế và đảm bảo chất lượng phần mềm**

**Thành viên : Vũ Việt Trường – N19DCCN220**

**Vi Minh Khôi – N19DCCN095**

**Phan Thanh Trụ - N19DCCN21**

**Nhóm : 54**

**Lớp : D19CQCNPM02-N**

**TP. HCM, tháng 07/2023**

**MỤC LỤC**

[**CHƯƠNG 1: GIỚI THIỆU 1**](#_heading=h.30j0zll)

[**1.1.**](#_heading=h.1fob9te) **Mục đích 1**

[**1.2.**](#_heading=h.3znysh7) **Mục tiêu 1**

[**1.3.**](#_heading=h.2et92p0) **Phương pháp tiến hành 1**

[**CHƯƠNG 2: CƠ SỞ KHOA HỌC CỦA ĐỀ TÀI 3**](#_heading=h.tyjcwt)

[**1.1.**](#_heading=h.3dy6vkm) **Giới thiệu về công nghệ sử dụng 3**

[1.1.1. ReactJs là gì? 3](#_heading=h.1t3h5sf)

[1.1.2. TypeScript là gì? 4](#_heading=h.17dp8vu)

[1.1.3. NodeJS là gì? 5](#_heading=h.3rdcrjn)

[1.1.4. ExpressJS là gì? 5](#_heading=h.26in1rg)

[1.1.5. PostgreSQL là gì? 6](#_heading=h.lnxbz9)

[1.1.6. Giới thiệu về Visual studio Code 6](#_heading=h.35nkun2)

[**1.1.7. Một số thư viện, package khác** 7](#_heading=h.1ksv4uv)

[**1.2.**](#_heading=h.44sinio) **Giới thiệu về kiến trúc 7**

[1.2.1. Kiến trúc hướng dịch vụ (SOA) 7](#_heading=h.2jxsxqh)

[1.2.2. Giới thiệu về RESTful API 8](#_heading=h.3j2qqm3)

[**CHƯƠNG 3: Phân tích thiết kế hệ thống 10**](#_heading=h.4i7ojhp)

[**3.1. Bối cảnh/ hiện trạng của hệ thống 10**](#_heading=h.2xcytpi)

[a. Định nghĩa vấn đề (mục tiêu) mà đề tài sẽ giải quyết 10](#_heading=h.1ci93xb)

[**b. Hiện trạng trước khi sử dụng phần mềm** 10](#_heading=h.3whwml4)

[**c. Đề xuất giải pháp đề tài** 11](#_heading=h.2bn6wsx)

[**3.2. Định nghĩa các tương tác trên phần mềm 12**](#_heading=h.qsh70q)

[**a. Xem công thức** 12](#_heading=h.3as4poj)

[**b. Quản lý công thức** 13](#_heading=h.1pxezwc)

[**c. Quản lý trang cá nhân** 13](#_heading=h.49x2ik5)

[**d. Quản lý tài khoản** 14](#_heading=h.2p2csry)

[**3.3. Định nghĩa yêu cầu và ràng buộc đối với phần mềm 15**](#_heading=h.147n2zr)

[a. Yêu cầu từ môi trường nghiệp vụ 15](#_heading=h.3o7alnk)

[b. Yêu cầu từ môi trường vận hành 17](#_heading=h.23ckvvd)

[c. Yêu cầu từ môi trường phát triển 18](#_heading=h.ihv636)

**DANH MỤC CÁC KÝ HIỆU VÀ CHỮ VIẾT TẮT**

| **Chữ viết tắt** | **Từ** | **Ý nghĩa** |
| --- | --- | --- |
| API | Application programming Interface | Nó cho phép kết nối và trao đổi dữ liệu giữa hai hệ thống phần mềm riêng biệt |
| SPA | Single page application |  |
| ERD | Entity Relationship Diagram | Mô hình thực thể kết hợp |
| HTML | Hyper text markup language | Ngôn ngữ đánh dấu siêu văn bản |

# CHƯƠNG 1: GIỚI THIỆU

Việc ăn uống là nhu cầu không thể thiếu đối với mỗi chúng ta. Trong thời đại hiện nay, ngoài việc tự do mua những món ăn ngoài hàng quán, xem công thức nấu ăn từ sách dạy, ta có thể dễ dàng trao đổi, chia sẻ cho nhau những công thức, mẹo vặt nấu nướng dễ hơn bao giờ hết. Đó là nhờ sự phát triển của Internet.

Tuy nhiên hiện nay, hàng loạt các công thức được đưa ra trên các trang web nhưng lại khó đánh giá được công thức này có ngon không, hay thậm chí chỉ là những công thức câu tương tác, chưa từng qua thử nghiệm.

Ứng dụng Food Blog được phát triển để người dùng có thể học hỏi hoặc chia sẻ công thức của mình với mọi người, từ đó có thể nhận được đánh giá của mọi người về công thức của mình, từ đó có thể tăng khả năng nấu nướng. Bên cạnh đó, ứng dụng cũng là nơi lưu trữ công thức mình yêu thích, giúp người dùng có thể tìm lại những công thức mình muốn một cách nhanh nhất, không cần phải ghi chép ra sổ tay hay giấy note, những thứ mà dễ mất mát, hư hỏng.

* 1. **Mục đích**

Mục đích chính của đề tài là thiết kế một trang web blog chia sẻ công thức nấu ăn nhằm tạo ra một không gian trực tuyến hấp dẫn và hữu ích cho người đọc yêu thích nấu ăn. Trang web này sẽ cung cấp cho người dùng một nơi để tìm kiếm và chia sẻ các công thức nấu ăn độc đáo, ngon miệng, đơn giản và dễ thực hiện. Mục tiêu là tạo ra một cộng đồng nấu ăn trực tuyến, kết nối những người đam mê ẩm thực và tạo ra trải nghiệm đọc trực tuyến thú vị và giáo dục.

* 1. **Mục tiêu**

1. Nâng Cao Trải Nghiệm Người Dùng:

* Thiết kế giao diện trực quan và dễ sử dụng để tăng cường trải nghiệm người dùng.
* Hỗ trợ tìm kiếm linh hoạt để người dùng dễ dàng tìm thấy các công thức nấu ăn theo loại, nguyên liệu, hoặc khối lượng thời gian chuẩn bị.

1. Nâng Cao Chất Lượng Nội Dung:

* Khuyến khích người dùng chia sẻ các công thức nấu ăn cá nhân, đảm bảo chất lượng thông qua việc duyệt và xác nhận nội dung trước khi đăng tải.
* Hỗ trợ viết và hiển thị các bài viết hấp dẫn, đồng thời kèm theo hình ảnh và video hướng dẫn chi tiết.

