**BỘ GIÁO DỤC VÀ ĐÀO TẠO BỘ NÔNG NGHIỆP VÀ PTNT**

**TRƯỜNG ĐẠI HỌC THỦY LỢI**

****

PHÙNG KÍ TÀI

**XÂY DỰNG WEBSITE VÀ TÍCH HỢP VỚI CHATBOT PHỤC VỤ TRUYỀN THÔNG VÀ GIỚI THIỆU ĐỊA ĐIỂM DI TÍCH LỊCH SỬ CỦA THỦ ĐÔ HÀ NỘI**

ĐỒ ÁN TỐT NGHIỆP

HÀ NỘI, NĂM 2024

**BỘ GIÁO DỤC VÀ ĐÀO TẠO BỘ NÔNG NGHIỆP VÀ PTNT**

**TRƯỜNG ĐẠI HỌC THỦY LỢI**

PHÙNG KÍ TÀI

**XÂY DỰNG WEBSITE VÀ TÍCH HỢP VỚI CHATBOT PHỤC VỤ TRUYỀN THÔNG VÀ GIỚI THIỆU ĐỊA ĐIỂM DI TÍCH LỊCH SỬ CỦA THỦ ĐÔ HÀ NỘI**

|  |  |
| --- | --- |
| Ngành: | Công nghệ thông tin |
| Mã ngành: | 7480201 |

|  |  |
| --- | --- |
| NGƯỜI HƯỚNG DẪN: | **Ths. Vũ Thành Vinh** |

HÀ NỘI, NĂM 2024

|  |  |
| --- | --- |
|  | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh phúc**  --------------------  **NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP** |

|  |  |
| --- | --- |
| Họ tên sinh viên: **Phùng Kí Tài** | Hệ đào tạo: Đại học chính quy |
| Lớp: **62TH4** | Khoa: Công nghệ thông tin |
| Mã sinh viên: **2051063445** | Ngành: Công nghệ thông tin |

1. **TÊN ĐỀ TÀI**

XÂY DỰNG WEBSITE VÀ TÍCH HỢP VỚI CHATBOT PHỤC VỤ TRUYỀN THÔNG VÀ GIỚI THIỆU ĐỊA ĐIỂM DI TÍCH LỊCH SỬ CỦA THỦ ĐÔ HÀ NỘI

1. **CÁC TÀI LIỆU CƠ BẢN**

|  |  |
| --- | --- |
| [1] | *React*. (n.d.).<https://react.dev/> |
| [2] | *Index | Node.js v21.7.3 Documentation*. (n.d.). https://nodejs.org/docs/latest/api/ |
| [3] | *Frameworks built on Express*. (n.d.). https://expressjs.com/en/resources/frameworks.html |
| [4] | *MongoDB documentation*. (n.d.). MongoDB Documentation.https://www.mongodb.com/docs/ |
| [5] | *Cultural heritage: 7 successes of UNESCO’s preservation work*. (2023, July 18). UNESCO. https://www.unesco.org/en/cultural-heritage-7-successes-unescos-preservation-work |
| [6] | *Tourism at World Heritage Sites – Challenges and Opportunities: International tourism seminar, Çeşme (Izmir), Turkey, 26 March 2013*. (2015).<https://doi.org/10.18111/9789284416608> |
| [7] | *Cultural heritage: 7 successes of UNESCO’s preservation work*. (2023b, July 18). UNESCO. https://www.unesco.org/en/cultural-heritage-7-successes-unescos-preservation-work |
| [8] | Phan, D. (2022, December 30). *The best historical sites to visit in Vietnam*. Culture Trip.<https://theculturetrip.com/asia/vietnam/articles/the-10-best-historical-sites-to-visit-in-vietnam> |
| [9] | *OpenAI Platform*. (n.d.). https://platform.openai.com/docs/introduction |

1. **NỘI DUNG CÁC PHẦN THUYẾT MINH VÀ TÍNH TOÁN**

|  |
| --- |
| **NỘI DUNG** |
| Chương 1: Cơ sở lý thuyết |
| Chương 2: Thiết kế và xây dựng hệ thống |
| Chương 3: Kết quả và đánh giá kết luận |

1. **BẢNG BIỂU, SƠ ĐỒ, HÌNH:** Có 48 hình, 11 bảng
2. **NGÀY GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP**

Ngày ............ Tháng ......... Năm 2024

|  |  |
| --- | --- |
| **Trưởng Bộ môn**  *(Ký và ghi rõ Họ tên)* | **Giáo viên hướng dẫn chính**  *(Ký và ghi rõ Họ tên)* |

Nhiệm vụ Đồ án tốt nghiệp đã được Hội đồng thi tốt nghiệp của Khoa thông qua

Ngày. . . . . Tháng . . . . Năm 2024

**Chủ tịch Hội đồng**

*(Ký và ghi rõ Họ tên)*

Sinh viên đã hoàn thành và nộp bản Đồ án tốt nghiệp cho Hội đồng thi Ngày... Tháng... Năm 2024

**Sinh viên làm Đồ án tốt nghiệp**

***(KÝ VÀ GHI RÕ HỌ TÊN)***

|  |  |
| --- | --- |
|  | **TRƯỜNG ĐẠI HỌC THUỶ LỢI**  **KHOA CÔNG NGHỆ THÔNG TIN**  --------------------  **BẢN TÓM TẮT ĐỀ CƯƠNG ĐỒ ÁN TỐT NGHIỆP** |

**TÊN ĐỀ TÀI:** XÂY DỰNG WEBSITE VÀ TÍCH HỢP VỚI CHATBOT PHỤC VỤ TRUYỀN THÔNG VÀ GIỚI THIỆU ĐỊA ĐIỂM DI TÍCH LỊCH SỬ CỦA THỦ ĐÔ HÀ NỘI

Sinh viên thực hiện: Phùng Kí Tài

Lớp: 62TH4

Mã sinh viên: 2051063445

Số điện thoại: 0346909328

Email: 2051063445@e.tlu.edu.vn

Giáo viên hướng dẫn: Ths. Vũ Thành Vinh

*Thời gian thực hiện*: … *tuần:* *từ ngày*: ………… *đến ngày*: ……………….

**TÓM TẮT ĐỀ TÀI**

Cố đô ngàn năm văn hiến Hà Nội của Việt Nam là nơi lưu giữ những di sản văn hóa và lịch sử phong phú. Từ Lăng Chủ tịch Hồ Chí Minh, biểu tượng của lòng yêu nước và tự hào dân tộc, đến Nhà tù Hỏa Lò, nơi ghi dấu những trang sử đau thương nhưng cũng đầy kiêu hãnh, mỗi di tích ở Hà Nội đều là một chứng nhân sống động của lịch sử.

Tuy nhiên, việc tiếp cận và tìm hiểu sâu về những di tích này không phải lúc nào cũng thuận lợi cho du khách. Một số khó khăn mà du khách thường gặp phải bao gồm thông tin không đầy đủ hoặc không cập nhật trên internet, rào cản ngôn ngữ, và thiếu các tour du lịch được tổ chức chuyên nghiệp. Đặc biệt, trong bối cảnh du lịch đang phục hồi sau đại dịch, Hà Nội đang đối mặt với cả cơ hội và thách thức trong việc quản lý lượng khách du lịch tăng cao, đồng thời nâng cao chất lượng dịch vụ và trải nghiệm cho du khách.

Trên khắp thế giới, việc bảo tồn và tiếp cận các di tích lịch sử đang đối mặt với nhiều thách thức. Từ việc bảo vệ di sản văn hóa trước những tác động của biến đổi khí hậu, đến việc duy trì nguyên trạng của các địa danh lịch sử trong bối cảnh đô thị hóa và phát triển kinh tế. Các di sản này không chỉ là những bảo tàng sống động của quá khứ mà còn là nguồn cảm hứng cho sự sáng tạo và đổi mới trong tương lai.

Ở Việt Nam, sự phát triển nhanh chóng của đô thị đã tạo ra những áp lực lớn đối với việc bảo tồn các di tích lịch sử. Ví dụ, Khu di tích Cố đô Huế đang phải đối mặt với những thách thức từ sự mở rộng đô thị, ảnh hưởng đến cảnh quan và giá trị văn hóa của di sản. Bên cạnh đó, việc tiếp cận thông tin về các di tích cho du khách cũng gặp khó khăn do sự thiếu hụt thông tin đầy đủ và chính xác trên các phương tiện truyền thông.

Chính vì những thách thức này, việc xây dựng một website giới thiệu các di tích lịch sử ở Hà Nội bằng công nghệ hiện đại như React, ExpressJS, MongoDB, và NodeJS trở nên hết sức cần thiết. Website này không chỉ giúp cung cấp thông tin một cách đầy đủ và dễ dàng tiếp cận hơn cho du khách, mà còn góp phần quảng bá hình ảnh và giá trị văn hóa của Việt Nam ra thế giới. Đây sẽ là cầu nối giữa di sản văn hóa và công nghệ, giúp du khách có thể tìm hiểu sâu rộng và trải nghiệm phong phú khi khám phá Hà Nội, ngay cả trước khi họ đặt chân đến nơi đây.

Đề tài sẽ tập trung thực hiện các nội dung chính:

* Sử dụng React để phát triển giao diện người dùng (UI) cho trang web với các thành phần tái sử dụng, nhằm tạo ra trải nghiệm người dùng linh hoạt và thân thiện.
* Sử dụng ExpressJS để xây dựng một RESTful API cho việc tương tác với cơ sở dữ liệu và cung cấp dữ liệu cho phía frontend.
* Sử dụng NodeJS để xử lý yêu cầu từ phía client và tương tác với cơ sở dữ liệu nhằm truy xuất và cập nhật thông tin về các di tích lịch sử
* Sử dụng một cơ sở dữ liệu như MongoDB hoặc MySQL để lưu trữ thông tin về các di tích lịch sử, bao gồm các trường như tên, mô tả, vị trí địa lý, hình ảnh, v.v.
* Tích hợp AI Chatbot để hỗ trợ du khách.
* Đánh giá và tinh chỉnh hệ thống.
* Triển khai thử nghiệm.
* Tổng kết và báo cáo kết quả đề tài.

**CÁC MỤC TIÊU CHÍNH**

* Xây dựng trang web hoàn chỉnh: Phát triển một trang web đầy đủ chức năng và chất lượng cao để giới thiệu các di tích lịch sử ở Hà Nội.
* Sử dụng công nghệ hiện đại: Áp dụng các công nghệ mới nhất như MongoDB, React, ExpressJS và NodeJS để xây dựng ứng dụng web một cách hiệu quả và linh hoạt.
* Tích hợp các tính năng tương tác: Tạo ra các tính năng tương tác như tìm kiếm, lọc, đánh giá và bình luận để người dùng có thể tương tác với trang web một cách thuận tiện và sinh động.

**KẾT QUẢ DỰ KIẾN**

1. Lý thuyết:

* Hiểu biết sâu sắc về quy trình phát triển phần mềm và kiến thức vững về các công nghệ được sử dụng trong dự án, bao gồm MongoDB, React, ExpressJS và NodeJS.
* Thành thạo trong việc triển khai các nguyên lý thiết kế web và các kỹ thuật lập trình để tạo ra các trang web đáng tin cậy, hiệu quả và dễ bảo trì.
* Hoàn thiện bản đặc tả, phân tích yêu cầu thực tiễn
* Hoàn thiện báo cáo đồ án tốt nghiệp

1. Thực hiện:

* Xây dựng và triển khai trang web hoàn thiện và hoạt động một cách mượt mà và hiệu quả trên mọi thiết bị và mọi môi trường truy cập và tạo ra một trải nghiệm tương tác và thú vị cho người dùng.
* Tích hợp với AI Chatbot để để hỗ trợ du khách trong việc tìm hiểu về địa điểm du lịch của thủ đô Hà Nội.

**TÀI LIỆU THAM KHẢO**

| [1] | *React*. (n.d.).<https://react.dev/> |
| --- | --- |
| [2] | *Index | Node.js v21.7.3 Documentation*. (n.d.). https://nodejs.org/docs/latest/api/ |
| [3] | *Frameworks built on Express*. (n.d.). https://expressjs.com/en/resources/frameworks.html |
| [4] | *MongoDB documentation*. (n.d.). MongoDB Documentation.https://www.mongodb.com/docs/ |
| [5] | *Cultural heritage: 7 successes of UNESCO’s preservation work*. (2023, July 18). UNESCO. https://www.unesco.org/en/cultural-heritage-7-successes-unescos-preservation-work |
| [6] | *Tourism at World Heritage Sites – Challenges and Opportunities: International tourism seminar, Çeşme (Izmir), Turkey, 26 March 2013*. (2015).<https://doi.org/10.18111/9789284416608> |
| [7] | *Cultural heritage: 7 successes of UNESCO’s preservation work*. (2023b, July 18). UNESCO. https://www.unesco.org/en/cultural-heritage-7-successes-unescos-preservation-work |
| [8] | Phan, D. (2022, December 30). *The best historical sites to visit in Vietnam*. Culture Trip.<https://theculturetrip.com/asia/vietnam/articles/the-10-best-historical-sites-to-visit-in-vietnam> |
| [9] | *OpenAI Platform*. (n.d.). https://platform.openai.com/docs/introduction |

**TIẾN ĐỘ THỰC HIỆN**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Thời gian** | **Nội dung công việc** | **Kết quả dự đoán đạt được** |
| 1 | 03/05/2024-05/05/2024 | Tìm hiểu đề tài: “Xây dựng website và tích hợp với Chatbot phục vụ truyền thông và giới thiệu địa điểm di tích lịch sử của thủ đô Hà Nội”.  Lên kế hoạch làm việc. | - Phân tích các vấn đề chính của đề tài và hướng giải quyết.  - Lên kế hoạch cho từng công việc phải làm để đảm bảo đồ án thực hiện đúng tiến độ. |
| 2 | 06/05/2024-12/05/2024 | - Setting up routes  - Create user auth form  - Google authentication | - Thiết lập đường dẫn cho biểu mẫu xác thực người dùng  - Tạo biểu mẫu xác thực người dùng  - Cho phép người dùng sử dụng tài khoản Google để đăng nhập vào ứng dụng  - Viết API cho chức năng Sign-in, Sign-up |
| 3 | 13/05/2024-19/05/2024 | - Create editor Page  - Create content editor, navbar, banner  - Create content card, user card  - Create search Input  - Create pagination | - Tạo trang cho người dùng đăng tải nội dung về di tích lịch sử ở Hà Nội  - Xây dựng các thành phần giao diện như trình soạn thảo content, thanh điều hướng, phân trang và banner đáp ứng yêu cầu thiết kế  - Tạo ra các thành phần giao diện để hiển thị thông tin về bài đăng content và người dùng một cách rõ ràng và đẹp mắt  - Phát triển một trường nhập liệu cho phép người dùng nhập thông tin tìm kiếm và hiển thị kết quả tương ứng  - Viết API cho chức năng soạn thảo content, tìm kiếm content và user |
| 4 | 20/05/2024-26/05/2024 | - Create user profile page  - Create content detail for content page | - Tạo ra một trang hồ sơ cá nhân cho người dùng hiển thị thông tin cá nhân và hoạt động liên quan  - Tạo ra nội dung chi tiết và đầy đủ để hiển thị trên trang content  - Viết API cho chức năng hồ sơ cá nhân |
| 5 | 27/05/2024-02/06/2024 | - Create edit content  - Create comments cards | - Phát triển chức năng cho phép người dùng dễ dàng và hiệu quả chỉnh sửa các bài đăng trên trang nội dung.  - Tạo ra các thẻ hiển thị bình luận cho các bài đăng  - Viết API cho chức năng chỉnh sửa content |
| 6 | 03/06/2024-09/06/2024 | - Create like, comment post  - Create reply comment  - Hiding & Loading replies | - Xây dựng chức năng cho phép người dùng thích và bình luận các bài đăng  - Cho phép người dùng trả lời vào các bình luận trên bài đăng  - Chức năng cho phép người dùng ẩn và tải các câu trả lời của bình luận  - Viết API cho các chức năng thích và bình luận |
| 7 | 10/06/2024-16/06/2024 | - Delete comments  - Create edit profile page  - Fix: User Profile Page  - Change password form | - Xây dựng chức năng cho phép người dùng xóa các bình luận mà họ đã đăng  - Xây dựng một giao diện cho phép người dùng chỉnh sửa thông tin cá nhân của họ  - Sửa các lỗi và cải thiện trang hồ sơ người dùng để đảm bảo hoạt động mượt mà  - Xây dựng một biểu mẫu cho phép người dùng thay đổi mật khẩu đăng nhập của họ  - Viết API cho chức năng xóa comments, edit profile và đổi mật khẩu - Sửa API cho hồ sơ cá nhân |
| 8 | 17/06/2024-23/06/2024 | - Create notification alert  - Create notification page | - Xây dựng chức năng cho phép người dùng nhận thông báo khi có người phản hồi lại bình luận  - Xây dựng giao diện bình luận cho phép người dùng bình luận vào bài đăng |
| 9 | 24/06/2024-30/06/2024 | - Update api add-comment  - Update user access  - Fix InPageNavigation UI | - Chỉnh sửa API thêm bình luận mới  - Cập nhật quyền truy cập của người dùng  - Chỉnh sửa chức năng chuyển tab trong trang |
| 10 | 01/07/2024-07/07/2024 | - Integrate chatbot into website  - Fix Update UI for HomePage  - Update thesis | - Tích hợp AI Chatbot vào trang Web  - Chỉnh sửa giao diện cho trang chủ  - Cập nhật nội dung báo cáo |
| 11 | 08/07/2024-14/07/2024 | - Fix Comment update UI  - Fix get isAdmin from session  - Update thesis | - Sửa giao diện cho phần bình luận  - Sửa lỗi không lấy được isAdmin từ session  - Cập nhật nội dung báo cáo |

**LỜI CAM ĐOAN**

Tác giả khẳng định rằng đây là bản Đồ án tốt nghiệp của chính mình. Các kết quả trong Đồ án này được thực hiện một cách trung thực, không sao chép từ bất kỳ nguồn nào và dưới bất kỳ hình thức nào. Việc tham khảo các nguồn tài liệu đã được thực hiện bằng cách trích dẫn và ghi nguồn đúng theo quy trình.

|  |  |
| --- | --- |
|  | **Tác giả ĐATN**  ***Chữ ký*** |

**LỜI CẢM ƠN**

Sau bốn năm học tập và rèn luyện tại Khoa Công nghệ thông tin của Trường Đại học Thuỷ Lợi, em đã được rèn luyện trong môi trường học tập chất lượng và nghiêm túc, em đã được sự chỉ dạy nhiệt tình và hướng dẫn chu đáo của các thầy cô trong Khoa cũng như các thầy cô trong trường.

