**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO KẾT THÚC HỌC PHẦN**

**LẬP TRÌNH MẠNG**

**Xây dựng chương trình AI Chatbot**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | ThS. Phạm Phú Thiện |

|  |  |
| --- | --- |
| **Học viên thực hiện:** | Nhóm đề tài 7  - Phạm Quốc Huy - 22DH111305  - Phan Tất Thắng - 22DH113428  - Trần Anh Tài - 22DH113195  - Võ Văn Thần Thái - 22DH114737 |
|  |  |

*Tp. Hồ chí minh, Ngày 14 tháng 7 năm 2024*

Mục lục

[1. Mở đầu: 5](#_Toc172831731)

[1.1 Giới thiệu đề tài: Xây dựng AI Chatbot 5](#_Toc172831732)

[1.2 Mục tiêu nghiên cứu: Xác định mục tiêu chính của đồ án. 6](#_Toc172831733)

[1.3 Phạm vi nghiên cứu: Giới hạn và phạm vi của đề tài. 7](#_Toc172831734)

[2. Phân tích yêu cầu 8](#_Toc172831735)

[2.1 Yêu cầu chức năng: Các chức năng chính của hệ thống. 8](#_Toc172831736)

[2.2 Yêu cầu phi chức năng: Hiệu năng, bảo mật, độ tin cậy. 8](#_Toc172831737)

[2.3 Các kịch bản sử dụng (use case): Mô tả các tình huống sử dụng thực tế. 9](#_Toc172831738)

[3. Thiết kế hệ thống 14](#_Toc172831739)

[3.1 Kiến trúc hệ thống: Mô hình tổng thể của hệ thống. 14](#_Toc172831740)

[3.2 Thiết kế chi tiết: 16](#_Toc172831741)

[3.3 Thiết kế giao diện người dùng (nếu có). 18](#_Toc172831742)

[3.4 Thiết kế cơ sở dữ liệu. 27](#_Toc172831743)

[3.5 Thiết kế các module và các luồng dữ liệu. 29](#_Toc172831744)

[3.6 Thiết kế ngoài yêu cầu (nếu có). 32](#_Toc172831745)

[4. Mã nguồn 32](#_Toc172831746)

[4.1 Môi trường phát triển: Công cụ và môi trường phát triển phần mềm. 32](#_Toc172831747)

[4.2 Cấu trúc mã nguồn: Tổ chức mã nguồn. 32](#_Toc172831748)

[4.3 Quá trình cài đặt và triển khai: Hướng dẫn cài đặt và triển khai hệ thống. 33](#_Toc172831749)

[4.4 Các đoạn mã nguồn quan trọng: Chỉ rõ client - server hoạt động như thế nào. 33](#_Toc172831750)

[4.5 Mô tả các điểm nổi bật trong mã nguồn. 46](#_Toc172831751)

[5. Đánh giá và kết luận 50](#_Toc172831752)

[5.1 Kết quả thực hiện (hình ảnh - nếu có). 50](#_Toc172831753)

[5.2 Đánh giá hệ thống: Đánh giá về hiệu năng, bảo mật, độ tin cậy. 50](#_Toc172831754)

[5.3 Kết luận: Tổng kết lại các kết quả đạt được. 51](#_Toc172831755)

[5.4 Hạn chế và hướng phát triển: Những hạn chế của đề tài và các hướng phát triển trong tương lai. 51](#_Toc172831756)

[5.5 Bảng phân công công việc 51](#_Toc172831757)

[6. Tài liệu tham khảo 56](#_Toc172831758)

**Mục lục hình**

[Hình 1: Thiết kế tổng quát của người dùng 16](#_Toc172831667)

[Hình 2: Giao diện đăng ký 17](#_Toc172831668)

[Hình 3: Thiết kế đăng nhập người dùng 18](#_Toc172831669)

[Hình 4: Giao diện đăng nhập 19](#_Toc172831670)

[Hình 5: Báo đăng nhập thành công 19](#_Toc172831671)

[Hình 6: Giao diện chat khi đăng nhập thành công 20](#_Toc172831672)

[Hình 7: Thông báo lưu lịch sử chat thành công 20](#_Toc172831673)

[Hình 8: Các chức năng của chatbot 21](#_Toc172831674)

[Hình 9: Kết quả trả về khi xem thời tiết 22](#_Toc172831675)

[Hình 10: Kết quả trả về khi xem ip 23](#_Toc172831676)

[Hình 11: Kết quả trả về khi xem tarot 24](#_Toc172831677)

[Hình 12: Kết quả xem port 25](#_Toc172831678)

[Hình 13: Kết quả khi kiểm tra url 26](#_Toc172831679)

[Hình 14: Kết quả trả về khi muốn tìm video YouTube 26](#_Toc172831680)

[Hình 15: Kết quả khi tìm phim 27](#_Toc172831681)

[Hình 16: Lịch sử chat trong database 27](#_Toc172831682)

[Hình 17: Bảng người dùng 27](#_Toc172831683)

[Hình 18: Cấu trúc bảng người dùng 28](#_Toc172831684)

[Hình 19: Cấu trúc bảng lịch sử trò chuyện 28](#_Toc172831685)

**Lời nói đầu**

Kính thưa thầy và các bạn!

Chào mừng các bạn đến với thế giới của ChatbotAI nơi công nghệ trí tuệ nhân tạo đang làm thay đổi cách chúng ta giao tiếp và tương tác hàng ngày.

Thế giới càng hiện đại thì đi kèm theo đó là những phát triển vượt bậc của những công nghệ tiên tiến, những lĩnh vực như mạng, di động,…đều đem lại những cải tiến vô cùng to lớn đối với người dùng ở xã hội 4.0 ngày nay.Chúng không những giúp ích cho chúng ta nhanh hơn trong công việc mà còn hỗ trợ đem lại những lợi ích tốt nhất cho người dùng trong đời sống hàng ngày.

Không chỉ đơn giản là một công cụ trả lời câu hỏi, Chatbot AI có khả năng học hỏi và cải thiện qua từng tương tác. Với khả năng xử lý ngôn ngữ tự nhiên tiên tiến, Chatbot AI có thể hiểu ngữ cảnh, nhận biết cảm xúc và cung cấp phản hồi phù hợp, từ đó tạo nên những cuộc trò chuyện thân thiện và gần gũi.

Chatbot AI có thể được tích hợp trong nhiều lĩnh vực khác nhau, từ dịch vụ khách hàng, hỗ trợ kỹ thuật, đến chăm sóc sức khỏe và giáo dục. Nó giúp tiết kiệm thời gian, nâng cao hiệu suất làm việc và mang lại sự hài lòng tối đa cho người dùng.

Hãy tưởng tượng một tương lai nơi bạn có thể trò chuyện với máy móc một cách tự nhiên, giải quyết mọi vấn đề một cách nhanh chóng và tiện lợi. Đó chính là sứ mệnh của Chatbot AI – biến trí tưởng tượng thành hiện thực, mang lại những giá trị thiết thực cho cuộc sống.

**Lời cảm ơn**

Chúng em xin gửi lời cảm ơn chân thành đến Thầy Phạm Phú Thiện đã tận tình hướng dẫn, giúp đỡ và tạo điều kiện thuận lợi để chúng em hoàn thành đề tài này. Những kiến thức và kinh nghiệm mà Thầy chia sẻ không chỉ giúp chúng em hiểu sâu hơn về các phần mềm, ngôn ngữ java và những kiến thức chuyên sâu cũng như sự tận tâm của thầy khi muốn giải đáp hết những thắc mắc mà chúng em còn không biết mình sẽ gặp phải trong tương lai.

Chúng em cũng xin cảm ơn các thầy cô trong khoa Công nghệ thông tin đã cung cấp những kiến thức nền tảng và sự hỗ trợ quý báu trong suốt quá trình học tập và nghiên cứu tại trường.

Cuối cùng, chúng em xin gửi lời cảm ơn đến gia đình và bạn bè đã luôn đồng hành, động viên và ủng hộ chúng em trong quá trình thực hiện đề tài này.

**NHẬN XÉT, ĐÁNH GIÁ**

**(Của Người hướng dẫn)**

…………………………………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

**Điểm:** ……………………………………….…………(bằng chữ: …………...)

**Đồng ý/Không đồng ý** cho sinh viên bảo vệ trước hội đồng chấm đồ án?

………ngày tháng năm 20

**GIÁO VIÊN HƯỚNG DẪN**

(ký, họ tên)

# Mở đầu:

## Giới thiệu đề tài: Xây dựng AI Chatbot

**Lý do chọn đề tài:**

* **Xu thế phát triển của AI và Chatbot:**
  + Trí tuệ nhân tạo (AI) đang trở thành một trong những công nghệ tiên tiến nhất hiện nay, ứng dụng rộng rãi trong nhiều lĩnh vực như y tế, tài chính, giáo dục, và dịch vụ khách hàng. Trong số đó, chatbot là một ứng dụng phổ biến và hữu ích của AI, được sử dụng để tương tác tự động với người dùng thông qua văn bản hoặc giọng nói.
* **Nhu cầu thực tiễn của doanh nghiệp và người dùng:**
  + Nhiều doanh nghiệp hiện nay đang tìm kiếm các giải pháp chatbot để cải thiện dịch vụ khách hàng, giảm chi phí và tăng hiệu quả hoạt động. Chatbot có thể hoạt động 24/7, cung cấp thông tin tức thì và hỗ trợ giải đáp thắc mắc cho khách hàng một cách nhanh chóng và hiệu quả.
* **Ưu điểm của việc sử dụng Java:**
  + Java là một ngôn ngữ lập trình mạnh mẽ, đa nền tảng và có hiệu suất cao, thích hợp cho việc xây dựng các ứng dụng phức tạp như AI chatbot. Ngoài ra, cộng đồng Java rất lớn mạnh và có nhiều thư viện, công cụ hỗ trợ cho việc phát triển chatbot.

