**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**********

# BÁO CÁO THỰC TẬP TỐT NGHIỆP

# KHOA HỌC DỮ LIỆU

Giảng viên hướng dẫn: TS. Trương Vĩnh Linh

Sinh viên thực hiện: Ngô Hồng Thông

Mã sinh viên: 22649011 Lớp: DHKHDL18A

TP. Hồ Chí Minh, Năm 2025

# LỜI NÓI ĐẦU

Trong thời đại công nghệ số, dữ liệu không chỉ là tài sản quý giá mà còn là nguồn lực chiến lược, đóng vai trò then chốt giúp doanh nghiệp đưa ra các quyết định quan trọng. Việc thu thập, lưu trữ và xử lý dữ liệu hiệu quả trở thành nền tảng vững chắc cho các hệ thống phân tích và ứng dụng trí tuệ nhân tạo (AI) hiện đại.

Được sự phân công của Khoa Công nghệ Thông tin – Trường Đại học Công nghiệp Thành phố Hồ Chí Minh và sự đồng ý tiếp nhận của **Công Ty Cổ Phần Công Nghệ Tia Chớp Xanh**, em đã có cơ hội thực tập trong thời gian 12 tuần tại công ty. Trong suốt quá trình thực tập, em được tham gia trực tiếp vào các dự án thực tế liên quan đến xây dựng hệ thống thu thập và xử lý dữ liệu lớn, triển khai các kho dữ liệu nhằm tối ưu hóa quy trình phân tích và hỗ trợ ra quyết định.

Bên cạnh việc trau dồi chuyên môn, em còn học hỏi và rèn luyện được tác phong làm việc chuyên nghiệp, tinh thần kỷ luật và đạo đức nghề nghiệp – những yếu tố quan trọng để trở thành một kỹ sư công nghệ thông tin trong tương lai.

Em xin chân thành cảm ơn **Khoa Công nghệ Thông tin – Trường Đại học Công nghiệp TP.HCM** cùng quý thầy cô đã tạo điều kiện thuận lợi để em thực hiện kỳ thực tập này. Đồng thời, em đặc biệt biết ơn Ban lãnh đạo **Công Ty Cổ Phần Công Nghệ Tia Chớp Xanh**, các anh/chị đồng nghiệp và đặc biệt là **anh Nguyễn Đình Thanh** – người đã tận tình hướng dẫn, hỗ trợ em trong suốt quá trình thực tập.

Mặc dù đã nỗ lực hoàn thành tốt nhiệm vụ được giao, nhưng do kinh nghiệm thực tế còn hạn chế và kiến thức còn đang trong quá trình rèn luyện, báo cáo này chắc chắn không tránh khỏi những thiếu sót. Em rất mong nhận được sự góp ý của quý thầy cô để báo cáo được hoàn thiện hơn và em có thể nâng cao kiến thức, kỹ năng của bản thân.

Em xin chân thành cảm ơn.

Sinh viên

Ngô Hồng Thông

**CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**
**Độc lập – Tự do – Hạnh phúc**

# NHẬN XÉT CỦA ĐƠN VỊ THỰC TẬP

**Họ và tên sinh viên : Ngô Hồng Thông**

**Mã sinh viên : 22649011**

**1. Nhận xét chung :**

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

*Ngày ..... tháng ..... năm .....*
**NGƯỜI HƯỚNG DẪN**
*(Ký tên và đóng dấu)*

# NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

*Ngày ..... tháng ..... năm .....*

**NGƯỜI HƯỚNG DẪN**

*(Ký tên và đóng dấu)*

# MỤC LỤC

# CHƯƠNG 1. GIỚI THIỆU CHUNG VỀ ĐƠN VỊ THỰC TẬP

## 1.1 Thông tin về đơn vị thực tập

Bluebolt Software là công ty chuyên cung cấp giải pháp công nghệ toàn diện cho doanh nghiệp. Công ty sẽ tư vấn và cung cấp giải pháp về công nghệ, nhân sự phù hợp với nhu cầu chuyển đổi số riêng biệt của từng doanh nghiệp.

Trụ sở chính: 32 Đường Số 9, Phường 10, Gò Vấp, Hồ Chí Minh

Website: https://blueboltsoftware.com

Các lĩnh vực hoạt động của công ty:

- Phần mềm Công nghệ Thông tin
- Cho thuê nhân sự IT
- Tư vấn, triển khai chuyển đổi số
- Tư vấn, xây dựng kế hoạch kinh doanh công nghệ

## 1.2 Thông tin về vị trí sinh viên tham gia thực tập

Vị trí công tác: Thực tập sinh team AI Engineer

Công việc chính ban đầu tham gia xây dựng các trình thu thập dữ liệu từ các nền tảng mạng xã hội (crawler), sau đó đảm nhận vai trò chính trong việc thiết kế, phát triển, tối ưu và triển khai hệ thống backend phục vụ việc phân tích cảm xúc bài viết, bình luận và báo cáo dữ liệu.

Đặc điểm:

- Xây dựng và vận hành các trình thu thập dữ liệu từ nhiều nguồn (Facebook, TikTok, Youtube, Website, …).
- Tham gia phát triển, tối ưu và mở rộng hệ thống thu thập – phân tích dữ liệu thời gian thực

Yêu cầu:

- Kiến thức tốt về Python, Structured Query Language (SQL), Database (PostgreSQL, MongoDB), Data Visualization.
- Hiểu biết về Data Architecture, Data Driven, Computer Network, DevOps.
- Kiến thức cơ bản về thu thập dữ liệu (crawler) và xử lý dữ liệu lớn

# CHƯƠNG 2: VẤN ĐỀ SINH VIÊN THAM GIA GIẢI QUYẾT / THỰC HIỆN TẠI ĐƠN VỊ / DOANH NGHIỆP THỰC TẬP

## 2.1 Tóm tắt vấn đề sinh viên tham gia giải quyết/thực hiện tại đơn vị/Doanh nghiệp thực tập

Giai đoạn 1 – Khởi động và thu thập dữ liệu ban đầu:

- Tìm hiểu nghiệp vụ, yêu cầu công ty và tham gia đào tạo hội nhập
- Nghiên cứu phương pháp thu thập dữ liệu từ các nền tảng mạng xã hội, đặc biệt là Facebook
- Triển khai cào dữ liệu Facebook theo từ khóa người dùng nhập vào, sử dụng các công cụ như Selenium, Scrapy, BeautifulSoup

