**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**

**<<Tên đồ án>>**

**BIỆN THÀNH CHUNG**

chung.bt150393@sis.hust.edu.vn

**Ngành Kỹ thuật phần mềm**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | TS. Nguyễn Thị Thu Trang |
| **Bộ môn:** | Công nghệ phần mềm |
| **Viện:** | Công nghệ thông tin – Truyền thông |
| **HÀ NỘI, 07/2020** | |

# Lời cam kết

Họ và tên sinh viên: Biện Thành Chung

Điện thoại liên lạc: (+84) 971702707 Email: chung.bt150393@sis.hust.edu.vn

Lớp: CNTT 2.02 – K60 Hệ đào tạo: Đại học chính quy

Tôi – *Biện Thành Chung* – cam kết Đồ án Tốt nghiệp (ĐATN) là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của *TS. Nguyễn Thị Thu Trang*. Các kết quả nêu trong ĐATN là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong ĐATN – bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

|  |  |
| --- | --- |
|  | *Hà Nội, ngày tháng năm*  Tác giả ĐATN  *Biện Thành Chung* |

# Lời cảm ơn

a

# Tóm tắt

Trước tình hình diễn biến phức tạp của dịch Covid-19, các doanh nghiệp đã chuyển hướng sang kinh doanh online, việc thay đổi này đã mang đến nhiều cơ hội cũng như thách thức. Ngày nay, với sự phát triển nhanh chóng của khoa học công nghệ, sự ra đời của chatbot, một ứng dụng có thể tương tác với người dùng bằng ngôn ngữ tự nhiên dưới một giao diện đơn giản, âm thanh hoặc dưới dạng tin nhắn.Khi người dùng nhắn tin đến, bot sẽ phân tích ý định, sau đó sẽ thực hiện chuỗi hành động và sẽ phản hồi lại cho người dùng. dùng. Với mục tiêu xây dựng một nền tảng cung cấp các chức năng để tạo ra những con chatbot dễ dàng và tích hợp nhanh chóng mà khôngyếu cầu người dùng có chuyên môn kĩ thuật cao thì giải pháp đưa ra là cung cấp một ứng dụng có thể tạo ra những chatbot sử dụng luật. Kết quả đồ án là một ứng dụng hoàn chỉnh, cung cấp một giao diện trực quan với đầy đủcác chức năng cơ bản để người dùng có thể thực hiện được những mục tiêu, yêu cầu đã đề ra. Ứng dụng được xây dựng dựa trên sự kết hợp của ReactJs, NodeJs với cơ sở dữ liệu là MongoDB. Đến thời điểm hiện tại, ứng dụng đã được triển khai trên server với tên miền là <https://chungbt.vbee.vn/>. Mặc dù có thể đáp ứng được với những mục tiêu đã đề ra nhưng ứng dụng vẫn đang trong quá trình phát triển, em luôn luôn lắng nghe và góp nhặt những ý kiến đóng góp của người dùng để hướng tới phiên bản hoàn thiện nhất.

# Mục lục

[Lời cam kết ii](#_Toc42489346)

[Lời cảm ơn iii](#_Toc42489347)

[Tóm tắt iv](#_Toc42489348)

[Mục lục v](#_Toc42489349)

[Danh mục hình vẽ viii](#_Toc42489350)

[Danh mục các từ viết tắt ix](#_Toc42489351)

[Danh mục thuật ngữ x](#_Toc42489352)

[Chương 1 Giới thiệu đề tài 1](#_Toc42489353)

[1.1 Đặt vấn đề 1](#_Toc42489354)

[1.2 Mục tiêu và phạm vi đề tài 2](#_Toc42489355)

[1.3 Định hướng giải pháp 2](#_Toc42489356)

[1.4 Bố cục đồ án 3](#_Toc42489357)

[Chương 2 Phân tích yêu cầu 4](#_Toc42489358)