1. Tạo Cộng Đồng:

* Tạo diễn đàn hoặc các tính năng xã hội để người dùng có thể thảo luận, đánh giá và chia sẻ ý kiến với nhau.
* Tổ chức các sự kiện, thử thách nấu ăn để kích thích tương tác và sự tham gia của cộng đồng.
  1. **Phương pháp tiến hành**

1. Phân Tích:

* Tiến hành nghiên cứu thị trường để hiểu nhu cầu và mong muốn của người dùng về các trang web chia sẻ công thức nấu ăn.
* Phân loại các người dùng mục tiêu (ví dụ: người mới học nấu ăn, người đam mê ẩm thực chuyên nghiệp) để hiểu rõ nhu cầu và mong muốn của từng đối tượng.

1. Thiết Kế Theo Hướng Đối Tượng:

* Xây dựng các biểu đồ use case để hiểu quy trình hoạt động của trang web từ góc độ người dùng.
* Thiết kế các lớp đối tượng như Người Dùng, Công Thức Nấu Ăn, Bình Luận, v.v., để tổ chức thông tin một cách hợp lý.
* Sử dụng các nguyên tắc thiết kế hướng đối tượng như kế thừa, đa hình và đóng gói để xây dựng một cấu trúc dễ bảo trì và mở rộng.

1. Phát Triển và Kiểm Thử:

* Sử dụng các công cụ và ngôn ngữ lập trình phù hợp để phát triển trang web, ví dụ như HTML, CSS, JavaScript, và các framework như React hoặc Angular.
* Tiến hành kiểm thử chất lượng để đảm bảo rằng trang web hoạt động một cách mượt mà, đáng tin cậy trên các thiết bị và trình duyệt khác nhau.
* Liên tục thu thập phản hồi từ người dùng và cải thiện trang web dựa trên những phản hồi này.

1. Triển Khai và Duy Trì:

* Triển khai trang web trên một máy chủ web hoặc nền tảng đám mây để trang web có thể truy cập trên internet.
* Duy trì trang web bằng cách cập nhật nội dung, sửa lỗi và bổ sung tính năng mới dựa trên phản hồi của người dùng và xu hướng thị trường mới.

# CHƯƠNG 2: CƠ SỞ KHOA HỌC CỦA ĐỀ TÀI

## Giới thiệu về công nghệ sử dụng

### 1.1.1. ReactJs là gì?

ReactJS là một thư viện JavaScript phổ biến và mạnh mẽ được sử dụng rộng rãi trong việc phát triển giao diện người dùng động cho ứng dụng web. Nó được tạo ra bởi Facebook và có tính năng tái sử dụng thành phần, hiệu suất cao và khả năng xây dựng ứng dụng đơn trang (Single page application - SPA) để tạo ra các ứng dụng web động mà không cần tải lại trang.

ReactJS sử dụng cú pháp JSX (JavaScript XML) để xây dựng các thành phần giao diện. JSX kết hợp cú pháp JavaScript và HTML (Hyper text markup language), cho phép lập trình viên viết mã gọn gàng và dễ đọc. Cấu trúc ReactJS dựa trên mô hình khối xây dựng thành phần, trong đó mỗi thành phần độc lập và có thể được sử dụng lại trong ứng dụng.

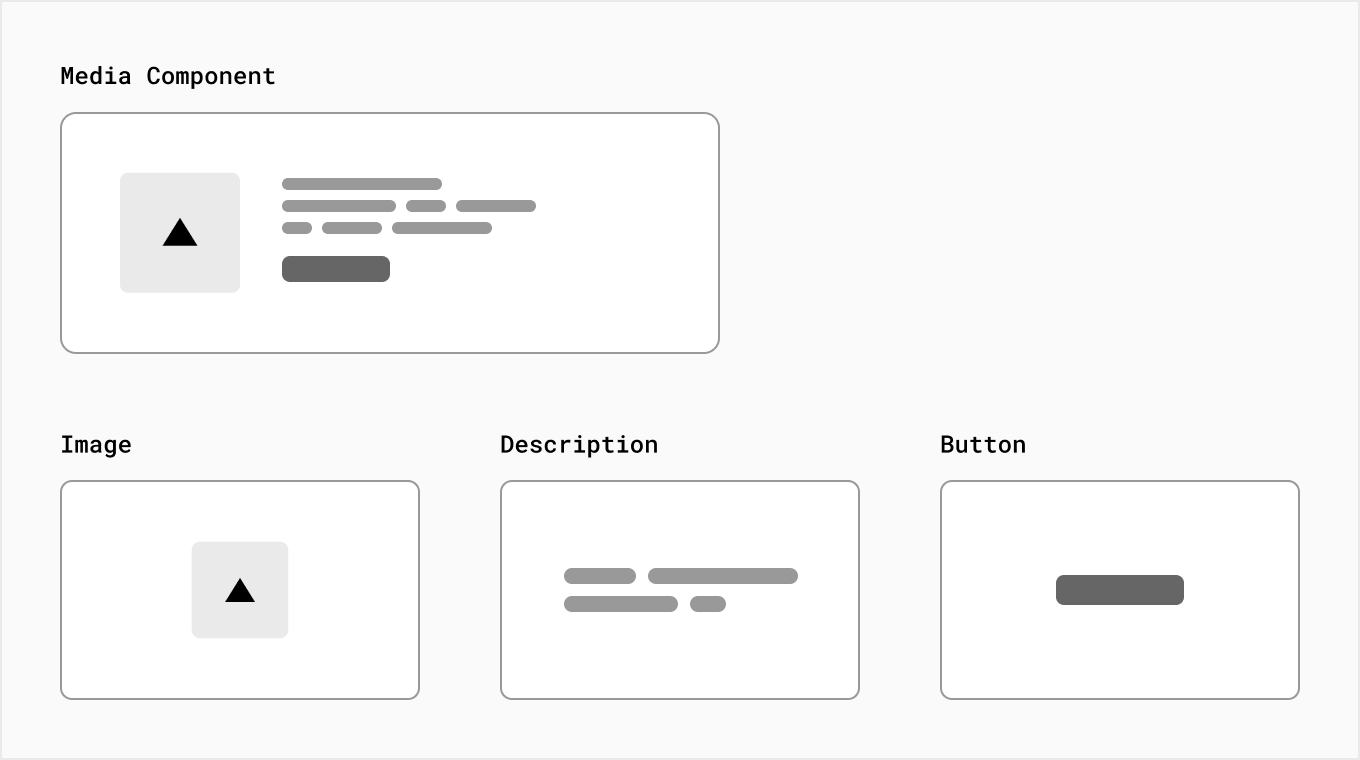
Có 3 khái niệm cốt lõi của ReactJS cần nắm được khi muốn xây dựng một ứng dụng ReactJS:

State

Props

Component

Component đại diện cho một phần giao diện người dùng độc lập và có thể được tái sử dụng trong ứng dụng. Component là một khái niệm quan trọng trong React, giúp tạo ra giao diện linh hoạt và dễ dàng quản lý

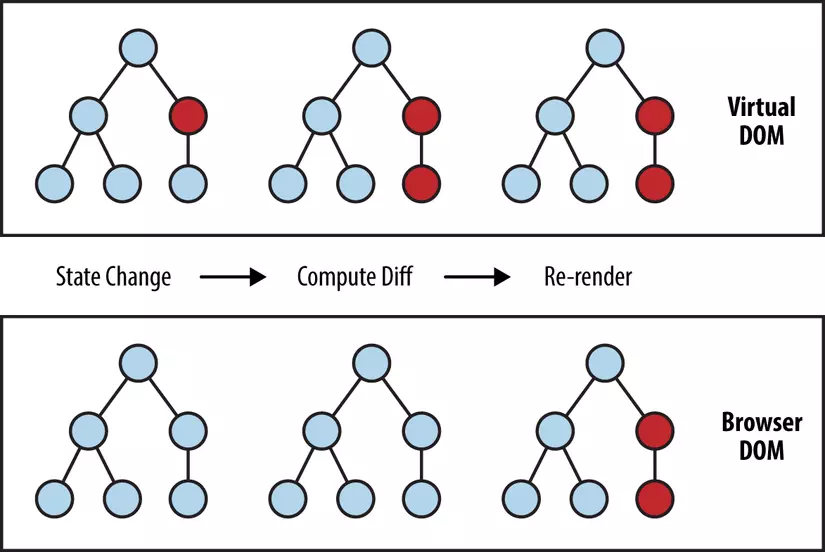


Hình 1. 1. Ví dụ về các component

Props (viết tắt của properties) là một cách để truyền dữ liệu từ component cha đến component con thông qua các thuộc tính. Props là một đối tượng không thay đổi.

State là một đối tượng chứa các dữ liệu nội bộ của một component. State được sử dụng để lưu trữ và quản lý các thông tin có thể thay đổi trong quá trình hoạt động của component. Khi state thay đổi, React tự động cập nhật lại giao diện của component đó, giúp đảm bảo tính tương tác và đồng bộ trong ứng dụng React.

ReactJS có hiệu suất cao nhờ vào cách làm việc hiệu quả của nó. ReactJS sử dụng Virtual DOM (Document Object Model) để quản lý cập nhật giao diện. Khi có sự thay đổi trong trạng thái của ứng dụng, ReactJS sẽ tạo ra một phiên bản Virtual DOM mới và so sánh nó với phiên bản cũ để xác định những thay đổi cần được áp dụng vào DOM thực tế. Quá trình này giúp giảm tối đa việc truy cập và cập nhật trực tiếp vào DOM, từ đó tăng hiệu suất ứng dụng.



Hình 1. 2. Quá trình so sánh và cập nhật của ReactJS

### 1.1.2. TypeScript là gì?

TypeScript là một dự án mã nguồn mở được phát triển bởi Microsoft, nó có thể được coi là một phiên bản nâng cao của JavaScript bởi việc bổ sung tùy chọn kiểu tĩnh và lớp hướng đối tượng mà điều này không có ở JavaScript. TypeScript có thể sử dụng để phát triển các ứng dụng chạy ở client-side (Angular, ReactJs) và server-side (NodeJs).

**1.1.2.1. Tại sao nên sử dụng TypeScript?**

Dễ phát triển dự án lớn: Với việc sử dụng các kỹ thuật mới nhất và lập trình hướng đối tượng nên TypeScript giúp chúng ta phát triển các dự án lớn một cách dễ dàng.

Nhiều Framework lựa chọn: Hiện nay các JavaScript Framework đã dần khuyến khích sử dụng TypeScript.

Hỗ trợ các tính năng của JavaScript phiên bản mới nhất: TypeScript luôn đảm bảo việc sử dụng đầy đủ các kỹ thuật mới nhất của JavaScript.

Là mã nguồn mở: TypeScript là một mã nguồn mở nên có thể sử dụng mà không mất phí, bên cạnh đó còn được cộng đồng hỗ trợ.

TypeScript và JavaScript: Bản chấn của TypeScript là biên dịch tạo ra các đoạn mã JavaScript nên có thể chạy ở bất kì đâu miễn ở đó có hỗ trợ biên dịch JavaScript.

### 1.1.3. NodeJS là gì?

NodeJs là một môi trường runtime chạy JavaScript đa nền tảng và có mã nguồn mở, được sử dụng để chạy các ứng dụng web bên ngoài trình duyệt của client. Nền tảng này được phát triển bởi Ryan Dahl vào năm 2009, được xem là một giải pháp hoàn hảo cho các ứng dụng sử dụng nhiều dữ liệu nhờ mô hình hướng sự kiện không đồng bộ.

Ưu điểm của NodeJS

* IO hướng sự kiện không đồng bộ, cho phép xử lý nhiều yêu cầu đồng thời.
* Sử dụng JavaScript – một ngôn ngữ lập trình dễ học.
* Chia sẻ cùng code ở cả phía client và server.
* NPM (Node Package Manager): và module Node đang ngày càng phát triển.
* Cộng đồng hộ trợ tích cực.
* Cho phép stream các file có kích thước lớn.

Nhược điểm của NodeJS

* Không có khả năng mở rộng, vì vậy không thể tận dụng lợi thế mô hình đa lõi trong các phần cứng cấp server hiện nay.
* Mỗi callback sẽ đi kèm với rất nhiều callback lồng nhau khác.
* Không phù hợp với các tác vụ đòi hỏi nhiều CPU.

### 1.1.4. ExpressJS là gì?

ExpressJS là một framework được xây dựng trên nền tảng của NodeJS. Nó cung cấp các tính năng mạnh mẽ để phát triển web hoặc mobile. ExpressJS hỗ trợ các method HTTP (Hyper Text Transfer Protocol) và middleware tạo ra API (Application programming Interface) vô cùng mạnh mẽ là dễ sử dụng.

Một số chức năng chính của ExpressJS như sau:

* + Thiết lập các lớp trung gian để trả về các HTTP request.
  + Định nghĩa các router cho phép sử dụng với các hành động khác nhau dựa trên phương thức HTTP và URL (Uniform Resource Locator).
  + Cho phép trả về các trang HTML dựa vào các tham số.