Giai đoạn 2 – Tích hợp xử lý dữ liệu realtime:

- Ghép nối service Kafka vào hệ thống crawler để nhận và truyền dữ liệu thời gian thực.
- Xử lý dữ liệu thu được từ Facebook (bài viết, bình luận…) và lưu trữ vào MongoDB hoặc cơ sở dữ liệu phù hợp

Giai đoạn 3 – Phát triển và tối ưu hệ thống Backend:

- Phân tích, thiết kế cơ sở dữ liệu (PostgreSQL/MongoDB) tối ưu cho việc mở rộng.
- Xây dựng và cập nhật các API Backend phục vụ dự án Mention.vn.
- Tái cấu trúc hệ thống Backend để cải thiện hiệu suất và khả năng chịu tải lớn

Giai đoạn 4 – Hoàn thiện và bàn giao sản phẩm:

- Tối ưu tốc độ thu thập dữ liệu, cải thiện quy trình xử lý.
- Triển khai API lên production cho bên sản phẩm để đưa ra cho người dùng đâu cuối.
- Viết báo cáo tổng kết thực tập và trình bày kết quả với giảng viên hướng dẫn

### 2.2 Tiến độ công việc thực hiện

Thực hiện 12 tuần: Từ ngày 01/06/2025

| Tuần | Thời gian | Nội dung công việc thực hiện |
|---|---|---|
| Tuần 1 | 01/06/2025– 07/06/2025 | - Tìm hiểu nghiệp vụ và các yêu cầu mà công ty đặt ra cho sinh viên thực tập.<br>- Tham gia đào tạo, hội nhập công ty. |
| Tuần 2 | 08/06/2025 – 14/06/2025 | - Nghiên cứu phương pháp thu thập dữ liệu cho nền tảng Facebook. |
| Tuần 3 | 15/06/2025 – 21/06/2025 | - Triển khai crawler thu thập dữ liệu Facebook theo từ khóa do người dùng nhập. |
| Tuần 4 | 22/06/2025 – 28/06/2025 | - Tích hợp service Kafka vào crawler Facebook để nhận keyword realtime từ người dùng. |
| Tuần 5 | 29/06/2025 – 05/07/2025 | - Phân tích và thiết kế cơ sở dữ liệu cho hệ thống. |
| Tuần 6 | 06/07/2025 – 12/07/2025 | - Xây dựng hệ thống Backend API cho dự án Mention.vn. |
| Tuần 7 | 13/07/2025 – 19/07/2025 | - Tái cấu trúc backend để cải thiện hiệu suất và khả năng chịu tải. |
| Tuần 8 | 20/07/2025 – 26/07/2025 | - Thiết kế lại cơ sở dữ liệu để hỗ trợ khả năng mở rộng dài hạn. |
| Tuần 9 | 27/07/2025 – 02/08/2025 | - Cập nhật và bổ sung các tính năng qua API backend. |
| Tuần 10 | 03/08/2025 – 09/08/2025 | - Đóng gói hệ thống backend trên môi trường Docker. |
| Tuần 11 | 10/08/2025 – 16/08/2025 | - Triển khai hệ thống lên môi trường production thông qua Nginx trên server Linux. |
| Tuần 12 | 17/08/2025 – 23/08/2025 | - Viết báo cáo thực tập và trình bày kết quả với giảng viên hướng dẫn. |

### 2.3 Tóm tắt công việc thực hiện theo tiến độ

#### 2.3.1 Tuần 1 - (01/06/2025 – 07/06/2025) - Tìm hiểu nghiệp vụ và các yêu cầu mà công ty đặt ra cho sinh viên thực tập và tham gia đào tạo, hội nhập công ty.

Làm quen môi trường làm việc tại Công ty Cổ phần Công nghệ Tia Chớp Xanh (BlueBolt Software).

Nghe phổ biến nội dung, quy định thực tập và mục tiêu của dự án Mention.vn.
Tìm hiểu mô hình tổng quan hệ thống: crawler, Kafka, message queue, backend API, database.

Được giới thiệu các công cụ và công nghệ sẽ sử dụng: Python, PostgreSQL, MongoDB, Docker, Kafka, Nginx, Git.
Cài đặt môi trường phát triển: Python environment, IDE (VSCode / PyCharm), Git config, kết nối VPN nội bộ.

#### 2.3.2 Tuần 2 - (08/06/2025 – 14/06/2025) - Nghiên cứu phương pháp thu thập dữ liệu cho nền tảng Facebook.

Nghiên cứu đặc điểm của nền tảng Facebook để phục vụ việc thu thập dữ liệu: cách cấu trúc DOM của bài viết, bình luận, tương tác.

Tìm hiểu các kỹ thuật và thư viện thu thập dữ liệu: Selenium (xử lý trang động), BeautifulSoup (parse HTML), requests API.

Kiểm thử thử nghiệm một script nhỏ để cào dữ liệu bài viết công khai từ Facebook, gồm các trường:

1. Nội dung bài viết
2. Thời gian đăng
3. Người đăng
4. Số lượng tương tác (thích, chia sẻ, bình luận)

### 2.3.3 Tuần 3 - (15/06/2025 – 21/06/2025) - Triển khai crawler thu thập dữ liệu Facebook theo từ khóa do người dùng nhập.

**Nghiên cứu và xác định yêu cầu đầu vào:**

Đầu tiên, xác định rõ thông số đầu vào cho crawler là *keyword* do người dùng cung cấp. Mỗi lần nhận được keyword, crawler sẽ thực hiện quy trình truy xuất, tổ chức dữ liệu toàn bộ bài viết trên nền tảng Facebook có chứa từ khóa này.

Thiết kế luồng crawl và lọc dữ liệu

Sử dụng thư viện Selenium để tự động mở trình duyệt, đăng nhập Facebook (sử dụng tài khoản test hoặc tài khoản API, đảm bảo bảo mật).

Triển khai module tự động nhập keyword vào ô tìm kiếm, thực hiện quét các bài viết, nhóm công khai và trang liên quan đến keyword đó.

Với mỗi kết quả hiển thị, sử dụng BeautifulSoup để parse và trích xuất thông tin bài viết (nội dung, tác giả, thời gian đăng, số lượng tương tác …).

**Xử lý kỹ thuật phân trang:**

Facebook hiển thị kết quả dạng infinite scroll, do đó sử dụng Selenium lặp lại thao tác cuộn trang (driver.execute_script("window.scrollTo...")) đến khi tải hết dữ liệu.