[2.1 Khảo sát hiện trạng 4](#_Toc42489359)

[2.1.1 Chatbot 4](#_Toc42489360)

[2.1.2 Hệ thống Chatfuel 4](#_Toc42489361)

[2.1.3 Hệ thống Messnow 5](#_Toc42489362)

[2.2 Tổng quan chức năng 6](#_Toc42489363)

[2.2.1 Biểu đồ use case tổng quan 6](#_Toc42489364)

[2.2.2 Biểu đồ use case phân dã Quản lí chuỗi hành động 7](#_Toc42489365)

[2.2.3 Biểu đồ use case phân dã Quản lí luật 7](#_Toc42489366)

[2.2.4 Biểu đồ use case phân dã nhắn tin 8](#_Toc42489367)

[2.3 Đặc tả chức năng 9](#_Toc42489368)

[2.3.1 Đặc tả use case Sửa nội dung chuỗi hành động 10](#_Toc42489369)

[2.3.2 Đặc tả use case Nhắn tin 14](#_Toc42489370)

[2.3.3 Đặc tả chức năng gửi tin nhắn cho nhiều người 15](#_Toc42489371)

[2.4 Yêu cầu phi chức năng 16](#_Toc42489372)

[2.4.1 Tính dễ dùng 16](#_Toc42489373)

[2.4.2 Hiệu năng 16](#_Toc42489374)

[2.4.3 Tính tin cậy 16](#_Toc42489375)

[2.4.4 Tính dễ bảo trì 16](#_Toc42489376)

[Chương 3 Công nghệ sử dụng 17](#_Toc42489377)

[3.1 Front-end 17](#_Toc42489378)

[3.1.1 ReactJs 17](#_Toc42489379)

[3.1.2 Redux 18](#_Toc42489380)

[3.1.3 Redux Saga 20](#_Toc42489381)

[3.1.4 Material-UI 20](#_Toc42489382)

[3.2 Back-end 21](#_Toc42489383)

[3.2.1 NodeJs 21](#_Toc42489384)

[3.2.2 ExpressJs 21](#_Toc42489385)

[3.2.3 MongoDB 22](#_Toc42489386)

[3.2.4 Rest API 22](#_Toc42489387)

[3.2.5 Elasticsearch 23](#_Toc42489388)

[Chương 4 Phát triển và triển khai ứng dụng 25](#_Toc42489389)

[4.1 Thiết kế kiến trúc 25](#_Toc42489390)

[4.1.1 Thiết kế tổng quan 25](#_Toc42489391)

[4.1.2 Kiến trúc front-end 26](#_Toc42489392)

[4.1.3 Kiến trúc back-end 27](#_Toc42489393)

[4.2 Thiết kế chi tiết 28](#_Toc42489394)

[4.2.1 Thiết kế chi tiết 1 số use case chính 28](#_Toc42489395)

[4.2.2 Thiết kế giao diện front-end 32](#_Toc42489396)

[4.2.3 Thiết kế thành phần back-end 33](#_Toc42489397)

[4.2.4 Thiết kế cơ sở dữ liệu 34](#_Toc42489398)

[4.3 Xây dựng ứng dụng 39](#_Toc42489399)

[4.3.1 Thư viện và công cụ sử dụng 39](#_Toc42489400)

[4.3.2 Kết quả đạt được 40](#_Toc42489401)

[4.3.3 Minh hoạ các chức năng chính 40](#_Toc42489402)

[4.4 Kiểm thử 44](#_Toc42489403)

[4.5 Triển khai 51](#_Toc42489404)

[Chương 5 Các giải pháp và đóng góp nổi bật 52](#_Toc42489405)

[5.1 Sử dụng Elasticsearch và chỉ mục để cải thiện kết quả trả về 52](#_Toc42489406)

[5.1.1 Chỉ mục trong MongoDB 52](#_Toc42489407)

[5.1.2 Sử dụng Elastic Search 53](#_Toc42489408)