**Tầm quan trọng của đề tài trong thực tế:**

* **Nâng cao chất lượng dịch vụ khách hàng:**
  + Chatbot có thể cung cấp hỗ trợ kịp thời và liên tục cho khách hàng, từ đó nâng cao trải nghiệm người dùng và tăng sự hài lòng của khách hàng. Điều này đặc biệt quan trọng trong các lĩnh vực như thương mại điện tử, dịch vụ tài chính, và hỗ trợ kỹ thuật.
* **Tiết kiệm chi phí và tài nguyên:**
  + Sử dụng chatbot giúp doanh nghiệp giảm thiểu chi phí thuê nhân viên trực tổng đài và giải đáp thắc mắc của khách hàng. Chatbot có thể xử lý nhiều yêu cầu cùng lúc mà không cần nghỉ ngơi, từ đó tiết kiệm thời gian và tài nguyên nhân lực.
* **Tăng cường khả năng tương tác và giao tiếp:**
  + Chatbot không chỉ giúp doanh nghiệp giao tiếp hiệu quả với khách hàng mà còn thu thập và phân tích dữ liệu từ các cuộc trò chuyện để hiểu rõ hơn về nhu cầu và mong muốn của khách hàng, từ đó đưa ra các chiến lược kinh doanh phù hợp.
* **Hỗ trợ trong giáo dục và y tế:**
  + Chatbot có thể được sử dụng trong giáo dục để cung cấp hỗ trợ học tập, giải đáp câu hỏi của học sinh và sinh viên. Trong y tế, chatbot có thể giúp bệnh nhân đặt lịch hẹn, nhắc nhở uống thuốc và cung cấp thông tin y tế cơ bản.
* **Đóng góp vào sự phát triển công nghệ:**
  + Nghiên cứu và phát triển AI chatbot không chỉ giải quyết các vấn đề hiện tại mà còn mở ra những hướng đi mới trong lĩnh vực trí tuệ nhân tạo và tương tác người-máy, thúc đẩy sự tiến bộ của công nghệ.

Tóm lại, việc xây dựng AI chatbot sử dụng Java không chỉ mang lại lợi ích thiết thực cho doanh nghiệp và người dùng mà còn góp phần vào sự phát triển của công nghệ AI và ứng dụng của nó trong đời sống hàng ngày. Đây là lý do và tầm quan trọng của việc chọn đề tài nghiên cứu này.

## Mục tiêu nghiên cứu: Xác định mục tiêu chính của đồ án.

* **Khả năng hiểu ngôn ngữ tự nhiên (Natural Language Processing - NLP):** Chatbot cần có khả năng phân tích và hiểu được các câu hỏi và yêu cầu từ người dùng bằng tiếng Việt một cách chính xác.
* **Khả năng trả lời tự động:** Chatbot phải có khả năng cung cấp các câu trả lời và phản hồi phù hợp dựa trên ngữ cảnh và nội dung của câu hỏi từ người dùng.
* **Giao diện người dùng (UI/UX):** Thiết kế một giao diện người dùng trực quan, dễ sử dụng, giúp người dùng dễ dàng tương tác với chatbot.
* **Khả năng tích hợp:** Đảm bảo chatbot có thể tích hợp vào các nền tảng phổ biến như website, ứng dụng di động, và các mạng xã hội.
* **Hiệu suất hệ thống:** Đảm bảo chatbot hoạt động ổn định, có thể xử lý nhiều yêu cầu đồng thời mà không gặp sự cố.
* **Bảo mật thông tin:** Bảo vệ dữ liệu người dùng và thông tin trao đổi, đảm bảo tính riêng tư và bảo mật.
* **Tính ứng dụng:** Đảm bảo chatbot có thể được sử dụng trong các lĩnh vực khác nhau như dịch vụ khách hàng, giáo dục, y tế, và thương mại điện tử.
* **Hướng dẫn triển khai:** Cung cấp hướng dẫn chi tiết về cách cài đặt và triển khai chatbot trên các hệ thống thực tế.
* **Đánh giá hiệu suất:** Đo lường và đánh giá hiệu suất của chatbot thông qua các chỉ số như độ chính xác của câu trả lời, tốc độ phản hồi, và mức độ hài lòng của người dùng.
* **Phản hồi và cải tiến:** Thu thập phản hồi từ người dùng và liên tục cải tiến hệ thống dựa trên phản hồi đó.

## Phạm vi nghiên cứu: Giới hạn và phạm vi của đề tài.

* **Ngôn ngữ:** Chatbot sẽ được phát triển chủ yếu để tương tác bằng tiếng Việt. Các ngôn ngữ khác có thể được xem xét trong các nghiên cứu và phát triển tiếp theo.
* **Nền tảng:** Chatbot sẽ được triển khai và thử nghiệm trên các nền tảng web và ứng dụng di động. Tích hợp trên các nền tảng khác như mạng xã hội (Facebook Messenger, Zalo, v.v.) có thể được xem xét trong tương lai.
* **Khả năng học hỏi:** Chatbot sẽ sử dụng các kỹ thuật học máy (Machine Learning) cơ bản, tập trung vào học có giám sát và không giám sát. Các kỹ thuật học sâu (Deep Learning) nâng cao sẽ không phải là trọng tâm chính trong phạm vi nghiên cứu này.
* **Tích hợp dữ liệu:** Chatbot sẽ chủ yếu sử dụng các dữ liệu tĩnh và cập nhật định kỳ. Việc tích hợp dữ liệu động theo thời gian thực sẽ được xem xét trong các giai đoạn phát triển sau.
* **Bảo mật:** Hệ thống sẽ bao gồm các biện pháp bảo mật cơ bản để bảo vệ thông tin người dùng và dữ liệu trao đổi. Các biện pháp bảo mật nâng cao sẽ không phải là trọng tâm chính của đề tài này.

# Phân tích yêu cầu

## Yêu cầu chức năng: Các chức năng chính của hệ thống.

* Về phía client:
  + Tra cứu thời tiết: gửi yêu cầu là tên một tỉnh/thành phố hoặc địa danh (bằng tiếng Việt). Kết quả phản hồi là thời tiết ngày hiện tại và một số ngày/tuần (do SV quyết định) kế tiếp của địa điểm đó.
  + Xác định vị trí IP: gửi yêu cầu là 1 địa chỉ IP bất kỳ. Kết quả phản hồi là tọa độ địa điểm tương ứng với địa chỉ IP đó. Tham khảo kết quả từ <https://www.iplocation.net/de> biết các thông tin cần trả về.
  + Quét port: gửi yêu cầu là một địa chỉ IP. Kết quả phản hồi là các port đang mở trong giới hạn từ port x đến port y (với x, y là dữ liệu người dùng nhập).
* Về phía server:
  + Nhân các yêu cầu từ client. sử dụng Java socket kết hợp API hoặc (các tools/ngôn ngữ khác) để lấy kết quả trả về cho client. Tất cả xử lý phải nằm ở phía server.

## Yêu cầu phi chức năng: Hiệu năng, bảo mật, độ tin cậy.

1. **Hiệu suất (Performance):**

* **Khả năng đáp ứng (Responsiveness):** Hệ thống phải phản hồi nhanh chóng và kịp thời các yêu cầu của người dùng.
* **Thông lượng (Throughput):** Hệ thống phải có khả năng xử lý nhiều yêu cầu cùng lúc mà không bị quá tải.
* **Mức tiêu thụ tài nguyên (Resource utilization):** Hệ thống phải sử dụng hiệu quả các tài nguyên hệ thống như CPU, bộ nhớ, mạng,...
* **Khả năng mở rộng (Scalability):** Hệ thống phải có khả năng mở rộng để đáp ứng nhu cầu tăng trưởng của người dùng và dữ liệu.

1. **Bảo mật (Security):**

* **Bảo mật dữ liệu (Data security):** Dữ liệu của người dùng phải được bảo vệ khỏi truy cập trái phép, sử dụng, thay đổi hoặc xóa.
* **Bảo mật hệ thống (System security):** Hệ thống phải được bảo vệ khỏi các cuộc tấn công mạng và phần mềm độc hại.
* **Xác thực và ủy quyền (Authentication and authorization):** Người dùng chỉ có thể truy cập vào các tài nguyên mà họ được phép truy cập.
* **Bảo mật giao tiếp (Communication security):** Giao tiếp giữa hệ thống và người dùng phải được mã hóa để bảo vệ tính bí mật và toàn vẹn của dữ liệu.

1. **Độ tin cậy (Reliability):**

* **Khả năng sẵn sàng (Availability):** Hệ thống phải luôn sẵn sàng hoạt động và đáp ứng các yêu cầu của người dùng.
* **Độ tin cậy (Reliability):** Hệ thống phải hoạt động chính xác và không bị lỗi hay sự cố.
* **Khả năng phục hồi (Recoverability):** Hệ thống phải có khả năng phục hồi nhanh chóng từ các lỗi hoặc sự cố.
* **Bảo trì (Maintainability):** Hệ thống phải dễ dàng bảo trì và sửa chữa.