Kiểm tra và lưu lại điểm dừng (nếu hết kết quả hoặc không còn new post).

**Kiểm soát trùng lặp dữ liệu:**

Xây dựng cơ chế kiểm tra post ID để không lưu trùng cùng một bài hoặc comment nhiều lần.

Áp dụng bộ lọc set/hashtable trước khi ghi xuống file JSON tạm thời.

**Giới hạn tốc độ crawl để tránh bị block:**

Đặt delay hợp lý giữa các lần truy xuất (5-10s tuỳ từng thao tác), sử dụng mô-đun sleep của Python để mô phỏng hành động người dùng thật.

**Lưu trữ dữ liệu tạm thời:**

Dữ liệu thu thập được sẽ lưu từng post và comment dưới dạng file JSON, mỗi file sẽ đại diện cho một keyword để phục vụ các bước xử lý/truyền tiếp sau này.

Đảm bảo dữ liệu đạt chuẩn đầu ra: đầy đủ trường thông tin, loại bỏ dữ liệu thừa, chuẩn hóa mã hóa ký tự (UTF-8).

**Test chức năng và đánh giá kết quả:**

Chạy thử với một số keyword mẫu (ví dụ: "AI", "khởi nghiệp", "marketing") và kiểm tra đầu ra JSON.

Đánh giá mức độ đầy đủ, tính chính xác của thông tin thu thập; thống kê số lượng bài viết và bình luận được crawl thành công cho từng keyword.

#### 2.3.3.2 Cào dữ liệu từ các nguồn đã phân tích

Để thu thập dữ liệu từ các nguồn đã được phân tích ở phần trên, em đã tiến hành xây dựng các giải pháp cào dữ liệu phù hợp cho từng loại dữ liệu cụ thể. Quá trình này sử dụng kết hợp các thư viện mạnh mẽ trong Python như Selenium, và BeautifulSoup, giúp tối ưu hóa khả năng truy xuất dữ liệu từ các trang web.

***Selenium***

Selenium là công cụ tự động hóa trình duyệt web, cho phép điều khiển các hành động trên trang web như nhấp chuột, điền dữ liệu vào biểu mẫu và tương tác với các phần tử động. Đây là một giải pháp hiệu quả cho việc thu thập dữ liệu từ các trang web có nội dung được render bằng JavaScript.

**Ưu điểm**:

- Selenium hỗ trợ nhiều ngôn ngữ lập trình như Python, Java, C#, Ruby, JavaScript, Perl, v.v.
- Selenium có khả năng chạy các trình duyệt web như Firefox, Chrome, Safari, Edge, Opera, v.v.
- Selenium có thể xử lý các trang web động và thực hiện các tương tác người dùng trên trình duyệt.
- Selenium cho phép người dùng tự động hóa các thao tác trên trình duyệt như điền thông tin vào các biểu mẫu, tải xuống các tệp, v.v.

**Nhược điểm**:

- Selenium tốn tài nguyên hệ thống hơn so với các công cụ khác.
- Selenium chậm hơn so với các công cụ khác trong việc xử lý dữ liệu lớn.

***Scrapy***

Scrapy là một framework mạnh mẽ và linh hoạt trong Python, được thiết kế để xây dựng các spider thu thập dữ liệu từ các trang web và các nguồn dữ liệu khác. Khác với Selenium, Scrapy tập trung vào việc xử lý dữ liệu trên quy mô lớn, tối ưu tốc độ thu thập và quản lý dữ liệu dưới dạng pipeline.

**Ưu điểm**:

- Scrapy được thiết kế đặc biệt cho việc thu thập dữ liệu từ các trang web.
- Scrapy có thể chạy đồng thời nhiều luồng, tăng tốc độ thu thập dữ liệu.
- Scrapy có thể lưu trữ dữ liệu vào các định dạng phổ biến như JSON, CSV, XML, v.v.
- Scrapy có thể tự động xác định cấu trúc của trang web và trích xuất dữ liệu theo cấu trúc đó.

**Nhược điểm**:

- Scrapy chỉ hỗ trợ Python.

***Beautiful Soup***

BeautifulSoup là một thư viện Python mạnh mẽ cho phép trích xuất dữ liệu từ các trang web bằng cách phân tích cú pháp HTML hoặc XML. Đây là công cụ rất phù hợp để xử lý các trang web có cấu trúc HTML phức tạp hoặc cần xử lý dữ liệu chi tiết từ từng thẻ HTML cụ thể.

**Ưu điểm**:

- Beautiful Soup có khả năng trích xuất thông tin từ các tài liệu HTML và XML.
- Beautiful Soup có thể phân tích cú pháp của tài liệu và giúp trích xuất dữ liệu dễ dàng hơn.
- Beautiful Soup hỗ trợ Python và có thể được kết hợp với các thư viện Python khác để xử lý dữ liệu.

**Nhược điểm**:

- Beautiful Soup không thể xử lý các trang web động và thực hiện các tương tác người dùng trên trình duyệt.
- Beautiful Soup không hỗ trợ các tính năng như đồng bộ hóa dữ liệu, tăng tốc độ thu thập dữ liệu, v.v.

### 2.3.4 Tuần 4 - (22/06/2025 – 28/06/2025) - Tích hợp service Kafka vào crawler Facebook để nhận keyword realtime từ người dùng.

Trong tuần này, em tập trung triển khai tích hợp dịch vụ Kafka vào cụm máy cào dữ liệu (crawler) cho Facebook nhằm xây dựng một luồng thu thập dữ liệu theo thời gian thực. Mục tiêu là để hệ thống có thể ngay lập tức phản hồi và tiến hành thu thập dữ liệu khi người dùng nhập vào một từ khóa mới.

**Thiết kế luồng xử lý**

- Producer: Sau khi crawler hoàn thành việc cào dữ liệu Facebook theo các keyword có sẵn, dữ liệu thô (bài viết, bình luận, thông tin tương tác) sẽ được đóng gói thành các thông điệp JSON và đẩy vào Kafka topic chuyên dụng.
- Consumer (keyword listener): Một nhóm consumer sẽ luôn lắng nghe topic chứa từ khóa. Khi có một từ khóa mới được gửi từ phía người dùng (frontend hoặc service quản lý từ khóa), consumer sẽ lập tức nhận nhiệm vụ và kích hoạt crawler tương ứng để thu thập dữ liệu theo từ khóa đó.
- Topic cấu hình: Sử dụng ít nhất 2 topic — một cho từ khóa (keywords-topic) và một cho dữ liệu đã cào (facebook-posts-topic) để tách biệt luồng điều khiển và luồng dữ liệu, giúp dễ dàng mở rộng và bảo trì.