Đầu tiên, em muốn bày tỏ lòng biết ơn sâu sắc **đến Ths. Vũ Thành Vinh** vì đã quan tâm và hướng dẫn em nhiệt tình trong quá trình thực hiện đồ án tốt nghiệp.

Em cũng muốn chân thành gửi lời cảm ơn tới các giáo viên tại Khoa Công nghệ thông tin và Trường Đại học Thuỷ Lợi vì đã trang bị cho em kiến thức quý báu để chuẩn bị cho thời gian làm việc sắp tới.

Trong quá trình làm đồ án tốt nghiệp, mặc dù có sự hướng dẫn tận tình từ **Ths. Vũ Thành Vinh** và kinh nghiệm cá nhân, nhưng đồ án không thể tránh khỏi những sai sót. Em hy vọng nhận được sự thông cảm và đóng góp từ Thầy Cô cũng như các bạn để nâng cao chất lượng và hoàn thiện đề tài hơn.

**Em xin chân thành cảm ơn!**

**MỤC LỤC**

[DANH MỤC HÌNH ẢNH v](#_Toc171787324)

[DANH MỤC BẢNG BIỂU vi](#_Toc171787325)

[DANH MỤC CÁC TỪ VIẾT TẮT VÀ GIẢI THÍCH CÁC THUẬT NGỮ vii](#_Toc171787326)

[LỜI MỞ ĐẦU 1](#_Toc171787327)

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 5](#_Toc171787328)

[1.1. Giới thiệu về bài toán xây dựng website tích hợp Chatbot 5](#_Toc171787329)

[1.2. Ngôn ngữ sử dụng 5](#_Toc171787330)

[1.2.1. Tổng quan về Javascript 5](#_Toc171787331)

[1.2.2. Tổng quan về React 7](#_Toc171787332)

[1.2.4. Tổng quan về ExpressJS 14](#_Toc171787333)

[1.2.5. Tổng quan về MongoDB 16](#_Toc171787334)

[1.3. Coze AI Bot 21](#_Toc171787335)

[CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 23](#_Toc171787336)

[2.1. Thu thập và phân tích yêu cầu 23](#_Toc171787337)

[2.2. Các thành phần hệ thống 24](#_Toc171787338)

[2.3. Chức năng hệ thống 25](#_Toc171787339)

[2.3.1 Sơ đồ use-case 26](#_Toc171787340)

[2.3.2. Đặc tả Use Case 28](#_Toc171787341)

[2.3.3. Biểu đồ tuần tự 36](#_Toc171787342)

[2.3.4. Biểu đồ hoạt động 41](#_Toc171787343)

[2.4. Cài đặt mô hình Website tích hợp Chatbot AI Coze 46](#_Toc171787344)

[2.4.1. Công nghệ sử dụng và môi trường phát triển 46](#_Toc171787345)

[2.4.2. Chuẩn bị dữ liệu 47](#_Toc171787346)

[2.4.3. Chuyển đổi dữ liệu sang Excel 47](#_Toc171787347)

[2.4.4. Huấn luyện chatbot 48](#_Toc171787348)

[CHƯƠNG 3: KẾT QUẢ VÀ ĐÁNH GIÁ KẾT QUẢ 50](#_Toc171787349)

[3.1. Cài đặt chương trình 50](#_Toc171787350)

[3.1.1 Cài đặt chương trình hệ thống (Ngôn ngữ Javascript - NodeJS) 50](#_Toc171787351)

[3.1.2. Cài đặt giao diện chương trình (Ngôn ngữ Javascript - ReactJS) 50](#_Toc171787352)

[3.1.3. Cài đặt cơ sở dữ liệu (Cơ sở dữ liệu MongoDB) 50](#_Toc171787353)

[3.2. Kết quả chương trình 50](#_Toc171787354)

[3.3. Kiểm thử 56](#_Toc171787355)

[KẾT LUẬN 58](#_Toc171787356)

[TÀI LIỆU THAM KHẢO 60](#_Toc171787357)

**DANH MỤC HÌNH ẢNH**

[Hình 1.1. Javascript có thể tạo ra những chuyển động đẹp mắt cho website 7](#_Toc171787376)

[Hình 1.2. ReactJS có thể tạo ra bất kỳ trang web nào mượt mà, nhanh chóng, khả năng mở rộng cao 8](#_Toc171787377)

[Hình 1.3. Ưu điểm của ReactJS 10](#_Toc171787378)

[Hình 1.4. Mô hình hoạt động của NodeJS 12](#_Toc171787380)

[Hình 1.5. Tính năng của ExpressJS là gì? 15](#_Toc171787381)

[Hình 1.6. MongoDB là một phần mềm cơ sở dữ liệu mã nguồn mở NoSQL 18](#_Toc171787382)

[Hình 1.7. Truy vấn ad hoc được xem là tính năng tốt nhất 19](#_Toc171787383)

[Hình 1.8. Coze AI tích hợp được nhiều công việc khác nhau 21](#_Toc171787384)

[Hình 2.1. Biểu đồ use-case của người đọc 26](#_Toc171787385)

[Hình 2.2. Biểu đồ use-case của người đọc - thành viên 27](#_Toc171787386)

[Hình 2.3. Biểu đồ use-case của người quản trị 27](#_Toc171787387)

[Hình 2.4. Biểu đồ phân rã chức năng 36](#_Toc171787388)

[Hình 2.5. Biểu đồ tuần tự đăng nhập cho người dùng hoặc quản trị viên 36](#_Toc171787389)

[Hình 2.6. Biểu đồ tuần tự đăng ký thành viên cho người dùng 37](#_Toc171787390)

[Hình 2.7. Biểu đồ tuần tự tìm kiếm địa danh di tích lịch sử 37](#_Toc171787391)

[Hình 2.8. Biểu đồ tuần tự xem thông tin di tích lịch sử 38](#_Toc171787392)

[Hình 2.9. Biểu đồ tuần tự bình luận bài viết về di tích lịch sử 38](#_Toc171787393)

[Hình 2.10. Biểu đồ tuần tự thêm mới nội dung về di tích lịch sử 39](#_Toc171787394)

[Hình 2.11. Biểu đồ tuần tự chỉnh sửa nội dung về di tích lịch sử 39](#_Toc171787395)

[Hình 2.12. Biểu đồ tuần tự xóa nội dung về di tích lịch sử 40](#_Toc171787396)

[Hình 2.13. Biểu đồ tuần tự xóa tài khoản người dùng 40](#_Toc171787397)

[Hình 2.14. Biểu đồ tuần tự xóa thông báo về cảm xúc và bình luận 41](#_Toc171787398)

[Hình 2.15. Biểu đồ hoạt động đăng nhập cho người dùng hoặc quản trị viên 41](#_Toc171787399)

[Hình 2.16. Biểu đồ hoạt động đăng ký cho người dùng 42](#_Toc171787400)

[Hình 2.17. Biểu đồ hoạt động tìm kiếm địa danh di tích lịch sử 42](#_Toc171787401)

[Hình 2.18. Biểu đồ hoạt động xem thông tin di tích lịch sử 43](#_Toc171787402)

[Hình 2.19. Biểu đồ hoạt động bình luận bài viết về di tích lịch sử 43](#_Toc171787403)

[Hình 2.20. Biểu đồ hoạt động thêm mới nội dung về di tích lịch sử 44](#_Toc171787404)

[Hình 2.21. Biểu đồ hoạt động chỉnh sửa nội dung về di tích lịch sử 44](#_Toc171787405)

[Hình 2.22. Biểu đồ hoạt động xóa nội dung về di tích lịch sử 45](#_Toc171787406)

[Hình 2.23. Biểu đồ hoạt động xóa tài khoản người dùng 45](#_Toc171787407)

[Hình 2.25. Mẫu dữ liệu về tên di tích lịch sử và mô tả sau khi xuất ra từ database 47](#_Toc171787409)

[Hình 2.26. Dữ liệu sau khi chuyển đổi sang excel 47](#_Toc171787410)

[Hình 2.27. Dữ liệu sau khi tích hợp vào Chatbot 48](#_Toc171787411)

[Hình 2.28. Logic phản hồi của chatbot 48](#_Toc171787412)

[Hình 3.1. Giao diện trang chủ 51](#_Toc171787413)

[Hình 3.2. Giao diện trang chi tiết về địa điểm di tích lịch sử 51](#_Toc171787414)

[Hình 3.3. Giao diện trang đăng nhập 52](#_Toc171787415)

[Hình 3.4. Giao diện trang đăng ký 52](#_Toc171787416)

[Hình 3.5. Giao diện trang soạn thảo 52](#_Toc171787417)

[Hình 3.6. Giao diện tìm kiếm địa điểm di tích lịch sử 53](#_Toc171787418)

[Hình 3.7. Giao diện trang hồ sơ cá nhân 53](#_Toc171787419)

[Hình 3.8. Giao diện trang chỉnh sửa hồ sơ 53](#_Toc171787420)

[Hình 3.9. Giao diện trang quản lý nội dung 54](#_Toc171787421)

[Hình 3.10. Giao diện trang quản lý người dùng 54](#_Toc171787422)

[Hình 3.11. Giao diện trang quản lý thông báo 54](#_Toc171787423)

[Hình 3.12. Giao diện trang đổi mật khẩu 55](#_Toc171787424)

[Hình 3.13. Giao diện hộp thoại với Chatbot 55](#_Toc171787425)

**DANH MỤC BẢNG BIỂU**

[Bảng 2.1. Bảng đặc tả hệ thống 25](#_Toc171787426)

[Bảng 2.2. Bảng đặc tả đăng nhập 28](#_Toc171787427)

[Bảng 2.3. Bảng đặc tả đăng ký 29](#_Toc171787428)

[Bảng 2.4. Bảng đặc tả đổi mật khẩu, cập nhật thông tin 30](#_Toc171787429)

[Bảng 2.5. Bảng đặc tả thả cảm xúc bài đăng 31](#_Toc171787430)

[Bảng 2.6. Bảng đặc tả bình luận bài đăng 32](#_Toc171787431)

[Bảng 2.7. Bảng đặc tả thêm mới bài đăng 33](#_Toc171787432)

[Bảng 2.8. Bảng đặc tả chỉnh sửa bài đăng 34](#_Toc171787433)

[Bảng 2.9. Bảng đặc tả xóa bài đăng 35](#_Toc171787434)

[Bảng 3.1. Kiểm thử đăng nhập & đăng ký 56](#_Toc171787435)

[Bảng 3.2. Bảng kiểm thử quản lý nội dung, người dùng, thông báo 57](#_Toc171787436)

**DANH MỤC CÁC TỪ VIẾT TẮT VÀ GIẢI THÍCH CÁC THUẬT NGỮ**

|  |  |
| --- | --- |
| **Từ viết tắt** | **Viết đầy đủ** |
| **API** | **A**pplication **P**rogramming **I**nterface |
| **BSON** | Binary JSON |
| **CSDL** | Cơ sở dữ liệu |
| **CPU** | Central Processing Unit |
| **DB** | Database |
| **I/O** | Input/Output |
| **JSON** | **J**ava**S**cript **O**bject **N**otation |
| **MERN** | MongoDB, Express.js, React.js, Node.js |
| **NoSQL** | non-SQL |
| **REST** | **RE**presentational **S**tate **T**ransfer |
| **RDBMS** | Relational Database Management System |
| **SQL** | Structured Query Language |
| **SEO** | **Search Engine Optimization** |
| **XSS** | Cross-Site Scripting |

**LỜI MỞ ĐẦU**

1. **Giới thiệu đề tài**

Trên thị trường hiện nay, với sự phát triển nhanh chóng của công nghệ thông tin, việc sử dụng trang web và chatbot đang trở thành một xu hướng quan trọng trong lĩnh vực truyền thông và quảng cáo du lịch. Đặc biệt, khi khám phá các di tích lịch sử, việc kết hợp cả hai yếu tố này không chỉ giúp tăng cường hiệu quả quảng cáo mà còn mang lại trải nghiệm tương tác tốt hơn cho khách hàng.

Việc xây dựng một nền tảng kết hợp trang web và chatbot cho du lịch không chỉ giúp các doanh nghiệp thu hút khách du lịch mà còn tạo ra một cách tiếp cận hiện đại và thuận tiện cho người dùng. Nhờ vào tính linh hoạt của chatbot, khách hàng có thể nhận thông tin ngay lập tức và hỗ trợ khi cần mà không cần phải chờ đợi.

Với sự phổ biến của điện thoại thông minh và internet, việc sử dụng trang web và chatbot trong lĩnh vực du lịch giúp tạo ra một kênh giao tiếp trực tuyến hiệu quả, giúp doanh nghiệp tiếp cận đến đông đảo khách hàng tiềm năng một cách nhanh chóng và tiện lợi. Đồng thời, việc áp dụng công nghệ này cũng giúp nâng cao trải nghiệm của khách hàng khi tìm hiểu thông tin về điểm đến du lịch và dễ dàng tương tác để có được những thông tin cần thiết.

1. **Lý do chọn đề tài**

Hiện nay, em nhận thấy các trang web thường không đáp ứng đủ nhu cầu của người dùng như việc chia sẻ cảm xúc và bình luận để trao đổi với những người khác về di sản lịch sử Hà Nội mà họ quan tâm. Điều này gây khó khăn khi họ cần thông tin chi tiết từ các bài viết mà không tìm thấy câu trả lời chính xác.

Vấn đề này đã truyền cảm hứng cho em từ sự yêu thích và quan tâm đặc biệt đến việc bảo tồn và phát huy giá trị của di sản văn hóa lịch sử của Hà Nội.

Việc phát triển một trang web kết hợp với chatbot sẽ không chỉ giúp chia sẻ thông tin một cách chính xác và dễ dàng về các địa điểm di tích, mà còn thúc đẩy sự tương tác giữa người dùng và văn hóa lịch sử sâu sắc của dân tộc. Điều này mở ra cơ hội để khám phá và truyền đạt những giá trị văn hóa lịch sử một cách sinh động và hiệu quả nhất.

Sử dụng công nghệ trong việc bảo tồn di sản văn hóa không chỉ giúp duy trì mà còn phát triển các giá trị truyền thống theo cách hiện đại và hấp dẫn hơn đối với giới trẻ.

Kết hợp giữa truyền thống và công nghệ không chỉ là một cách tiếp cận mới mà còn là yếu tố quan trọng trong việc thúc đẩy sự phát triển toàn diện cho cộng đồng, từ việc bảo tồn di sản đến việc tăng cường tinh thần quốc gia và nhận thức văn hóa trong cả nước. Đây là một cách tiếp cận sáng tạo và mang lại nhiều lợi ích vượt ra ngoài phạm vi dự án đơn lẻ, hướng tới một cộng đồng phát triển và sáng tạo hơn.

1. **Mục tiêu của đề tài**

* Xây dựng trang web hoàn chỉnh: Phát triển một trang web đầy đủ chức năng và chất lượng cao để giới thiệu các di tích lịch sử ở Hà Nội.
* Sử dụng công nghệ hiện đại: Áp dụng kiến trúc hệ thống MERN (MongoDB, ExpressJS, ReactJS và NodeJS) để xây dựng ứng dụng web một cách hiệu quả và linh hoạt.
* Tích hợp các tính năng tương tác: Tạo ra các tính năng tương tác như tìm kiếm, lọc, đánh giá và bình luận để người dùng có thể tương tác với trang web một cách thuận tiện và sinh động.
* Mục đích cuối cùng là tăng cường hiểu biết về văn hóa lịch sử, quảng bá vẻ đẹp và giá trị du lịch của Hà Nội cũng như thúc đẩy sự phát triển của công nghệ và du lịch.

1. **Phương pháp nghiên cứu**

**4.1. Phân tích yêu cầu và thiết kế hệ thống:**

Xác định và phân tích yêu cầu cụ thể cho website và chatbot nhằm mục đích truyền thông và giới thiệu các địa điểm di tích lịch sử của Hà Nội.