## Các kịch bản sử dụng (use case): Mô tả các tình huống sử dụng thực tế.

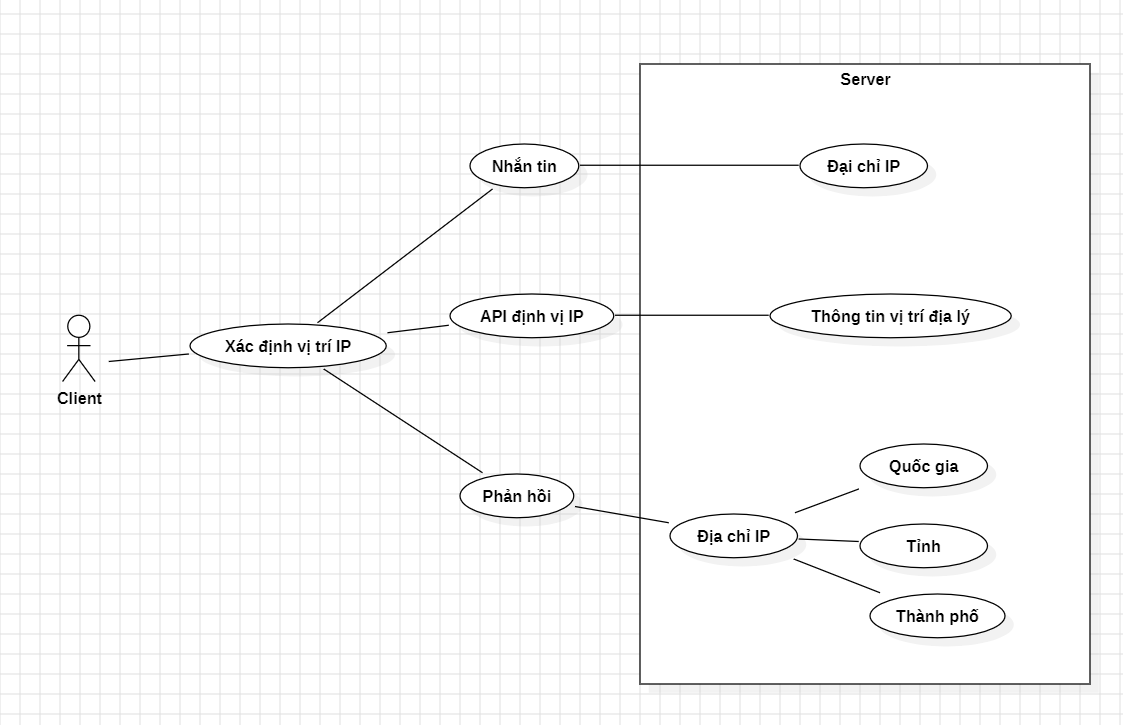
1. **Tra cứu thời tiết:**

* **Tình huống:** Người dùng muốn biết thời tiết hiện tại và dự báo cho một số ngày tới ở một địa điểm cụ thể.
* **Hành động:**
  + Người dùng nhắn tin cho chatbot bằng tiếng Việt, bao gồm tên tỉnh/thành phố hoặc địa danh họ muốn tra cứu.
  + Chatbot sử dụng API dự báo thời tiết OpenWeatherMap để lấy thông tin thời tiết hiện tại và dự báo cho một số ngày/tuần tiếp theo.
  + Chatbot phản hồi cho người dùng bằng tiếng Việt, bao gồm:
    - Nhiệt độ hiện tại, cảm giác, điều kiện thời tiết (ví dụ: nắng, mưa, râm mây).
    - Dự báo thời tiết cho một số ngày/tuần tiếp theo, bao gồm:
      * Ngày: Thứ Hai, Thứ Ba, ...
      * Nhiệt độ cao nhất, thấp nhất.
      * Điều kiện thời tiết chủ đạo (ví dụ: nắng, mưa rào, dông).
* **Ví dụ:**
  + **Người dùng:** “/thoitiet quận 10"
  + **Chatbot:** " 2024-07-20 06:00:00: light rain, Nhiệt độ: 26.28°C"



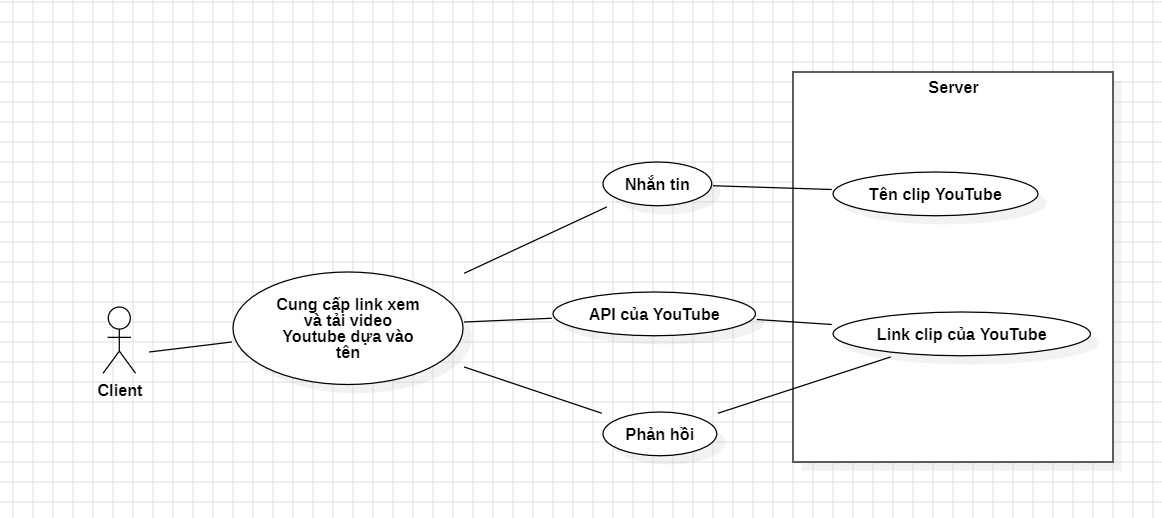
1. **Xác định vị trí IP:**

* **Tình huống:** Người dùng muốn biết vị trí địa lý tương ứng với một địa chỉ IP cụ thể.
* **Hành động:**
  + Người dùng nhắn tin cho chatbot bằng tiếng Việt, bao gồm địa chỉ IP họ muốn tra cứu.
  + Chatbot sử dụng API định vị IP (ví dụ: FreeGeoIP) để lấy thông tin vị trí địa lý tương ứng với địa chỉ IP đó.
  + Chatbot phản hồi cho người dùng bằng tiếng Việt, bao gồm:
    - Quốc gia, tỉnh/thành phố, vĩ độ, kinh độ của địa điểm tương ứng với địa chỉ IP.
* **Ví dụ:**
  + **Người dùng:** "/ip 8.8.8.8?"
  + Bot: Thông tin IP cho 8.8.8.8:
  + Bot: - Trạng thái: success
  + Bot: - Châu lục: North America (NA)
  + Bot: - Quốc gia: United States (US)
  + Bot: - Vùng: VA (Virginia)
  + Bot: - Thành phố: Ashburn
  + Bot: - Mã bưu điện: 20149
  + Bot: - Vĩ độ: 39.03
  + Bot: - Kinh độ: -77.5
  + Bot: - Múi giờ: America/New\_York
  + Bot: - Chênh lệch múi giờ: -14400
  + Bot: - Tiền tệ: USD
  + Bot: - ISP: Google LLC
  + Bot: - Tổ chức: Google Public DNS
  + Bot: - AS: AS15169 Google LLC (GOOGLE)
  + Bot: - Tên miền ngược: dns.google
  + Bot: - Mobile: false
  + Bot: - Proxy: false
  + Bot: - Hosting: true
  + Bot: - Google Maps URL: <https://www.google.com/maps?q=39.03,-77.5>
  + **Tình huống:** Người dùng muốn biết vị trí địa lý tương ứng với một địa chỉ IP cụ thể.



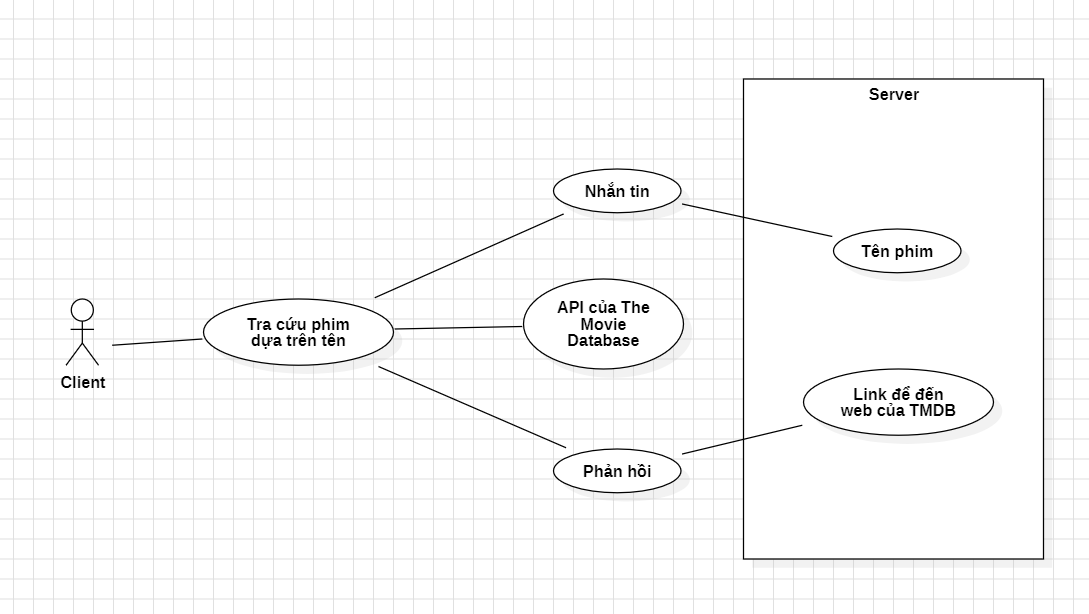
1. **Cung cấp link xem và tải video YouTube dựa vào tên:**

