**BỘ LAO ĐỘNG THƯƠNG BINH VÀ XÃ HỘI**

**TRƯỜNG CAO ĐẲNG CÔNG NGHỆ THÔNG TIN THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN – ĐIỆN TỬ**

**BÁO CÁO MÔN HỌC: CHUYÊN ĐỀ BACK-END**

**ĐỀ TÀI: XÂY DỰNG WEBSITE BLOG ONLINE**

**.....................................................................................................................................**

Giáo viên hướng dẫn : **Thầy.Nguyễn Minh Hải**

Sinh viên thực hiện: **Vũ Văn Tuệ Ninh**

**Trần Quang Mạnh**

Lớp : **501-CĐ B-E\_Hk2.2345\_CD23CT.N6**

Khoá  **: 2023-2025**

***TP. Hồ Chí Minh, ngày 6 tháng 12 năm 2025***

Lời Mở Đầu

Trong thời đại số, blog không chỉ là nơi lưu giữ suy nghĩ cá nhân mà còn là nền tảng quan trọng để chia sẻ tri thức, kinh nghiệm và kết nối cộng đồng. Đặc biệt, đối với sinh viên ngành Công nghệ Thông tin, blog là công cụ hữu ích giúp hệ thống hóa kiến thức, chia sẻ kinh nghiệm lập trình, thảo luận các vấn đề kỹ thuật và ghi lại hành trình phát triển bản thân. Ngoài ra, việc duy trì một blog còn giúp sinh viên nâng cao tư duy logic, kỹ năng trình bày và mở rộng cơ hội kết nối trong ngành.

Nhận thức được tầm quan trọng của blog, nhóm chúng em quyết định lựa chọn đề tài **"Website Blog"** cho dự án lần này. Mặc dù đây không phải là một ý tưởng mới, nhưng chúng em đã tập trung nghiên cứu, phát triển để mang đến một sản phẩm hoàn thiện, tối ưu hóa trải nghiệm người dùng. Website blog sẽ có giao diện thân thiện, dễ sử dụng, tích hợp các chức năng như đăng bài, bình luận, chia sẻ nội dung và quản lý bài viết. Đồng thời, hệ thống cũng được chú trọng đến hiệu suất, bảo mật và khả năng mở rộng để đáp ứng nhu cầu của nhiều người dùng.

Thông qua dự án này, chúng em không chỉ áp dụng kiến thức về lập trình web mà còn rèn luyện kỹ năng thiết kế giao diện, tối ưu hệ thống và quản lý dữ liệu. Quan trọng hơn, chúng em mong muốn blog trở thành một nền tảng hữu ích, giúp mọi người kết nối, chia sẻ tri thức và lan tỏa đam mê trong cộng đồng. Đây cũng sẽ là một bước đệm quan trọng, giúp chúng em tự tin hơn khi bước vào thế giới công nghệ đầy thử thách nhưng cũng vô cùng tiềm năng.

Lời Cảm Ơn

Nhóm chúng em xin gửi lời cảm ơn chân thành đến Quý Thầy Cô khoa Công nghệ Thông tin, đặc biệt là giảng viên hướng dẫn Thầy Nguyễn Minh Hải, người đã tận tình chỉ bảo, định hướng và góp ý để chúng em có thể hoàn thành đề tài “Website Blog” một cách tốt nhất.

Chúng em cũng xin bày tỏ lòng biết ơn đến Thầy bộ môn chuyên đề Back End đã trang bị kiến thức và chia sẻ kinh nghiệm quý báu trong suốt quá trình học tập, giúp chúng em có đủ nền tảng để thực hiện dự án này.

Bên cạnh đó, chúng em xin gửi lời cảm ơn đến các bạn sinh viên đã góp ý, chia sẻ ý tưởng và hỗ trợ trong quá trình phát triển website.

Mặc dù đã cố gắng hoàn thiện, nhưng chắc chắn sản phẩm vẫn còn nhiều thiếu sót. Chúng em rất mong nhận được những góp ý quý báu từ Quý Thầy Cô và mọi người để có thể cải thiện và phát triển hơn nữa.

**NHẬN XÉT**

**(Giảng viên hướng dẫn)**

**Kết luận: Đồng ý hoặc Không đồng ý cho sinh viên nộp báo cáo.**

TP. Hồ Chí Minh, ngày .... tháng năm 20

GIẢNG VIÊN HƯỚNG DẪN

(Ký tên)

MỞ ĐẦU

# 1.GIỚI THIỆU (PHÂN TÍCH ) DỰ ÁN

## 1.1 GIỚI THIỆU:

-Trang blog là một nền tảng trực tuyến nơi các cá nhân hoặc tổ chức chia sẻ thông tin, kiến thức, quan điểm và kinh nghiệm về các chủ đề cụ thể. Dự án này nhằm mục đích tạo ra một trang blog chất lượng, thu hút độc giả và cung cấp giá trị thực sự thông qua nội dung hữu ích, đa dạng và được cập nhật thường xuyên.

-Mục tiêu của dự án:

1. Xây dựng cộng đồng độc giả trung thành: Tạo ra một không gian tương tác, nơi độc giả có thể đọc, bình luận và chia sẻ bài viết.
2. Cung cấp nội dung chất lượng: Đảm bảo các bài viết được nghiên cứu kỹ lưỡng, chính xác và phù hợp với nhu cầu của độc giả.
3. Tối ưu hóa trải nghiệm người dùng: Thiết kế giao diện thân thiện, dễ sử dụng và tương thích với các thiết bị di động.
4. Tăng khả năng hiển thị trên công cụ tìm kiếm (SEO): Áp dụng các kỹ thuật SEO để thu hút lượng truy cập tự nhiên từ các công cụ tìm kiếm như Google.
5. Tạo nguồn thu nhập thụ động: Thông qua quảng cáo, affiliate marketing, hoặc các dịch vụ trả phí.

* Phân tích hệ thống

1. CMS: WordPress, Blogger hoặc Wix.
2. Cơ sở dữ liệu: Lưu trữ bài viết, bình luận, người dùng.
3. Bảo mật: SSL, plugin chống spam, sao lưu dữ liệu.
4. Công cụ phân tích: Google Analytics, Search Console.