Thiết kế kiến trúc hệ thống MERN (MongoDB, ExpressJS, ReactJS và NodeJS) để đáp ứng các yêu cầu và tích hợp chatbot một cách hiệu quả.

**4.2. Phát triển website:**

Sử dụng ReactJS để thiết kế một giao diện người dùng trực quan và dễ dàng sử dụng.

Xây dựng backend bằng các công nghệ NodeJS và ExpressJS để kiểm soát các luồng nghiệp vụ và tương tác với cơ sở dữ liệu MongoDB.

**4.3. Phát triển chatbot:**

Tích hợp chatbot vào website bằng cách sử dụng Chat Coze Web SDK

Huấn luận chatbot dựa vào data từ database

Lập trình chatbot để giới thiệu và hỗ trợ người dùng tra cứu thông tin về các địa điểm di tích lịch sử của Hà Nội một cách nhanh chóng và hiệu quả.

**4.4. Kiểm thử và đánh giá:**

Thực hiện việc kiểm tra, sự đánh giá nhằm đảm bảo rằng độ ổn định, an toàn và hiệu suất của website và chatbot luôn được duy trì.

Thu thập phản hồi từ người dùng để cải thiện và điều chỉnh sản phẩm.

1. **Đối tượng phạm vi nghiên cứu**

**Người dùng cuối:** Các độc giả, khách tham quan và du khách quan tâm đến các di tích lịch sử của Hà Nội.

**Người quản trị hệ thống:** Quản trị viên chịu trách nhiệm duy trì và quản lý website và chatbot.

**Người quản lý dự án:** Những người có trách nhiệm điều hành và giám sát tiến độ của dự án.

Phạm vi của đề tài giới hạn trong khuân khổ nội dung phục vụ truyền thông và giới thiệu địa điểm di tích lịch sử của thủ đô Hà Nội sử dụng mô hình công nghệ MERN

1. **Nhiệm vụ và kết quả đạt được**

Nghiên cứu và tìm hiểu một số trang web, phần mềm quảng bá di tích, địa điểm tham quan

Phân tích yêu cầu và đề xuất của phần mềm

Thiết kế đặc tả hệ thống

Xây dựng được cơ sở dữ liệu

Lập trình các module của hệ thống

1. **Bố cục của đồ án**

Dựa trên các nội dung đã khảo sát, để thực hiện được mục tiêu đề ra và đảm bảo tính logic, ngoài phần mở đầu và phần kết luận, đồ án sẽ được tổ chức theo các chương sau đây:

*Chương 1: Cơ sở lý thuyết*. Chương này trình bày các cơ sở lý thuyết về internet và website; trình bày kết quả nghiên cứu, tìm hiểu về các ngôn ngữ lập trình và kiến trúc hệ thống MERN (MongoDB, ExpressJS, ReactJS và NodeJS) được sử dụng trong việc xây dựng đề tài.

*Chương 2: Phân tích và thiết kế hệ thống*. Chương này giới thiệu bài toán và đưa ra các mô hình hệ thống; đề xuất các ý tưởng, chức năng cho phần mềm; trình bày các mô hình thực thể quan hệ, sơ đồ phân cấp chức năng. Phân tích chi tiết các nhiệm vụ của người sử dụng và người quản lý; thiết kế giao diện cho người dùng và thiết kế cấu trúc dữ liệu, các chức năng cho hệ thống.

*Chương 3: Kết quả và đánh giá kết quả.* Trong chương này, nêu kết quả thực hiện được của đề tài và trình bày cách cài đặt chương trình.

1. **Đóng góp của đồ án**

Trong quá trình thực hiện đồ án tốt nghiệp, em đã sử dụng những kiến thức đã học được từ trường để tiếp cận đề tài một cách linh động. Không chỉ vậy, em còn tự tìm hiểu, nghiên cứu thêm về các kiến thức mới và ngôn ngữ lập trình mới để phát triển dự án một cách nhanh chóng và hoàn thiện hơn:

*Phân tích rõ ràng vai trò của người dùng trong hệ thống*. Đối với người quản *trị*, khách hàng đã đăng nhập hay chưa đăng nhập đều được giới hạn một số chức năng nhất định.

*Đưa ra các chức năng điều khiển dành cho người quản trị*. Nghiên cứu và triển khai thực nghiệm các chức năng nhằm đáp ứng nhu cầu cho người quản lý như quản lý các thông tin về bài viết, thông tin các người dùng.

*Đưa ra chức năng thả cảm xúc và bình luận cho người dùng*. Nghiên cứu và triển khai thực nghiệm chức năng thả cảm xúc và bình luận cho người dùng. Chức năng này cho phép người dùng thể hiện cảm xúc và ý kiến cá nhân về các địa danh, cũng như trao đổi với người dùng khác về những địa điểm họ quan tâm.

**CHƯƠNG 1: CƠ SỞ LÝ THUYẾT**

**1.1. Giới thiệu về bài toán xây dựng website tích hợp Chatbot**

Xây dựng một website tích hợp Chatbot bằng MERN (MongoDB, Express.js, React.js, Node.js) là một bài toán hấp dẫn và phức tạp trong lĩnh vực phát triển web và trí tuệ nhân tạo. Mục tiêu chính của bài toán này là tạo ra một ứng dụng web hiện đại với khả năng tương tác tự động thông qua chatbot, giúp cải thiện trải nghiệm người dùng và cung cấp thông tin hoặc dịch vụ một cách hiệu quả.

MERN là một thuật ngữ rút gọn của MongoDB, Express, React và Node; Stack MERN là một stack Javascript được thiết kế để giúp phát triển ứng dụng web toàn ngăn xếp dễ dàng hơn và nhanh hơn.

Tất cả bốn công nghệ này cung cấp một khuôn khổ hoàn chỉnh cho các nhà phát triển để tạo ra bất kỳ ứng dụng web nào. MERN đang tuân theo kiến ​​trúc 3 tầng truyền thống, bao gồm tầng hiển thị front-end (ReactJS), tầng ứng dụng (ExpressJS và NodeJS) và tầng cơ sở dữ liệu (MongoDB)

**1.2. Ngôn ngữ sử dụng**

***1.2.1. Tổng quan về Javascript***

JavaScript là một ngôn ngữ lập trình phổ biến và quan trọng trong lĩnh vực phát triển web. JavaScript (JS) ban đầu được phát triển để thêm tính năng tương tác vào các trang web. Ngày nay, nó đã trở thành một ngôn ngữ lập trình phổ biến không chỉ cho front-end mà còn cho back-end development (với Node.js), mobile app development (Với React Native), và nhiều ứng dụng khác.

So sánh với các ngôn ngữ khác

**Với Python:**

* **Điểm mạnh:** Python dễ học, cú pháp rõ ràng, phù hợp cho AI/ML, và có thư viện phong phú.
* **Điểm yếu:** Không phù hợp cho front-end web development mà cần phải kết hợp với JavaScript hoặc các framework khác.

**Với Java:**

* **Điểm mạnh:** Được sử dụng rộng rãi trong enterprise applications, hệ thống lớn, hỗ trợ đa luồng.
* **Điểm yếu:** Cú pháp phức tạp hơn JavaScript, khởi động chậm hơn cho các ứng dụng nhỏ.

**Với Ruby:**

* **Điểm mạnh:** Đơn giản, dễ đọc, phù hợp cho web development.
* **Điểm yếu:** Hiệu suất không cao bằng JavaScript, hạn chế trong các ứng dụng cần xử lý nhanh.

**Điểm mạnh của JavaScript**

1. **Thiết kế cho web:** JavaScript được thiết kế đặc biệt cho việc tương tác trực tiếp trên trình duyệt.
2. **Không đồng bộ:** Xử lý không đồng bộ (asynchronous processing) làm cho việc làm web trở nên mượt mà hơn.
3. **Community lớn:** Có một cộng đồng lớn, hỗ trợ mạnh mẽ và nhiều thư viện/framework phát triển.

Điểm yếu của JavaScript

1. **Bảo mật:** JavaScript dễ bị tấn công XSS (Cross-Site Scripting) nếu không được bảo vệ cẩn thận.
2. **Không đồng nhất:** Sự khác biệt giữa các trình duyệt có thể gây ra các vấn đề tương thích.
3. **Performance:** Hiệu suất không thể so sánh với các ngôn ngữ như C++ hoặc Rust cho các ứng dụng yêu cầu tính toán nặng.

|  |
| --- |
|  |
| *Hình 1.1. Javascript có thể tạo ra những chuyển động đẹp mắt cho website* |

***1.2.2. Tổng quan về React***

**Giới thiệu React**

ReactJS được phát triển bởi Facebook và được giới thiệu lần đầu tiên vào năm 2011. Ban đầu, ReactJS được phát triển để xây dựng giao diện người dùng trên trang web Facebook, nhằm cải thiện tốc độ và hiệu suất của ứng dụng web.

Tuy nhiên, ReactJS không được công bố cho cộng đồng phát triển cho đến năm 2013, khi Facebook công bố mã nguồn mở của nó và giới thiệu cho cộng đồng lập trình viên. Từ đó, ReactJS nhanh chóng trở thành một trong những thư viện phổ biến nhất để phát triển các ứng dụng web động.

Sau đó, vào năm 2015, Facebook giới thiệu phiên bản React Native, một framework phát triển ứng dụng di động sử dụng ReactJS. React Native cho phép các nhà phát triển xây dựng các ứng dụng di động cho cả iOS và Android sử dụng cùng một mã nguồn, tương tự như ReactJS trên web.

Hiện nay, ReactJS đã trở thành một trong những thư viện phát triển web phổ biến nhất, được sử dụng rộng rãi bởi các công ty lớn và nhỏ trên toàn thế giới. Facebook cũng tiếp tục đầu tư phát triển và nâng cấp ReactJS để đáp ứng nhu cầu của cộng đồng phát triển.

|  |
| --- |
| React use case |
| *Hình 1.2. ReactJS có thể tạo ra bất kỳ trang web nào mượt mà, nhanh chóng, khả năng mở rộng cao* |

**Lợi ích của ReactJS**

ReactJS mang đến nhiều lợi ích cho việc phát triển ứng dụng web, bao gồm:

**Hiệu suất cao:** ReactJS sử dụng Virtual DOM để tối ưu hóa hiệu suất của ứng dụng. Virtual DOM cho phép ReactJS cập nhật các thay đổi trên trang web một cách nhanh chóng và hiệu quả hơn so với cách truyền thống, giúp tăng tốc độ và hiệu suất của ứng dụng.

**Tái sử dụng:** ReactJS cho phép tái sử dụng các thành phần UI, giúp giảm thiểu thời gian và chi phí phát triển. Các thành phần UI có thể được sử dụng lại trong nhiều phần khác nhau của ứng dụng, giúp tăng tính linh hoạt và khả năng mở rộng của ứng dụng.

**Dễ dàng quản lý trạng thái:** ReactJS giúp quản lý trạng thái của ứng dụng một cách dễ dàng. Sử dụng State và Props, ReactJS cho phép các nhà phát triển quản lý trạng thái của các thành phần UI một cách chính xác và dễ dàng.

**Hỗ trợ tốt cho SEO:** ReactJS cho phép các nhà phát triển xây dựng ứng dụng web với khả năng tương thích tốt với SEO. Với sự hỗ trợ của các thư viện như React Helmet, ReactJS cho phép các nhà phát triển tùy chỉnh và quản lý các phần tử meta và title cho từng trang web.

**Hỗ trợ đa nền tảng:** ReactJS không chỉ được sử dụng để phát triển các ứng dụng web, mà còn được sử dụng để phát triển các ứng dụng di động với React Native. Sử dụng React Native, các nhà phát triển có thể xây dựng ứng dụng di động cho cả iOS và Android sử dụng cùng một mã nguồn.

Tính năng nổi bật của ReactJS

ReactJS có nhiều tính năng hữu ích cho việc phát triển ứng dụng web, bao gồm:

**Components**: ReactJS cho phép phát triển ứng dụng web theo mô hình component. Các component là các phần tử UI độc lập có thể được tái sử dụng trong nhiều phần khác nhau của ứng dụng.

**Virtual DOM**: ReactJS sử dụng Virtual DOM để tối ưu hóa hiệu suất của ứng dụng. Virtual DOM là một bản sao của DOM được lưu trữ trong bộ nhớ và được cập nhật một cách nhanh chóng khi có thay đổi, giúp tăng tốc độ và hiệu suất của ứng dụng.

**JSX**: JSX là một ngôn ngữ lập trình phân biệt được sử dụng trong ReactJS để mô tả các thành phần UI. JSX kết hợp HTML và JavaScript, giúp cho việc viết mã dễ hiểu và dễ bảo trì hơn.

**State và Props**: ReactJS cho phép quản lý trạng thái của các thành phần UI thông qua State và Props. State là trạng thái của một thành phần được quản lý bởi nó chính, trong khi Props là các giá trị được truyền vào từ bên ngoài để tùy chỉnh hoặc điều khiển hành vi của một thành phần.

**Hỗ trợ tốt cho SEO**: ReactJS hỗ trợ tốt cho việc tối ưu hóa SEO. Với các thư viện như React Helmet, các nhà phát triển có thể quản lý các phần tử meta và title cho từng trang web, giúp tăng khả năng tìm kiếm và tăng cường trải nghiệm người dùng.

**Hỗ trợ đa nền tảng**: ReactJS không chỉ được sử dụng để phát triển ứng dụng web, mà còn được sử dụng để phát triển ứng dụng di động với React Native. Sử dụng React Native, các nhà phát triển có thể xây dựng ứng dụng di động cho cả iOS và Android sử dụng cùng một mã nguồn.

**Redux**: Redux là một thư viện quản lý trạng thái cho các ứng dụng ReactJS. Nó giúp quản lý trạng thái của ứng dụng một cách chính xác và dễ dàng, đồng thời giúp tăng tính linh hoạt và khả năng mở rộng của ứng dụng.

|  |
| --- |
| Ưu điểm của ReactJS |
| *Hình 1.3. Ưu điểm của ReactJS* |

Các dự án sử dụng ReactJS thành công

ReactJS là một trong những công nghệ phát triển web phổ biến nhất hiện nay, được sử dụng trong nhiều dự án thành công của các công ty lớn trên thế giới. Dưới đây là một số dự án thành công sử dụng ReactJS:

* Facebook: ReactJS là một trong những công nghệ chính của Facebook, được sử dụng để xây dựng nhiều sản phẩm của họ như Facebook, Instagram, Messenger...
* Netflix: Netflix sử dụng ReactJS để xây dựng giao diện người dùng cho trang web của họ và cho các ứng dụng di động.
* Airbnb: Airbnb sử dụng ReactJS để xây dựng giao diện người dùng cho trang web của họ và cho các ứng dụng di động.
* Uber: Uber sử dụng ReactJS để xây dựng giao diện người dùng cho ứng dụng di động của họ.
* Dropbox: Dropbox sử dụng ReactJS để xây dựng giao diện người dùng cho trang web và cho các ứng dụng di động của họ.
* Atlassian: Atlassian sử dụng ReactJS để xây dựng giao diện người dùng cho nhiều sản phẩm của họ như Jira, Confluence...

Tất cả những dự án trên đều cho thấy rằng ReactJS là một công nghệ rất phổ biến và thành công trong việc xây dựng các ứng dụng web và di động.

***1.2.3. Tổng quan về NodeJS***

Node.js được phát triển từ Javascript vào năm 2009 bởi Ryan Dahl. NodeJS có cách thức hoạt động chủ yếu trên Server sử dụng để xây dựng cho các ứng dụng realtime. NodeJS dùng mô hình I/O lập trình dựa theo sự kiện non-blocking.

Chính vì vậy, NodeJS tương đối gọn nhẹ, hiệu quả và là một công cụ hoàn hảo dành cho mọi ứng dụng chuyên sâu về dữ liệu dựa theo khoảng thời gian thực khi chạy trên những thiết bị phân tán. NodeJs thường xuyên được dùng cho mục đích xây dựng một số ứng dụng như:  Ad Server, Websocket server, Fast File Upload Client, RESTful API, Cloud Services, Any Real-time Data Application.

Nguyên lý hoạt động của NodeJS

Node.js là một nền tảng phát triển phía server được xây dựng trên nền tảng JavaScript. Khi một yêu cầu mạng đến từ một client, Node.js sẽ xử lý yêu cầu đó bằng cách thực hiện các bước như sau:

* Node.js tạo một event loop để theo dõi các yêu cầu mạng đến và đi.
* Khi một yêu cầu mạng đến, Node.js sẽ tạo một worker thread (luồng làm việc) để xử lý yêu cầu đó.
* Trong worker thread, Node.js sẽ thực hiện các tác vụ xử lý yêu cầu, chẳng hạn như đọc và ghi vào cơ sở dữ liệu, đọc và ghi file, tương tác với API, ...
* Khi worker thread đã hoàn thành các tác vụ, Node.js sẽ trả về kết quả cho client qua mạng.
* Nếu có yêu cầu mạng mới đến, Node.js sẽ tạo một worker thread mới để xử lý yêu cầu đó.