* **Tình huống:** Người dùng muốn tìm một video YouTube cụ thể và đã biết tên clip.
* **Hành động:**
  + Người dùng nhắn tin cho chatbot bằng tiếng Việt, bao gồm tên clip YouTube họ muốn tra cứu.
  + Chatbot sử dụng API của YouTube để lấy link của clip YouTube ở đầu phần suggestion khi nhập tên clip vào thanh search của YouTube
  + Chatbot phản hồi cho người dùng bằng tiếng Việt, bao gồm:
    - Link của clip YouTube đó.
* **Ví dụ:**
  + **Người dùng:** /video đừng làm trái tim anh đau
  + **Bot: Đây là link để nghe bài hát "đừng làm trái tim anh đau": https://www.youtube.com/watch?v=abPmZCZZrFA**
  + **Bot: Đây là link tải video: https://www.ss youtube.com/watch?v=abPmZCZZrFA**



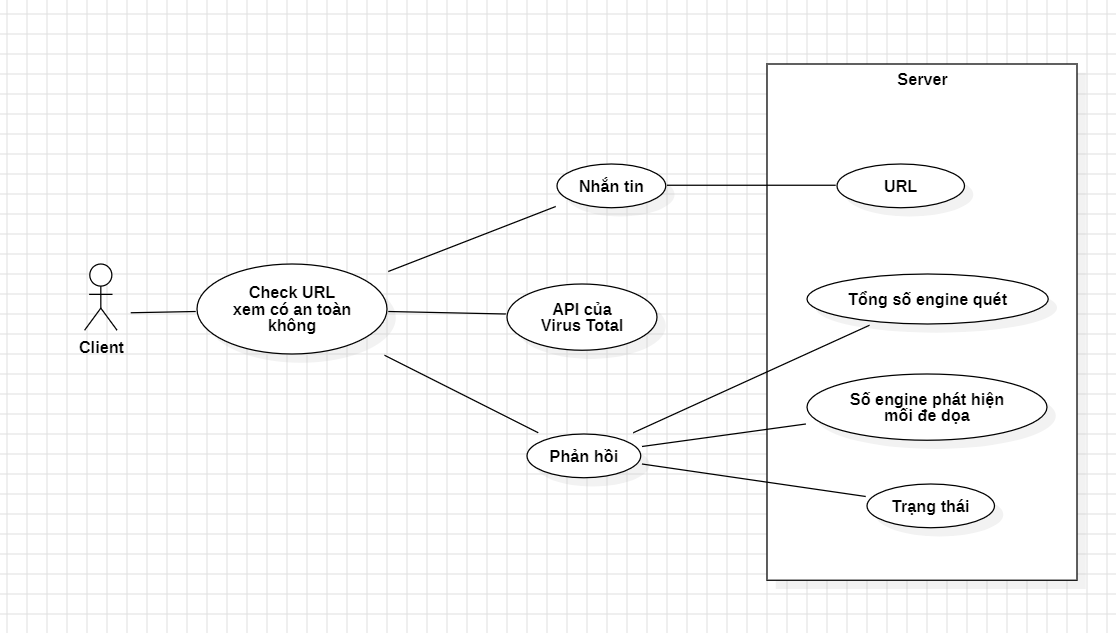
1. **Tra cứu phim dựa trên tên**

* **Tình huống:** Người dùng muốn tìm thông tin về 1 bộ phim đã biết tên.
* **Hành động:**
  + Người dùng nhắn tin cho chatbot bằng tiếng Việt, bao gồm tên phim họ muốn tra cứu.
  + Chatbot sử dụng API của The Movie Database (TMDB)
  + Chatbot phản hồi cho người dùng bằng tiếng Việt, bao gồm:
    - Link để đến web của TMDB để thấy thông tin của phim cần tìm.
* **Ví dụ:**
  + **Người dùng:** /phim Despicable me 4
  + **Bot: Đây là link cho phim "despicable me":** [**https://www.themoviedb.org/movie/519182**](https://www.themoviedb.org/movie/519182)



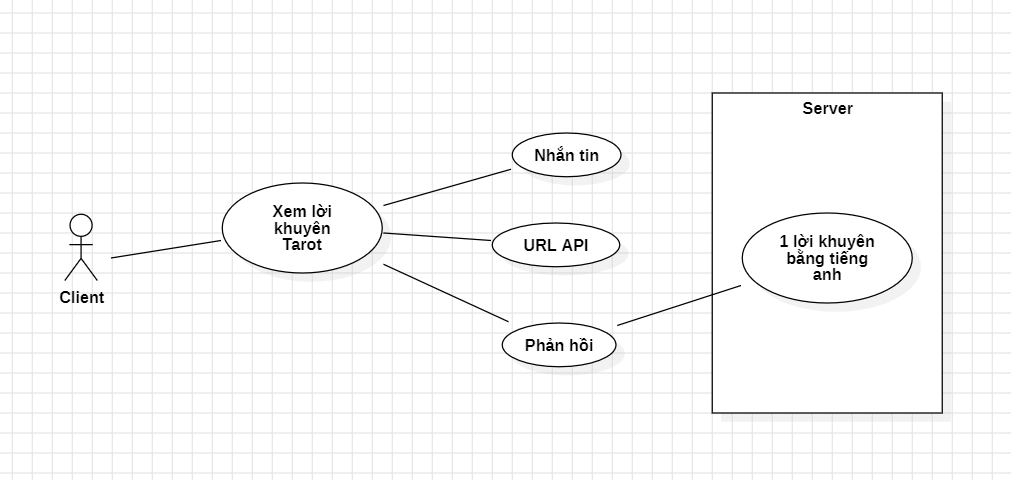
1. Check URL xem có an toàn không

* **Tình huống:** Người dùng muốn kiểm tra xem 1 url có an toàn không.
* **Hành động:**
  + Người dùng nhắn tin cho chatbot bằng tiếng Việt, bao gồm url mà họ muốn kiểm tra
  + Chatbot sử dụng API của Virus Total
  + Chatbot phản hồi cho người dùng bằng tiếng Việt, bao gồm:
    - Tổng số engine quét
    - Số engine phát hiện mối đe dọa
    - Trạng thái
* **Ví dụ:**
  + **Người dùng:** /check https://www.YouTube.com/watch?v=j9bX4Nvtk9M&list=RDdiOo4AZ7KFU&index=5
  + **Bot: Kết quả quét cho URL https://www.YouTube.com/watch?v=j9bX4Nvtk9M&list=RDdiOo4AZ7KFU&index=5:**
  + **Bot: - Ngày quét: 2024-07-23 10:07:10**
  + **Bot: - Tổng số engines quét: 94**
  + **Bot: - Số engines phát hiện mối đe dọa: 0**
  + **Bot: - Trạng thái: An toàn. Không có engine nào phát hiện mối đe dọa.**



1. Xem lời khuyên Tarot

* **Tình huống:** Người dùng muốn xem lời khuyên tarot
* **Hành động:**
  + Người dùng nhắn tin cho chatbot /tarot
  + Chatbot sử dụng URL API free https://api.adviceslip.com/advice
  + Chatbot phản hồi cho người dùng bằng tiếng Việt, bao gồm:
    - 1 lời khuyên bằng tiếng anh.
* **Ví dụ:**
  + **Người dùng:** /tarot
  + **Bot: If you have the chance, take it!**



1. **Quét port:**

* **Tình huống:** Người dùng muốn biết các port đang mở trên một máy tính cụ thể được xác định bằng địa chỉ IP.
* **Hành động:**
  + **Cảnh báo:** Chức năng quét port có thể tiềm ẩn nguy cơ bảo mật, do đó cần cân nhắc kỹ lưỡng trước khi triển khai và sử dụng.
  + Người dùng nhắn tin cho chatbot bằng tiếng Việt, bao gồm địa chỉ IP họ muốn quét port và giới hạn port (từ port x đến port y).
  + **Lưu ý:** Do tính chất nhạy cảm của chức năng này, việc triển khai thực tế có thể cần được điều chỉnh và kiểm soát chặt chẽ để đảm bảo an toàn và tuân thủ các quy định về bảo mật.
* **Ví dụ:**
  + Người dùng: /port 192.168.1.15 262 284 ( /port <địa chỉ IP> <cổng bắt đầu> <cổng kết thúc> )
  + Bot: Quét cổng cho IP 192.168.1.15 từ cổng 262 đến 284:
  + Bot: Cổng 262 đóng
  + Bot: Cổng 263 đóng
  + Bot: Cổng 264 đóng
  + Bot: Cổng 265 đóng
  + Bot: Cổng 266 đóng
  + Bot: Cổng 267 đóng
  + Bot: Cổng 268 đóng
  + Bot: Cổng 269 đóng
  + Bot: Cổng 270 đóng
  + Bot: Cổng 271 đóng
  + Bot: Cổng 272 đóng
  + Bot: Cổng 273 đóng
  + Bot: Cổng 274 đóng
  + Bot: Cổng 275 đóng
  + Bot: Cổng 276 đóng
  + Bot: Cổng 277 đóng
  + Bot: Cổng 278 đóng
  + Bot: Cổng 279 đóng
  + Bot: Cổng 280 đóng
  + Bot: Cổng 281 đóng
  + Bot: Cổng 282 đóng
  + Bot: Cổng 283 đóng
  + Bot: Cổng 284 đóng



# Thiết kế hệ thống

## Kiến trúc hệ thống: Mô hình tổng thể của hệ thống.

1. **Giao diện người dùng (User Interface - UI):**

* Là phần tương tác trực tiếp với người dùng, cho phép người dùng nhập yêu cầu và nhận phản hồi từ chatbot.
* Có thể được xây dựng bằng giao diện web, ứng dụng di động hoặc giao diện giọng nói.
* Cần đảm bảo dễ sử dụng, trực quan và phù hợp với thói quen của người dùng.