**Các bước triển khai kỹ thuật**

1. **Khởi tạo topic trong Kafka**:
    - keywords-topic: lưu các từ khóa người dùng nhập theo thời gian thực.
    - facebook-posts-topic: lưu dữ liệu bài viết/bình luận đã thu thập.
2. **Kết nối crawler với Kafka**:
    - Thêm module Kafka producer vào crawler để gửi dữ liệu cào được sang facebook-posts-topic.
    - Tích hợp Kafka consumer chạy song song, luôn lắng nghe keywords-topic để nhận các keyword mới.
3. **Cơ chế phản ứng realtime**:
    - Khi consumer nhận keyword mới, crawler ngay lập tức khởi chạy job thu thập dữ liệu với keyword đó, thay vì đợi lịch chạy định kỳ.
    - Áp dụng consumer group để phân phối keyword tới nhiều crawler song song, đảm bảo mỗi keyword chỉ xử lý một lần.
4. **Chống trùng và xử lý lỗi**:
    - Kiểm tra post_id trước khi lưu để tránh trùng dữ liệu.
    - Thiết lập cơ chế retry/backoff khi gặp lỗi mạng, bị giới hạn API.
    - Ghi log chi tiết mỗi khi có sự cố để hỗ trợ gỡ lỗi nhanh.

**Kết quả đạt được**

- Hệ thống đã có thể nhận keyword realtime từ người dùng và triển khai thu thập dữ liệu ngay lập tức.

- Luồng dữ liệu được đẩy qua Kafka bảo đảm tính ổn định, giảm độ trễ và cho phép mở rộng sang nhiều loại crawler khác trong tương lai.
- Thử nghiệm với các keyword mẫu ("AI", "marketing", "du lịch") cho thấy thời gian từ khi nhập keyword đến khi bắt đầu thu thập chỉ mất vài giây, dữ liệu được lưu trữ ngay vào MongoDB để sẵn sàng xử lý.

### 2.3.5 Tuần 5 – (29/06/2025 – 05/07/2025) - Phân tích và thiết kế cơ sở dữ liệu cho hệ thống.

Sau khi hệ thống crawler và Kafka đã vận hành ổn định, tuần này em tập trung vào thiết kế cơ sở dữ liệu nhằm lưu trữ, tổ chức và tối ưu truy vấn cho dữ liệu thu thập được từ Facebook và các nguồn khác.

**Mục tiêu thiết kế**

1. **Xây dựng mô hình dữ liệu có khả năng:**
    - Lưu trữ dữ liệu thô (raw data) và dữ liệu phân tích đã xử lý.
    - Hỗ trợ **truy vấn nhanh** cho dashboard, báo cáo và API backend.
    - Dễ dàng mở rộng khi bổ sung thêm nguồn dữ liệu hoặc loại phân tích mới.
2. **Áp dụng Star Schema để tối ưu cho các truy vấn phân tích (OLAP), đặc biệt trong bài toán thống kê keyword, xu hướng, sentiment.**
3. **Quy trình thực hiện**

    **Bước 1 – Thu thập yêu cầu:**

    - Làm việc với team frontend để hiểu rõ:
        - Các truy vấn chính sẽ thực hiện (ví dụ: "Top bài viết có tương tác cao theo keyword và thời gian").
        - Dạng báo cáo cần xuất (biểu đồ trend, bảng thống kê).
        - Khối lượng dữ liệu dự kiến (hàng triệu bản ghi mỗi tháng).
    - Phân loại dữ liệu:
        - Fact data: số liệu định lượng (lượt like, share, comment, số bài viết).
        - Dimension data: thông tin mô tả (keyword, nguồn dữ liệu, thời gian, tác giả…).

**Bước 2 – Lựa chọn công nghệ**

- PostgreSQL: lưu trữ dữ liệu đã xử lý (theo Star Schema) để phục vụ phân tích, báo cáo.
- MongoDB: lưu dữ liệu thô dạng JSON từ crawler để phục vụ kiểm tra hoặc xử lý lại khi cần.
- ETL pipeline: chuyển dữ liệu từ MongoDB sang PostgreSQL theo chu kỳ (batch hoặc streaming).

**Bước 3 – Thiết kế Star Schema**

Star Schema gồm:

- Bảng fact chính: fact_social_interaction
    - Chứa số liệu thống kê tương tác (likes, shares, comments, views…).
    - Khóa ngoại tới các bảng dimension.
    - Các cột chính: fact_id, date_key, keyword_key, source_key, author_key, likes, shares, comments, views, sentiment_score.
- Bảng dimension:
    - dim_keyword: từ khóa người dùng theo dõi (có thể gắn category).
    - dim_author: tác giả/nick đăng bài, kèm thông tin profile cơ bản.
    - dim_post: thông tin mô tả bài viết/video (title, url, content_type).

**Bước 4 – Quy tắc chuẩn hóa & indexing**

- Chuẩn hóa dữ liệu trước khi insert (remove null, format datetime UTC).
- Tạo index cho các trường truy vấn thường xuyên:
    - date_key, keyword_key, source_key trên bảng fact.
    - keyword_name trên bảng dim_keyword.
- Partition bảng fact theo tháng để tối ưu truy vấn thời gian.

**3. Kết quả đạt được**

- Hoàn thành bản **ERD** của Star Schema cho hệ thống Mention.vn.
- Tạo script **DDL PostgreSQL** cho toàn bộ bảng fact & dimension.
- Thử nghiệm truy vấn mẫu cho báo cáo:
    - Top 10 keyword có lượt tương tác cao nhất trong 7 ngày qua.
    - Thống kế bài viết theo keyword

### 2.3.6 Tuần 6 – (06/07/2025 – 12/07/2025) - Xây dựng hệ thống Backend API cho Mention.vn (FastAPI)

Sau khi đã thiết kế xong cơ sở dữ liệu (PostgreSQL – Star Schema), tuần này em tập trung vào việc xây dựng hệ thống Backend API để phục vụ các tính năng chính của nền tảng Mention.vn.