* Các chức năng

1. Người dùng: Đọc bài, bình luận, chia sẻ, đăng ký nhận thông báo.

* . Phương pháp nghiên cứu

Nghiên cứu thị trường, phân tích đối thủ. Khảo sát người dùng, phân tích dữ liệu. A/B testing để tối ưu hóa.

* . Mục tiêu Ngắn hạn:

20 bài viết, 1.000 lượt truy cập/tháng. Dài hạn: 10.000 lượt truy cập/tháng, tạo doanh thu từ quảng cáo và affiliate.

* Kỹ thuật sử dụng:

SEO, Tiếp thị nội dung, Tiếp thị truyền thông xã hội, Tiếp thị qua email, Thử nghiệm A / B.

* Kết quả thực hiện

Tăng lượt truy cập, giảm tỷ lệ thoát. Tăng tương tác (bình luận, chia sẻ). Đạt doanh thu từ quảng cáo và affiliate. Nâng cao độ phủ thương hiệu.**Mục lục**

[LỜI MỞ ĐẦU 1](#_Toc108458782)-5

[Chương 1. TỔNG QUAN TÀI LIỆU 5](#_Toc108458783)-7

**1.1 Khảo sát bài toán 8 1.1.1 Đặt vấn đề 9**

**1.1.2.Giải pháp 10**

**CHƯƠNG 2: GIỚI THIỆU VỀ NPM & EXPRESS 12**

**2.2. Một số cài đặt npm 12-14**

**2.2.1.Giới thiệu Express 14-15**

**2.2.2.Xử lý yêu cầu với express 15**

**CHƯƠNG 3: BẮT ĐẦU DỰ ÁN BLOG 17**

**3.1.Phân tích hệ thống 18**

**3.1.1.Tổng quát quy tình nghiệp vụ 18**

**3.1.2.Chi tiết quy trình nghiệp vụ 19**

**3.1.3.Thông tin yêu cầu 20**

**CHƯƠNG 4: SƠ ĐỒ CHỨC NĂNG HỆ THỐNG 21**

**4.1.Mô tả sơ đồ chức năng 21**

**4.1.2.Đặc tả usecase đăng nhập vào hệ thống 22**

**4.1.3.Đặc tả usecase đăng xuất hệ thống 22**

**4.1.4.Đặc tả usecase đăng kí tài khoản 23**

**4.1.5.Usecase hiện thị bài viết 23**

**4.1.6.Usecase Quản lý nội dung 23**

**4.1.7.Usecase tìm kiếm và hiện thị nội dung 24**

**4.1.8.UseCase quản lý khách hàng 25**

**4.1.9.Usecase quản lsy bài viết 25**

**4.1.10.Quản lý tài khoản admin 26**

**CHƯƠNG 5: LƯỢC ĐỒ ACTIVITY**

**5.1.Lược đồ chức năng đăng nhập 27**

**5.1.2.Lược đồ chức năng đăng kí 28**

**5.1.3.Lược đồ reset mật khẩu 29**

**5.1.4.Luộc đồ chức năng tìm kiếm 30**

**5.1.5.Lược đồ chức năng tạo bài viết 31**

**5.2.Biểu đồ thành phần hệ thống 32**

**5.2.1.Sơ đồ khách hàng 32**

**5.2.2.Sơ đồ admin 33**

**CHƯƠNG 6: ÁP DỤNG MONGODB VÀO DỰ ÁN CỦA CHÚNG TÔI 34-35**

**CHƯƠNG 7: TẢI LÊN ẢNH BẰNG EXPRESS 35-39**

**CHƯƠNG 8: TỔNG KẾT DỰ ÁN 40**

# Chương 1: Tổng Quan Tài Liệu:

1.1.Kháo sát bài toán :

Trước khi xây dựng một trang web blog, cần tiến hành khảo sát để hiểu rõ nhu cầu và sở thích của người dùng. Một số yếu tố cần khảo sát bao gồm:

* **Mục đích sử dụng:** Blog được tạo ra để chia sẻ kiến thức, kinh nghiệm cá nhân, tin tức hay phục vụ mục đích kinh doanh?
* **Đối tượng người dùng:** Ai sẽ là người đọc chính của blog? (Học sinh, sinh viên, người làm nghề, chuyên gia...)
* **Thể loại nội dung:** Người dùng quan tâm đến chủ đề nào? (Công nghệ, ẩm thực, du lịch, giáo dục, tài chính, đời sống,...)
* **Hình thức trình bày:** Người dùng thích đọc bài viết dài hay ngắn? Họ có quan tâm đến hình ảnh, video, infographics không?
* **Chức năng mong muốn:** Người dùng muốn có bình luận, chia sẻ bài viết lên mạng xã hội, hoặc đăng ký nhận thông báo không?
  + 1. Đặt vấn đề:

Sau khi khảo sát, cần xác định vấn đề chính mà blog sẽ giải quyết:

* **Vấn đề 1: Khó tìm kiếm thông tin chất lượng trên mạng**
  + Giải pháp: Blog sẽ tổng hợp thông tin từ nhiều nguồn uy tín, trình bày dễ hiểu.
* **Vấn đề 2: Người đọc thiếu động lực quay lại blog**
  + Giải pháp: Cập nhật nội dung thường xuyên, tạo cộng đồng tương tác.
* **Vấn đề 3: Thiếu trải nghiệm người dùng tốt trên blog**
  + Giải pháp: Giao diện thân thiện, tối ưu tốc độ tải trang, có chức năng tìm kiếm và phân loại bài viết.
    1. Giải Pháp :

### **1. Mục Tiêu**

* Tạo nền tảng giúp khách hàng dễ dàng đặt món, thanh toán online.
* Hỗ trợ doanh nghiệp quản lý đơn hàng, doanh thu, tồn kho.

### **2. Chức Năng Chính**

* Cho khách hàng: Đặt món, thanh toán online, theo dõi đơn hàng, đánh giá sản phẩm.
* Cho doanh nghiệp: Quản lý sản phẩm, đơn hàng, khách hàng, khuyến mãi.
* Cho admin: Quản lý tài khoản, bảo mật, cấu hình hệ thống.