[5.2 Xây dựng cấu trúc thư mục rõ ràng 56](#_Toc42489409)

[5.3 Thiết kế thành phần có tính tái sử dụng cao và dễ mở rộng 58](#_Toc42489410)

[Chương 6 Kết luận 59](#_Toc42489411)

[6.1 Kết luận 59](#_Toc42489412)

[6.2 Hướng phát triển 60](#_Toc42489413)

[Tài liệu tham khảo 61](#_Toc42489414)

# Danh mục hình vẽ

[**Hình 1** Các hoạt động của chatbot 4](#_Toc42497520)

[**Hình 2** Giao diện Chatfuel 5](#_Toc42497521)

[**Hình 3** Giao diện Messnow 5](#_Toc42497522)

[**Hình 4** Biểu đồ use case tổng quan 6](#_Toc42497523)

[**Hình 5** Biểu đồ use case phân rã Quản lí chuỗi hành động 7](#_Toc42497524)

[**Hình 6** Biểu đồ phân dã Use case Quản lí luật 8](#_Toc42497525)

[**Hình 7** Biển đồ phân rã Use case Nhắn tin 9](#_Toc42497526)

[**Hình 8** Cấu trúc cây thư mục ứng dụng giả định 19](#_Toc42497527)

[**Hình 9** Luồng dữ liệu khi sử dụng One-way data binding 20](#_Toc42497528)

[**Hình 10** Luồng dữ liệu khi áp dụng Redux 20](#_Toc42497529)

[**Hình 11** Sơ đồ tổng quan nguyên lí hoạt động của Redux Saga 21](#_Toc42497530)

[**Hình 12** Kiến trúc tổng quan 26](#_Toc42497531)

[**Hình 13** Kiến trúc front-end 27](#_Toc42497532)

[**Hình 14** Mô hình MVC 28](#_Toc42497533)

[**Hình 15** Biểu đồ trình tự Sửa nội dung chuỗi hành động 30](#_Toc42497534)

[**Hình 16** Biểu đồ trình tự Nhắn tin 32](#_Toc42497535)

[**Hình 17** Gửi tin nhắn cho nhiều người 32](#_Toc42497536)

[**Hình 18** Thiết kế thành phần giao diện 33](#_Toc42497537)

[**Hình 19** Thành phần back-end 34](#_Toc42497538)

[**Hình 20** Biểu đồ thực thể liên kết của ứng dụng 35](#_Toc42497539)

[**Hình 21** Giao điện sau khi tạo mới 1 chuỗi hành động 41](#_Toc42497540)

[**Hình 22** Giao điện sau khi sửa nội dung chuỗi hành động 42](#_Toc42497541)

[**Hình 23** Giao điện danh sách luật đã được thiết lập 42](#_Toc42497542)

[**Hình 24** Chuỗi hành động trả về khi nhắn tin 43](#_Toc42497543)

[**Hình 25** Giao diện trang quản lí người dùng 43](#_Toc42497544)

[**Hình 26** Giao diện gửi tin nhắn dưới dạng văn bản thông thường 44](#_Toc42497545)

[**Hình 27** Kết quả khi sau khi gửi tin nhắn dạng văn bản 44](#_Toc42497546)

[**Hình 28** Giao diện gửi tin nhắn cho nhiều người dưới dạng chuỗi hành động 45](#_Toc42497547)

[**Hình 29** Kết quả khi sau khi gửi tin nhắn dạng chuỗi hành động 45](#_Toc42497548)

[**Hình 30** Ví dụ về đánh chỉ mục trong MongoDB 54](#_Toc42497549)

[**Hình 31** Chỉ mục trong Rule Collection 55](#_Toc42497550)

[**Hình 32** Lưu đồ biểu diễn quá trình xử lí khi tích hợp Elasticsearch 56](#_Toc42497551)