1. **Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP):**

* Phân tích và hiểu các yêu cầu của người dùng bằng tiếng Việt.
* Sử dụng các kỹ thuật như phân loại ý định, trích xuất thực thể, v.v. để xác định thông tin quan trọng trong yêu cầu.
* Chuyển đổi yêu cầu của người dùng thành một dạng mà hệ thống có thể xử lý.

1. **Logic xử lý:**

* Xử lý các yêu cầu của người dùng dựa trên thông tin thu thập từ NLP.
* Truy cập và sử dụng các nguồn dữ liệu thời tiết, vị trí IP và quét port.
* Áp dụng các thuật toán và quy tắc logic để đưa ra kết quả phù hợp.

1. **Cổng kết nối dữ liệu (Data Access Layer):**

* Tương tác với nguồn dữ liệu thời tiết OpenWeatherMap
* Truy cập API định vị IP (ví dụ: FreeGeoIP).
* Thực hiện quét port trên các máy chủ cụ thể.
* Virustotal
* YouTube
* TMDB

1. **Hệ thống quản lý cơ sở dữ liệu (Database Management System - DBMS):**

* MySQL
* Lưu trữ dữ liệu cấu hình hệ thống, thông tin người dùng và lịch sử tương tác.
* Đảm bảo tính toàn vẹn, bảo mật và truy cập dữ liệu hiệu quả.

1. **Động cơ phản hồi (Response Engine):**

* Tạo ra các phản hồi phù hợp cho người dùng dựa trên kết quả xử lý.
* Sử dụng ngôn ngữ tự nhiên để tạo câu văn trôi chảy, dễ hiểu và phù hợp với ngữ cảnh.
* Có thể tích hợp các kỹ thuật tổng hợp văn bản, dịch thuật để hỗ trợ đa ngôn ngữ.

1. **Lớp quản lý hội thoại (Conversation Management Layer):**

* Quản lý trạng thái hội thoại giữa người dùng và chatbot.
* Theo dõi bối cảnh cuộc trò chuyện để đưa ra phản hồi phù hợp.
* Lưu trữ lịch sử hội thoại để hỗ trợ học hỏi và cải thiện hiệu suất chatbot.

1. **Hệ thống quản lý kiến thức (Knowledge Management System - KMS):**

* Lưu trữ kiến thức về thời tiết, vị trí địa lý và các lĩnh vực liên quan.
* Cập nhật kiến thức mới một cách thường xuyên.
* Hỗ trợ chatbot truy cập và sử dụng kiến thức để trả lời các câu hỏi phức tạp hơn.

1. **Hệ thống phân tích (Analytics System):**

* Thu thập dữ liệu về tương tác giữa người dùng và chatbot.
* Phân tích dữ liệu để đánh giá hiệu suất chatbot và xác định các điểm cần cải thiện.
* Hỗ trợ tối ưu hóa chatbot và nâng cao trải nghiệm người dùng.

1. **Hệ thống triển khai (Deployment System):**

* Triển khai chatbot lên môi trường thực tế để người dùng có thể truy cập và sử dụng.
* Có thể triển khai trên web, ứng dụng di động hoặc các nền tảng đám mây.
* Đảm bảo tính sẵn sàng, khả năng mở rộng và bảo mật của hệ thống.

## Thiết kế chi tiết:

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phông chữ

Mô tả được tạo tự động

Hình 1: Thiết kế tổng quát của người dùng

Ảnh có chứa văn bản, ảnh chụp màn hình, hàng, biểu đồ

Mô tả được tạo tự động

Hình 2: Giao diện đăng ký

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

Hình 3: Thiết kế đăng nhập người dùng

## Thiết kế giao diện người dùng (nếu có).

#### Giao diện đăng nhập:

* Người dùng đăng nhập bằng tài khoản đã đăng ký trước đó.

A screenshot of a login box

Description automatically generated

Hình 4: Giao diện đăng nhập

A screenshot of a computer

Description automatically generated

Hình 5: Báo đăng nhập thành công

A screenshot of a computer

Description automatically generated

Hình 6: Giao diện chat khi đăng nhập thành công

A screenshot of a computer

Description automatically generated

Hình 7: Thông báo lưu lịch sử chat thành công

* Người dùng sử dụng dấu : “/” để sử dụng chức năng của chat bot.

A screenshot of a computer

Description automatically generated

Hình 8: Các chức năng của chatbot

* Kết quả trả về khi người dùng muốn xem thời tiết quận cụ thể:

A screenshot of a computer

Description automatically generated

Hình 9: Kết quả trả về khi xem thời tiết

* Kết quả trả về khi người dùng xem ip:

A screenshot of a computer

Description automatically generated

Hình 10: Kết quả trả về khi xem ip

A screenshot of a phone

Description automatically generated

Hình 11: Kết quả trả về khi xem tarot

A screenshot of a computer

Description automatically generated

Hình 12: Kết quả xem port

A screenshot of a computer

Description automatically generated

Hình 13: Kết quả khi kiểm tra url

A screenshot of a computer screen

Description automatically generated

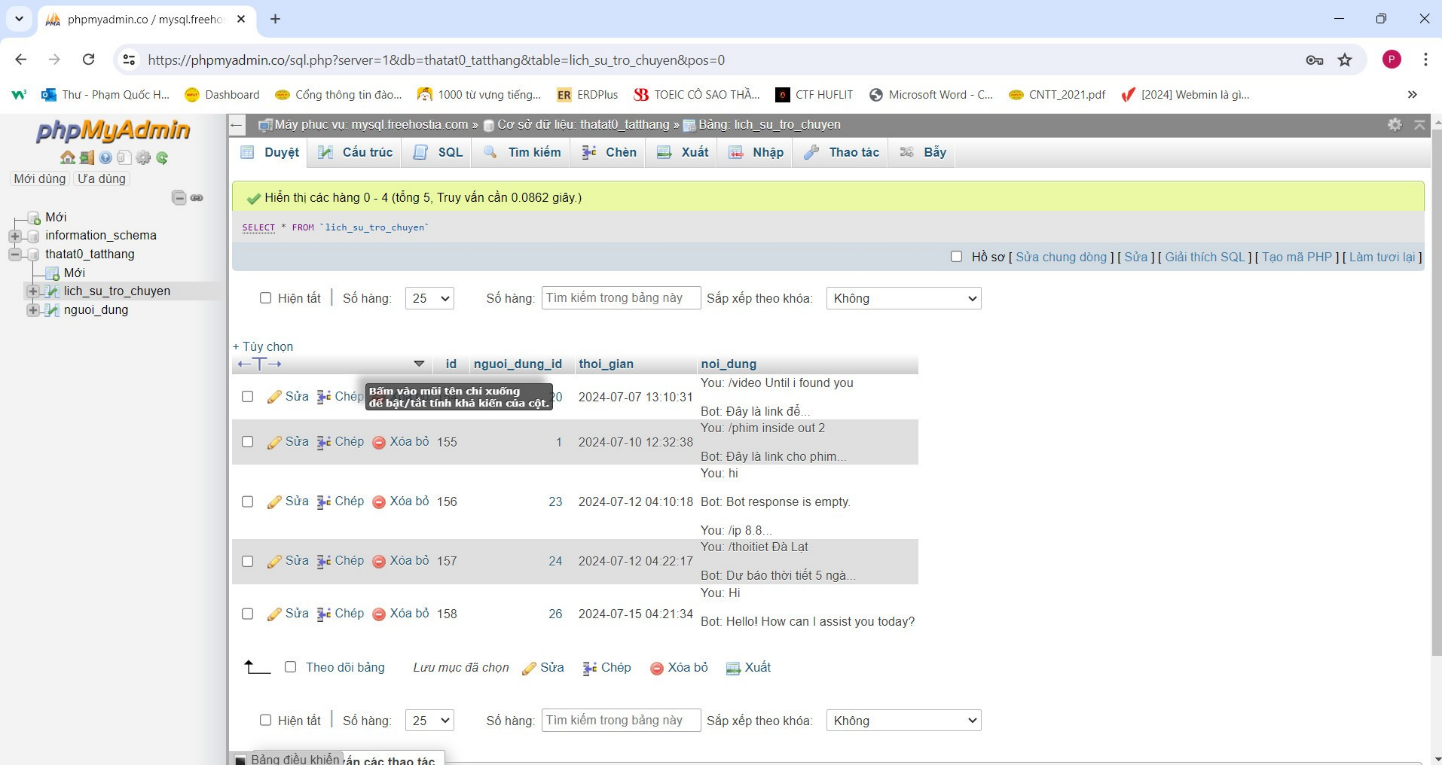
Hình 14: Kết quả trả về khi muốn tìm video YouTube

A close up of a website

Description automatically generated

Hình 15: Kết quả khi tìm phim

## Thiết kế cơ sở dữ liệu.



Hình 16: Lịch sử chat trong database

Ảnh có chứa văn bản, phần mềm, Biểu tượng máy tính, Trang web

Mô tả được tạo tự động

Hình 17: Bảng người dùng

A close-up of a message

Description automatically generated

Hình 18: Cấu trúc bảng người dùng

A close-up of a white page

Description automatically generated

Hình 19: Cấu trúc bảng lịch sử trò chuyện

* **user\_id**: Đây là ID của người dùng, giúp xác định duy nhất mỗi người dùng trong hệ thống. Nó có thể được sử dụng để liên kết các tin nhắn với người dùng cụ thể.
* **time**: Trường này ghi lại thời gian mà tin nhắn được gửi. Thời gian này thường được lưu dưới định dạng ngày tháng năm, giúp theo dõi thứ tự và thời gian của các cuộc trò chuyện.
* **content**: Đây là nội dung của tin nhắn. Nó có thể là câu hỏi của người dùng hoặc phản hồi từ chatbot. Nội dung này là phần quan trọng nhất, vì nó chứa thông tin mà người dùng muốn truyền đạt hoặc nhận được.
* **is\_user**: Trường này có thể cho biết liệu tin nhắn được gửi từ người dùng hay từ chatbot. Nếu giá trị là 1 (hoặc true), điều đó có nghĩa là tin nhắn đến từ người dùng; nếu là 0 (hoặc false), tin nhắn đến từ chatbot.