### **3. Công Nghệ Phát Triển**

* Giao diện người dùng: Angular, React, Vue.js
* Phần phụ trợ: Node.js, Laravel, Django
* Cơ sở dữ liệu: MySQL, PostgreSQL, MongoDB
* Thanh toán: VNPAY, MoMo, ZaloPay

### **4. Chiến Lược Triển Khai**

* Giai đoạn 1: Nghiên cứu thị trường, xác định mô hình.
* Giai đoạn 2: Thiết kế & phát triển website.
* Giai đoạn 3: Kiểm thử & triển khai.
* Giai đoạn 4: Marketing, mở rộng thị trường.

Ứng dụng sẽ xây dựng

Chúng em xây dựng một ứng dụng blog cho phép người dùng viết bài đăng trên blog sau khi họ đăng ký tài khoản (hình 1.1, 1.2, 1.3).

Hình 1.1 trang chủ

A person in a white dress

AI-generated content may be incorrect.

Hình 1.2 – Trang đăng nhập

A screenshot of a computer screen

AI-generated content may be incorrect.

Hình 1.2 trang đăng kí

A screenshot of a login form

AI-generated content may be incorrect.

Hình 1.3 – Tạo biểu mẫu bài đăng trên blog

A screenshot of a computer

AI-generated content may be incorrect.

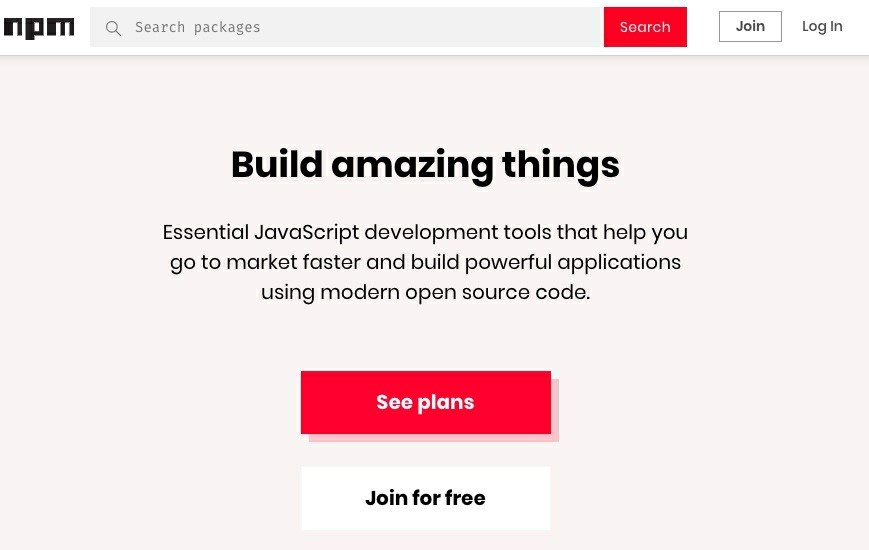
Sau khi người dùng đăng ký, họ có thể vào trang chủ và đăng nhập bằng thông tin đăng nhập của mình. Thanh điều hướng sẽ hiển thị động các mục khác nhau tùy thuộc vào người dùng đang đăng nhập hay đăng xuất. Chúng ta sẽ thực hiện điều này bằng cách sử dụng

Công cụ tạo mẫu EJS. Sau khi đăng nhập, thanh điều hướng sẽ có mục ‘Đăng xuất’. Cũng sẽ có mục ‘Bài đăng mới’ nơi người dùng có thể tạo bài đăng blog mới và tải lên tệp hình ảnh liên quan. Khi cô ấy quay lại Trang chủ, các mục nhập blog của cô ấy sẽ được đăng ở đó với tên người dùng và ngày đăng. Trong suốt ứng dụng này, chúng ta sẽ học được nhiều khái niệm và củng cố kiến ​​thức về Node.js, Express và MongoDB của mình.

CHƯƠNG 2: GIỚI THIỆU VỀ *NPM* & EXPRESS

# 2.2.Cài đặt gói tùy chỉnh với *npm:*

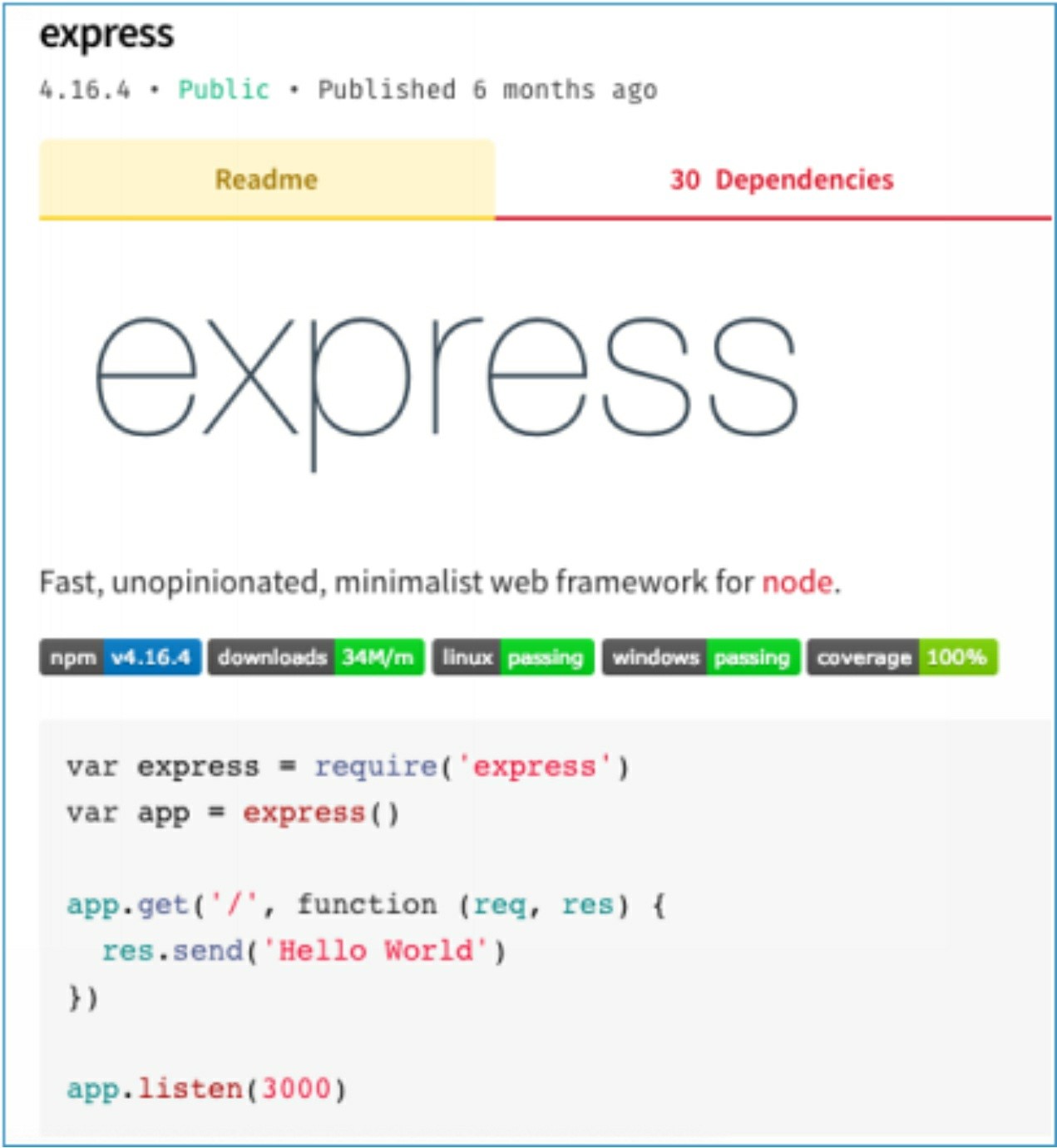
Trong ứng dụng của chúng em từ chương một, chúng em đã nhập các gói từ chính Node.js, ví dụ: *http*, *fs*. Bây giờ cộng đồng Node.js có rất nhiều nhà phát triển viết các mô-đun/gói tùy chỉnh mà chúng ta có thể sử dụng trong các ứng dụng của riêng mình. Các gói này được lưu trữ trên một trang web có tên là *npmjs.com* (hình 2.1) nơi bạn có thể tìm kiếm để xem gói nào phù hợp để sử dụng trong mã của riêng bạn. Chúng em sử dụng *npm* (hoặc Node Package Manager) để quản lý các gói mà chúng em tải xuống. *npm* là một trợ giúp chính thức cho các dự án Node đi kèm với Node khi chúng em cài đặt nó.