[**Hình 33** Danh sách các luật đã thiết lập 57](#_Toc42497552)

[**Hình 34** Kết quả trả về với trường hợp sai chính tả 57](#_Toc42497553)

[**Hình 35** Cấu trúc thư mục front-end 58](#_Toc42497554)

[**Hình 36** Cấu trúc thư mục back-end 59](#_Toc42497555)

[**Hình 37** Giao diện trang chủ của ứng dụng 60](#_Toc42497556)

[**Hình 38** Cách tổ chức code trong file Dashboard 60](#_Toc42497557)

# Danh mục các từ viết tắt

|  |  |
| --- | --- |
| **API** | Application Programming Interface  Giao diện lập trình ứng dụng |
| **UI** | User Interface  Giao diện người dùng |
| **UX** | User Experience  Trải nghiệm người dùng |
| **ĐATN** | Đồ án tốt nghiệp |
| **CRUD** | Create, Read, Update, Delete  Thêm, xem, sửa, xóa |
| **HTTP** | HyperText Transfer Protocol Giao thức truyền tải siêu  văn bản |
| **URL** | Uniform Resource Locator  Đường dẫn |
| **ĐATN** | Đồ án tốt nghiệp |
| **CSDL** | Cơ sở dữ liệu |
| **NoSQL** | Non SQL  Cơ sở dữ liệu phi quan hệ |
| **RDBMS** | Relational Database Management System  Hệ quản trị cơ sỡ dữ liệu quan hệ |
| **OA** | Official Account  Tài khoản Official của bên thứ 3 (Zalo) |
| **SMS** | Short Message Services  Dịch vụ tin nhắn ngắn |

# Danh mục bảng

[**Bảng 1** Danh sách các use case 9](#_Toc42497895)

[**Bảng 2** Đặc tả use case “Sửa nội dung chuỗi hành động 10](#_Toc42497896)

[**Bảng 3** Dữ liệu đầu ra use case “Sửa nội dung chuỗi hành động” 11](#_Toc42497897)

[**Bảng 4** Dữ liệu đầu ra use case “Sửa nội dung chuỗi hành động” 14](#_Toc42497898)

[**Bảng 5** Đặc tả use case “Nhắn tin” 14](#_Toc42497899)

[**Bảng 6** Đặc tả chức năng gửi tin nhắn cho nhiều người 15](#_Toc42497900)

[**Bảng 7** Dữ liệu đầu cho use case nhắn tin cho nhiều người 16](#_Toc42497901)

[**Bảng 8** Chi tiết thuộc tính của Account 35](#_Toc42497902)

[**Bảng 9** Chi tiết thuộc tính của Bot 36](#_Toc42497903)

[**Bảng 10** Chi tiết thuộc tính của Group 36](#_Toc42497904)

[**Bảng 11** Chi tiết thuộc tính của Block 37](#_Toc42497905)

[**Bảng 12** Chi tiết thuộc tính của Element 37](#_Toc42497906)

[**Bảng 13** Chi tiết thuộc tính của Rule 38](#_Toc42497907)

[**Bảng 14** Chi tiết thuộc tính của Attribute 38](#_Toc42497908)

[**Bảng 15** Chi tiết thuộc tính của User 39](#_Toc42497909)

[**Bảng 16** Chi tiết thuộc tính của BadMessage 39](#_Toc42497910)

[**Bảng 17** Danh sách thư viện và công cụ sử dụng 40](#_Toc42497911)

[**Bảng 18** Kiểm thử chức năng Sửa nội dung chuỗi hành động 45](#_Toc42497912)

[**Bảng 19** Kiểm thử chức năng Nhắn tin 50](#_Toc42497913)

[**Bảng 20** Kiểm thử chức năng Gửi tin nhắn cho nhiều người 51](#_Toc42497914)

[**Bảng 21** Thông số cấu hình của ứng dụng và server triển khai 52](#_Toc42497915)