Mục tiêu là xây dựng một dịch vụ backend hiệu suất cao, cung cấp API cho frontend/dashboard và các service khác, đồng thời kết nối trực tiếp với cơ sở dữ liệu và pipeline xử lý dữ liệu.

**Công nghệ và lý do lựa chọn**

- **FastAPI**: framework Python hiện đại, hỗ trợ async/await, hiệu suất cao, tích hợp Swagger UI tự động cho tài liệu API.
- **Uvicorn**: ASGI server chạy ứng dụng FastAPI.
- **SQLAlchemy + asyncpg**: ORM/driver kết nối PostgreSQL dạng bất đồng bộ.
- **Pydantic**: định nghĩa và validate schema dữ liệu vào/ra.
- **Docker**: đóng gói service backend để triển khai nhanh.

**Cấu trúc dự án Backend**

Thư mục chính:

```
/backend
├── app
│   ├── main.py          # Entry point FastAPI
│   ├── config.py        # Cấu hình env, DB connection
│   ├── models/          # Định nghĩa ORM model (SQLAlchemy)
│   ├── schemas/         # Định nghĩa Pydantic schema
│   ├── routers/         # Các router API
│   ├── services/        # Business logic
│   └── utils/           # Hàm tiện ích, middleware
├── requirements.txt
└── Dockerfile
```

**Các nhóm API chính**

1. **Quản lý từ khóa (Keyword Management)**

- POST /keywords → thêm từ khóa mới, đồng thời publish vào Kafka keywords-topic.
- GET /keywords → lấy danh sách từ khóa đang theo dõi.
- DELETE /keywords/{id} → xóa từ khóa.

2. **Truy xuất dữ liệu đã thu thập**
   - GET /posts → lấy danh sách bài viết theo bộ lọc (keyword, thời gian, nguồn).
   - GET /posts/{id} → chi tiết một bài viết, bao gồm nội dung, tác giả, số liệu tương tác.
3. **Báo cáo & thống kê**
   - GET /stats/keyword-trend → thống kê số bài viết, tương tác theo keyword.
   - GET /stats/sentiment → thống kê sentiment (positive/negative/neutral) theo keyword và nguồn dữ liệu.
   - GET /stats/top-authors → tác giả có tương tác cao nhất theo keyword.
4. **Hỗ trợ kiểm thử**
   - GET /health → kiểm tra tình trạng service.

**Các bước triển khai**

1. **Khởi tạo dự án FastAPI**
   - pip install fastapi uvicorn sqlalchemy asyncpg motor pydantic python-dotenv
   - fastapi startproject app
2. **Kết nối cơ sở dữ liệu**
   - PostgreSQL (fact & dimension tables): dùng SQLAlchemy + asyncpg.
   - MongoDB (raw data): dùng Motor.
3. **Xây dựng model & schema**
   - ORM model mapping bảng fact/dimension từ Star Schema.
   - Pydantic schema cho request/response đảm bảo dữ liệu vào/ra chuẩn hóa.
4. **Tạo router API**
   - Chia thành các module (keywords.py, posts.py, stats.py) trong /routers.
   - Sử dụng dependency injection của FastAPI để quản lý DB session.
5. **Tích hợp Kafka**
   - Khi thêm từ khóa mới, backend publish keyword vào keywords-topic.

- Tạo background task hoặc service riêng để lắng nghe kết quả crawl.

6. **Triển khai bảo mật**
    - Thêm JWT Auth cho API quan trọng.
    - CORS middleware cho phép frontend truy cập.
7. **Docker hóa service**
    - Tạo Dockerfile cho backend.
    - Chạy cùng Kafka, DB qua docker-compose.

**Kết quả đạt được**

- Hoàn thiện bộ API nền tảng cho Mention.vn:
- CRUD từ khóa, truy vấn dữ liệu, thống kê, báo cáo.
- API tài liệu tự động qua Swagger UI (/docs).
- Kết nối hoàn chỉnh tới PostgreSQL
- Service backend đóng gói Docker, sẵn sàng triển khai production.

### 2.3.7 Tuần 7 – (13/07/2025 – 19/07/2025) - Tái cấu trúc backend để cải thiện hiệu suất và khả năng chịu tải

Sau khi hoàn thiện API cơ bản ở tuần 6 bằng FastAPI, tuần này em tập trung vào tối ưu kiến trúc backend để hệ thống có thể xử lý lượng truy cập và dữ liệu lớn hơn, đồng thời đảm bảo độ ổn định khi chạy lâu dài.

**Mục tiêu tái cấu trúc**

- Giảm độ trễ phản hồi của API.
- Tăng khả năng chịu tải khi số lượng request tăng đột biến.
- Hạn chế tình trạng nghẽn tài nguyên khi truy vấn dữ liệu lớn.
- Cải thiện khả năng scale-out (mở rộng ngang) khi triển khai nhiều instance.

**Các vấn đề gặp phải trước khi tối ưu**

- Truy vấn PostgreSQL chưa được tối ưu, đặc biệt với các bảng fact lớn.
- Xử lý logic phức tạp trong request chính (blocking), chưa tách sang background tasks.
- Chưa có caching layer → các truy vấn thống kê bị lặp lại nhiều lần.
- Ứng dụng chạy single worker trên Uvicorn → chưa khai thác hết tài nguyên CPU.

**Các bước tái cấu trúc**

**1. Tối ưu truy vấn và cấu trúc DB**

- Tạo index cho các trường lọc nhiều nhất (date_key, keyword_key, source_key).
- Áp dụng partitioning cho bảng fact theo tháng để giảm dung lượng mỗi truy vấn.
- Sử dụng materialized view cho các thống kê nặng (vd: tổng tương tác theo keyword mỗi ngày) và refresh định kỳ.

**2. Tách logic xử lý nặng ra background**

- Sử dụng Redis để xử lý các tác vụ tốn thời gian (vd: tổng hợp báo cáo, phân tích sentiment).
- API chỉ nhận request, push task vào queue và trả về response ngay, kết quả được lưu vào DB để frontend truy xuất sau.

**3. Thêm caching layer**

- Cài Redis để cache kết quả truy vấn thống kê hoặc dữ liệu ít thay đổi (vd: danh sách keyword, top tác giả tuần).
- Thiết lập TTL (Time To Live) hợp lý để tránh dữ liệu cũ quá lâu.