Hình 2.1

Một gói tùy chỉnh rất quan trọng mà chúng em sẽ cài đặt là Express. Bởi vì các API Node.js vani có thể dài dòng, khó hiểu và hạn chế về các tính năng, Express là một framework hoạt động như một lớp nhẹ trên máy chủ web Node.js giúp phát triển các ứng dụng web Node.js dễ dàng hơn. Chuyển phát nhanh được nhiều công ty sử dụng. Sau này chúng ta sẽ thấy cách nó đơn giản hóa các API của Node.js, thêm các tính năng hữu ích, giúp tổ chức chức năng của ứng dụng của chúng tôi với phần mềm trung gian và định tuyến, thêm các tiện ích hữu ích vào các đối tượng HTTP của Node.js và tạo điều kiện hiển thị các chế độ xem HTML động. Trong cuốn sách này, chúng ta sẽ khám phá sâu những đặc điểm này.

Vì vậy, hãy vào *npmjs.com* và tìm kiếm 'express ', sẽ có hướng dẫn về cách cài đặt và sử dụng gói *express* (hình 2.2).



Hình 2.2

Như đã đề cập trong trang web dưới ' Cài đặt ', chúng ta phải chạy *npm install express* để cài đặt gói *express.*

Trước khi chúng em chạy *npm install express*, *npm* yêu cầu chúng em phải có một tệp *package.json* để theo dõi tất cả các gói và phiên bản của chúng được sử dụng trong ứng dụng. Nếu bạn chạy *npm install express* ngay bây giờ mà không có tệp, bạn sẽ nhận được một loạt cảnh báo như " không *có tệp hoặc thư mục như vậy, hãy mở ... package.json* " .

Để tạo *tệp package.json*, hãy chạy ' npm init ' sẽ nhắc một loạt câu hỏi về dự án của chúng emi (ví dụ: tên dự án, tác giả, phiên bản) để tạo package.json cho chúng em.

*node\_modules*. Nếu bạn nhìn vào thư mục ứng dụng của mình, thư mục node\_modules sẽ được tạo cho bạ (hình 2.3). Đây là nơi các phụ thuộc tùy chỉnh được lưu cho dự án của chúng ta.



Hình 2.3

Nếu bạn mở và khám phá *node\_modules*, bạn sẽ có thể xác định vị trí gói chuyển *phát nhanh*. Lý do tại sao chúng tôi thấy nhiều gói khác trong node\_modules mặc dù chúng em chỉ cài đặt *express* là vì *express* phụ thuộc vào các gói khác này và chúng đã được cài đặt khi *cài* đặt express. Các gói khác này là phụ thuộc của Express. Tệp *package-lock.json* theo dõi các phiên bản của tất cả các phụ thuộc của Express.

# 2.2.1.Giới thiệu về Express

Trong phần này, chúng em giới thiệu Express và cách nó giúp phát triển ứng dụng trong Node dễ dàng và nhanh hơn. Đầu tiên, chúng em nhập *express* vào ứng dụng của mình bằng cách thêm mã dưới đây vào *index.js*:

const express = yêu cầu ('express')

Thao tác này kéo gói từ  *thư mục* node\_modules.

Để đánh giá cao việc Express giúp phát triển dễ dàng hơn bao nhiêu cho chúng ta, chúng ta sẽ sử dụng Express để lặp lại những gì chúng ta đã làm trước đây trong chương một.

Express xử lý *các đối tượng http*, *yêu cầu* và *phản hồi* đằng sau hậu trường. Hàm callback được cung cấp trong đối số thứ 2 trong *app.listen()* được thực thi khi máy chủ bắt đầu lắng nghe.

## Chạy ứng dụng của bạn

Sao chép mã Express ở trên vào index.js, chạy *nút index.js* trong lệnh và xem nhật ký "Ứng dụng đang lắng nghe trên cổng 3000" được in. Chúng ta sẽ đọc thêm về những ưu điểm của việc sử dụng Express trong phần tiếp theo.