# Danh mục thuật ngữ

|  |  |
| --- | --- |
| Render | Chuyển đổi các React component thành DOM node để trình duyệt hiểu và hiển thị |
| **Server** | Máy chủ |
| **Client** | Máy khách |
| **Front-End** | Hệ thống giao diện người dùng |
| **Back-End** | Thành phần lập trình trên server |
| **Routing** | Điều hướng |
| **Framework** | Một bộ khung cung cấp các chức năng, giải pháp được cài đặt sẵn giúp tiết kiệm thời gian trong quá trình phát triển ứng dụng |
| **Component** | Thành phần trong 1 trang web |
| **Middleware** | Thành phần trung gian giữa request và response |

# Giới thiệu đề tài

Chương 1 giới thiệu những vấn đề thực tế dẫn tới lý do chọn đề tài, tổng quan về công cụ quản lý trong chatbot. Sau đó đưa ra phạm vi và mục tiêu của đồ án, định hướng giải pháp và bố cục trình bày của đồ án.

## Đặt vấn đề

Ngày nay để ứng phó với tình hình dịch Covid-19 thì chính phủ đã đưa ra chỉ thị như cách ly xã hội đóng cửa các cửa hàng không cần thiết. Để tồn tại trong mùa dịch các doanh nghiệp đang dần chuyển hướng từ kinh doanh “offline” sang kinh doanh “online” và kinh doanh nền tảng mạng xã hội như Zalo, Facebook… là một lựa chọn không thể bỏ qua. Một vấn đề đáng để các doanh nghiệp và các tổ chức quan tâm đó là chatbot. Định nghĩa một cách đơn giản, chatbot là một chương trình máy tính mô phỏng và xử lý cuộc hội thoại của con người (bằng văn bản hoặc lời nói), cho phép con người tương tác, trao đổi thông tin với các thiết bị kỹ thuật số một cách hiệu quả và nhanh gọn. Được áp dụng những công nghệ hàng đầu như: trí tuệ nhân tạo (Artificial Intelligence - AI), xử lí ngôn ngữ tự nhiên (Natural Language Processing - NLP), chatbot có thể trả lời tự động từ những kiến thức học được trong quá trình giao tiếp với con người. Chatbot giúp giảm tải 1 phần khối lượng công việc mà người quản lý phải làm. Việc tích hợp chatbot với 1 số nền tảng mạng xã hội hiện nay Zalo, Facebook hay bất kì một webchat nào sẽ đa dạng hóa và đưa chatbot đến gần hơn với người dùng. Có thể kể đến 1 số nền tảng chatbot nổi tiếng như Chatfuel, Dialogflow, Messnow…

Trong thực tế, việc tích hợp 1 hệ thống chatbot với những công nghệ hàng đầu như AI, NLP đối với các doanh nghiệp nhỏ cũng gặp nhiều bất cập về chi phí và yêu cầu người quản lí có phải có hiểu biết về chuyên môn kĩ thuật. Ngược lại những chatbot được sử dụng hiện nay thì thường yêu cầu người dùng thao tác một cách cứng nhắc với các nút bấm trong cửa sổ chat. Việc bấm nút này giống như người dùng đang sử dụng một chương trình máy tính mất đi tính tương tác của người dùng và hệ thống.

Ứng dụng hoàn thành đã giải quyết tổng hòa được các vấn đề là chi phí, sử dụng dễ dàng và giảm tải yếu tố kĩ quá trình tương tác với người dùng.