Các yêu cầu mạng đến và đi trong Node.js được xử lý bằng cách sử dụng các hàm callback, Promise, async/await để đảm bảo tính phi đồng bộ và tăng hiệu suất của ứng dụng. Các yêu cầu mạng được xử lý một cách độc lập, giúp tránh tình trạng "blocking" (chặn) trong quá trình xử lý yêu cầu. Node.js cũng có thể hoạt động với các module và thư viện khác để hỗ trợ cho việc phát triển ứng dụng web.

|  |
| --- |
| Cơ bản về NodeJS - Thầy Long Web |
| *Hình 1.4. Mô hình hoạt động của NodeJS* |

**Ưu điểm của NodeJS**

Hiệu suất cao: Node.js được xây dựng trên nền tảng JavaScript, được thiết kế để xử lý các yêu cầu mạng phức tạp với tốc độ nhanh và hiệu quả.

Lập trình phi đồng bộ (asynchronous programming): Node.js sử dụng các callback, Promise, async/await để xử lý các yêu cầu mạng phi đồng bộ, giúp tăng hiệu suất của ứng dụng và giảm thiểu thời gian đáp ứng cho người dùng.

Có nhiều thư viện và module hỗ trợ: Node.js có rất nhiều thư viện và module hỗ trợ cho việc phát triển ứng dụng web, giúp các nhà phát triển nhanh chóng xây dựng ứng dụng với đầy đủ tính năng.

Dễ dàng mở rộng: Node.js có khả năng mở rộng tốt, cho phép các nhà phát triển tăng hiệu suất của ứng dụng bằng cách thêm các máy chủ và máy chủ công việc (worker servers).

Đơn giản và dễ học: Vì Node.js được xây dựng trên nền tảng JavaScript, nên các nhà phát triển có thể nhanh chóng học và sử dụng nó.

**Nhược điểm của NodeJS**

Không phù hợp cho các ứng dụng CPU-bound: Node.js thường được sử dụng cho các ứng dụng I/O-bound (yêu cầu đọc và ghi vào cơ sở dữ liệu, đọc và ghi file, tương tác với API, ...) nên nó không phù hợp cho các ứng dụng CPU-bound (như xử lý ảnh, phân tích dữ liệu lớn, ...).

Có thể gây ra vấn đề khi xử lý lỗi: Khi xử lý lỗi trong Node.js, các nhà phát triển cần quản lý callback và exception để tránh gây ra vấn đề trong quá trình xử lý các yêu cầu mạng.

Phiên bản thay đổi liên tục: Node.js thường có các phiên bản thay đổi liên tục, điều này đôi khi gây khó khăn cho các nhà phát triển khi cập nhật các thư viện và module để đảm bảo tính tương thích của ứng dụng.

Những ứng dụng phổ biến của Node.js hiện nay

Node.js là một nền tảng phát triển phía server, có thể được sử dụng trong nhiều ứng dụng lập trình web, bao gồm:

Phát triển các ứng dụng web động (dynamic web applications): Node.js cho phép phát triển các ứng dụng web động như các trang web tương tác cao hoặc các ứng dụng phức tạp, cung cấp tính năng xử lý yêu cầu đồng thời và xử lý dữ liệu theo thời gian thực.

Phát triển các ứng dụng web thời gian thực (real-time web applications): Node.js cung cấp tính năng xử lý sự kiện theo thời gian thực, giúp phát triển các ứng dụng web thời gian thực như các ứng dụng chat, các trò chơi trực tuyến hoặc các ứng dụng tương tác cao.

Xử lý các yêu cầu API (APIs): Node.js là một nền tảng phát triển tuyệt vời để xử lý các yêu cầu API, cho phép phát triển các ứng dụng web dựa trên API.

Xử lý các tác vụ đồng bộ và không đồng bộ (synchronous và asynchronous tasks): Node.js cho phép phát triển các ứng dụng web đồng bộ và không đồng bộ, giúp tăng tốc độ xử lý của ứng dụng.

Phát triển các ứng dụng web đơn trang (single-page applications): Node.js cung cấp các thư viện và framework như Express.js, Sails.js, Meteor.js để phát triển các ứng dụng web đơn trang.

Phát triển các ứng dụng web liên kết nhiều hệ thống (microservices): Node.js cho phép phát triển các ứng dụng web liên kết với nhiều hệ thống khác nhau như cơ sở dữ liệu, các hệ thống phân tán, các dịch vụ bên thứ ba, giúp phát triển các ứng dụng web phức tạp và có tính mở rộng cao.

***1.2.4. Tổng quan về ExpressJS***

**Giới thiệu ExpressJS**

ExpressJS là một framework mạnh mẽ và phổ biến được xây dựng trên nền tảng **NodeJS**. Nó giúp đơn giản hóa quá trình phát triển ứng dụng web bằng cách cung cấp các công cụ và thư viện cần thiết để xử lý yêu cầu và phản hồi HTTP một cách dễ dàng.

ExpressJS cho phép bạn xây dựng các ứng dụng web độc lập, linh hoạt và có hiệu suất cao. Với cú pháp đơn giản và dễ hiểu, ExpressJS giúp lập trình viên tập trung vào việc xây dựng logic ứng dụng thay vì phải lo lắng về các chi tiết cơ bản của việc xử lý yêu cầu và phản hồi.

Điều đặc biệt về ExpressJS là khả năng sử dụng **middleware**. Middleware là các chức năng trung gian được thực thi trước khi yêu cầu của người dùng được xử lý hoặc sau khi phản hồi được gửi đi. Điều này giúp bạn thực hiện các tác vụ như xác thực người dùng, ghi log, xử lý lỗi và nhiều hơn nữa một cách dễ dàng.

**Tính năng của ExpressJS là gì?**

|  |
| --- |
| **A purple and white diagram  Description automatically generated** |
| *Hình 1.5. Tính năng của ExpressJS là gì?* |

ExpressJS là một nền tảng lập trình web phổ biến trên NodeJS, mã nguồn mở và được áp dụng rộng rãi. Nó đi kèm với một loạt tính năng mạnh mẽ, cho phép lập trình viên xây dựng và triển khai các ứng dụng web nhanh chóng và dễ dàng. Dưới đây là một vài tính năng chính của ExpressJS:

**Phát triển máy chủ nhanh hơn:** ExpressJS tối ưu hóa cú pháp và cung cấp các phương thức và hàm tiện ích để xử lý các tác vụ phổ biến trong lập trình web. Nhờ đó, bạn có thể viết code ngắn gọn và tối giản hóa quy trình phát triển.

**Định hướng (Routing):** ExpressJS cung cấp một hệ thống định hướng hiệu quả, giúp bạn định nghĩa các đường dẫn (routes) để điều khiển các yêu cầu từ người dùng và trả lời phù hợp. Điều này giúp tổ chức và quản lý các thành phần của ứng dụng một cách dễ dàng.

**Middleware:** ExpressJS hỗ trợ middleware, cho phép bạn thêm các chức năng trung gian vào quy trình xử lý yêu cầu và phản hồi. Middleware giúp xác thực người dùng, ghi log, xử lý lỗi, nén dữ liệu và thực hiện nhiều tác vụ khác một cách linh hoạt.

**Cấu hình môi trường:** ExpressJS cung cấp một cách để cấu hình môi trường phát triển và môi trường sản xuất. Bạn có thể thiết lập các biến môi trường, cấu hình định dạng và quy tắc, tùy chỉnh ứng dụng của mình theo các môi trường khác nhau.

**Xử lý lỗi:** ExpressJS cung cấp cơ chế xử lý lỗi cho phép bạn kiểm soát và xử lý các lỗi xảy ra trong quá trình xử lý yêu cầu. Bạn có thể tạo ra các middleware để xử lý lỗi và phản hồi với các thông báo lỗi tùy chỉnh.

**Ưu điểm của ExpressJS**

Ngôn ngữ đơn giản và sử dụng dễ dàng: ExpressJS sở hữu cú pháp rõ ràng và dễ hiểu, cho phép các lập trình viên nhanh chóng hiểu rõ và triển khai những tính năng mới một cách dễ dàng.

Linh hoạt: ExpressJS không áp đặt một cấu trúc cụ thể, cho phép lập trình viên tự do tùy chỉnh và xây dựng ứng dụng theo ý muốn.

Hỗ trợ middleware: ExpressJS cung cấp hệ thống middleware mạnh mẽ, cho phép thực hiện các chức năng như xác thực, ghi log, nén dữ liệu và xử lý lỗi một cách linh hoạt và dễ dàng.

Hiệu suất cao: ExpressJS được xây dựng trên NodeJS, nền tảng có hiệu suất cao, cho phép xử lý nhanh chóng các yêu cầu web đồng thời và có khả năng mở rộng tốt.

**Nhược điểm của ExpressJS**

Thiếu cấu trúc: Do ExpressJS không áp đặt một cấu trúc nghiêm ngặt, việc tổ chức dự án và quản lý mã nguồn có thể trở nên khó khăn, đặc biệt khi ứng dụng phát triển lớn và phức tạp.

Khả năng mở rộng: Khi ứng dụng phát triển lớn và phức tạp, việc quản lý mã nguồn và mở rộng có thể trở nên khó khăn với ExpressJS. Cần có sự kiểm soát cẩn thận để tránh sự phức tạp và rối rắm trong việc quản lý các module và tương tác giữa chúng.

Cộng đồng hỗ trợ: Mặc dù ExpressJS có một cộng đồng lớn và đầy đủ tài liệu, tuy nhiên, không đạt được mức độ hỗ trợ như các framework web khác như **Angular** hoặc **React**.

***1.2.5. Tổng quan về MongoDB***

**Giới thiệu MongoDB**

MongoDB là một cơ sở dữ liệu hướng tài liệu (document), khác với các cơ sở dữ liệu quan hệ như MySQL sử dụng các bảng (table) để phân chia dữ liệu và dùng các trường (column) để định nghĩa dữ liệu, MongoDB không yêu cầu bạn phải chỉ định dạng dữ liệu của mình. Thay vào đó, nó sử dụng một schema rất linh hoạt được gọi là BSON. BSON có thể được hiểu đơn giản là dạng document JSON. Chính vì dữ liệu không bị ràng buộc nên các đơn vị lưu trữ của MongoDB, hay còn được gọi là các collection sẽ có kích thước và các document khác nhau. Ngoài ra, việc lưu trữ các document dưới dạng JSON là một lợi thế giúp tăng hiệu suất cho các quá trình truy vấn và duyệt dữ liệu.

MongoDB được sử dụng rất nhiều bởi các doanh nghiệp lớn, chúng ta có thể kể đến ở đây như: eBay, SourceForge và The New York Times. Được tin dùng như vậy bởi các tập đoàn lớn là vì DB này mang lại rất nhiều tính năng ưu việt đáp ứng được nhu cầu tải trọng cao và cần sự ổn định - các yếu tố tiên quyết để có thể xây dựng nên các hệ thống lớn.

Một số tính năng của MongoDB có thể được kể đến như:

**Các ad hoc query:** hỗ trợ search bằng field, các phép search thông thường, regular expression searches, và range queries.

**Indexing:**bất kì field nào trong BSON document cũng có thể được index.

**Replication:** có ý nghĩa là “nhân bản”, là có một phiên bản giống hệt phiên bản đang tồn tại, đang sử dụng. Với cơ sở dữ liệu, nhu cầu lưu trữ lớn, đòi hỏi cơ sở dữ liệu toàn vẹn, không bị mất mát trước những sự cố ngoài dự đoán là rất cao. Vì vậy, người ta nghĩ ra khái niệm “nhân bản”, tạo một phiên bản cơ sở dữ liệu giống hệt cơ sở dữ liệu đang tồn tại, và lưu trữ ở một nơi khác, đề phòng có sự cố.

**Aggregation:**Các Aggregation operation xử lý các bản ghi dữ liệu và trả về kết quả đã được tính toán. Các phép toán tập hợp nhóm các giá trị từ nhiều Document lại với nhau, và có thể thực hiện nhiều phép toán đa dạng trên dữ liệu đã được nhóm đó để trả về một kết quả duy nhất. Trong SQL, count(\*) và GROUP BY là tương đương với Aggregation trong MongoDB.

**Lưu trữ file:**MongoDB được dùng như một hệ thống file tận dụng những function trên và hoạt động như một cách phân phối qua sharding.

|  |
| --- |
| mongodb la gi 1 TECHVIFY Careers |
| *Hình 1.6.* MongoDB là một phần mềm cơ sở dữ liệu mã nguồn mở NoSQL |

**Tìm hiểu về MongoDB: Khái niệm NoSQL**

NoSQL là 1 dạng CSDL mã nguồn mở và được viết tắt bởi: Non-Relational SQL hay có nơi thường gọi là Not-Only SQL.

NoSQL được phát triển trên Javascript Framework với kiểu dữ liệu là JSON và dạng dữ liệu theo kiểu key và value.

NoSQL ra đời như là 1 mảnh vá cho những khuyết điểm và thiếu xót cũng như hạn chế của mô hình dữ liệu quan hệ RDBMS (Relational Database Management System - Hệ quản trị cơ sở dữ liệu quan hệ) về tốc độ, tính năng, khả năng mở rộng,...

Với NoSQL bạn có thể mở rộng dữ liệu mà không lo tới những việc như tạo khóa ngoại, khóa chính, kiểm tra ràng buộc ...

NoSQL bỏ qua tính toàn vẹn của dữ liệu và transaction để đổi lấy hiệu suất nhanh và khả năng mở rộng.

NoSQL được sử dụng ở rất nhiều công ty, tập đoàn lớn, ví dụ như FaceBook sử dụng Cassandra do FaceBook phát triển, Google phát triển và sử dụng BigTable,...

#### NoSQL hoạt động như thế nào

Cơ sở dữ liệu NoSQL sử dụng nhiều mô hình dữ liệu để truy cập và quản lý dữ liệu. Các loại cơ sở dữ liệu này được tối ưu hóa dành riêng cho các ứng dụng yêu cầu mô hình dữ liệu linh hoạt có lượng dữ liệu lớn và độ trễ thấp, có thể đạt được bằng cách giảm bớt một số hạn chế về tính nhất quán của dữ liệu của các cơ sở dữ liệu khác.

|  |
| --- |
| mongodb la gi 2 TECHVIFY Careers |
| *Hình 1.7.* Truy vấn ad hoc được xem là tính năng tốt nhất |

**Ưu điểm của MongoDB**

Do MongoDB sử dụng lưu trữ dữ liệu dưới dạng Document JSON nên mỗi một collection sẽ có các kích cỡ và các document khác nhau, linh hoạt trong việc lưu trữ dữ liệu, nên bạn muốn gì thì cứ insert vào thoải mái.

Dữ liệu trong MongoDB không có sự ràng buộc lẫn nhau, không có join như trong RDBMS nên khi insert, xóa hay update nó không cần phải mất thời gian kiểm tra xem có thỏa mãn các ràng buộc dữ liệu như trong RDBMS.

MongoDB rất dễ mở rộng (Horizontal Scalability). Trong MongoDB có một khái niệm cluster là cụm các node chứa dữ liệu giao tiếp với nhau, khi muốn mở rộng hệ thống ta chỉ cần thêm một node với vào cluster:

Trường dữ liệu “\_id” luôn được tự động đánh index (chỉ mục) để tốc độ truy vấn thông tin đạt hiệu suất cao nhất.

Khi có một truy vấn dữ liệu, bản ghi được cached lên bộ nhớ Ram, để phục vụ lượt truy vấn sau diễn ra nhanh hơn mà không cần phải đọc từ ổ cứng.

Hiệu năng cao: Tốc độ truy vấn (find, update, insert, delete) của MongoDB nhanh hơn hẳn so với các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS). Với một lượng dữ liệu đủ lớn thì thử nghiệm cho thấy tốc độ insert của MongoDB có thể nhanh tới gấp 100 lần so với MySQL.

**Nhược điểm của MongoDB**

Không ứng dụng được cho các mô hình giao dịch nào có yêu cầu độ chính xác cao do không có ràng buộc.

Không có cơ chế transaction (giao dịch) để phục vụ các ứng dụng ngân hàng.

Dữ liệu lấy RAM làm trọng tâm hoạt động vì vậy khi hoạt động yêu cầu một bộ nhớ RAM lớn.

Mọi thay đổi về dữ liệu mặc định đều chưa được ghi xuống ổ cứng ngay lập tức vì vậy khả năng bị mất dữ liệu từ nguyên nhân mất điện đột xuất là rất cao.

**Nên dùng MongoDB trong trường hợp nào?**

**Khi quản lý và truyền tải content:**quản lý đa dạng nhiều sản phẩm và có nội dung khác nhau chỉ trong một kho lưu trữ dữ liệu cho phép thay đổi và phản hồi nhanh chóng, không làm phức tạp hóa thêm từ hệ thống content

**Cấu trúc mobile và social:** MongoDB cung cấp một platform có sẵn, phản xạ nhanh, và dễ mở rộng cho phép rất nhiều khả năng đột phá, phân tích real-time, và hỗ trợ toàn cầu.

**Quản lý data khách hàng:**tận dụng khả năng query nhanh chóng cho phân tích real-time trên cơ sở dữ liệu người dùng cực lớn với các mô hình data phức tạp bằng các schema linh hoạt và tự động sharding cho mở rộng chiều ngang.