# 2.2.2.Xử lý yêu cầu với Express

Express cho phép linh hoạt hơn trong việc phản hồi các yêu cầu ' get ' hoặc ' post ' của trình duyệt. Chúng tôi minh họa điều này trong mã sau in **đậm:**

const express = require('express') const app = express()

app.listen(3000,()=>{ console.log("Ứng dụng nghe trên cổng 3000")

})

**app.get('/',(req,res)=>{ res.json({ tên: 'Greg Lim'**

**})**

**})**

Trong đoạn code trên, bạn có thể thấy rằng Express đã điền nhiều hoạt động hơn để chúng em phản hồi tốt hơn các yêu cầu của trình duyệt. Ví dụ: chúng tôi trả về phản hồi JSON cho trình duyệt với res.json. Do đó, điều này cho phép chúng ta xây dựng API với Node. Chúng ta có thể xác định các tuyến đường cụ thể và phản hồi của nó khi một tuyến đường bị tấn công, ví dụ:

**app.get('/about',(req,res)=>{ res.json({ name: 'Greg Lim'**

**})**

**})**

Điều này còn được gọi là *Định tuyến,* nơi chúng em, ánh xạ các yêu cầu đến các trình xử lý cụ thể tùy thuộc vào URL của chúng. Trước đây không có Express, chúng tôi phải phản hồi các tuyến riêng lẻ bằng một  *câu lệnh if-else* mở rộng trong một trình xử lý yêu cầu lớn:

const server = http.createServer((req, res) =>{ if(req.url === '/about') res.end(aboutPage)

else if(req.url === '/contact') res.end(contactPage)

else if(req.url === '/') res.end(trang chủ)

else { res.writeHead(404)

res.end(không tìm thấy)

}

})

Với Express, chúng ta có thể tái cấu trúc một hàm xử lý yêu cầu lớn thành nhiều trình xử lý yêu cầu nhỏ hơn mà mỗi bộ xử lý xử lý một hàm cụ thể. Điều này cho phép chúng tôi xây dựng ứng dụng của mình theo cách mô-đun và dễ bảo trì hơn.

Chúng ta cũng có thể xác định loại phản hồi mà máy chủ phục vụ. Ví dụ: để phục vụ các tệp HTML với express, chúng ta thêm như sau:

**const path = require('đường dẫn')**

…

app.get('/',(req,res)=>{ **res.sendFile(path.resolve(\_\_dirname,'index.html'))**

})

## Giải thích mã

const path = require('đường dẫn')

Chúng em giới thiệu một mô-đun mới có tên là *path*. *path* là một mô-đun được tích hợp sẵn khác trong Node. Nó giúp chúng tôi có được đường dẫn cụ thể đến tệp này, bởi vì nếu chúng tôi thực hiện *res.sendFile( ' index.html ' )*, sẽ có lỗi được ném ra vì cần một đường dẫn tuyệt đối.

path.resolve(\_\_dirname,'index.html') giúp chúng ta có được đường dẫn tuyệt đối đầy đủ mà nếu không sẽ thay đổi dựa trên các Hệ điều hành khác nhau. Ví dụ: trên Mac và Linux, chúng tôi đề cập đến một thư mục bằng cách sử dụng */directory*, nhưng trên Windows, chúng tôi đề cập đến nó bằng \ *directory*. Mô-đun *đường dẫn* đảm bảo rằng mọi thứ chạy trơn tru bất kể hệ điều hành là Windows, Mac hay Linux.

app.get('/',(req,res)=>{ **res.sendFile(path.resolve(\_\_dirname,'index.html'))**

})

Express cũng thêm những thứ như phương thức *sendFile* mà nếu chúng ta phải thực hiện mà không có Express, sẽ có khoảng 45 dòng mã tệp phức tạp. Vì vậy, hy vọng bạn sẽ dần đánh giá cao cách Express làm cho việc viết các chức năng xử lý yêu cầu này dễ dàng hơn và nói chung đơn giản hóa việc phát triển trong Node.

CHƯƠNG 3: BẮT ĐẦU DỰ ÁN BLOG

3.1 Phân tích hệ thống:

3.1.1 Tổng quát quy trình nghiệp vụ(chia bước)

\*Đầu tiên quy trình đăng ký và đăng nhập của web blog :

-Khách hàng đăng ký tài khoản với email và thông tin cá nhân

-Xác thực email

\*Hiển thị bài viết

* Người dùng có thể xem danh sách bài viết theo danh mục hoặc tìm kiếm.
* Mỗi bài viết có tiêu đề, nội dung, hình ảnh, và thời gian đăng.

\*Quản lý nội dung:

* Tạo, chỉnh sửa, xuất bản, và xóa bài viết.
* Phân loại theo danh mục, từ khóa.

\*Tương tác người dùng:

* Bình luận, thích, chia sẻ bài viết.
* Đánh giá bài viết.

\*Tối ưu tốc độ & SEO:

* Trang tải nhanh, hiển thị tốt trên mọi thiết bị.
* Nội dung chuẩn SEO để tăng khả năng tìm kiếm.

\*Tìm kiếm & phân loại nội dung

* Lọc bài viết theo từ khóa, danh mục, hoặc thời gian.

3.1.2.Chi tiết quy trình nghiệp vụ:

\* Đăng ký tài khoản:

-Khách hàng bấm vào mục đăng ký

-Nhập thông tin:Họ tên,Email (bắt buộc),Số điện thoại,Mật khẩu,Xác nhận mật khẩu

\* Đăng nhập:

-Nhập email/số điện thoại và mật khẩu

-Hệ thống xác thực là vào trang web bán hàng online

\*Hiện thị bài viết :

-Xem các bài viết ,like và chia sẻ bài viết

\*Quản lý nội dung:

-Người dùng chọn vào mục tạo bài viết

-Người dùng tạo tiêu đề và thêm hình ảnh từ máy lên

-Người dùng có thể chỉnh sửa bài viết hoặc xóa bài viết

\*Tương tác người dùng:

-Người dùng bấm vào bài post có thể tim và comment vào bình luận

\*Tìm kiếm & phân loại nội dung:

-Người dùng bấm vào thanh tìm kiếm

-Người dùng có thể tìm kiếm nội dung bằng hastag hoặc nội dung

3.1.3.Thông tin yêu cầu :