### 1.1.5. PostgreSQL là gì?

PostgreSQL là một hệ quản trị cơ sở dữ liệu quan hệ và đối tượng miễn phí và mã nguồn mở tiên tiến nhất hiện nay. Khả năng mở rộng cao và tuân thủ các tiêu chuẩn kỹ thuật. Nó được thiết kế để xử lý một loạt các khối lượng công việc lớn, từ các máy tính cá nhân đến kho dữ liệu hoặc dịch vụ web có nhiều người dùng đồng thời

PostgreSQL tích hợp nhiều tính năng tuyệt vời giúp hỗ trợ nhà phát triển xây dựng phần mềm đáp ứng các chức năng phức tạp, truy vấn nhanh chóng và bảo mật duy trì tính toàn vẹn và độ tin cậy. Để đáng tin cậy hơn, PostgreSQL cung cấp các tùy chọn bảo mật, xác thực và khôi phục thảm họa khác nhau. PostgreSQL được chứng minh là có khả năng mở rộng cao trả về số lượng dữ liệu và số lượng người dùng có thể thao tác cùng lúc.

Một số tính năng nổi bật:

* Câu truy vấn phức hợp (complex query).
* Thủ tục sự kiện (trigger).
* Các khung nhìn (view).
* Tính toàn vẹn của các giao dịch (integrity transactions).
* Việc kiểm tra truy cập đồng thời đa phiên bản (multiversion concurrency control).
* Truy vấn xử lý song song (parallel query).
* Sao chép dữ liệu dạng luồng (Streaming replication).

### 1.1.6. Giới thiệu về Visual studio Code

Visual Studio Code (VS Code hay VSC) là một trong những trình soạn thảo mã nguồn phổ biến nhất được sử dụng bởi các lập trình viên. Nhanh, nhẹ, hỗ trợ đa nền tảng, nhiều tính năng và là mã nguồn mở chính là những ưu điểm vượt trội khiến VS Code ngày càng được ứng dụng rộng rãi.

Visual Studio Code hỗ trợ chức năng debug, đi kèm với Git, có syntax highlighting, tự hoàn thành mã thông minh, snippets, và cải tiến mã nguồn. Nhờ tính năng tùy chỉnh, Visual Studio Code cũng cho phép người dùng thay đổi theme, phím tắt, và các tùy chọn khác.

**1.1.7. Một số thư viện, package khác**

* **React hook form**: là một thư viện giúp quản lý và xác thực các form trong React. Nó cung cấp các cách tiếp cận đơn giản và hiệu quả để xử lý các và kiểm tra các dữ liệu trong form trong ứng dụng React.
* **React-query**: là một thư viện quản lý trạng thái và dữ liệu trong React. Nó được thiết kế để giúp bạn quản lý dữ liệu từ xa một cách dễ dàng và hiệu quả. Nó có thể xử lý việc tìm nạp, lưu vào bộ nhớ đệm, đồng bộ hoá và cập nhật trạng thái.
* **Axios**: Axios là một thư viện HTTP Client dựa trên Promise. Cơ bản thì nó cung cấp một API cho việc xử lý XHR (XMLHttpRequests).

## Giới thiệu về kiến trúc

### 1.2.1. Kiến trúc hướng dịch vụ (SOA)

SOA là một hướng tiếp cận với việc thiết kế và tích hợp các phần mềm hay chức năng hệ thống theo dạng module, trong đó mỗi module đóng vai trò là một “dịch vụ có tính kết nối lỏng lẻo – Loose coupling” và có khả năng truy cập thông qua môi trường mạng. SOA là hướng tiếp cận xây dựng phần mềm dựa trên sự kết hợpcác dịch vụ.

A diagram of a service

Description automatically generated

Hình 1. 3. Sơ đồ kiến trúc hướng dịch vụ

Ưu điểm của SOA

* Sử dụng lại những thành phần hoặc dịch vụ có sẵn giúp tiết kiệm chi phí và thời gian xây dựng ứng dụng.
* Cung cấp giải pháp ứng dụng tổng hợp cho doanh nghiệp (tích hợp các ứng dụng rời rạc thành hệ thống đồng nhất).
* Tính loose coupling (kết nối lỏng lẻo) giúp tăng tính linh hoạt và khả năng triển khai cài đặt.
* Thích ứng với những thay đổi trong tương lai (giảm rủi ro do có thay đổi hoặc lỗi trong thiết kế hệ thống).
* Hỗ trợ đa thiết bị và đa nền tảng.
* Tăng khả năng mở rộng và khả năng sẵn sàng cung cấp.

### 1.2.2. Giới thiệu về RESTful API

**1.2.2.1. API là gì ?**

API là từ viết tắt của Application programming Interface. Nó cho phép kết nối và trao đổi dữ liệu giữa hai hệ thống phần mềm riêng biệt.

Một hệ thống phần mềm có thể nhúng các API bao gồm các hàm/ thủ tục con (functions/ sub-routines) mà có thể chạy bởi một hệ thống phần mềm khác.

Một sử dụng phổ biến của một API là khi bạn muốn để có được dữ liệu từ một ứng dụng (chẳng hạn như một công thức làm bánh) mà không cần phải thực sự truy cập các ứng dụng riêng của mình.

Để cho phép các hành động này diễn ra, ứng dụng đã xuất bản một API mà cụ thể cho phép các ứng dụng bên ngoài để thực hiện truy vấn đến các dữ liệu của nó và trả lại cho người dùng.

**1.2.2.2. RESTful API**

RESTful API là một giao diện lập trình ứng dụng (API) mà tuân thủ các ràng buộc và quy ước kiến trúc REST được sử dụng trong việc giao tiếp giữa client và server. REST là viết tắt của Representational State Transfer, nó được tạo ra bởi nhà khoa học máy tính Roy Fielding.

A diagram of a api

Description automatically generated

*Hình 1. 4. Cách hoạt động của RESTful API*

RESTful API thường vẫn sử dụng các HTTP method (như GET, POST, PUT, PATCH, DELETE…) và cách định dạng các URL cho ứng dụng web để quản lý các resource

# CHƯƠNG 3: Phân tích thiết kế hệ thống

**3.1. Bối cảnh/ hiện trạng của hệ thống**

### a. Định nghĩa vấn đề (mục tiêu) mà đề tài sẽ giải quyết

Trước khi có mạng xã hội chia sẻ công thức nấu ăn, việc tìm kiếm và lưu trữ công thức nấu ăn là một vấn đề khá khó khăn. Người nấu ăn thường phải tìm kiếm thông tin trên sách báo, tạp chí, hoặc từ người thân, bạn bè. Điều này khiến cho việc tìm kiếm một công thức nấu ăn cụ thể trở nên khó khăn và mất thời gian. Ngoài ra, việc lưu trữ công thức nấu ăn cũng không dễ dàng, người nấu ăn thường phải viết lại công thức nấu ăn vào một cuốn sổ tay hoặc lưu trữ trên máy tính. Điều này khiến cho công thức nấu ăn dễ bị thất lạc hoặc hư hỏng.