**1.3. Coze AI Bot**

Coze là một nền tảng phát triển bot trò chuyện AI đa năng. Bất kể bạn có kỹ năng lập trình như thế nào, Coze cho phép bạn nhanh chóng tạo ra các loại bot trò chuyện phong phú được trang bị AI, có khả năng xử lý từ những câu hỏi đơn giản đến các cuộc đối thoại phức tạp. Hơn nữa, bạn có thể xuất bản bot của mình lên các ứng dụng khác nhau, tạo điều kiện cho tương tác động với người dùng trên những ứng dụng đó.

A screenshot of a computer

Description automatically generated

*Hình 1.8.* Coze AI tích hợp được nhiều công việc khác nhau

**Tại sao nên chọn Coze?**

Coze đơn giản hóa và tối ưu hóa quá trình phát triển bot AI với những tính năng sau:

**Plugins:** mở rộng khả năng của bot của bạn với bộ sưu tập plugin đa dạng.

**Built-in plugins**: tích hợp gần một trăm plugin hiện tại, giúp nâng cao tính năng của bot ở nhiều lĩnh vực như trợ giúp thông tin, hỗ trợ du lịch, nâng cao năng suất, hiểu hình ảnh, và nhiều hơn nữa.

**Custom plugins:** nhanh chóng tạo ra plugin tùy chỉnh bằng cách chỉ định đầu vào và đầu ra từ API hiện có.

**Knowledge:** thêm dữ liệu của bạn vào bot với cách quản lý và lưu trữ dữ liệu thân thiện và dễ hiểu.

**Database & variable:** sử dụng bộ nhớ dài hạn với cơ sở dữ liệu và biến số.

**Scheduled tasks:** tự động thực hiện các nhiệm vụ theo lịch trình.

**Workflow:** tạo luồng công việc linh hoạt với giao diện kéo và thả đơn giản.

**Multi agents:** chạy đa nhiệm trong một bot để xử lý các nhiệm vụ phức tạp của người dùng.

**CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**

**2.1. Thu thập và phân tích yêu cầu**

Hệ thống có chức năng ủy quyền người dùng, do đó:

**Đối với người dùng là người đọc - không phải thành viên:**

* Được xem thông tin về các địa danh di tích lịch sử của thủ đô Hà Nội, xem thông của người đăng bài
* Được xem lượt tương tác và các bình luận của các bài đăng
* Được tìm kiếm các bài đăng và người đăng
* Được đăng ký tài khoản, đăng nhập

**Đối với người dùng là người đọc - thành viên:**

* Được xem thông tin về các địa danh di tích lịch sử của thủ đô Hà Nội, xem thông của người đăng bài
* Được xem lượt tương tác và các bình luận của các bài đăng
* Được tìm kiếm các bài đăng và người đăng
* Được đăng ký tài khoản, đăng nhập
* Được thả cảm xúc và viết bình luận
* Được nhận thông báo khi người khác phản hồi lại bình luận
* Được quản lý tài khoản thành viên của mình
* Sử dụng chatbot để tìm kiếm thông tin về di tích lịch sử của thủ đô Hà Nội

**Đối với người dùng là người quản trị:**

Ngoài các tính năng của người đọc - thành viên:

* Được thêm bài viết về di tích lịch sử của thủ đô Hà Nội
* Được chỉnh sửa thông tin của bài viết: Khi có sự thay đổi về thông tin di tích, lịch sử, người quản lý có thể thay đổi thông tin để người dùng cập nhật kịp thời
* Được xóa bài viết: Có thể vì một lý do nào đó hoặc buộc bị xóa khỏi hệ thống thì người quản lý cũng có thể xóa được bài đăng đó
* Được xóa tài khoản các thành viên: Khi người dùng vi phạm các điều luật không gian mạng hoặc comment gây ảnh hưởng đến các người dùng khác hoặc vì một lý do nào đó yêu cầu xóa tài khoản, người quản lý có thể xóa toàn bộ thông tin liên quan đến tài khoản đó
* Được xem các thống kê về các bài đăng như số lượt like, comment, lượt đọc

**2.2. Các thành phần hệ thống**

**- Giao diện người dùng (Front-end)**:

* Trang chủ: Cung cấp thông tin giới thiệu về website và các di tích lịch sử nổi bật.
* Trang chi tiết di tích: Hiển thị thông tin chi tiết về từng di tích lịch sử, bao gồm hình ảnh, mô tả, lịch sử, và các sự kiện liên quan.
* Trang liên hệ: Cho phép người dùng liên hệ với ban quản trị hoặc đưa ra ý kiến, phản hồi.
* Trang đăng ký/đăng nhập: Cung cấp cho người dùng khả năng tạo tài khoản và đăng nhập để truy cập vào các tính năng mở rộng.
* Chatbot: Giao diện để người dùng tương tác với chatbot, bao gồm cả cửa sổ chat và các nút tương tác.

- Hệ thống quản lý nội dung (CMS):

* Quản lý di tích lịch sử: Cho phép thêm, chỉnh sửa và xóa thông tin về các di tích lịch sử.
* Quản lý thông tin khách hàng: Giám sát chi tiết về khách hàng, cấp phép quyền truy nhập và theo dõi các hoạt động của họ trên trang web.
* Quản lý chatbot: Cấu hình và quản lý các câu hỏi, câu trả lời và kịch bản tương tác của chatbot.

- **Backend/Server-side**:

* Máy chủ API: Cung cấp những API để giao diện người sử dụng có khả năng tác động đến cơ sở dữ liệu và hệ thống quản trị nội dung.

- **Cơ sở dữ liệu (Database)**:

* Thông tin di tích lịch sử: Lưu trữ thông tin chi tiết về các di tích lịch sử.
* Thông tin người dùng: Lưu trữ thông tin đăng ký và hoạt động của người dùng.

- **Chatbot**:

* **Kịch bản tương tác**: Xây dựng các kịch bản tương tác để hướng dẫn người dùng tìm kiếm thông tin về các di tích lịch sử.

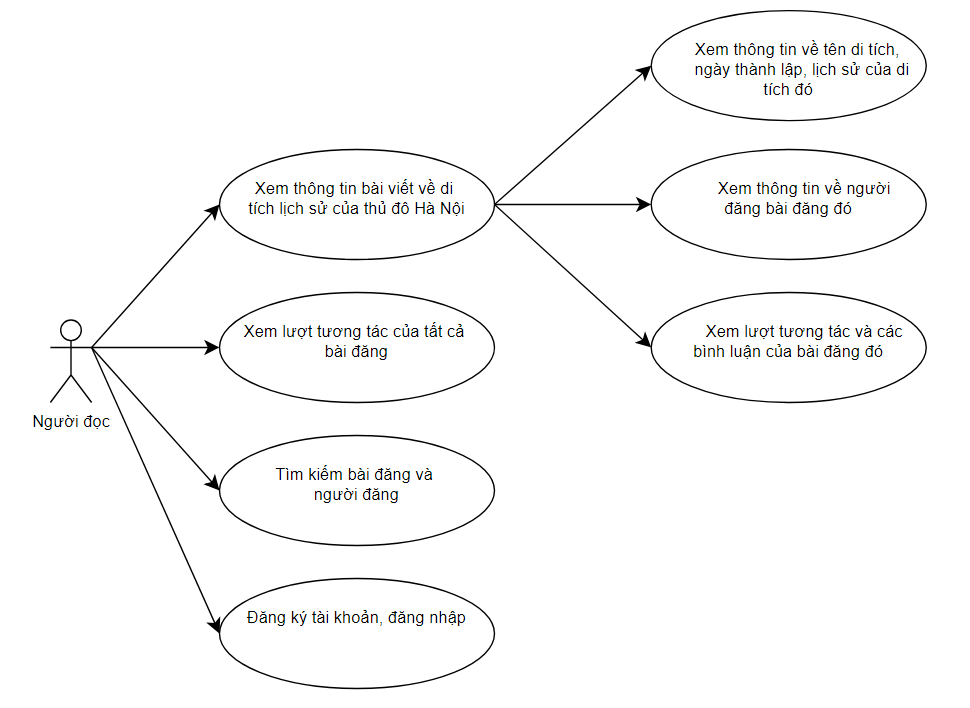
**2.3. Chức năng hệ thống**

Bảng 2.1. Bảng đặc tả hệ thống

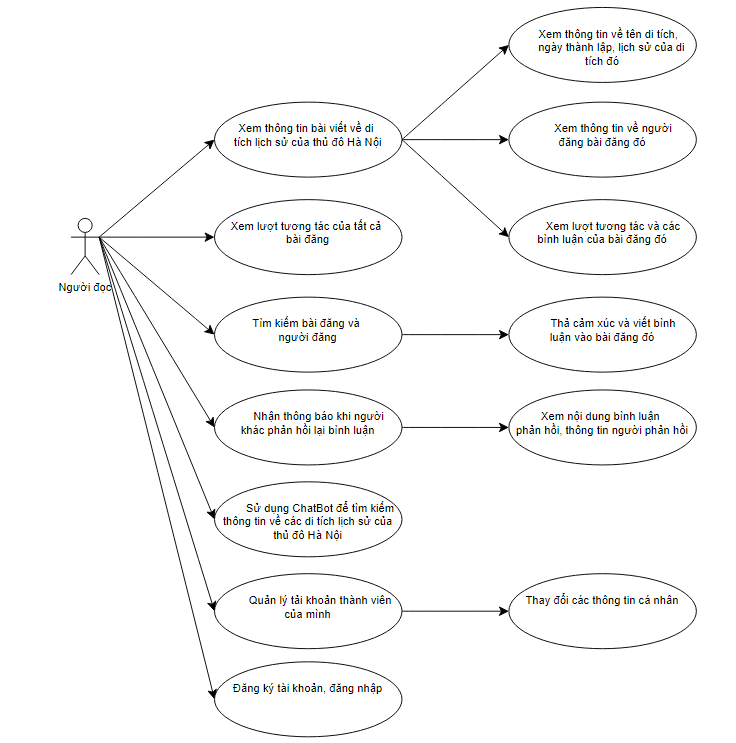
|  |  |  |
| --- | --- | --- |
| **Vai trò** | | **Mô tả** |
| Người quản trị | | Người giữ vai trò quan trọng nhất trong hệ thống, quản lý có quyền thêm, chỉnh sửa và xóa tất cả các thông tin của hệ thống. |
| Khách hàng | Không phải thành viên | Thao tác trực tiếp trên hệ thống người dùng, với các chức năng: xem các post về di tích lịch sử của thủ đô Hà Nội, xem lượt đánh giá và các bình luận về bài viết đó, tìm kiếm các bài đăng và người dùng khác, đăng ký tài khoản, đăng nhập |
| Thành viên | Vận hành các chức năng của người đọc - không phải thành viên, ngoài ra người đọc thành viên có thể thả cảm xúc, viết bình luận về các bài đăng mà người đọc quan tâm, có thông báo khi người khác phản hồi bình luận của mình, quản lý tài khoản thành viên của mình, sử dụng chatbot để tìm kiếm thông tin |

***2.3.1 Sơ đồ use-case***

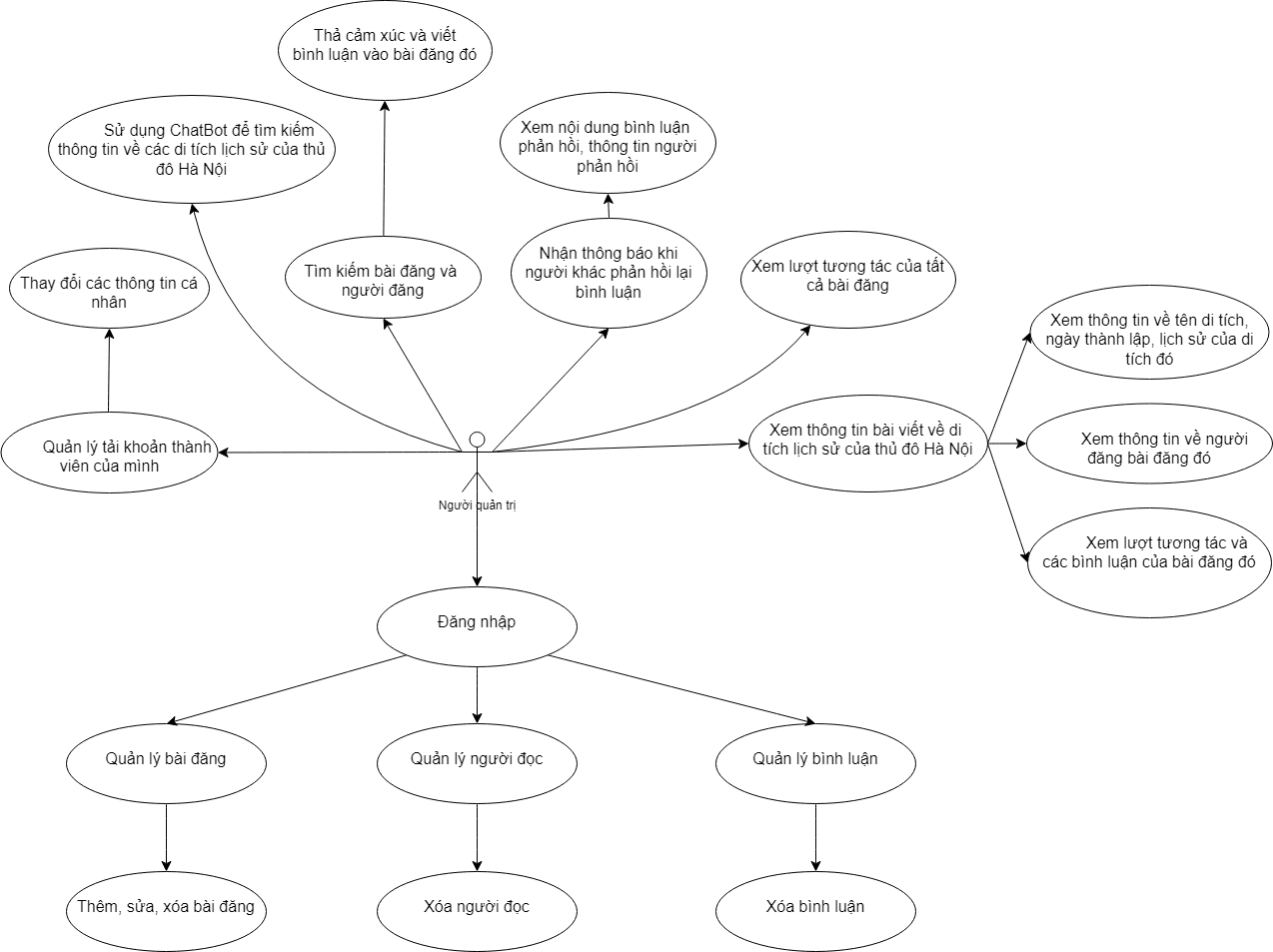
Biểu đồ trường hợp sử dụng (use-case) là một phương pháp được sử dụng trong kỹ thuật phần mềm và hệ thống để nhận diện các yêu cầu chức năng của hệ thống. Hoặc, Sơ đồ trường hợp sử dụng là một công cụ quan trọng trong lập trình phần mềm và thiết kế hệ thống, giúp xác định các yêu cầu chức năng cần thiết cho hệ thống. Use case mô tả sự tương tác đặc trưng giữa người dùng bên ngoài và hệ thống. Use case cũng mô tả các yêu cầu đối với hệ thống.