**Ví dụ về dữ liệu**

* **2024-01-01 12:00:00**: Thời gian mà tin nhắn được gửi.
* **"Hello, how can I help you?"**: Nội dung tin nhắn từ chatbot.
* **user\_id 156**: ID của người dùng đã gửi tin nhắn.

**Chức năng**

* **Theo dõi cuộc trò chuyện**: Các trường này giúp theo dõi và quản lý các cuộc trò chuyện giữa người dùng và chatbot, từ đó cải thiện khả năng tương tác và phản hồi của chatbot.
* **Phân tích dữ liệu**: Dữ liệu trong bảng có thể được sử dụng để phân tích hành vi người dùng, từ đó tối ưu hóa trải nghiệm người dùng.

## Thiết kế các module và các luồng dữ liệu.

* **Thiết kế các module**
* **java.net**:
  + **Mô tả**: Được sử dụng để xử lý kết nối mạng, đặc biệt là khi ứng dụng cần thiết lập và quản lý các kết nối socket giữa các client và server trong hệ thống chatbot.
  + **Chức năng**:
    - Tạo và quản lý kết nối socket.
    - Quản lý giao tiếp giữa client và server.
    - Xử lý các yêu cầu và phản hồi giữa client và server.
* **java.io**:
  + **Mô tả**: Dùng để đọc và ghi dữ liệu từ và vào các luồng (streams), file và các tài nguyên hệ thống, phục vụ cho việc truyền tin nhắn giữa các client và server.
  + **Chức năng**:
    - Đọc và ghi dữ liệu từ file.
    - Xử lý luồng dữ liệu từ client đến server và ngược lại.
    - Lưu trữ lịch sử trò chuyện và các dữ liệu liên quan khác.
* **java.util**:
  + **Mô tả**: Các cấu trúc dữ liệu như ArrayList, HashMap được sử dụng để lưu trữ và quản lý danh sách người dùng, tin nhắn, danh sách các phòng chat, v.v.
  + **Chức năng**:
    - Quản lý danh sách người dùng.
    - Lưu trữ và truy xuất tin nhắn.
    - Quản lý danh sách phòng chat và các thông tin liên quan.
* **java.awt và javax.swing**:
  + **Mô tả**: Được sử dụng để xây dựng giao diện người dùng đồ họa (GUI) cho ứng dụng chat. Thông thường, sử dụng Swing để tạo các thành phần như cửa sổ chat, ô nhập tin nhắn, danh sách người dùng, v.v.
  + **Chức năng**:
    - Tạo và quản lý cửa sổ giao diện người dùng.
    - Tạo các thành phần giao diện như khung chat, nút gửi tin nhắn, danh sách người dùng.
    - Xử lý sự kiện từ người dùng (ví dụ: gửi tin nhắn, chọn phòng chat).
* Thiết kế các luồng dữ liệu

Đăng ký

* Client: Người dùng nhập thông tin đăng ký (username, password, v.v.).
  + Model: Register.java ghi thông tin vào cơ sở dữ liệu.
* Server: Nhận yêu cầu từ client và xử lý thông tin.
  + Kết nối với cơ sở dữ liệu qua DataBaseConnection.java.
* Client: Nhận phản hồi từ server về kết quả đăng ký.

Đăng nhập

* Client: Người dùng nhập thông tin đăng nhập (username, password).
  + Model: Login.java ghi thông tin vào cơ sở dữ liệu.
* Server: Nhận yêu cầu từ client và xác thực thông tin.
  + Kết nối với cơ sở dữ liệu qua DataBaseConnection.java.
* Client: Nhận phản hồi từ server về kết quả đăng nhập.

Gửi tin nhắn

* Client: Người dùng nhập tin nhắn và gửi.
  + Model: Message.java ghi thông tin tin nhắn.
* Server: Nhận tin nhắn từ client và xử lý.
  + Kết nối với cơ sở dữ liệu qua DataBaseConnection.java.
* Client: Nhận phản hồi từ server về kết quả gửi tin nhắn.

Nhận tin nhắn

* Client: Nhận tin nhắn từ server.
  + Model: Receive\_Message.java để hiển thị tin nhắn.
* Server: Gửi tin nhắn đến client qua UI Module.

**Gửi nhận ảnh và file**

**Client:**

* **Mô tả**: Client là nơi người dùng tương tác với chatbot.
* **Chức năng**:
  + Nhận đầu vào từ người dùng (tin nhắn hoặc file).
  + Gửi dữ liệu đến server.
  + Nhận phản hồi từ server và hiển thị trên giao diện người dùng.

**Server:**

* **Mô tả**: Server xử lý các yêu cầu từ client và phản hồi.
* **Chức năng**:
  + Nhận tin nhắn từ client.
  + Xử lý tin nhắn (sử dụng AI để tạo phản hồi).
  + Gửi phản hồi trở lại client.

**Module File:**

* **Mô tả**: Quản lý việc gửi và nhận file giữa client và server.
* **Chức năng**:
  + **Model\_File\_Sender.java**: Gửi file từ client đến server.
  + **Model\_File\_Receiver.java**: Nhận file từ server và lưu trữ.
  + **module java fxbrowser:** Hiển thị nội dung web trong giao diện ứng dụng JavaFX mà không cần mở trình duyệt bên ngoài.

## Thiết kế ngoài yêu cầu (nếu có).

# Mã nguồn

## Môi trường phát triển: Công cụ và môi trường phát triển phần mềm.

1. Visual code studio
2. MySQL

## Cấu trúc mã nguồn: Tổ chức mã nguồn.

1. ChatSession.java
2. Client.java
3. DisintegrationPanel.java
4. Registration.java
5. LoginDialog.java
6. Server,java

## Quá trình cài đặt và triển khai: Hướng dẫn cài đặt và triển khai hệ thống.

## Các đoạn mã nguồn quan trọng: Chỉ rõ client - server hoạt động như thế nào.

1. Khởi tạo Server và Xử lý Kết nối Client

Lý thuyết:

- Server sử dụng mô hình đa luồng để xử lý nhiều kết nối client đồng thời.

- ServerSocket được sử dụng để lắng nghe các kết nối đến trên một cổng cụ thể.

- Mỗi kết nối client mới được xử lý trong một thread riêng biệt.

Code tương ứng:

```java

 public static void main(String[] args) {

        try (ServerSocket serverSocket = new ServerSocket(12345)) {

            System.out.println("Server is listening on port 12345");

            while (true) {

                Socket socket = serverSocket.accept();

                System.out.println("New client connected");

                new ClientHandler(socket).start();

            }

        } catch (IOException ex) {

            ex.printStackTrace();

        }

    }

2. Xử lý Yêu cầu Client

Lý thuyết:

- Mỗi ClientHandler là một thread riêng biệt, cho phép xử lý đồng thời nhiều client.

- Sử dụng BufferedReader và PrintWriter để đọc và ghi dữ liệu từ/đến client.

- Yêu cầu được phân loại dựa trên lệnh đầu tiên (ví dụ: /thoitiet, /ip, /tarot).

- ExecutorService được sử dụng để quản lý các tác vụ chạy bất đồng bộ, cho phép hủy tác vụ đang chạy khi cần.

Code tương ứng:

private static class ClientHandler extends Thread {

        private final Socket socket;

        private PrintWriter out;

        private BufferedReader in;

        private Future<?> currentTask;

        public ClientHandler(Socket socket) {

            this.socket = socket;

        }

        public void run() {

            try {

                in = new BufferedReader(new InputStreamReader(socket.getInputStream(), StandardCharsets.UTF\_8));

                out = new PrintWriter(new OutputStreamWriter(socket.getOutputStream(), StandardCharsets.UTF\_8), true);

                String inputLine;

                while ((inputLine = in.readLine()) != null) {

                    System.out.println("Received: " + inputLine);

                    final String finalInputLine = inputLine;

                    if (inputLine.equalsIgnoreCase("STOP") && currentTask != null && !currentTask.isDone()) {

                        currentTask.cancel(true);

                        out.println("Request cancelled.");

                        continue;

                    }

                    switch (finalInputLine.split(" ")[0]) {

                        case "/thoitiet":

                            handleRequest(() -> handleWeatherRequest(finalInputLine));

                            break;

                        case "/ip":

                            handleRequest(() -> handleIPRequest(finalInputLine));

                            break;

                        case "/tarot":

                            handleRequest(() -> handleTarotRequest());

                            break;

                        case "/port":

                            handleRequest(() -> handlePortRequest(finalInputLine));

                            break;

                        case "/phim":

                            handleRequest(() -> handleMovieRequest(finalInputLine));

                            break;

                        case "/video":

                            handleRequest(() -> handleVideoRequest(finalInputLine));

                            break;

                        case "/check":

                            handleRequest(() -> handleCheckUrlSafety(finalInputLine));

                            break;

                        case "thoát":

                        case "quit":

                            out.println("Goodbye!");

                            return;

                        default:

                            handleRequest(() -> handleBotResponse(finalInputLine));

                            break;

                    }

                }

            } catch (IOException ex) {

                ex.printStackTrace();

            } finally {

                try {

                    socket.close();

                } catch (IOException ex) {

                    ex.printStackTrace();

                }

            }

        }

3. Tương tác với API Bên ngoài

Lý thuyết:

- Server tương tác với nhiều API bên ngoài để cung cấp các dịch vụ khác nhau.

- HttpsURLConnection được sử dụng để gửi yêu cầu HTTPS an toàn.

- Phản hồi JSON từ API được phân tích bằng thư viện Gson.

- Xử lý lỗi và ngoại lệ được thực hiện để đảm bảo tính ổn định của server.

Code tương ứng (ví dụ với API kiểm tra an toàn URL):

 private void handleCheckUrlSafety(String inputLine) {

            try {

                if (Thread.currentThread().isInterrupted()) {

                    out.println("Request cancelled.");

                    return;

                }

                String url = inputLine.substring("/check".length()).trim();

                String safetyInfo = checkUrlSafety(url);

                if (Thread.currentThread().isInterrupted()) {

                    out.println("Request cancelled.");

                    return;

                }

                out.println(safetyInfo);

            } catch (Exception e) {

                if (!Thread.currentThread().isInterrupted()) {

                    out.println("Error checking URL safety: " + e.getMessage());

                }

            }

        }

        private void handleBotResponse(String userMessage) {

            try {

                if (Thread.currentThread().isInterrupted()) {

                    out.println("Request cancelled.");

                    return;

                }

                String botResponse = getBotResponse(userMessage);

                if (Thread.currentThread().isInterrupted()) {

                    out.println("Request cancelled.");

                    return;

                }

                out.println(botResponse);

            } catch (Exception e) {

                if (!Thread.currentThread().isInterrupted()) {

                    out.println("Error fetching bot response: " + e.getMessage());

                }

            }

        }