Dưới đây là một số vấn đề cụ thể khi chưa có mạng xã hội chia sẻ công thức nấu ăn:

* **Khó khăn trong việc tìm kiếm công thức nấu ăn**: Trước khi có mạng xã hội chia sẻ công thức nấu ăn, người nấu ăn thường phải tìm kiếm thông tin trên sách báo, tạp chí, hoặc từ người thân, bạn bè. Điều này khiến cho việc tìm kiếm một công thức nấu ăn cụ thể trở nên khó khăn và mất thời gian. Ví dụ, nếu bạn muốn tìm một công thức nấu ăn cho món bánh kem, bạn phải tìm kiếm trên nhiều nguồn khác nhau, và có thể bạn sẽ không tìm thấy công thức nấu ăn phù hợp với sở thích của mình.
* **Khó khăn trong việc lưu trữ công thức nấu ăn**: Việc lưu trữ công thức nấu ăn cũng không dễ dàng, người nấu ăn thường phải viết lại công thức nấu ăn vào một cuốn sổ tay hoặc lưu trữ trên máy tính. Điều này khiến cho công thức nấu ăn dễ bị thất lạc hoặc hư hỏng. Ví dụ, nếu bạn có một cuốn sổ tay lưu trữ công thức nấu ăn, cuốn sổ tay đó có thể bị thất lạc hoặc bị rách, khiến cho bạn mất đi những công thức nấu ăn yêu thích.
* **Thiếu sự tương tác giữa người nấu ăn**: Việc chia sẻ công thức nấu ăn giữa người nấu ăn với nhau là một cách để học hỏi và chia sẻ kinh nghiệm nấu ăn. Tuy nhiên, trước khi có mạng xã hội chia sẻ công thức nấu ăn, việc này rất khó khăn. Người nấu ăn thường phải gặp gỡ trực tiếp để chia sẻ công thức nấu ăn.

**b. Hiện trạng trước khi sử dụng phần mềm**

A diagram of a computer

Description automatically generated

Hình 3.1. Lược đồ cộng tác tìm kiếm công thức khi chưa có phần mềm

Trước khi có mạng xã hội chia sẻ công thức nấu ăn, người nấu ăn thường phải tìm kiếm thông tin trên sách báo, tạp chí, hoặc từ internet. Điều này khiến cho việc tìm kiếm một công thức nấu ăn cụ thể trở nên khó khăn và mất thời gian. Ví dụ, nếu muốn tìm một công thức nấu ăn cho món bánh kem, bạn phải tìm kiếm trên nhiều nguồn khác nhau, không chắc chắn công thức đó có thật sự ngon và có thể bạn sẽ không tìm thấy công thức nấu ăn phù hợp với sở thích của mình.

A white rectangle with black text

Description automatically generated

Hình 3.2. Lược đồ cộng tác lưu trữ công thức khi chưa sử dụng phần mềm

Trước khi có mạng xã hội chia sẻ công thức nấu ăn thì người nấu ăn thường phải viết lại công thức nấu ăn vào một cuốn sổ tay hoặc lưu trữ trên máy tính, điện thoại. Việc ghi chép hay nhập liệu đôi khi mất nhiều thời gian và việc lưu trữ thủ công như vậy khiến cho công thức nấu ăn dễ bị thất lạc hoặc hư hỏng. Ví dụ, nếu bạn có một cuốn sổ tay lưu trữ công thức nấu ăn, cuốn sổ tay đó có thể bị thất lạc hoặc bị rách, khiến cho bạn mất đi những công thức nấu ăn yêu thích của mình.

A diagram of a diagram

Description automatically generated

Hình 3.3. Lược đồ cộng tác cho việc chia sẻ công thức nấu ăn của các bạn bếp

Trước khi có mạng xã hội chia sẻ công thức nấu ăn, các bạn bếp muốn học hỏi kiến thức nấu ăn thường phải tham gia các khóa học nấu ăn hoặc cộng đồng đam mê nấu ăn. Tuy nhiên việc này khiến cho các bạn thường khá đắn đo vì chi phí học hoặc không thể sắp xếp thời gian tham gia.

**c. Đề xuất giải pháp đề tài**

**A diagram of a graph

Description automatically generated**

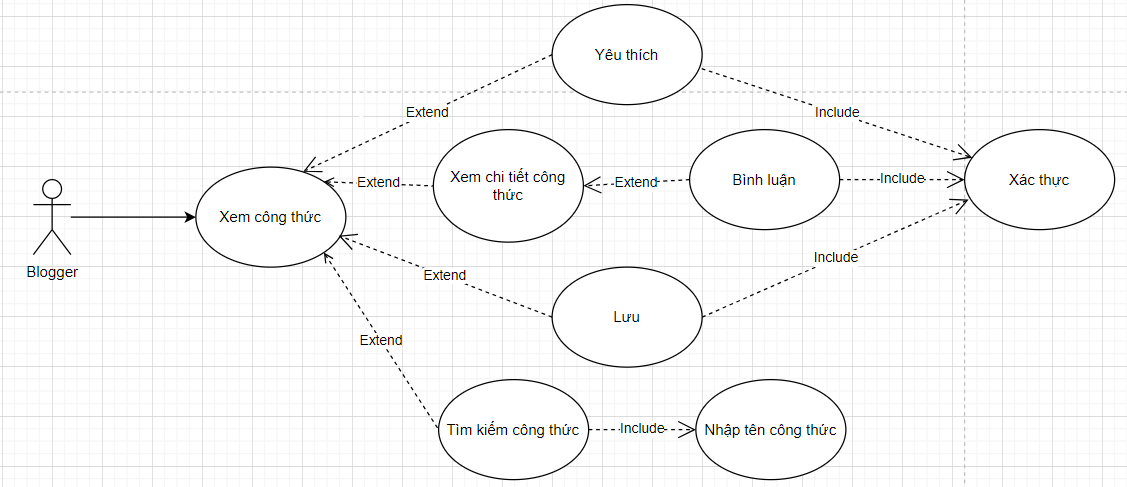
Hình 3.4. Lược đồ cộng tác giữa các bạn bếp chia sẻ công thức nấu ăn khi sử dụng phần mềm