****

Hình 2.1. Biểu đồ use-case của người đọc

****

Hình 2.2. Biểu đồ use-case của người đọc - thành viên

****

Hình 2.3. Biểu đồ use-case của người quản trị

**2.3.2. Đặc tả Use Case**

|  |  |  |
| --- | --- | --- |
| *Bảng 2.2. Bảng đặc tả đăng nhập* | | |
| Tên use case | Đăng nhập | |
| Tác nhân chính | Quản trị viên và người dùng có tài khoản | |
| Mục đích | Người dùng hoặc quản trị viên đã đăng nhập thành công vào tài khoản tương ứng. | |
| Mức độ ưu tiên (Priority) | Bắt buộc phải có | |
| Mức | Mức 0 | |
| Điều kiện kích hoạt (Trigger) | Khi người dùng hoặc quản trị viên dùng muốn đăng nhập vào tài khoản và ấn nút Đăng Nhập | |
| Điều kiện tiên quyết (Pre-condition) | Người dùng hoặc quản trị viên phải có tài khoản | |
| Điều kiện thành công (Post-condition) | Người dùng hoặc quản trị viên đăng nhập thành công vào tài khoản của mình | |
| Điều kiện thất bại | Người dùng không đăng nhập được vào tài khoản của mình | |
| Luồng sự kiện chính (Basic flow) | Bước | Hành động |
| 1 | Người dùng nhập email và password |
| 2 | Hệ thống xác minh email và password của người dùng là đúng |
| 3 | Người dùng hoặc quản trị viên đăng nhập thành công |
| Luồng sự kiện ngoại lệ (Exception Flow) | 2a. Hệ thống xác minh email hoặc password bị sai | |
| 2a1. Hệ thống hiển thị thông báo và người dùng nhập lại email và password | |

|  |  |  |
| --- | --- | --- |
| *Bảng 2.3. Bảng đặc tả đăng ký* | | |
| Tên use case | Đăng ký | |
| Tác nhân chính | Người dùng mới | |
| Mục đích | Cho phép người dùng mới tạo tài khoản để sử dụng dịch vụ | |
| Mức độ ưu tiên (Priority) | Bắt buộc phải có | |
| Mức | Mức 0 | |
| Điều kiện kích hoạt (Trigger) | Khi người dùng mới muốn tạo tài khoản và ấn nút Đăng Ký | |
| Điều kiện tiên quyết (Pre-condition) | Người dùng mới chưa có tài khoản | |
| Điều kiện thành công (Post-condition) | Người dùng mới đăng ký tài khoản thành công và được chuyển hướng đến trang đăng nhập | |
| Điều kiện thất bại | Người dùng không thể đăng ký tài khoản | |
| Luồng sự kiện chính (Basic flow) | Bước | Hành động |
| 1 | Người dùng nhập thông tin cần thiết (fullname, email, password) |
| 2 | Hệ thống kiểm tra tính hợp lệ của thông tin |
| 3 | Hệ thống lưu thông tin người dùng và tạo tài khoản mới |
| 4 | Người dùng được chuyển hướng đến trang đăng nhập |
| Luồng sự kiện ngoại lệ (Exception Flow) | 2a. Hệ thống xác minh thông tin không hợp lệ (ví dụ: email đã tồn tại, password không khớp) | |
| 2a1. Hệ thống thông báo lỗi và yêu cầu người dùng cung cấp lại thông tin | |

|  |  |  |
| --- | --- | --- |
| *Bảng 2.4. Bảng đặc tả đổi mật khẩu, cập nhật thông tin* | | |
| Tên use case | Đổi mật khẩu, cập nhật thông tin | |
| Tác nhân chính | Người dùng có tài khoản | |
| Mục đích | Cho phép người dùng cập nhật thông tin cá nhân của mình | |
| Mức độ ưu tiên (Priority) | Không bắt buộc | |
| Mức | Mức 1 | |
| Điều kiện kích hoạt (Trigger) | Khi người dùng muốn cập nhật thông tin và ấn nút Lưu Thay Đổi | |
| Điều kiện tiên quyết (Pre-condition) | Người dùng phải đăng nhập vào tài khoản của mình và đang ở màn chỉnh sửa hồ sơ | |
| Điều kiện thành công (Post-condition) | Thông báo cập nhật thành công | |
| Điều kiện thất bại | Thông báo cập nhật thất bại | |
| Luồng sự kiện chính (Basic flow) | Bước | Hành động |
| 1 | Người dùng nhập thông tin cá nhân mới |
| 2 | Hệ thống kiểm tra tính hợp lệ của thông tin mới |
| 3 | Hệ thống cập nhật thông tin cá nhân của người dùng |
| 4 | Hệ thống hiển thị thông báo cập nhật thông tin thành công |
| Luồng sự kiện ngoại lệ (Exception Flow) | 2a. Dữ liệu bị bỏ trống không cập nhật được | |

|  |  |  |
| --- | --- | --- |
| *Bảng 2.5. Bảng đặc tả thả cảm xúc bài đăng* | | |
| Tên use case | Thả cảm xúc bài đăng | |
| Tác nhân chính | Người dùng có tài khoản | |
| Mục đích | Cho phép người dùng thể hiện cảm xúc với bài viết | |
| Mức độ ưu tiên (Priority) | Không bắt buộc | |
| Mức | Mức 1 | |
| Điều kiện kích hoạt (Trigger) | Khi người dùng muốn thả cảm xúc cho một bài viết và ấn vào biểu tượng cảm xúc | |
| Điều kiện tiên quyết (Pre-condition) | Người dùng phải đăng nhập vào tài khoản của mình | |
| Điều kiện thành công (Post-condition) | Cảm xúc của người dùng được ghi nhận thành công và hiển thị trên bài viết | |
| Điều kiện thất bại | Người dùng không thể thả cảm xúc cho bài viết | |
| Luồng sự kiện chính (Basic flow) | Bước | Hành động |
| 1 | Người dùng nhấn vào biểu tượng cảm xúc trên bài viết |
| 2 | Hệ thống ghi nhận cảm xúc của người dùng cho bài viết |
| 3 | Hệ thống cập nhật và hiển thị cảm xúc của người dùng trên các bài đăng |
| Luồng sự kiện ngoại lệ (Exception Flow) | 2a. Hệ thống không thể ghi nhận cảm xúc (ví dụ: lỗi kết nối) | |

|  |  |  |
| --- | --- | --- |
| *Bảng 2.6. Bảng đặc tả bình luận bài đăng* | | |
| Tên use case | Thả cảm xúc bài đăng | |
| Tác nhân chính | Người dùng có tài khoản | |
| Mục đích | Cho phép người dùng thêm bình luận vào bài đăng | |
| Mức độ ưu tiên (Priority) | Không bắt buộc | |
| Mức | Mức 1 | |
| Điều kiện kích hoạt (Trigger) | Khi người dùng muốn bình luận vào một bài đăng và ấn nút Bình Luận | |
| Điều kiện tiên quyết (Pre-condition) | Người dùng phải đăng nhập vào tài khoản của mình | |
| Điều kiện thành công (Post-condition) | Bình luận của người dùng được ghi nhận thành công và hiển thị dưới bài đăng | |
| Điều kiện thất bại | Người dùng không thể bình luận vào bài đăng | |
| Luồng sự kiện chính (Basic flow) | Bước | Hành động |
| 1 | Người dùng nhập nội dung bình luận vào bài đăng |
| 2 | Người dùng ấn nút Bình Luận |
| 3 | Hệ thống ghi nhận bình luận của người dùng |
| 4 | Hệ thống cập nhật và hiển thị bình luận của người dùng dưới bài đăng |
| Luồng sự kiện ngoại lệ (Exception Flow) | 2a. Hệ thống không thể ghi nhận bình luận (ví dụ: lỗi kết nối) | |

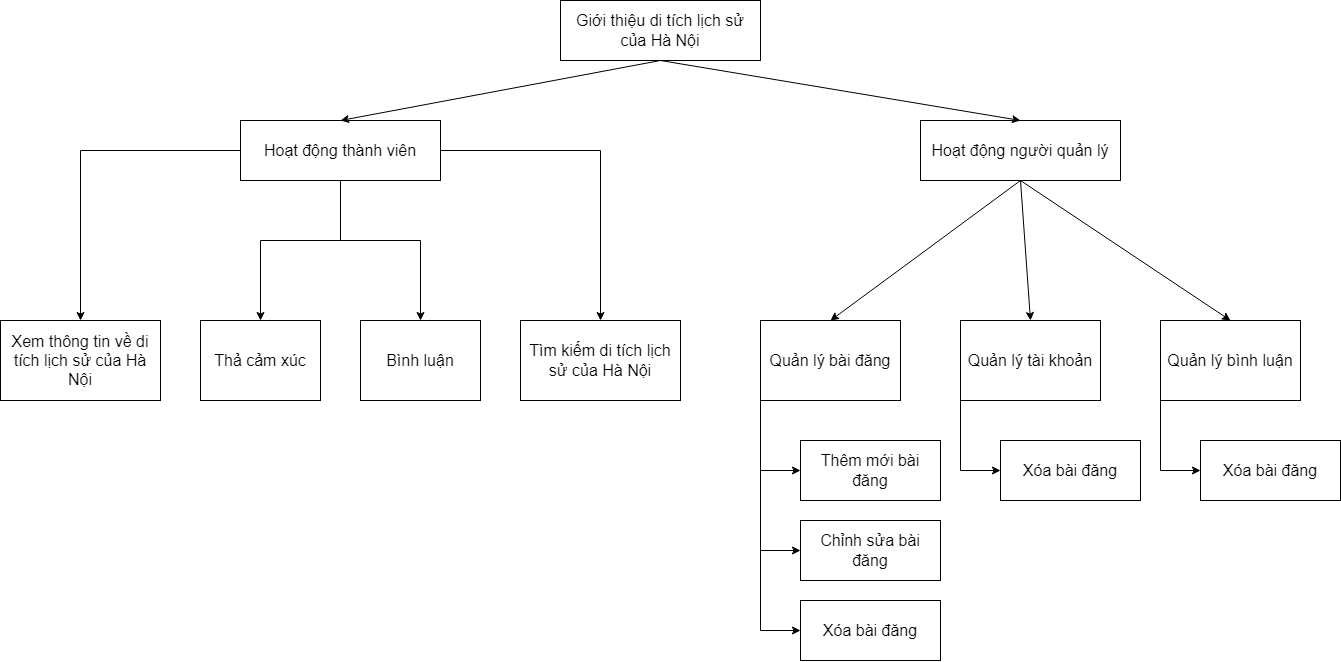
|  |  |  |
| --- | --- | --- |
| *Bảng 2.7. Bảng đặc tả thêm mới bài đăng* | | |
| Tên use case | Thêm mới bài đăng | |
| Tác nhân chính | Quản trị viên | |
| Mục đích | Cho phép quản trị viên thêm mới bài đăng trên hệ thống | |
| Mức độ ưu tiên (Priority) | Bắt buộc phải có | |
| Mức | Mức 0 | |
| Điều kiện kích hoạt (Trigger) | Khi quản trị viên muốn thêm mới một bài đăng và ấn nút Đăng Bài | |
| Điều kiện tiên quyết (Pre-condition) | Quản trị viên phải đăng nhập vào tài khoản quản trị | |
| Điều kiện thành công (Post-condition) | Bài đăng mới được thêm thành công và hiển thị trên hệ thống | |
| Điều kiện thất bại | Quản trị viên không thể thêm mới bài đăng | |
| Luồng sự kiện chính (Basic flow) | Bước | Hành động |
| 1 | Quản trị viên nhập ảnh banner, tiêu đề và nội dung bài đăng |
| 2 | Quản trị viên thêm hình ảnh, video hoặc các tài liệu đính kèm (nếu có) |
| 3 | Quản trị viên ấn nút Đăng Bài |
| 4 | Hệ thống ghi nhận thông tin bài đăng mới |
| 5 | Hệ thống cập nhật và hiển thị bài đăng mới trên hệ thống |
| Luồng sự kiện ngoại lệ (Exception Flow) | 2a. Hệ thống không thể ghi nhận thông tin bài đăng mới (ví dụ: lỗi kết nối) | |

|  |  |  |
| --- | --- | --- |
| *Bảng 2.8. Bảng đặc tả chỉnh sửa bài đăng* | | |
| Tên use case | Chỉnh sửa bài đăng | |
| Tác nhân chính | Quản trị viên | |
| Mục đích | Cho phép quản trị viên chỉnh sửa nội dung bài đăng trên hệ thống | |
| Mức độ ưu tiên (Priority) | Không bắt buộc | |
| Mức | Mức 1 | |
| Điều kiện kích hoạt (Trigger) | Khi quản trị viên muốn chỉnh sửa bài đăng và ấn nút Lưu Chỉnh Sửa | |
| Điều kiện tiên quyết (Pre-condition) | Quản trị viên phải đăng nhập vào tài khoản quản trị và bài đăng cần chỉnh sửa đã tồn tại trên hệ thống | |
| Điều kiện thành công (Post-condition) | Bài đăng được chỉnh sửa thành công và cập nhật trên hệ thống | |
| Điều kiện thất bại | Quản trị viên không thể chỉnh sửa bài đăng | |
| Luồng sự kiện chính (Basic flow) | Bước | Hành động |
| 1 | Quản trị viên chọn bài đăng cần chỉnh sửa |
| 2 | Quản trị viên chỉnh sửa tiêu đề, nội dung bài đăng và các tài liệu đính kèm (nếu cần) |
| 3 | Quản trị viên ấn nút Lưu Chỉnh Sửa |
| 4 | Hệ thống ghi nhận thông tin chỉnh sửa mới |
| 5 | Hệ thống cập nhật và hiển thị bài đăng đã chỉnh sửa trên hệ thống |
| Luồng sự kiện ngoại lệ (Exception Flow) | 2a. Hệ thống không thể ghi nhận thông tin chỉnh sửa mới (ví dụ: lỗi kết nối) | |

|  |  |  |
| --- | --- | --- |
| *Bảng 2.9. Bảng đặc tả xóa bài đăng* | | |
| Tên use case | Xóa bài đăng | |
| Tác nhân chính | Quản trị viên | |
| Mục đích | Cho phép quản trị viên xóa bài đăng khỏi hệ thống | |
| Mức độ ưu tiên (Priority) | Không bắt buộc | |
| Mức | Mức 1 | |
| Điều kiện kích hoạt (Trigger) | Khi quản trị viên muốn xóa một bài đăng và ấn nút Xóa | |
| Điều kiện tiên quyết (Pre-condition) | Quản trị viên phải đăng nhập vào tài khoản quản trị và bài đăng cần xóa đã tồn tại trên hệ thống | |
| Điều kiện thành công (Post-condition) | Bài đăng được xóa thành công khỏi hệ thống | |
| Điều kiện thất bại | Quản trị viên không thể xóa bài đăng | |
| Luồng sự kiện chính (Basic flow) | Bước | Hành động |
| 1 | Quản trị viên chọn bài đăng cần xóa |
| 2 | Quản trị viên xác nhận việc xóa bài đăng |
| 3 | Hệ thống ghi nhận yêu cầu xóa bài đăng |
| 4 | Hệ thống xóa bài đăng khỏi hệ thống |
| 5 | Hệ thống hiển thị thông báo xóa bài đăng thành công |
| Luồng sự kiện ngoại lệ (Exception Flow) | 2a. Hệ thống không thể xóa bài đăng (ví dụ: lỗi kết nối) | |

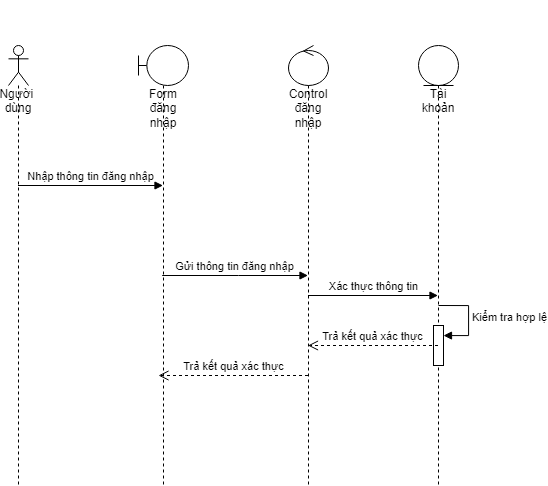
*2.3.2. Biểu đồ phân rã chức năng*

Mô hình phân tách chức năng (BFD - Business Function Diagram) là một công cụ để biểu diễn việc phân tách một cách đơn giản các nhiệm vụ cần thực hiện. Mỗi nhiệm vụ được chia thành nhiều nhiệm vụ con, mức độ phân chia này sẽ phụ thuộc vào kích thước và độ phức tạp của hệ thống.