**4. Tối ưu cấu hình FastAPI + Uvicorn/Gunicorn**

- Chạy multi-worker bằng Gunicorn với uvicorn.workers.UvicornWorker, cấu hình worker theo số core CPU.
- Bật keep-alive và tối ưu timeout để tránh kết nối treo.

**5. Chuẩn bị cho scale-out**

- Thiết lập Docker để có thể chạy nhiều instance backend và load balancing.
- Sử dụng Nginx reverse proxy để phân phối request giữa các instance.

**Kết quả đạt được**

- Thời gian phản hồi API giảm trung bình 35–50% cho các endpoint thống kê.
- Hệ thống có thể chịu tải gấp ~3 lần so với trước (từ ~200 req/s → ~600 req/s trong thử nghiệm nội bộ).
- Giảm tình trạng timeout khi truy vấn dữ liệu lớn.
- Kiến trúc backend trở nên module hóa, dễ dàng mở rộng tính năng hoặc scale thêm node mới.

### 2.3.8 Tuần 8 – (20/07/2025 – 26/07/2025) - Thiết kế lại cơ sở dữ liệu để hỗ trợ khả năng mở rộng dài hạn.

Sau khi backend đã được tái cấu trúc để đạt hiệu suất và khả năng chịu tải tốt hơn ở tuần 7, tuần này em tập trung mở rộng hệ thống API nhằm đáp ứng các yêu cầu mới từ phía sản phẩm và cải thiện trải nghiệm người dùng.

**Mục tiêu**

- Bổ sung các endpoint mới phục vụ báo cáo, phân tích nâng cao.
- Mở rộng khả năng tìm kiếm, lọc dữ liệu trên các API hiện có.
- Tích hợp thêm dữ liệu từ các nguồn mới vào luồng backend.
- Cải thiện bảo mật và tài liệu API.

**Các tính năng mới được bổ sung**

1. **API báo cáo & thống kê nâng cao**
    - GET /stats/keyword-trend: Thống kê xu hướng bài viết và tương tác theo keyword trong khoảng thời gian tuỳ chọn.
    - GET /stats/sentiment: Thống kê phân loại sentiment (positive, neutral, negative) theo keyword và nguồn dữ liệu.
    - GET /stats/top-authors: Liệt kê các tác giả có tương tác cao nhất cho từng keyword.
    - GET /stats/source-distribution: Biểu đồ phân bổ dữ liệu theo nguồn (Facebook, YouTube, Website…).
2. **API quản lý & tìm kiếm dữ liệu**
    - Bổ sung bộ lọc nâng cao cho GET /posts:
        - Lọc theo khoảng thời gian (from_date, to_date).
        - Lọc theo nguồn (source).
        - Lọc theo sentiment (sentiment).
    - Phân trang & sắp xếp: thêm tham số page, size, sort_by (vd: newest, interaction_count).
    - Tìm kiếm toàn văn (full-text search) theo nội dung bài viết trong PostgreSQL và MongoDB.
3. **Tích hợp dữ liệu từ nguồn mới**
    - Thêm hỗ trợ dữ liệu từ Twitter/X vào pipeline backend.

- Cập nhật dim_source và các bảng liên quan để nhận diện nguồn mới.

**4. Bảo mật & quyền truy cập**

- Áp dụng JWT Authentication cho các API nhạy cảm (quản lý từ khóa, xuất dữ liệu).
- Thêm Role-based Access Control (RBAC): phân quyền admin, analyst, viewer.
- Kích hoạt CORS policy chặt chẽ hơn, chỉ cho phép domain frontend của Mention.vn.

**5. Cải thiện tài liệu API**

- Tự động generate tài liệu Swagger UI tại /docs và ReDoc tại /redoc.
- Gắn ví dụ response cho từng endpoint để hỗ trợ team frontend.

**Các bước triển khai**

- Làm việc với team sản phẩm để xác định yêu cầu tính năng mới.
- Cập nhật schema Pydantic để phản ánh dữ liệu mới (ví dụ thêm trường sentiment_score).
- Viết migration script cho PostgreSQL (thêm bảng, cột mới).
- Cập nhật service layer để xử lý logic nghiệp vụ mới.
- Viết test case cho từng API (pytest + HTTPX).
- Triển khai phiên bản mới lên môi trường staging, kiểm thử, sau đó lên production.

**Kết quả đạt được**

- Hoàn thiện 4 nhóm API mới phục vụ phân tích, báo cáo và quản trị dữ liệu.
- API hiện tại được mở rộng bộ lọc và phân trang, tăng tốc độ truy vấn nhờ index mới.
- Hệ thống hỗ trợ thêm nguồn Twitter/X, mở rộng phạm vi thu thập dữ liệu.
- Nâng cao bảo mật backend với JWT + RBAC.
- Bộ tài liệu API rõ ràng, hỗ trợ tốt cho team frontend và khách hàng nội bộ.

### 2.3.9 Tuần 9 – (27/07/2025 – 02/08/2025) – Cập nhật và bổ sung các tính năng qua API Backend

Trong tuần này, em tập trung vào việc xây dựng và hoàn thiện các API còn lại nhằm bổ sung đầy đủ tính năng cho phía Frontend. Các công việc chính bao gồm:

- Hoàn thiện các API còn thiếu cho module quản lý từ khóa, bài viết và thống kê, đảm bảo frontend có thể gọi trực tiếp để hiển thị dữ liệu.
- Bổ sung tham số lọc và phân trang (theo từ khóa, nguồn dữ liệu, thời gian) để đáp ứng nhu cầu hiển thị dữ liệu động và trực quan trên giao diện.
- Xây dựng thêm các API thống kê như: xu hướng theo keyword, top tác giả, phân bố nguồn dữ liệu, hỗ trợ các biểu đồ báo cáo.
- Cập nhật tài liệu Swagger UI cho toàn bộ API, giúp team frontend dễ dàng kiểm thử và tích hợp.

Kết quả: Đến cuối tuần, hệ thống backend đã có đầy đủ các endpoint cần thiết, các API trả về dữ liệu đúng yêu cầu và được frontend tích hợp thành công, tạo nền tảng cho việc hoàn thiện giao diện người dùng ở giai đoạn sau.

### 2.3.10 Tuần 10 – (03/08/2025 – 09/08/2025) – Đóng gói hệ thống Backend trên môi trường Docker

Sau khi hoàn thiện các API backend và tối ưu hiệu suất, tuần này em tập trung đóng gói toàn bộ hệ thống backend (FastAPI + Kafka + PostgreSQL + MongoDB + Redis) trên môi trường Docker. Mục tiêu là giúp triển khai nhanh chóng, dễ dàng mở rộng và đảm bảo môi trường chạy đồng nhất giữa development, staging và production.