Từ hiện trạng hiện tại, mạng xã hội chia sẻ công thức nấu ăn cung cấp một số giải pháp cho người dùng, bao gồm:

* Tính năng tìm kiếm nâng cao: Các nền tảng mạng xã hội chia sẻ công thức nấu ăn cung cấp các tính năng tìm kiếm nâng cao để giúp người dùng dễ dàng tìm kiếm công thức nấu ăn phù hợp. Ví dụ, người dùng có thể tìm kiếm công thức nấu ăn theo tên.
* Tính năng lưu trữ và chia sẻ: Các nền tảng mạng xã hội chia sẻ công thức nấu ăn cung cấp các tính năng lưu trữ và chia sẻ để giúp người dùng dễ dàng lưu trữ và chia sẻ công thức nấu ăn của mình. Ví dụ, người dùng có thể lưu trữ công thức nấu ăn dưới dạng danh sách, hoặc chia sẻ công thức nấu ăn trên trang cá nhân.
* Tính năng tương tác: Các nền tảng mạng xã hội chia sẻ công thức nấu ăn cung cấp các tính năng tương tác để giúp người dùng học hỏi và chia sẻ kinh nghiệm nấu ăn với nhau. Ví dụ, người dùng có thể bình luận, đánh giá, hoặc chia sẻ kinh nghiệm nấu ăn của mình.

**3.2. Định nghĩa các tương tác trên phần mềm**

**a. Xem công thức**



Hình 3.5. Usecase xem công thức

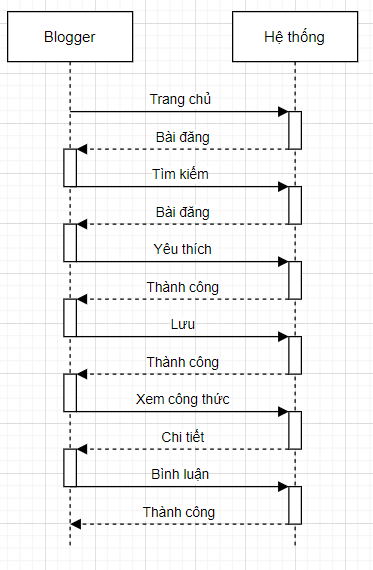
Use Case ID: U01

- Ý nghĩa: Khi blogger xem công thức trên trang chủ, có thể nhấn Yêu thích hoặc lưu công thức mình muốn. Cần phải nhập tên công thức để phần mềm tìm kiếm trong cơ sở dữ liệu để trả về kết quả đề xuất. Ấn vào khung ảnh đại diện của công thức để xem chi tiết đầy đủ của công thức đó, trong giao diện chi tiết này, blogger có thể bình luận về món được đăng.

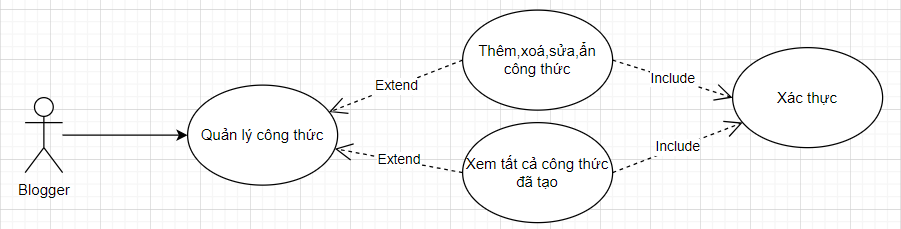
- Actors: Blogger, hệ thống.

- Điều kiện: Blogger đã có tài khoản trên phần mềm.

- Tương tác:



**b. Quản lý công thức**



Hình 3.6. Usecase quản lý công thức

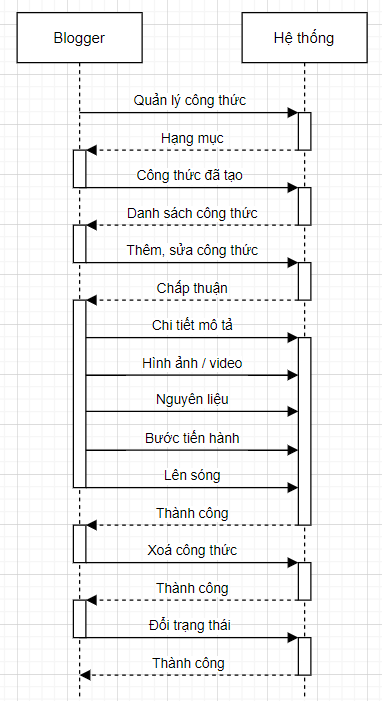
Use Case ID: U02

- Ý nghĩa: Blogger có thể tùy ý tạo 1 công thức mới hoặc xóa, sửa, public/private công thức đã tạo. Các công thức đã tạo sẽ được đăng lên trang chủ và trang cá nhân, blogger có thể dễ dàng xem lại bất cứ lúc nào.

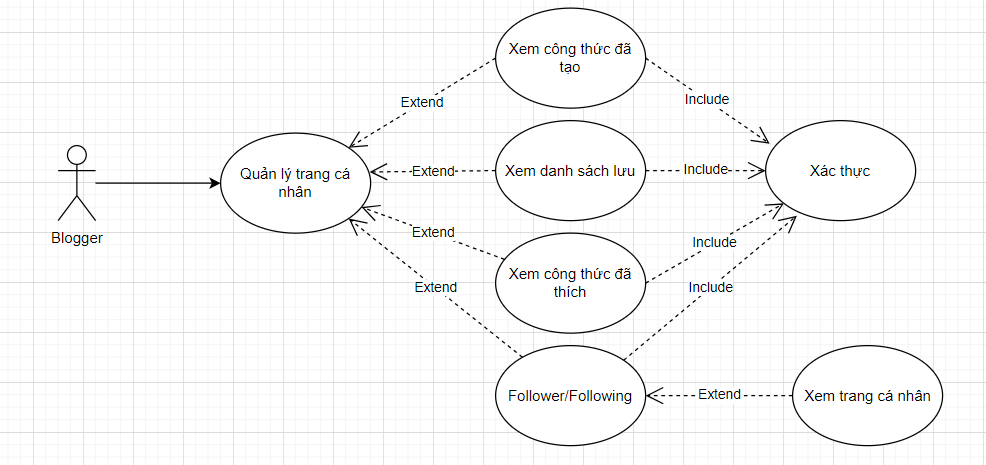
- Actors: Blogger, hệ thống.