Yêu cầu chức năng:

Người dùng:

-Đăng kí và đăng nhập

-xem nội dung bài post

-thêm hình ảnh, sửa nội dung bài post

-thích nội dung và bình luận

Admin:

-Đăng nhập

-Quản lý tài khoản người dùng

-Quản lý bài viết

# CHƯƠNG 4: Sơ Đồ Chức Năng Của Hệ Thống

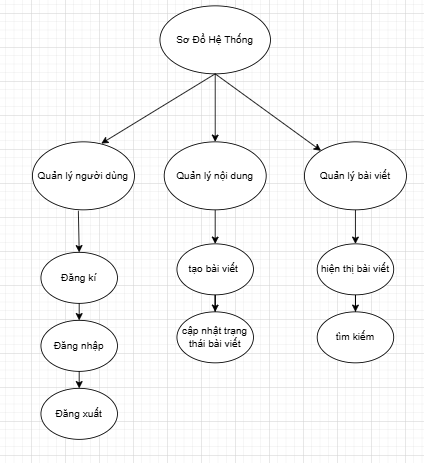
4.1.Mô Tả Sơ Đồ Chức Năng:

Hệ thống bao gồm các nhóm chức năng chính:

-Quản lý nội dung:tạo bài post ,cập nhật trạng thái

-Quản lý người dùng:Đăng kí , Đăng nhập , quản lý thông tin tài khoản

-Quản lý bài viết:hiện thị bài viết , tìm kiếm



Hình 2.4:Sơ đồ hệ thống

4.1.2.đặc tả usecase đăng nhập vào hệ thống:

Tên Use case: Đăng nhập vào hệ thống  
Tác nhân chính: Khách hàng, Quản trị viên  
Mô tả tóm tắt:  
Khách hàng hoặc quản trị viên nhập thông tin tài khoản để đăng nhập vào hệ thống.

Luồng sự kiện chính:

1. người nhập tên đăng nhập và mật khẩu.
2. Hệ thống kiểm tra thông tin.
3. Nếu thông tin hợp lệ, hệ thống cho phép truy cập vào tài khoản.

Luồng sự kiện phụ:

* Nếu thông tin sai, hệ thống hiển thị thông báo lỗi và yêu cầu nhập lại.

Điều kiện tiền đề:

* Tài khoản đã được đăng ký.

Điều kiện thoải mãn:

* Người dùng được truy cập vào hệ thống.

4.1.3 Đặc tả Use case đăng xuất vào hệ thống

Tên Use case: Đăng xuất hệ thống  
Tác nhân chính: Khách hàng, Quản trị viên  
Mô tả tóm tắt:  
Người dùng thoát khỏi hệ thống.

Luồng sự kiện chính:

1. Người dùng nhấn nút đăng xuất.
2. Hệ thống đóng phiên làm việc và chuyển về trang đăng nhập.

Điều kiện tiền đề:

* Người dùng đã đăng nhập.

Điều kiện hậu mãn:

* Phiên làm việc được kết thúc.

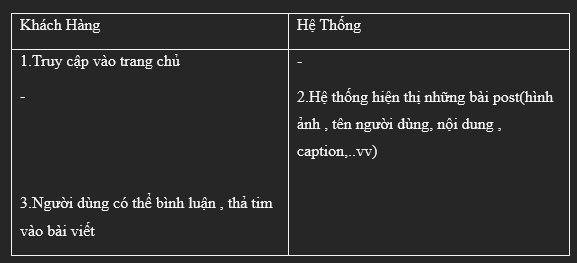
4.1.4. Use Case: Đăng ký tài khoản

* Actor: Khách hàng
* Mô tả: Người dùng tạo tài khoản mới để sử dụng các tính năng của website.
* A white page with black text

  Description automatically generated**Luồng chính:**
* Luồng phụ:
  + Nếu thông tin không hợp lệ (email đã tồn tại, mật khẩu yếu, v.v.), hệ thống sẽ hiển thị lỗi.

4.1.5.Use case Hiển thị bài viết:

* Actor: Khách hàng
* Mô tả: Người dùng xem nội dung các bài post

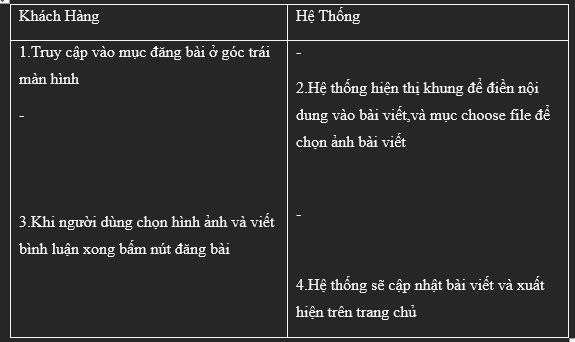


* Luồng phụ:

.Người dùng thoát ra bài post xem những bài post khác

4.1.6.Use Case :Quản lý Nội dụng

* Actor: Khách hàng
* Mô tả: Người dùng đăng bài viết
* Luồng chính:



* Luồng phụ :

.Người dùng có thể chỉnh sửa bài viết ở mục tài khoản

4.1.7. Use Case:Tìm kiếm & phân loại nội dung

* Actor:Khách hàng
* Mô tả:khách hàng bấm vào tìm kiếm ở góc trái màn hình
* Luồng chính:



* Luồng phụ:Nếu không có nội dung tìm kiếm hệ thống sẽ xuất thông báo không thấy nội dung tìm kiếm

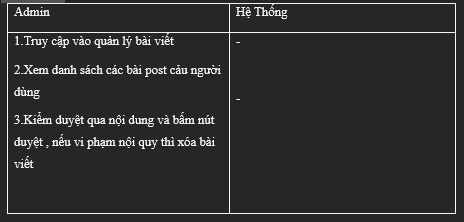
4.1.8. Use Case: Quản lý khách hàng

* Actor: Admin
* Mô tả: Admin xem, cập nhật thông tin tài khoản khách hàng hoặc xóa tài khoản.
* Luồng chính: A white paper with black text

  Description automatically generated

4.1.9.Use Case:Quản lý bài viết

* Actor: Admin
* Mô tả: Admin kiểm duyệt và xóa các đánh giá không phù hợp.
* Luồng chính:



4.1.10.Use Case: Quản lý tài khoản Admin

* Actor: Admin
* Mô tả: Admin cấp quyền hoặc xóa tài khoản admin khác.
* Luồng chính:

A screenshot of a computer

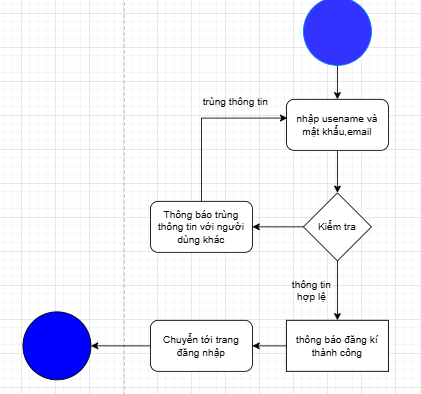
Description automatically generated

# Chương 5:Lược đồ Activity

# 5.1.Lược đồ chức năng đăng nhập:

A diagram of a company

Description automatically generated with medium confidence

5.1.2.Lược đồ chức năng đăng kí

# 5.1.3.Lược đồ resest mật khẩu:

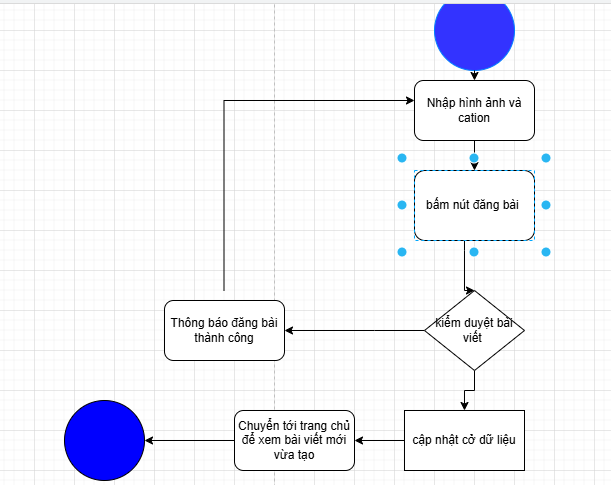
A screenshot of a diagram

Description automatically generated

5.1.4.Lược đồ chức năng tìm kiếm:

A diagram of a work flow

Description automatically generated  
5.1.5 Lược đồ chức năng tạo bài bài viết:

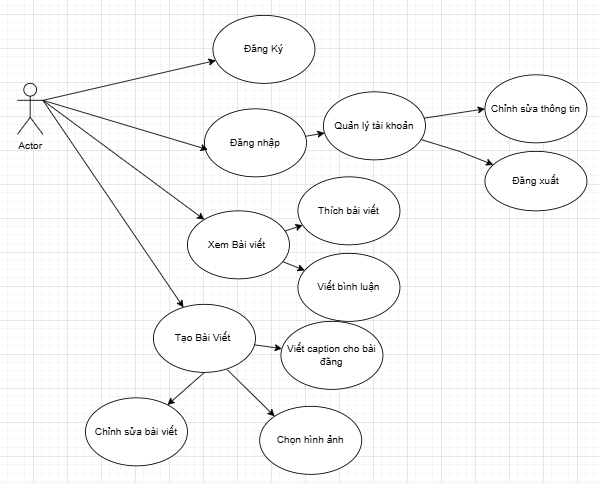


5.2.Biểu đồ thành phần hệ thống:

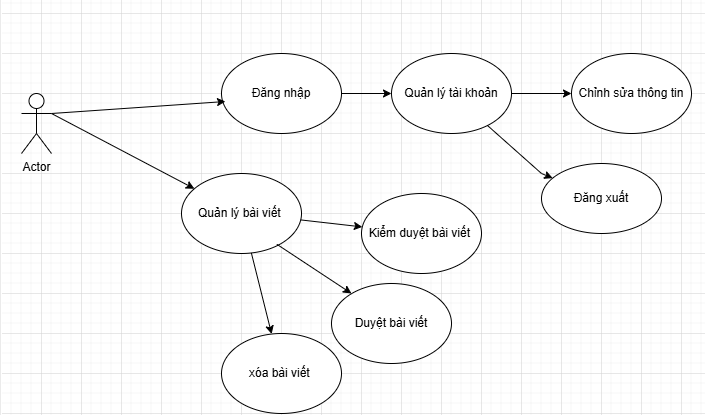
A diagram of a network

Description automatically generated with medium confidence

5.2.1.Sơ đồ khách hàng



5.2.2.Sơ Đồ admin



# CHƯƠNG 6: ÁP DỤNG MONGODB VÀO DỰ ÁN

### **1. Giới thiệu:**

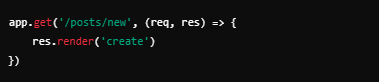
Trong chương này, **chúng em** sẽ tích hợp **MongoDB** vào ứng dụng **Node.js**, giúp người dùng có thể **tạo bài đăng mới** và lưu vào cơ sở dữ liệu. Việc này sẽ làm cho trang web có nội dung động, thay vì chỉ hiển thị dữ liệu tĩnh như trước.

### **2. Tạo Giao Diện Nhập Bài Đăng Mới:**

Đầu tiên, **chúng em** cần tạo trang cho phép người dùng nhập bài đăng mới. Để làm nhanh, **chúng em** **sao chép file** contact.ejs **thành** create.ejs, sau đó sửa nội dung:

* Thay tiêu đề thành **"Tạo bài đăng mới"**.
* Xóa các trường không cần thiết như **email, số điện thoại**, chỉ giữ lại **tiêu đề bài viết** và **nội dung**.

**Thêm đường dẫn để hiển thị trang nhập bài đăng**:  
Trong index.js, thêm đoạn sau để khi vào /posts/new, trang sẽ hiển thị:



Đồng thời, cập nhật thanh điều hướng (navbar.ejs) để có nút **"Bài đăng mới"**, giúp người dùng dễ dàng truy cập.