**Mục tiêu**

- Đóng gói backend service thành container Docker để dễ triển khai trên mọi máy chủ.
- Tích hợp các service phụ trợ (DB, Kafka, Redis…) trong cùng một môi trường điều phối.
- Đảm bảo cấu hình backend có thể mở rộng (scale-out) mà không xung đột môi trường.

**Công nghệ sử dụng**

- Docker: Đóng gói ứng dụng backend thành container.
- Docker Compose: Quản lý multi-container (backend, DB, Kafka, Redis).

- .env file: Cấu hình biến môi trường (DB URI, Kafka broker, Redis host…).

**Các bước triển khai**

- **Viết Dockerfile cho Backend**

```
FROM python:3.11-slim
# Cài đặt thư viện cần thiết
WORKDIR /app
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt
# Copy code vào container
COPY . .
# Expose port và chạy app
EXPOSE 8000
CMD ["uvicorn", "app.main:app", "--host", "0.0.0.0", "--port", "8000"]
```

- **Cấu hình docker-compose.yml**

```
version: '3.8'
services:
  backend:
    build: .
    container_name: mention_backend
    env_file: .env
    ports:
      - "8000:8000"
    depends_on:
      - postgres
      - redis
  redis:
    image: redis:7
volumes:
  postgres_data:
  mongo_data:
```

**Tích hợp biến môi trường**

Sử dụng .env để lưu các thông số kết nối:

- POSTGRES_URI=postgresql+asyncpg://mention:mention_pass@postgres:5432/mention_db
- REDIS_URI=redis://redis:6379

**Build và chạy**

- docker compose up --build

**Kết quả đạt được**

- Backend và toàn bộ service phụ trợ chạy ổn định trong môi trường Docker.
- Thời gian triển khai trên máy chủ mới giảm từ vài giờ xuống vài phút.
- Môi trường dev/staging/production đồng nhất → giảm lỗi do khác cấu hình.

### 2.3.11 Tuần 11 – (10/08/2025 – 16/08/2025) – Triển khai hệ thống lên môi trường Production thông qua Nginx trên server Linux

Sau khi đã đóng gói backend và các service phụ trợ trong Docker (Tuần 10), tuần này em tập trung triển khai hệ thống lên server Linux ở môi trường production, sử dụng Nginx làm reverse proxy và load balancer.

Mục tiêu

- Đưa hệ thống backend và các service lên máy chủ production.
- Cấu hình Nginx để điều phối request tới các container backend.
- Đảm bảo bảo mật kết nối (HTTPS), khả năng chịu tải và khả năng mở rộng.

**Môi trường triển khai**

- OS: Ubuntu Server 22.04 LTS.
- Server: Cấu hình 8 vCPU, 16GB RAM, SSD 256GB.
- Docker Engine + Docker Compose: chạy toàn bộ stack backend, Redis.
- Nginx: reverse proxy, SSL termination, load balancing.
- Certbot: cài đặt SSL miễn phí từ Let’s Encrypt.

**Các bước triển khai**

**1. Chuẩn bị server**

```
# Cập nhật hệ thống
sudo apt update && sudo apt upgrade -y
# Cài đặt Docker và Docker Compose
sudo apt install docker.io docker-compose -y
# Cài đặt Nginx và Certbot
```

sudo apt install nginx certbot python3-certbot-nginx -y

**2. Triển khai stack Docker**

- Upload docker-compose.yml, .env, Dockerfile lên server.
- Chạy: docker compose up -d

**3. Cấu hình Nginx reverse proxy**

File /etc/nginx/sites-available/mention.conf:

```
server {
    listen 80;
    server_name mention.vn;
    location / {
        proxy_pass http://127.0.0.1:8000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
    }
}
```

**4. Kích hoạt SSL**

sudo certbot --nginx -d mention.vn

**5. Tối ưu Nginx cho hiệu suất**

- Bật gzip.
- Tăng worker_connections.
- Thêm caching layer cho static content.

**Kết quả đạt được**

- Hệ thống production hoạt động ổn định trên domain mention.vn.
- API backend trả kết quả nhanh, đáp ứng hàng trăm request/s.
- Triển khai bảo mật HTTPS thành công, đạt chuẩn A trên SSL Labs.
- Cấu trúc hạ tầng sẵn sàng scale-out khi cần.

### 2.3.12 Tuần 12 – (17/08/2025 – 23/08/2025) – Viết báo cáo thực tập và trình bày kết quả

**Mục tiêu**

- Hoàn thiện báo cáo thực tập theo chuẩn khoa.
- Chuẩn bị nội dung trình bày cho buổi bảo vệ.

**Các công việc thực hiện**

**1. Hoàn thiện báo cáo**

- Thu thập log, screenshot hệ thống, sơ đồ kiến trúc để đưa vào phụ lục.
- Viết các chương còn thiếu: Nhận xét – Đánh giá, Kết luận, Tài liệu tham khảo.
- Định dạng lại toàn bộ file Word: font Times New Roman, size 13, line spacing 1.5, canh lề chuẩn.

**2. Chuẩn bị slide thuyết trình**

- Nội dung gồm:
    - Giới thiệu công ty và dự án Mention.vn.
    - Quy trình thu thập và xử lý dữ liệu (Crawler – Kafka – DB).
    - Backend API với FastAPI, Star Schema DB.
    - Hạ tầng triển khai Docker + Nginx Production.
    - Kết quả đạt được và hướng phát triển.
- Slide thiết kế ngắn gọn, hình ảnh trực quan, dùng sơ đồ kiến trúc để minh họa.

**3. Luyện tập trình bày**

- Tập thuyết trình trong 10–12 phút.
- Chuẩn bị trả lời các câu hỏi liên quan đến:
    - Lý do chọn Star Schema thay vì Snowflake.
    - Cách Kafka đảm bảo realtime.
    - Chiến lược tối ưu backend để chịu tải.

**3. Kết quả đạt được**

- Báo cáo hoàn chỉnh, đúng chuẩn định dạng và nội dung yêu cầu.
- Slide thuyết trình rõ ràng, trực quan.
- Sẵn sàng trình bày và giải đáp thắc mắc của giảng viên.