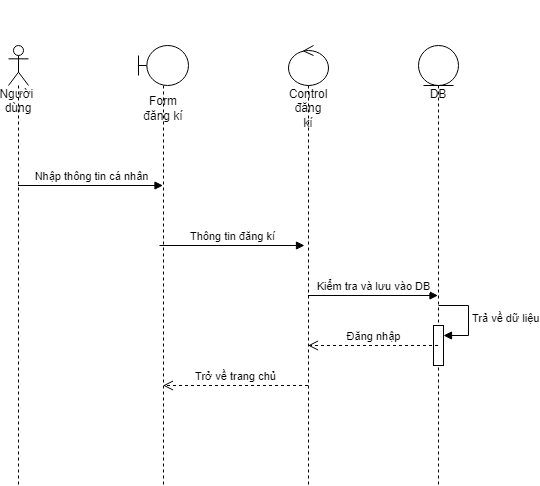


*Hình 2.4. Biểu đồ phân rã chức năng*

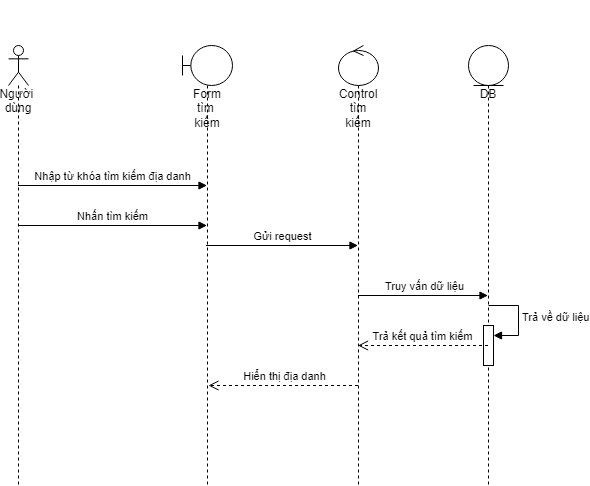
**2.3.3. Biểu đồ tuần tự**

****

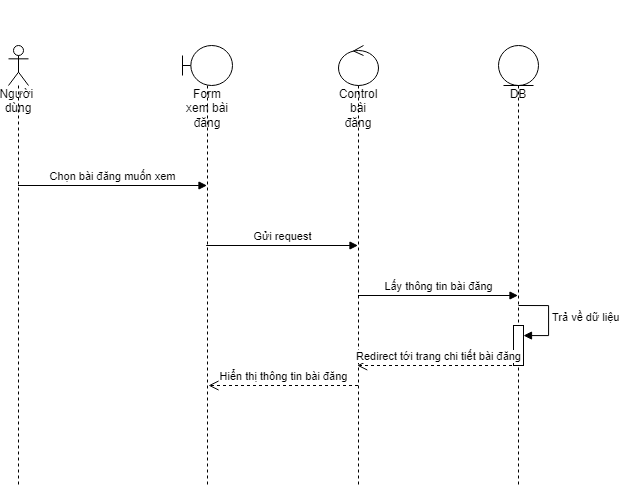
*Hình 2.5. Biểu đồ tuần tự đăng nhập cho người dùng hoặc quản trị viên*

****

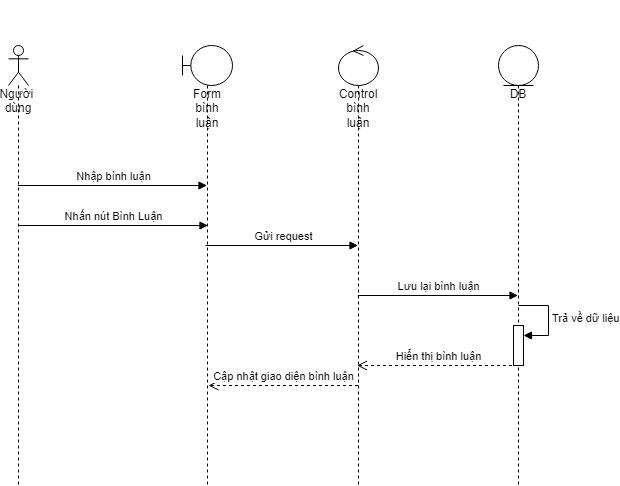
*Hình 2.6. Biểu đồ tuần tự đăng ký thành viên cho người dùng*

****

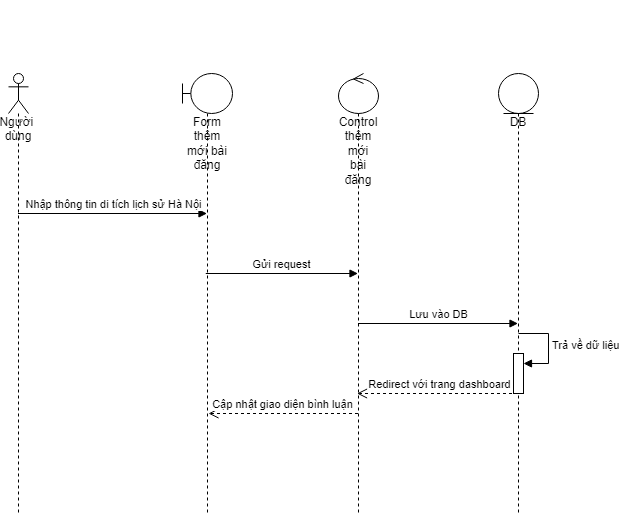
*Hình 2.7. Biểu đồ tuần tự tìm kiếm địa danh di tích lịch sử*

****

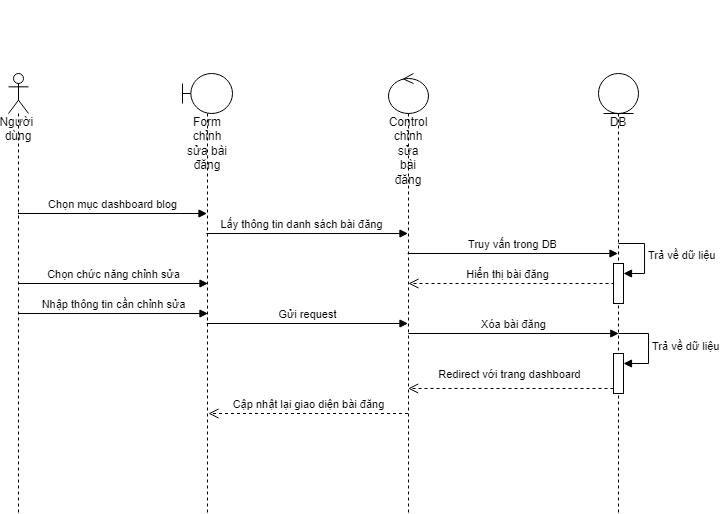
*Hình 2.8. Biểu đồ tuần tự xem thông tin di tích lịch sử*

****

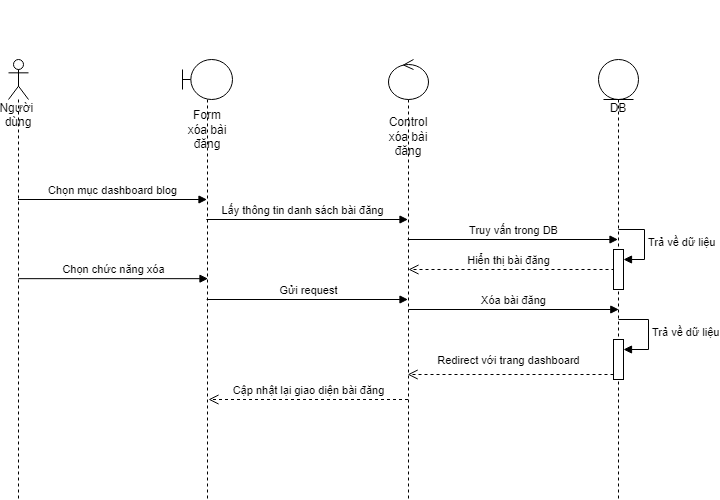
*Hình 2.9. Biểu đồ tuần tự bình luận bài viết về di tích lịch sử*

****

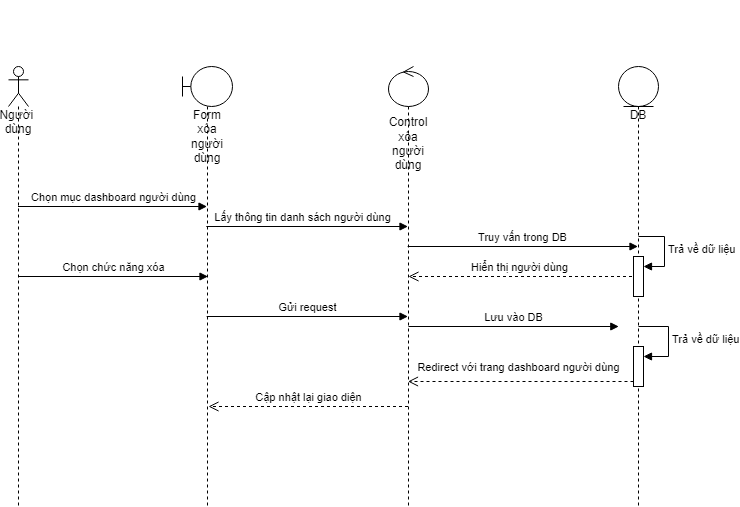
*Hình 2.10. Biểu đồ tuần tự thêm mới nội dung về di tích lịch sử*

****

*Hình 2.11. Biểu đồ tuần tự chỉnh sửa nội dung về di tích lịch sử*

****

*Hình 2.12. Biểu đồ tuần tự xóa nội dung về di tích lịch sử*

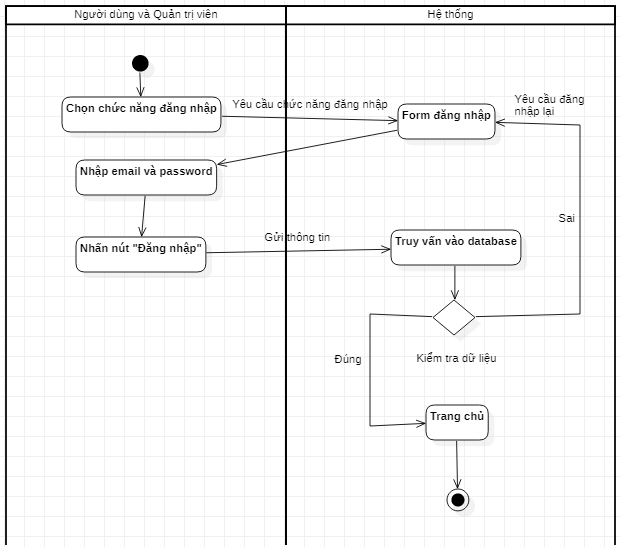
****

*Hình 2.13. Biểu đồ tuần tự xóa tài khoản người dùng*

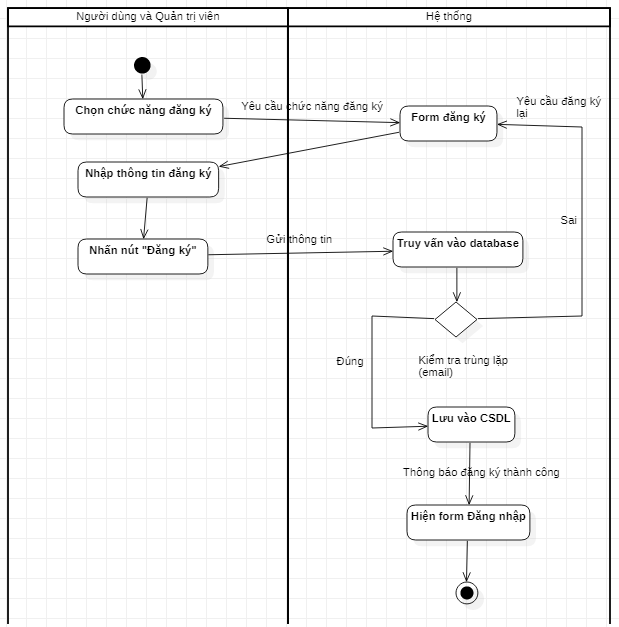
****

*Hình 2.14. Biểu đồ tuần tự xóa thông báo về cảm xúc và bình luận*

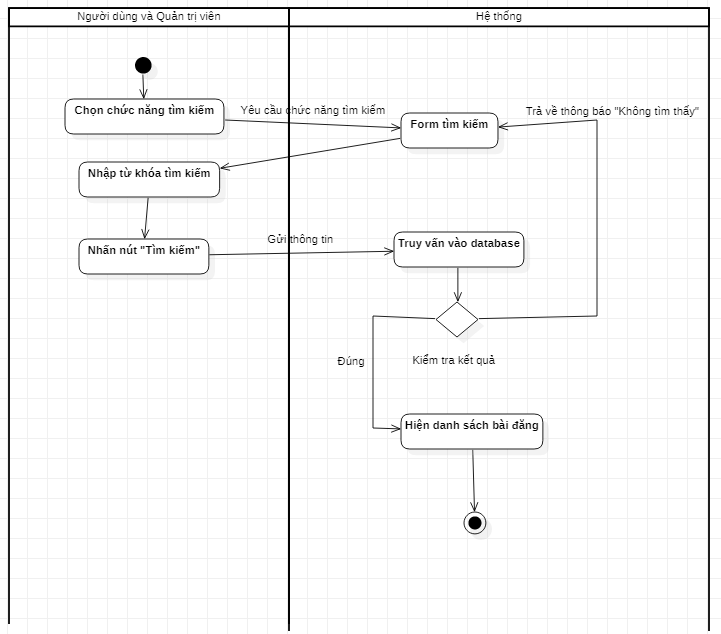
***2.3.4. Biểu đồ hoạt động***

****

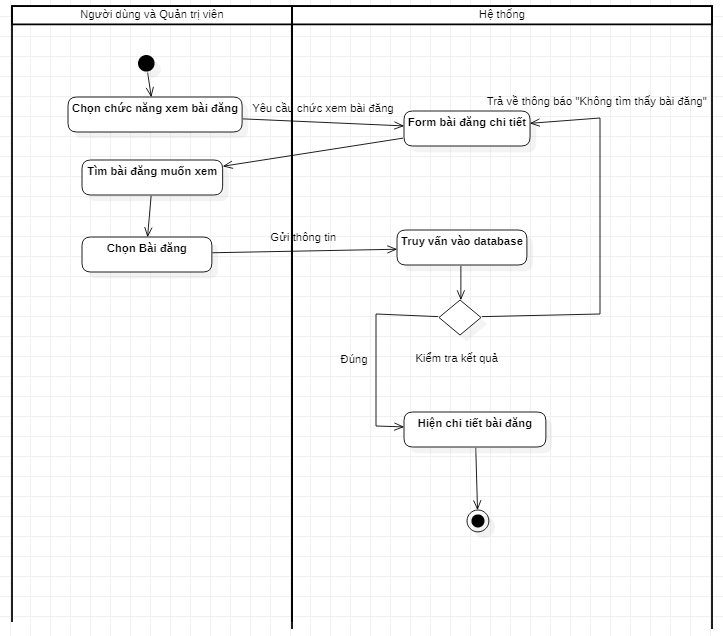
*Hình 2.15. Biểu đồ hoạt động đăng nhập cho người dùng hoặc quản trị viên*

****

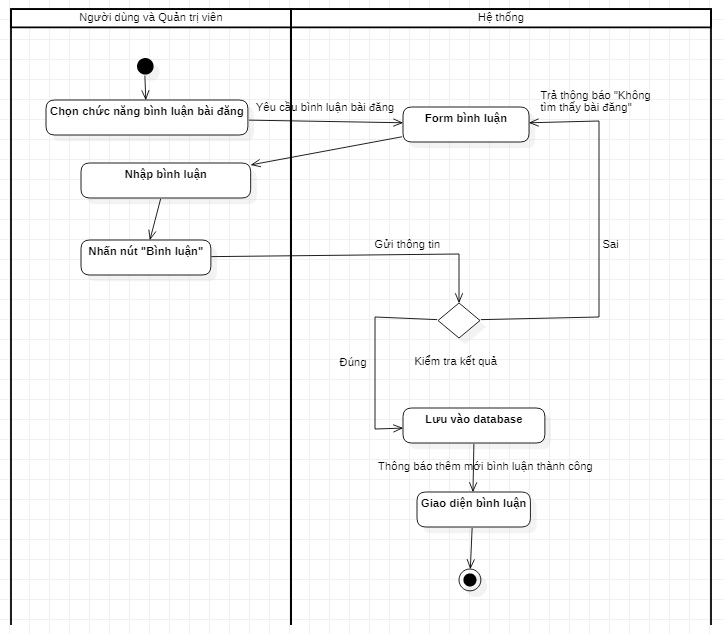
*Hình 2.16. Biểu đồ hoạt động đăng ký cho người dùng*

****

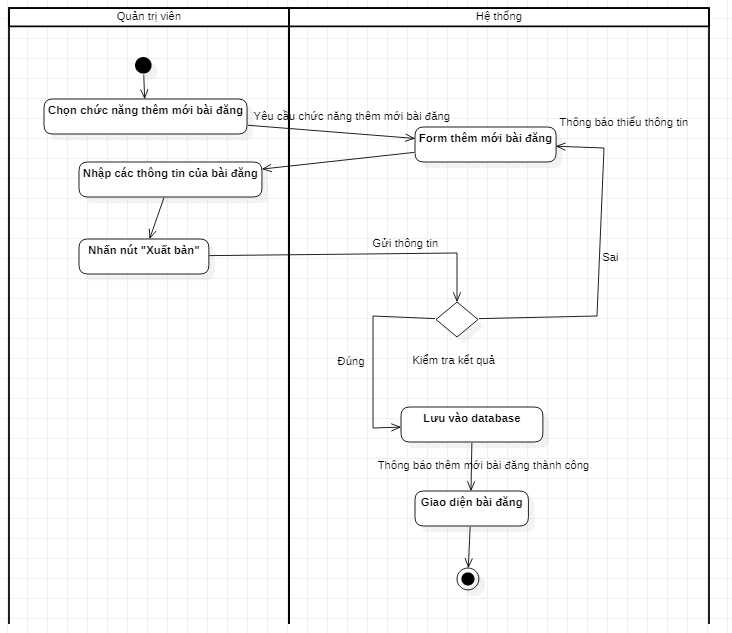
*Hình 2.17. Biểu đồ hoạt động tìm kiếm địa danh di tích lịch sử*

****

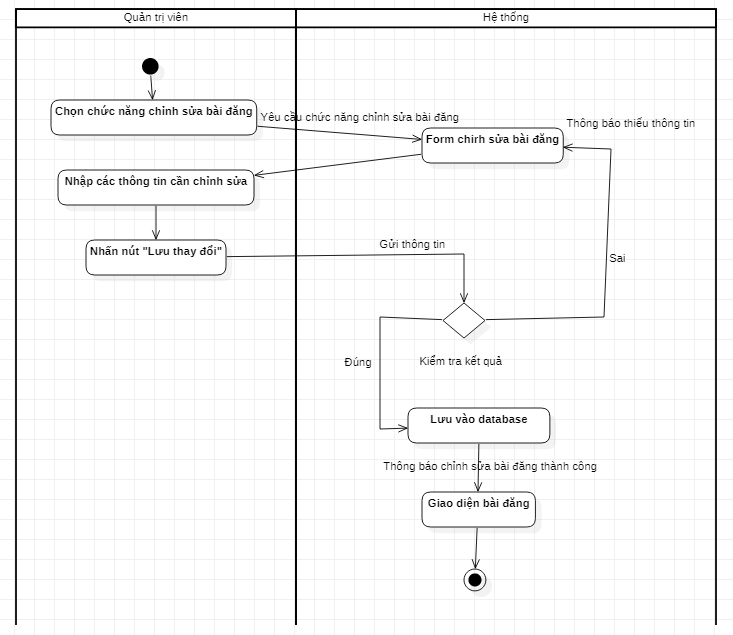
*Hình 2.18. Biểu đồ hoạt động xem thông tin di tích lịch sử*