- Điều kiện: Blogger đã có tài khoản trên phần mềm.

- Tương tác:



**c. Quản lý trang cá nhân**



Hình 3.7. Usecase quản lý trang cá nhân

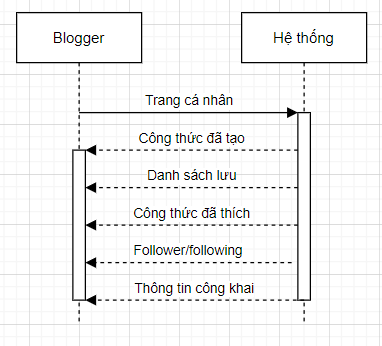
Use Case ID: U03

- Ý nghĩa: Mỗi blogger có một trang cá nhân riêng, trang cá nhân này sẽ hiển thị thông tin công khai của blogger đó. Bao gồm số người blogger đã follow, số người follow blogger đó, các công thức đã tạo của blogger sẽ được hiển thị công khai nếu được điều chỉnh công khai. Danh sách lưu chỉ được hiển thị với chủ sở hữu tài khoản, các công thức được lưu trong quá trình xem trang chủ sẽ được hiển thị ở đây. Danh sách công thức đã thích chỉ hiển thị với chủ sở hữu tài khoản, các công thức được yêu thích trong quá trình xem trang chủ sẽ được hiển thị ở đây. Khi xem trang cá nhân của blogger khác, một số chức năng sẽ bị ẩn để bảo mật thông tin người dùng.

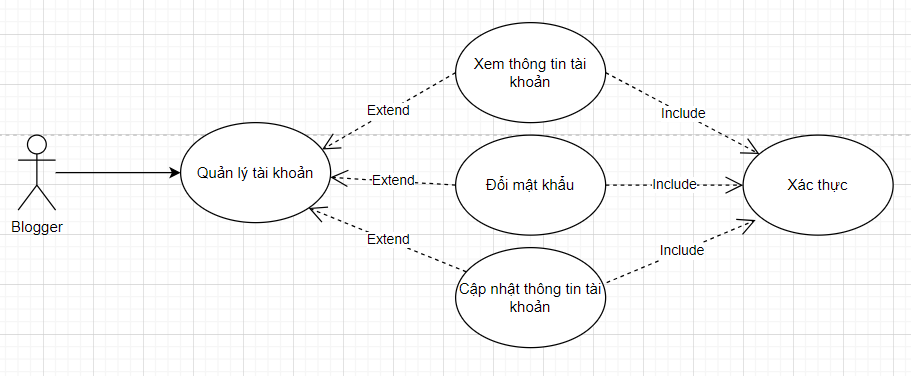
- Actors: Blogger, hệ thống.

- Điều kiện: Blogger đã đăng nhập trên phần mềm.

- Tương tác:



**d. Quản lý tài khoản**



Hình 3.8. Usecase quản lý tài khoản

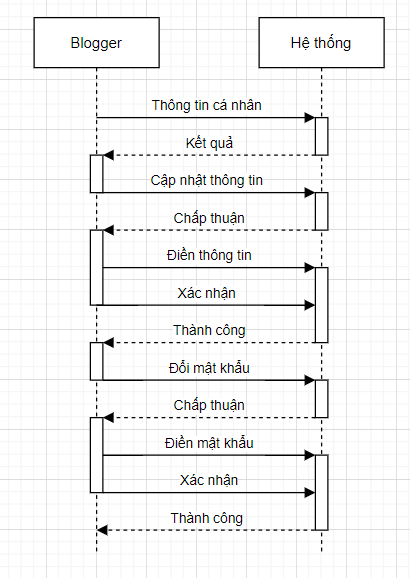
Use Case ID: U04

- Ý nghĩa: Thông tin đăng ký sẽ được hiển thị cho chủ sở hữu. Blogger có thể tuỳ thích chỉnh sửa thông tin cá nhân miễn đúng quy định. Mật khẩu cũng có thể thay đổi dựa vào mật khẩu cũ.

- Actors: Blogger, hệ thống.

- Điều kiện: Blogger đã đăng nhập trên phần mềm.

- Tương tác:



## 3.3. Định nghĩa yêu cầu và ràng buộc đối với phần mềm

### a. Yêu cầu từ môi trường nghiệp vụ

**1. Yêu cầu từ môi trường nghiệp vụ đối với usecase xem công thức**

| **UseCase** | **Req-ID** | **Nội dung yêu cầu** | **Stack-Holder** |
| --- | --- | --- | --- |
| U01 | B01.1 | **Yêu cầu về tính chính xác và đầy đủ thông tin**  Thông tin về công thức nấu ăn phải chính xác và đầy đủ các bước, nguyên liệu để người dùng có thể dễ dàng học hỏi | Khách hàng |
| B01.2 | **Người sử dụng có thể tương tác với nhau**  Mạng xã hội chia sẻ công thức nấu ăn phải có các tính năng để người dùng có thể tương tác với nhau, chẳng hạn như bình luận, yêu thích công thức. | Khách hàng |
| B01.3 | **Có thể tìm kiếm công thức**  Người dùng có thể tìm kiếm công thức dễ dàng thông qua tên công thức, tên nguyên liệu | Khách hàng |
| B01.4 | **Có thể lưu trữ**  Người dùng có thể lưu trữ công thức mình yêu thích để tiện xem lại nếu muốn | Khách hàng |

Bảng 3.1. Yêu cầu từ môi trường nghiệp vụ đối với usecase xem công thức

**2. Yêu cầu từ môi trường nghiệp vụ đối với usecase quản lý công thức**

| **UseCase** | **Req-ID** | **Nội dung yêu cầu** | **Stack-Holder** |
| --- | --- | --- | --- |
| U02 | B02.1 | **Yêu cầu về quyền riêng tư công thức**  Nếu công thức ở trạng thái riêng tư, chỉ chủ sở hữu công thức đó mới xem được công thức | Khách hàng |

Bảng 3.2. Yêu cầu từ môi trường nghiệp vụ đối với usecase quản lý công thức

**3. Yêu cầu từ môi trường nghiệp vụ đối với usecase quản lý trang cá nhân**

| **UseCase** | **ID** | **Nội dung yêu cầu** | **Stack-Holder** |
| --- | --- | --- | --- |
| U03 | B03.01 | **Yêu cầu về xem trang cá nhân của bản thân**  Hiển thị tất cả công thức (riêng tư và công khai)  Người dùng khác không xem được danh sách lưu trữ | Khách hàng |
| B03.02 | **Yêu cầu về xem trang cá nhân của người khác**  Chỉ hiển thị công thức ở trạng thái công khai | Khách hàng |