# CHƯƠNG 3. NHẬN XÉT, ĐÁNH GIÁ QUÁ TRÌNH THỰC TẬP

## 3.1 Kết quả đạt được

Sau khoảng thời gian thực tập tại Công ty Công nghệ Tia chớp xanh, em đã thu được nhiều kinh nghiệm làm việc cũng như kinh nghiệm sống tại môi trường doanh nghiệp. Tiếp thu được những kiến thức mới cũng như kinh nghiệm giải quyết các vấn đề. Đồng thời có thể vận dụng những kiến thực được dạy ở trường vào công việc. Các kỹ năng và kiến thức đã được trau dồi cụ thể như sau:

**Kiến thức:**

- Nắm vững kiến thức nền tảng về Data Engineer, bao gồm các nguyên tắc hoạt động, hệ thống, các kỹ thuật xử lý, lưu trữ dữ liệu, thu thập dữ liệu.
- Sử dụng FastAPI để xây dựng Backend API, kết nối với DB và Kafka, đồng thời cung cấp endpoint cho frontend, kiến thức về Database (PostgreSQL, MongoDB).
- Có kiến thức thực tế về triển khai hệ thống trên Linux server, cấu hình Nginx reverse proxy, SSL Certbot, và quản lý service bằng Docker Compose.

**Kỹ năng:**

- Kỹ năng lập trình: Phát triển crawler, xây dựng API backend với FastAP
- Kỹ năng dữ liệu: Thiết kế và tối ưu hóa cơ sở dữ liệu, sử dụng chỉ mục, partition, materialized view để tăng tốc truy vấn
- Kỹ năng hệ thống: Tạo và quản lý container bằng Docker, triển khai cụm dịch vụ (Kafka, MongoDB, PostgreSQL, Redis) trong môi trường Docker Compose.
- Kỹ năng giao tiếp: Có khả năng giao tiếp hiệu quả với các thành viên trong nhóm dự án, trình bày kết quả công việc một cách rõ ràng, súc tích.

**Kinh nghiệm:**

- Tham gia phát triển một hệ thống hoàn chỉnh, từ giai đoạn phân tích yêu cầu, thiết kế hệ thống, lập trình, huấn luyện đến triển khai và vận hành.
- Có kinh nghiệm làm việc với các công cụ và thư viện hỗ trợ thu thập dữ liệu, cũng như các công cụ về dữ liệu khác.
- Hiểu biết về quy trình phát triển phần mềm và các phương pháp quản lý dự án.

**Về kỹ năng mềm:**

- Kỹ năng làm việc nhóm: Có khả năng hợp tác hiệu quả với các thành viên trong nhóm để hoàn thành mục tiêu chung.
- Kỹ năng giải quyết vấn đề: Có khả năng phân tích và giải quyết các vấn đề phát sinh trong quá trình thực tập.
- Kỹ năng học tập: Có khả năng tự học hỏi và cập nhật kiến thức mới liên tục.
- Kỹ năng thích nghi: Có khả năng thích nghi với môi trường làm việc mới và hoàn thành tốt các nhiệm vụ được giao.

**Ngoài ra:**

- Tham gia các hoạt động ngoại khóa do công ty tổ chức, góp phần xây dựng môi trường làm việc năng động và gắn kết.
- Có cơ hội học hỏi từ những chuyên gia giàu kinh nghiệm trong lĩnh vực công nghệ thông tin.

## 3.2 Kết luận

Thời gian thực tập tại Công ty đã mang lại cho em những trải nghiệm vô cùng quý giá, không chỉ giúp em củng cố kiến thức đã học mà còn mở rộng thêm nhiều kỹ năng thực tế cần thiết cho công việc sau này. Từ việc triển khai hệ thống Backend cho đến thiết lập các dịch vụ Kafka và quản lý dữ liệu trong MongoDB, mỗi công việc đều là một bài học thực tế đầy thử thách và bổ ích.

Em xin gửi lời cảm ơn chân thành đến anh Nguyễn Đình Thanh – người đã luôn tận tình hướng dẫn, giải đáp những thắc mắc và chia sẻ những kinh nghiệm quý báu trong suốt quá trình thực tập. Sự hỗ trợ và động viên từ anh đã giúp em tự tin hơn trong từng bước hoàn thành các nhiệm vụ được giao.

Em xin chân thành cảm ơn công ty đã tạo điều kiện cho em được tham gia thực tập và trải nghiệm môi trường làm việc thực tế. Đây thực sự là bước đệm quan trọng để em có thể tiếp tục phát triển bản thân và chuẩn bị cho những thử thách lớn hơn trong tương lai.

# TÀI LIỆU THAM KHẢO


1. Docker, Inc. (không có năm). *"Docker Documentation"*. docs.docker.com.
2. Richardson, Leonard (không có năm). *"Beautiful Soup Documentation"*. crummy.com.
3. SeleniumHQ (không có năm). *"Selenium Documentation"*. selenium.dev.
4. Ramírez, Sebastián (không có năm). *"FastAPI Documentation – Fast, High-Performance Web Framework for Building APIs"*. fastapi.tiangolo.com.
5. Patairya, Dilip Kumar & Whitfield, Brennan (2024). *"Star Schema vs. Snowflake Schema Explained"*. builtin.com. builtin.com.
6. Tom (2024). *"Optimizing FastAPI for High Performance"*. medium.com.
7. tiangolo (không có năm). *"uvicorn-gunicorn-fastapi-docker"*. github.com.
8. Sematext (không có năm). *"PostgreSQL Performance Tuning: Partitioning and Indexing"*. sematext.com.
9. Bhuyan, Aditya Pratap (2024). *"Best Practices for PostgreSQL Database Design to Ensure Performance, Scalability, and Efficiency"*. dev.to dev.to.
10. Manoharan, Sivakumar (2024). *"Caching in FastAPI: Unlocking High-Performance Development"*. dev.to. dev.to.
11. F5 NGINX (không có năm). *"NGINX Reverse Proxy"*. docs.nginx.com. docs.nginx.comdocs.nginx.com.
12. Vivesh (2024). *"An In-Depth Look at Nginx: The Modern Web Server and Its Applications in Web and Application Backend Infrastructure"*. dev.to. dev.todev.to.
13. Katyal, Anchal (2024). *"Optimizing Nginx for High-Traffic Systems: A Guide to Configuration & Monitoring"*. hackernoon.com. hackernoon.com.