****

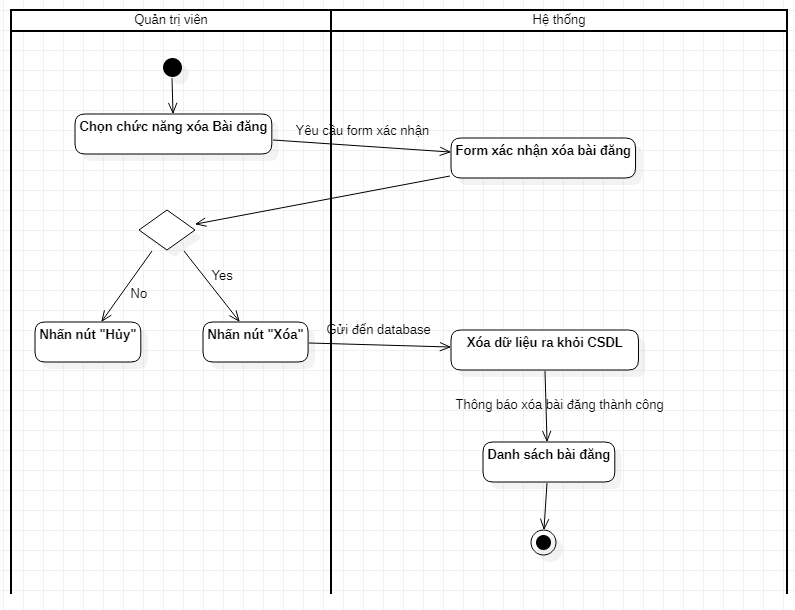
*Hình 2.19. Biểu đồ hoạt động bình luận bài viết về di tích lịch sử*

****

*Hình 2.20. Biểu đồ hoạt động thêm mới nội dung về di tích lịch sử*

****

*Hình 2.21. Biểu đồ hoạt động chỉnh sửa nội dung về di tích lịch sử*

****

*Hình 2.22. Biểu đồ hoạt động xóa nội dung về di tích lịch sử*

****

*Hình 2.23. Biểu đồ hoạt động xóa tài khoản người dùng*

****

*Hình 2.24. Biểu đồ hoạt động xóa thông báo về cảm xúc và bình luận*

**2.4. Cài đặt mô hình Website tích hợp Chatbot AI Coze**

***2.4.1. Công nghệ sử dụng và môi trường phát triển***

Sau khi nắm rõ bài toán và phân tích thiết kế hệ thống, em đã quyết định lựa chọn Web SDK để tích hợp Chatbot Coze này vì SDK giúp cho các nhà phát triển có thể phát triển ứng dụng một cách nhanh chóng và dễ dàng hơn. Hơn nữa, nó còn giảm thời gian và chi phí phát triển: SDK giúp giảm thời gian và chi phí phát triển bằng cách cung cấp các công cụ và tài nguyên cần thiết để phát triển ứng dụng một cách nhanh chóng và hiệu quả.

Môi trường phát triển trong đồ án tốt nghiệp này em đã sử dụng máy tính MSI GF63 Thin 9SCXR, Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz 2.40 GHz, , GPU NVIDIA GeForce GTX 1650, RAM 16GB, ổ sứng SSD 512GB, chạy hệ điều hành Window 11 Pro và sử dụng thêm các môi trường phát triển sau:

* Visual Studio Code
* NodeJS
* Source control (Quản lý mã nguồn) – Github
* Trình duyệt thử nghiệm Google Chrome

***2.4.2. Chuẩn bị dữ liệu***

Dữ liệu về các địa điểm di tích lịch sử của thủ đô Hà Nội được thu nhập từ data trong database MongoDB. Dữ liệu được xuất ra dưới dạng .json và có cấu trúc gồm 2 key: title, content. Số lượng dữ liệu khoảng **20** địa điểm và **80** dòng thông tin về các địa điểm đó

A screenshot of a computer

Description automatically generated

*Hình 2.25. Mẫu dữ liệu về tên di tích lịch sử và mô tả sau khi xuất ra từ database*

***2.4.3. Chuyển đổi dữ liệu sang Excel***

Chuyển đổi dữ liệu từ dạng .json sang dạng .xlsx để tiến hành huấn luyện cho Chatbot Coze

A screenshot of a computer

Description automatically generated

*Hình 2.26. Dữ liệu sau khi chuyển đổi sang excel*

**2.4.4. Huấn luyện chatbot**

Sử dụng bộ dữ liệu đã chuyển đổi để huấn luyện mô hình Coze.

A screenshot of a computer

Description automatically generated

*Hình 2.27. Dữ liệu sau khi tích hợp vào Chatbot*

Thiết lập logic phản hồi cho chatbot

**A screenshot of a white page

Description automatically generated**

*Hình 2.28. Logic phản hồi của chatbot*

**Nhận xét :**

**-** Dữ liệu đầu vào từ MongoDB được xuất ra dưới dạng .json với cấu trúc gồm hai key là "title" và "content". Việc này giúp đảm bảo tính nhất quán và dễ dàng chuyển đổi sang định dạng Excel.

- Việc chuyển đổi dữ liệu từ .json sang .xlsx đã được thực hiện một cách hiệu quả, giúp cho quá trình huấn luyện mô hình trở nên thuận lợi hơn.

- Mô hình Coze đã được huấn luyện thành công với bộ dữ liệu đã chuyển đổi. Mô hình có thể nhận biết và hiểu rõ hơn về các địa điểm di tích lịch sử của Hà Nội.

- Việc tích hợp dữ liệu vào chatbot được thực hiện một cách mượt mà, đảm bảo rằng chatbot có thể truy xuất thông tin một cách nhanh chóng và chính xác.

**CHƯƠNG 3: KẾT QUẢ VÀ ĐÁNH GIÁ KẾT QUẢ**

**3.1. Cài đặt chương trình**

Người dùng clone mã nguồn từ Github, sử dụng lệnh dưới đây trong terminal:

$ git clone https://github.com/TaiPhung02/Hanoi-content.git

***3.1.1 Cài đặt chương trình hệ thống (Ngôn ngữ Javascript - NodeJS)***

Người dùng truy cập vào thư mục server, sử dụng lệnh dưới đây trong terminal:

$ cd server

Người dùng tiến hành import các thư viện cho chương trình, sử dụng lệnh dưới đây trong terminal:

$ npm install

Để khởi chạy chương trình, sử dụng lệnh dưới đây trong terminal:

$ npm start

***3.1.2. Cài đặt giao diện chương trình (Ngôn ngữ Javascript - ReactJS)***

Người dùng truy cập vào thư mục frontend, sử dụng lệnh dưới đây trong terminal:

$ cd frontend

Người dùng tiến hành import các thư viện cho chương trình, sử dụng lệnh dưới đây trong terminal:

$ npm install

Để khởi chạy chương trình, sử dụng lệnh dưới đây trong terminal:

$ npm run dev

***3.1.3. Cài đặt cơ sở dữ liệu (Cơ sở dữ liệu MongoDB)***

Người dùng cần cấu hình lại url trong file .env của thư mục server thành địa chỉ url MongoDB của mình

**3.2. Kết quả chương trình**

A screenshot of a chat

Description automatically generated

*Hình 3.1. Giao diện trang chủ*

**

*Hình 3.2. Giao diện trang chi tiết về địa điểm di tích lịch sử*

*A screenshot of a building

Description automatically generated*

*Hình 3.3. Giao diện trang đăng nhập*

*A screenshot of a building

Description automatically generated*

*Hình 3.4. Giao diện trang đăng ký*

*A person standing in front of a white screen

Description automatically generated*

*Hình 3.5. Giao diện trang soạn thảo*

*A screenshot of a computer

Description automatically generated*

*Hình 3.6. Giao diện tìm kiếm địa điểm di tích lịch sử*

*A screenshot of a chat

Description automatically generated*

*Hình 3.7. Giao diện trang hồ sơ cá nhân*

**A screenshot of a chat

Description automatically generated**

*Hình 3.8. Giao diện trang chỉnh sửa hồ sơ*

*A screenshot of a computer

Description automatically generated*

*Hình 3.9. Giao diện trang quản lý nội dung*

*A screenshot of a computer

Description automatically generated*

*Hình 3.10. Giao diện trang quản lý người dùng*

**A screenshot of a chat

Description automatically generated**

*Hình 3.11. Giao diện trang quản lý thông báo*

*A screenshot of a computer

Description automatically generated*

*Hình 3.12. Giao diện trang đổi mật khẩu*

*A screenshot of a chat

Description automatically generated*

*Hình 3.13. Giao diện hộp thoại với Chatbot*

**3.3. Kiểm thử**

Bảng 3.1. Kiểm thử đăng nhập & đăng ký

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Description** | **Data input** | **Steps** | **Excuted output** | **Actual output** | **Status** |
| 1 | Hiển thị trang đăng nhập |  | Mở trang đăng nhập từ nút đăng nhập | Hiện thị trang đăng nhập | Hiện thị trang đăng nhập | Pass |
| 2 | Hiển thị trang đăng ký |  | Mở trang đăng ký từ nút đăng ký | Hiển thị trang đăng ký | Hiển thị trang đăng ký | Pass |
| 3 | Đăng nhập | Email: [admin@gmail.com](mailto:admin@gmail.com) Password: Admin@123 | Nhập email và password | Đăng nhập thành công và vào trang chủ | Đăng nhập thành công và vào trang chủ | Pass |
| 4 | Đăng ký | Fullname: TaiTester  Email: phungkitai2810@gmail.com  Password: Taitester@123 | Nhập fullname, email và password | Đăng ký thành công và tiến hành đăng nhập | Đăng ký thành công và tiến hành đăng nhập | Pass |

Bảng 3.2. Bảng kiểm thử quản lý nội dung, người dùng, thông báo

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Testcase id** | **Description** | **Data input** | **Steps** | **Excuted output** | **Actual output** | **Status** |
| 1 | Hiển thị danh sách nội dung |  | Mở danh sách nội dung từ dashboard | Hiện thị danh sách nội dung | Hiển thị danh sách nội dung | Pass |
| 2 | Thêm nội dung về địa danh di tích lịch sử của Hà Nội |  | Nhập đầy đủ thông về địa danh | Hiển thị về địa danh vừa thêm | Hiển thị về địa danh vừa thêm | Pass |
| 3 | Chỉnh sửa nội dung di tích lịch sử của Hà Nội |  | Ấn nút chỉnh sửa của nội dung | Cập nhật nội dung vừa chỉnh sửa | Cập nhật nội dung vừa chỉnh sửa | Pass |
| 4 | Xóa nội dung di tích lịch sử của Hà Nội |  | Ấn nút xóa của nội dung | Xóa nội dung khỏi danh sách | Xóa nội dung của danh sách | Pass |
| 5 | Xóa tài khoản của người dùng |  | Ấn nút xóa tài khoản người dùng | Xóa tài khoản của người dùng | Xóa tài khoản của người dùng | Pass |
| 6 | Xóa thông báo tương tác hoặc bình luận |  | Ấn nút xóa thông báo tương tác hoặc bình luận | Xóa nội dung thông báo tưởng tác hoặc bình luận | Xóa nội dung thông báo tưởng tác hoặc bình luận | Pass |

**KẾT LUẬN**

Sau khoảng thời gian 3 tháng từ ngày nhận được đề tài, trải qua quá trình phân tích, nghiên cứu các kiến thức cần thiết và phát triển sản phẩm. Đồ án tốt nghiệp đề tài “Xây dựng website và tích hợp với Chatbot phục vụ truyền thông và giới thiệu địa điểm di tích lịch sử của thủ đô Hà Nội” em đã thu được một số kết quả cũng như nhận thấy một vài điểm hạn chế sau:

1. Kết quả thu được

****Kiến thức về xây dựng website****: Bao gồm các kỹ năng phát triển và triển khai một trang web từ đầu, bao gồm cả việc thiết kế giao diện và lập trình các chức năng cần thiết.

****Tích hợp và vận hành Chatbot****: Em đã nắm được quy trình tích hợp Chatbot vào website và cấu hình để phù hợp với mục đích của đề tài, cũng như quản lý và duy trì hoạt động của Chatbot.

****Phân tích nghiên cứu và lựa chọn công nghệ****: Sau đồ án này em đã có kinh nghiệm trong việc phân tích yêu cầu, tìm hiểu và lựa chọn công nghệ phù hợp để xây dựng sản phẩm, bao gồm cả việc nghiên cứu các công nghệ hỗ trợ như AI để tích hợp vào Chatbot.

Phân tích thiết kế hệ thống: xây dựng bản vẽ biểu hoạt động, thiết kế database, xác định actor và use-case trong hệ thống

Cách triển khai một mô hình AI sau khi huấn luyện.

****Nâng cao kiến thức về truyền thông và giới thiệu****: Đã áp dụng các kiến thức về truyền thông và marketing để hiệu quả hóa việc giới thiệu các địa điểm di tích lịch sử thông qua trang web và Chatbot.

2. Các điểm hạn chế

Em luôn mong muốn tất cả các sản phẩm được sản xuất có chất lượng tốt nhất, mang lại lợi ích và giá trị cho cộng đồng. Tuy nhiên, do hạn chế về điều kiện và thời gian, em chưa thể hoàn thiện sản phẩm một cách chỉnh chu để đạt được mục tiêu ban đầu, dẫn tới một số hạn chế:

****Khả năng tích hợp và tương tác của Chatbot****: Gặp khó khăn trong việc tích hợp và cấu hình Chatbot sao cho phù hợp và hiệu quả với người dùng. Cần cải thiện để tăng tính thân thiện và hiệu quả của Chatbot.

****Độ phức tạp của hệ thống****: Việc xây dựng một hệ thống hoàn chỉnh gặp phải sự phức tạp trong quản lý và bảo trì sau khi triển khai. Điều này cần sự cải tiến trong thiết kế và quản lý hệ thống để đảm bảo tính ổn định và hiệu quả dài hạn.

****Kỹ năng quản lý dự án và thời gian****: Gặp phải khó khăn trong việc quản lý dự án và phân bổ thời gian hiệu quả giữa các giai đoạn phát triển, kiểm thử và triển khai sản phẩm.

****Công nghệ và kiến thức chuyên môn****: Cần cải thiện và nâng cao kiến thức về các công nghệ mới để áp dụng trong việc phát triển sản phẩm và tối ưu hóa hoạt động của hệ thống.

3. Hướng phát triển

Trong thời đại hiện nay, việc sử dụng công nghệ để xây dựng các nền tảng truyền thông như website và tích hợp Chatbot để giới thiệu các địa điểm di tích lịch sử của thủ đô Hà Nội đang ngày càng trở nên phổ biến và quan trọng hơn bao giờ hết. Em mong muốn sản phẩm này có thể mang đến trải nghiệm tuyệt vời nhất cho người dùng, cung cấp thông tin chi tiết và hiệu quả về các địa điểm di tích lịch sử của thủ đô Hà Nội nói riêng, và các địa danh trên cả nước chung.

Em đặt nặng vào việc tối ưu hóa trải nghiệm người dùng thông qua việc cải thiện tính tương tác của Chatbot và mở rộng phạm vi thông tin về các địa điểm di tích lịch sử của Hà Nội. Đồng thời, em mong muốn áp dụng các công nghệ mới để nâng cao hiệu quả của sản phẩm, đem đến cho người dùng những trải nghiệm thú vị và đáp ứng sâu sắc nhu cầu tìm kiếm thông tin về di sản văn hóa lịch sử của đất nước.

**TÀI LIỆU THAM KHẢO**

| [1] | *React*. (n.d.).<https://react.dev/> |
| --- | --- |
| [2] | *Index | Node.js v21.7.3 Documentation*. (n.d.). https://nodejs.org/docs/latest/api/ |
| [3] | *Frameworks built on Express*. (n.d.). https://expressjs.com/en/resources/frameworks.html |
| [4] | *MongoDB documentation*. (n.d.). MongoDB Documentation.https://www.mongodb.com/docs/ |
| [5] | *Cultural heritage: 7 successes of UNESCO’s preservation work*. (2023, July 18). UNESCO. https://www.unesco.org/en/cultural-heritage-7-successes-unescos-preservation-work |
| [6] | *Tourism at World Heritage Sites – Challenges and Opportunities: International tourism seminar, Çeşme (Izmir), Turkey, 26 March 2013*. (2015).<https://doi.org/10.18111/9789284416608> |
| [7] | *Cultural heritage: 7 successes of UNESCO’s preservation work*. (2023b, July 18). UNESCO. https://www.unesco.org/en/cultural-heritage-7-successes-unescos-preservation-work |
| [8] | Phan, D. (2022, December 30). *The best historical sites to visit in Vietnam*. Culture Trip.<https://theculturetrip.com/asia/vietnam/articles/the-10-best-historical-sites-to-visit-in-vietnam> |
| [9] | *OpenAI Platform*. (n.d.). https://platform.openai.com/docs/introduction |