### **3. Sửa Lỗi Không Tải Được Tệp Tĩnh**

Khi thêm trang mới, có thể gặp lỗi **CSS và hình ảnh không hiển thị**, do các tệp tĩnh bị tham chiếu sai. Để khắc phục, **chúng em** cần sửa đường dẫn thành **đường dẫn tuyệt đối**, ví dụ:



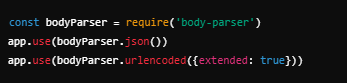
Việc này giúp trình duyệt luôn tìm đúng tệp, dù người dùng ở bất kỳ trang nào.

### 4. **Xử Lý Dữ Liệu Form Gửi Lên Server**

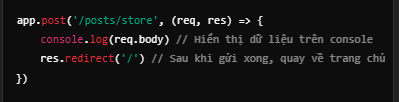
Khi người dùng nhập bài đăng và nhấn **"Gửi"**, dữ liệu sẽ được gửi lên server theo phương thức **POST**. Để xử lý dữ liệu này, ta cần cài đặt thư viện **body-parser**:



Sau đó, trong index.js, thêm:



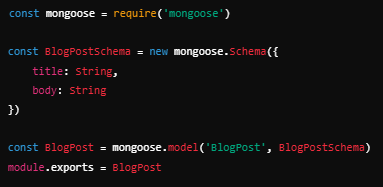
Tiếp theo, tạo route để nhận dữ liệu từ form và kiểm tra xem dữ liệu đã gửi lên chưa:



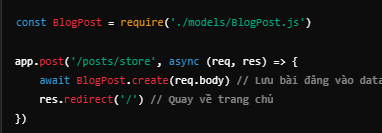
Bây giờ, khi nhập thông tin vào form và nhấn gửi, dữ liệu sẽ xuất hiện trên terminal của server.

### **5. Lưu Bài Đăng Vào MongoDB**

Sau khi kiểm tra dữ liệu đã gửi lên, bước tiếp theo là lưu bài đăng vào **MongoDB**. Trước tiên, đảm bảo đã có mô hình dữ liệu BlogPost.js:



Sau đó, trong index.js, nhập mô hình này và sửa lại route /posts/store để lưu bài viết vào MongoDB:



Chúng ta sử dụng **async/await** để đảm bảo việc lưu dữ liệu hoàn tất trước khi chuyển hướng.

### **6. Kết Quả**

* Người dùng có thể nhập bài đăng và lưu vào cơ sở dữ liệu.
* Giao diện hiển thị đúng, không bị lỗi CSS hay hình ảnh.
* Dữ liệu từ form được kiểm tra trên console trước khi lưu vào MongoDB.
* Mọi bài đăng sẽ được lưu vào **bộ sưu tập BlogPosts** và có thể được truy vấn khi cần.

Với những bước này, chúng ta đã hoàn thiện chức năng **tạo bài đăng mới** trong ứng dụng sử dụng **MongoDB và Node.js**

CHƯƠNG 7: Tải Lên Hình Ảnh Bằng Express

Trong chương này, chúng em sẽ tìm hiểu cách thêm tính năng tải ảnh lên cho bài đăng blog bằng **express-fileupload**. Điều này giúp blog sinh động hơn với những hình ảnh trực quan

### **7.1.Cài đặt express-fileupload**

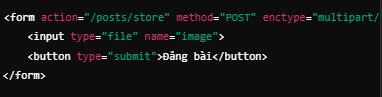
Trước tiên, cần cài đặt thư viện **express-fileupload** để xử lý việc tải ảnh: 

Sau đó,khai báo trong index.js đề sử dụng:

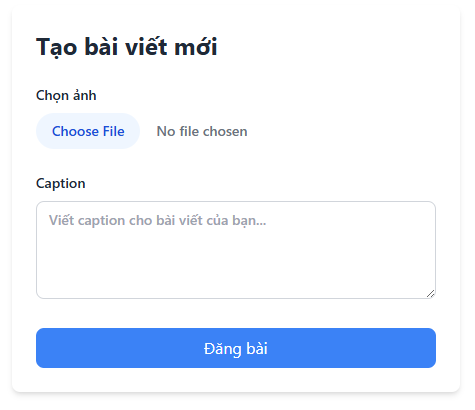


7.1.2. Thêm ô tải ảnh vào form:

Trong file create.ejs, chúng em thêm một trường để chọn ảnh:

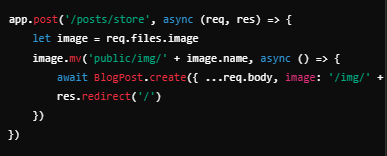


Hình ảnh ví dụ:



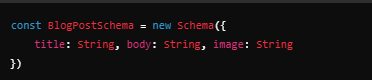
# **7.1.3.Xử lý ảnh trên server:**

Khi người dùng đăng bài, server sẽ nhận và lưu ảnh vào thư mục public/img/.



7.1.4. **Lưu đường dẫn ảnh vào database**

Trong file BlogPost.js, thêm một thuộc tính để lưu đường dẫn ảnh:

****

Bây giờ, khi tải ảnh lên, đường dẫn ảnh sẽ được lưu cùng bài viết:

7.1.5.**Hiển thị ảnh trong bài đăng**

Trong file post.ejs, thay đổi cách hiển thị ảnh để sử dụng đường dẫn từ database:

****

CHƯƠNG 8 :Tổng Kết Dự Án

Sau quá trình phát triển, website blog đã hoàn thiện với đầy đủ các tính năng cần thiết, từ đăng bài, chỉnh sửa nội dung cho đến tải ảnh minh họa, giúp cải thiện trải nghiệm người dùng. Việc áp dụng **Express.js** kết hợp với **MongoDB** giúp hệ thống hoạt động ổn định, hiệu quả và dễ mở rộng trong tương lai.

Dự án không chỉ giúp chúng em nâng cao kỹ năng lập trình web mà còn mang lại những trải nghiệm thực tế về cách thiết kế, tối ưu và triển khai một ứng dụng web hoàn chỉnh. Trong tương lai, chúng em có thể mở rộng blog với các tính năng như bình luận, tìm kiếm, phân quyền người dùng để nâng cao tính tương tác.

Mặc dù đã đạt được kết quả tốt, nhưng blog vẫn có thể được cải thiện về giao diện, bảo mật và hiệu suất. Đây là một bước tiến quan trọng trên hành trình học tập và phát triển kỹ năng lập trình web của chúng em.