Bảng 3.3. Yêu cầu từ môi trường nghiệp vụ đối với usecase quản lý trang cá nhân

**4.** **Yêu cầu từ môi trường nghiệp vụ đối với usecase quản lý tài khoản**

| **UseCase** | **ID** | **Nội dung yêu cầu** | **Stack-Holder** |
| --- | --- | --- | --- |
| U04 | B04.01 | **Yêu cầu về bảo mật dữ liệu**  Mật khẩu của người dùng sẽ được mã hóa, yêu cầu khách hàng đặt mật khẩu có độ an toàn cao | Khách hàng |
| B04.02 | **Đảm bảo về ràng buộc dữ liệu**  Mỗi email sẽ chỉ được đăng ký với 1 tài khoản duy nhất | Khách hàng |

Bảng 3.4. Yêu cầu từ môi trường nghiệp vụ đối với usecase quản lý tài khoản

### b. Yêu cầu từ môi trường vận hành

* Các yêu cầu chức năng

| **UseCase** | **ID** | **Nội dung yêu cầu** | **Stack-Holder** |
| --- | --- | --- | --- |
| U01 | F01.01 | **Yêu cầu về xem công thức**  Người dùng chỉ xem được công thức của người khác nếu công thức đó ở trạng thái công khai | Khách hàng |
| F01.02 | **Gợi ý công thức cho người dùng**  Gợi ý cho người dùng những công thức mới, công thức theo nguyên liệu đang trong mùa, các công thức có lượt tương tác cao | Khách hàng |

Bảng 3.5. Yêu cầu từ môi trường vận hành đối với usecase xem công thức

| **UseCase** | **ID** | **Nội dung yêu cầu** | **Stack-Holder** |
| --- | --- | --- | --- |
| U02 | F02.01 | **Đảm bảo tính nhất quán dữ liệu**  Khi thêm, sửa dữ liệu công thức, nguyên liệu, bước đều phải thêm thành công, nếu không phải rollback | Khách hàng |
| F02.02 | **Yêu cầu về trạng thái công thức**  Công thức chỉ được chia sẻ với người khác khi công thức ở trạng thái công khai | Khách hàng |

Bảng 3.6. Yêu cầu từ môi trường vận hành đối với usecase quản lý công thức

* Các yêu cầu chất lượng

| **UseCase** | **ID** | **Nội dung yêu cầu** | **Stack-Holder** |
| --- | --- | --- | --- |
| U01 | NF01.01 | **Performance efficiency: hệ thống**  Time-behavior:  ≤ 2 seconds / transaction  Resource utilization: cpu I5, hdd 1T, mem 16G  Capacity: ~ 50 transactions / day | Khách hàng |
| NF01.02 | **Maintainability: hệ thống**  Reusability: **yes**, Modifiability: **yes**, Testability: **yes** | Khách hàng |
| NF01.04 | **Giao diện thân thiện, dễ sử dụng** | Khách hàng |

Bảng 3.7. Yêu cầu chất lượng đối với phần mềm

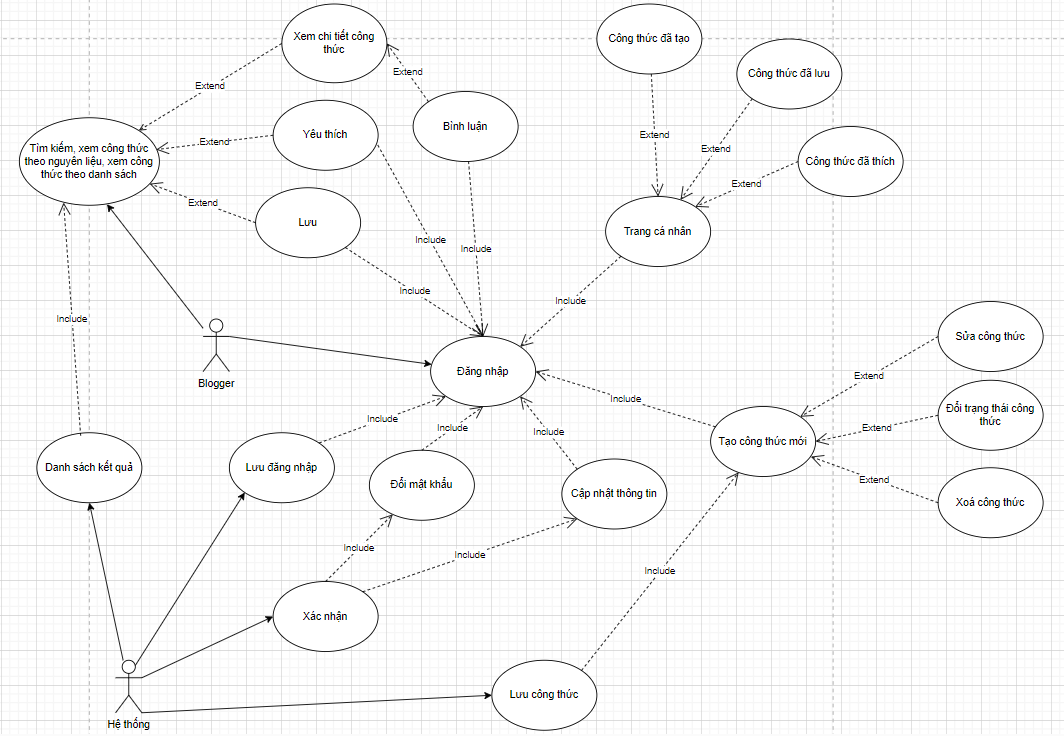
### c. Yêu cầu từ môi trường phát triển

| **ID** | **Đối tượng** | **Nội dung yêu cầu** | **Stack-Holder** |
| --- | --- | --- | --- |
| TA01 | Application | Server: Intel, Windows 10,  cpu I5, hdd 1T, mem 16G | Khách hàng |
| TD02 | Software | Viết code trên Visual Studio Code | Dev team |
| TD03 | Software | Dùng ReactJs 17, NodeJS v16.15.1 | Dev team |
| TD04 | Software | Sử dụng PostgreSQL để lưu trữ dữ liệu | Dev team |

Bảng 3.8. Yêu cầu từ môi trường phát triển đối với phần mềm

# CHƯƠNG 4: Thiết kế phần mềm

**4.1. Lược đồ usecase cho thiết kế**

****

Hình: Sơ đồ usecase cho ứng dụng web

**4.2. Các đối tượng trong thiết kế phần mềm**