```

4. Xử lý Đồng thời và Hủy Tác vụ

Lý thuyết:

- ExecutorService được sử dụng để quản lý pool of threads, tối ưu hóa việc sử dụng tài nguyên.

- Future<?> được sử dụng để theo dõi và kiểm soát các tác vụ đang chạy.

- Cơ chế hủy tác vụ cho phép server phản hồi nhanh với yêu cầu mới hoặc lệnh hủy từ client.

Code tương ứng:

private static final ExecutorService executor = Executors.newFixedThreadPool(5);

private void handleRequest(Runnable task) {

            if (currentTask != null && !currentTask.isDone()) {

                currentTask.cancel(true);

                out.println("Previous request cancelled.");

            }

            try {

                Thread.sleep(100);

            } catch (InterruptedException e) {

                Thread.currentThread().interrupt();

            }

            currentTask = executor.submit(() -> {

                try {

                    task.run();

                } catch (Exception e) {

                    out.println("Error processing request: " + e.getMessage());

                }

            });

        }

```

5. Kết nối và Tương tác với Cơ sở Dữ liệu

Lý thuyết:

- Server sử dụng JDBC để kết nối với cơ sở dữ liệu MariaDB.

- PreparedStatement được sử dụng để ngăn chặn SQL Injection và tối ưu hóa hiệu suất.

- Các phương thức tương tác với cơ sở dữ liệu được thiết kế để xử lý các hoạt động CRUD (Create, Read, Update, Delete).

Code tương ứng:

private static final String DB\_URL = "jdbc:mariadb://mysql.freehostia.com:3306/thatat0\_tatthang?connectTimeout=10000&socketTimeout=15000";

    private static final String USER = "thatat0\_tatthang";

    private static final String PASS = "Trangngu1402@";

public static void dangKyNguoiDung(String ho, String ten, String email, String matKhau) throws SQLException {

        String sql = "INSERT INTO nguoi\_dung (ho, ten, email, mat\_khau) VALUES (?, ?, ?, ?)";

        try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASS);

             PreparedStatement pstmt = conn.prepareStatement(sql)) {

            pstmt.setString(1, ho);

            pstmt.setString(2, ten);

            pstmt.setString(3, email);

            pstmt.setString(4, matKhau);

            pstmt.executeUpdate();

        }

    }

    public static boolean kiemTraEmailTonTai(String email) throws SQLException {

        String sql = "SELECT COUNT(\*) FROM nguoi\_dung WHERE email = ?";

        try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASS);

             PreparedStatement pstmt = conn.prepareStatement(sql)) {

            pstmt.setString(1, email);

            try (ResultSet rs = pstmt.executeQuery()) {

                if (rs.next()) {

                    return rs.getInt(1) > 0;

                }

            }

        }

        return false;

    }

public static int xacThucNguoiDung(String email, String matKhau) throws SQLException {

        String sql = "SELECT id FROM nguoi\_dung WHERE email = ? AND mat\_khau = ?";

        try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASS);

             PreparedStatement pstmt = conn.prepareStatement(sql)) {

            pstmt.setString(1, email);

            pstmt.setString(2, matKhau);

            try (ResultSet rs = pstmt.executeQuery()) {

                if (rs.next()) {

                    return rs.getInt("id");

                } else {

                    return -1;

                }

            }

        }

    }

6. Xử lý Yêu cầu Cụ thể

Lý thuyết:

- Mỗi loại yêu cầu (thời tiết, IP, tarot, v.v.) được xử lý trong một phương thức riêng biệt.

- Các phương thức này tương tác với API bên ngoài hoặc cơ sở dữ liệu nội bộ để lấy thông tin cần thiết.

- Kết quả được định dạng và gửi trả về cho client.

Code tương ứng (ví dụ với yêu cầu thời tiết):

private void handleWeatherRequest(String inputLine) {

            try {

                if (Thread.currentThread().isInterrupted()) {

                    out.println("Request cancelled.");

                    return;

                }

                String location = inputLine.substring("/thoitiet".length()).trim();

                String weatherInfo = getWeatherInfo(location);

                if (Thread.currentThread().isInterrupted()) {

                    out.println("Request cancelled.");

                    return;

                }

                out.println(weatherInfo);

            } catch (Exception e) {

                if (!Thread.currentThread().isInterrupted()) {

                    out.println("Error fetching weather information: " + e.getMessage());

                }

            }

        }

private String getWeatherInfo(String location) {

            try {

                String urlStr = String.format("https://api.openweathermap.org/data/2.5/forecast?q=%s&appid=%s&units=metric", URLEncoder.encode(location, "UTF-8"), OPENWEATHER\_API\_KEY);

                URL url = new URL(urlStr);

                HttpsURLConnection conn = (HttpsURLConnection) url.openConnection();

                conn.setRequestMethod("GET");

                if (conn.getResponseCode() == 200) {

                    BufferedReader reader = new BufferedReader(new InputStreamReader(conn.getInputStream(), StandardCharsets.UTF\_8));

                    StringBuilder content = new StringBuilder();

                    String inputLine;

                    while ((inputLine = reader.readLine()) != null) {

                        content.append(inputLine);

                    }

                    reader.close();

                    conn.disconnect();

                    JsonObject jsonObject = JsonParser.parseString(content.toString()).getAsJsonObject();

                    JsonArray forecastList = jsonObject.getAsJsonArray("list");

                    StringBuilder weatherInfo = new StringBuilder(String.format("Dự báo thời tiết 5 ngày tới cho %s:\n\n", location));

                    for (int i = 0; i < forecastList.size(); i++) {

                        JsonObject entry = forecastList.get(i).getAsJsonObject();

                        String dtTxt = entry.get("dt\_txt").getAsString();

                        JsonObject main = entry.getAsJsonObject("main");

                        JsonArray weather = entry.getAsJsonArray("weather");

                        JsonObject weatherDetails = weather.get(0).getAsJsonObject();

                        String description = weatherDetails.get("description").getAsString();

                        double temp = main.get("temp").getAsDouble();

                        weatherInfo.append(String.format("%s: %s, Nhiệt độ: %.2f°C\n", dtTxt, description, temp));

                    }

                    return weatherInfo.toString();

                } else {

                    return "Error fetching weather information.";

                }

            } catch (Exception e) {

                e.printStackTrace();

                return "Error fetching weather information.";

            }

        }

```

7. Bảo mật và Xử lý Lỗi

Lý thuyết:

- Sử dụng HTTPS cho các kết nối API bên ngoài để đảm bảo bảo mật.

- Xử lý ngoại lệ và lỗi ở nhiều cấp độ để tăng tính ổn định của server.

- Sử dụng PreparedStatement để ngăn chặn SQL Injection trong các truy vấn cơ sở dữ liệu.

- Kiểm tra và xác thực đầu vào từ client để ngăn chặn các cuộc tấn công có thể xảy ra.

Code tương ứng:

 private void handleRequest(Runnable task) {

            if (currentTask != null && !currentTask.isDone()) {

                currentTask.cancel(true);

                out.println("Previous request cancelled.");

            }

            try {

                Thread.sleep(100);

            } catch (InterruptedException e) {

                Thread.currentThread().interrupt();

            }

            currentTask = executor.submit(() -> {

                try {

                    task.run();

                } catch (Exception e) {

                    out.println("Error processing request: " + e.getMessage());

                }

            });

        }

```

Tóm lại, hệ thống server-client này được thiết kế để xử lý đồng thời nhiều yêu cầu từ client, tương tác với nhiều API bên ngoài và cơ sở dữ liệu nội bộ. Nó sử dụng các kỹ thuật như đa luồng, ExecutorService, và PreparedStatement để tối ưu hóa hiệu suất và bảo mật. Hệ thống cũng có khả năng xử lý lỗi và hủy tác vụ, giúp tăng tính ổn định và phản hồi nhanh với yêu cầu của người dùng.

private static final String COZE\_API\_KEY = "pat\_K44WnywXIj64PE5sUCljuerl1keBacaRICKRYlA1wB4n2wlcvV0cYR6gkTZAGtCd";

    private static final String COZE\_BOT\_ID = "7387388106103783425";

    private static final String OPENWEATHER\_API\_KEY = "d643d70366040b11e999b8670828c9f7";

    private static final String VIRUSTOTAL\_API\_KEY = "aaf27990c4a213aaf1563b499af252099972e5710839b4807f2cae0fb75eb068";

    private static final String YOUTUBE\_API\_KEY = "AIzaSyAXer0Oe2K4MyLrCL73G2XrHipBOIDxeVE";

    private static final String TMDB\_API\_KEY = "4f31cc15eae70cf00fa229f3915d93e9";

    private static final ExecutorService executor = Executors.newFixedThreadPool(5);

private void startReceivingResponse() {

        responseThread = new Thread(() -> {

            String response;

            try {

                while ((response = in.readLine()) != null) {

                    final String resp = response;

                    SwingUtilities.invokeLater(() -> {

                        String currentText = chatArea.getText();

                        int indicatorIndex = currentText.lastIndexOf(botResponseIndicator);

                        if (indicatorIndex != -1) {

                            currentText = currentText.substring(0, indicatorIndex)

                                    + currentText.substring(indicatorIndex + botResponseIndicator.length());

                            chatArea.setText(currentText);

                        }

                        String message = "Bot: " + resp + "\n";

                        chatArea.append(message);

                        currentSession.addMessage(message);

                        chatArea.setCaretPosition(chatArea.getDocument().getLength());

                        isSendingMessage = false;

                        sendButton.setEnabled(true);

                        stopButton.setEnabled(false);

                        if (resp.equals("Goodbye!")) {

                            closeClient();

                        }

                    });

                }

            } catch (IOException e) {

                e.printStackTrace();

                showErrorAndExit("Lost connection to the server.");

            }

        });

        responseThread.start();

    }

## Mô tả các điểm nổi bật trong mã nguồn.

Các hàm handle request: Gồm những hàm handle request từng chức năng chính như weather, ip, port, tarot, movie, url check,video YouTube.

A computer screen shot of code

Description automatically generated

Các hàm chính :

A screen shot of a computer code

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screen shot of a computer code

Description automatically generated

# Đánh giá và kết luận

## Kết quả thực hiện (hình ảnh - nếu có).

## Đánh giá hệ thống: Đánh giá về hiệu năng, bảo mật, độ tin cậy.

**1. Hiệu năng:**

* **Tốc độ phản hồi:** Một hệ thống chatbot AI thời tiết cần phải có khả năng phản hồi nhanh chóng và chính xác các yêu cầu của người dùng.

**2. Bảo mật:**

* **Bảo mật dữ liệu:** Dữ liệu người dùng, đặc biệt là thông tin cá nhân, cần được bảo mật nghiêm ngặt thông qua việc sử dụng mã hóa và các biện pháp bảo mật khác.
* **Xác thực và phân quyền:** Hệ thống cần có các cơ chế xác thực mạnh mẽ để đảm bảo rằng chỉ những người dùng hợp lệ mới có thể truy cập được dịch vụ. Phân quyền truy cập cũng là một phần quan trọng để đảm bảo an toàn.
* **Bảo mật mạng:** Việc bảo vệ hệ thống trước các cuộc tấn công từ bên ngoài như DDoS, SQL injection, và các cuộc tấn công mạng khác là vô cùng quan trọng.

**3. Độ tin cậy:**

* **Khả năng khôi phục sự cố:** Hệ thống cần có khả năng tự động khôi phục sau các sự cố để đảm bảo rằng dịch vụ không bị gián đoạn lâu dài. Điều này bao gồm việc sử dụng các cơ chế backup và recovery, và có thể triển khai ở nhiều khu vực địa lý khác nhau để đảm bảo tính liên tục.
* **Độ chính xác thông tin:** Chatbot cung cấp thông tin thời tiết chính xác và cập nhật liên tục.
* **Khả năng mở rộng:** Hệ thống có khả năng mở rộng để đáp ứng nhu cầu ngày càng tăng của người dùng mà không bị suy giảm hiệu suất.

## Kết luận: Tổng kết lại các kết quả đạt được.

1. Xây dựng được Chatbot AI đáp ứng được nhu cầu của người dùng
2. Triển khai thành công mô hình Server – Client
3. Có thêm các chức năng ngoài yêu cầu như: xem Tarot, kiểm tra URL, tìm video, phim.

## Hạn chế và hướng phát triển: Những hạn chế của đề tài và các hướng phát triển trong tương lai.

Hạn chế về mặt thời gian là một đặc thù của một đồ án và cần khả năng làm việc liên tục cùng với nhóm của mình để hoàn thành tốt nhất. Phần phát triển đồ án trong tương lai chưa được xác định rõ ràng

## Bảng phân công công việc

|  |  |  |
| --- | --- | --- |
| Thành viên | Phân công | Đánh giá |
| Phan Tất Thắng | 1.WEB DEMO:   * Viết code client sử dụng phương thức tcp * Viết code giao diện client   2.BÁO CÁO:   * Giới thiệu đề tài: Trình bày lý do chọn đề tài và tầm quan trọng của nó trong thực tế. * Mục tiêu nghiên cứu: Xác định mục tiêu chính của đồ án. * Phạm vi nghiên cứu: Giới hạn và phạm vi của đề tài. * Kiến trúc hệ thống: Mô hình tổng thể của hệ thống. * Thiết kế giao diện người dùng. * Thiết kế cơ sở dữ liệu. * Thiết kế các module và các luồng dữ liệu. * Thiết kế ngoài yêu cầu (nếu có). | 100% |
| Võ Văn Thần Thái | 1.WEB DEMO:   * Xây dựng AI bằng API * Chức năng tra cứu thời tiết * Chức năng lấy lời khuyên từ api   2.BÁO CÁO:   * Thiết kế cơ sở dữ liệu. * Thiết kế các module và các luồng dữ liệu. * Yêu cầu chức năng: Các chức năng chính của hệ thống. * Thiết kế giao diện người dùng. * Thiết kế giao diện người dùng. | 100% |
| Trần Anh Tài | 1.WEB DEMO:   * Chức năng tìm kiếm vị trí Ip * Chức năng quét url   2.BÁO CÁO:   * Các kịch bản sử dụng (use case): Mô tả các tình huống sử dụng thực tế. * Đánh giá hệ thống: Đánh giá về hiệu năng, bảo mật, độ tin cậy. * Kết luận: Tổng kết lại các kết quả đạt được. * Hạn chế và hướng phát triển: Những hạn chế của đề tài và các hướng phát triển trong tương lai. | 100% |
| Phạm Quốc Huy | 1.WEB DEMO:   * Viết code server.java sử dụng phương thức tcp để kết nối với client * Quét port   2.BÁO CÁO:   * Tài liệu tham khảo: Danh sách các tài liệu, mã nguồn (source code), bài báo tham khảo. * Quá trình cài đặt và triển khai: Hướng dẫn cài đặt và triển khai hệ thống. * Các đoạn mã nguồn quan trọng (chỉ rỏ client - server hoặt động như thế nào). * Mô tả các điểm nổi bật trong mã nguồn. | 100% |

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

# Tài liệu tham khảo

 **Sách**:

* **"Chatbots: 5 Ways to Help Your Business Grow"** của Matthew Barton: Cuốn sách này giải thích cách chatbots có thể cải thiện dịch vụ khách hàng và giúp doanh nghiệp phát triển.
* **"Designing Bots: Creating Conversational Experiences"** của Amir Shevat: Sách này cung cấp hướng dẫn chi tiết về cách thiết kế và triển khai chatbot hiệu quả.

 **Bài báo khoa học**:

* **"Chatbot Technologies and Applications for Enterprises"** của Jane E. Smith: Bài viết này tổng quan về công nghệ chatbot và cách áp dụng trong doanh nghiệp.
* **"Natural Language Processing for Chatbots: A Review"** của Thomas Miller: Bài báo này cung cấp cái nhìn sâu sắc về các kỹ thuật xử lý ngôn ngữ tự nhiên được sử dụng trong chatbot.

 **Trang web**:

* **Chatbots Magazine** ([chatbotsmagazine.com](https://chatbotsmagazine.com)): Đây là một nguồn tài liệu phong phú về các xu hướng, kỹ thuật và ứng dụng mới nhất của chatbot.
* **AI Chatbot Guide** ([aichatbotguide.com](https://aichatbotguide.com)): Trang web này cung cấp hướng dẫn chi tiết và tài liệu tham khảo về việc phát triển và triển khai chatbot.

 **Khóa học trực tuyến**:

* **Coursera**: Có nhiều khóa học về AI và chatbot như "Building Chatbots with Watson API" hoặc "Introduction to TensorFlow for Artificial Intelligence, Machine Learning, and Deep Learning".
* **Udacity**: Khóa học "AI Programming with Python" bao gồm các module về phát triển chatbot.

 **Blog và diễn đàn**:

* **Towards Data Science** ([towardsdatascience.com](https://towardsdatascience.com)): Một cộng đồng blog nơi các chuyên gia và nhà nghiên cứu chia sẻ kiến thức về AI và chatbot.
* **Stack Overflow** ([stackoverflow.com](https://stackoverflow.com)): Diễn đàn này có nhiều bài viết và câu hỏi liên quan đến việc lập trình và khắc phục sự cố khi phát triển chatbot.

**Tài liệu dùng để tham khảo nhiều trong bài đồ án:**

**Chatgpt,…**

Link clip demo chatbot của nhóm:

[https://www.YouTube.com/watch?v=v2Yp3PnPHIM](https://www.youtube.com/watch?v=v2Yp3PnPHIM)