## Mục tiêu và phạm vi đề tài

Mục tiêu của đề tài là xây dựng 1 hệ thống giao diện trực quan, thân thiện với người dùng. Hệ thống giúp người quản lí có thể cài đặt, tạo dữ liệu cũng như tích hợp những chatbot với nhiều nền tảng một cách dễ dàng, đơn giản

Để đạt được những mục tiêu đã đề ra ở trên thì phạm vi của ĐATN tập trung vào 2 phần việc chính ứng với client và server. Đối với client là xây dựng UI quản lí – nơi Quản trị viên có thể quản lí, cài đặt, tạo dữ liệu mẫu cho những chatbot được quản lí. Về phía server là Management Service, Upload Service và Chat Service. Trong đó, Management Service là dịch vụ chịu trách nhiệm cung cấp các API phục vụ cho việc quản lí bot, hành động, dữ liệu ở phía client. Upload Service là dịch vụ thực hiện nhiệm vụ lưu trữ hình ảnh, file khi người Quản trị viên muốn lưu trữ trong quá trình tạo dữ liệu. Trong phần Chatting Service, hệ thống bao gồm API sẽ thực hiện phân tích tin nhắn người dùng và sẽ hồi đáp lại theo đúng dữ liệu đã được tạo.

## Định hướng giải pháp

Đồ án đề xuất xây dựng một công cụ với giao diện trực quan, trực quan hóa quy trình, thao tác nghiệp vụ để tạo ra các hành động và các luật cho Chatbot. Việc quán lí và xây dựng các luật, hành động yêu cầu thao tác với nhiều dữ liệu, các trường hợp, khả năng có thể xảy ra khi người dùng sử dụng ứng dụng nên em đề xuất phát triển hệ thống thành gồm 2 phần là front-end và back-end

Phía front-end, ReactJs là một thư viện tập trung chủ yếu phát triển vào giao diện. ReactJs phân chia mọi thứ thành các component để thuận lợi cho việc tái sử dụng, mở rộng ứng dụng. ReactJs để quản lí trạng thái, cùng các thao tác mà người dùng đang thực hiện, đồng thời giúp người dùng thực hiện những thao tác với hệ thống 1 cách trực quan, sinh động. Ngoài ra em còn sử dụng Redux Saga là một thư viện Redux middleware, giúp quản lý những side effect trong ứng dụng Redux trở nên đơn giản hơn.

Phía back-end, đồ án sử dụng Expressjs là một framework được xây dựng trên nền tảng của Node.js. Expressjs cung cấp các tính năng mạnh mẽ để phát triển web hoặc mobile. Expressjs hỗ trợ các method HTTP và middleware tạo ra API vô cùng mạnh mẽ và dễ sử dụng. Cơ sở dữ liệu được chọn là Mongodb – cơ sở dữ liệu phân tán, phi cấu trúc, cho phép thực hiện các truy vấn đặc biệt, tích hợp thời gian thực và đánh index hiệu quả. Nó phù hợp với việc thay đổi dữ liệu thường xuyên và cung cấp xác thực dữ liệu bên phía server. MongoDB với Nodejs sẽ tạo nên một sự kết hợp tốt cho việc xây dựng ứng dụng web.

## Bố cục đồ án

Phần còn lại của báo cáo đồ án tốt nghiệp này được tổ chức như sau.

Trong Chương 2, em sẽ đưa ra một vài hệ thống tương tự hiện đang có trên thị trường, so sánh với sản phẩm của mình. Tiếp theo đó, em sẽ đưa ra tổng quan các chức năng của hệ thống mà em xây dựng thông qua biểu đồ use case và đặc tả một số chức năng chính củahệ thống cũng như các yêu cầu phi chức năng.

Trong Chương 3, em sẽ đưa ra công nghệ sử dụng, nguyên lý làm việc, ưu nhược điểm của các công nghệ sử dụng và đưa ra lý do tại sao lại lựa chọn công nghệ đó vào việc xây dựng hệ thống.

Trong Chương 4, dựa vào các công nghệ lựa chọn sử dụng đã trình bày ở Chương 3, em sẽ phân tích, lựa chọn các giải pháp thiết kế kiến trúc sao cho hợp lý. Sau đó đưa ra các kết quả đạt được và minh hoạ các chức năng xây dựng được.

Tiếp đến ở Chương 5, em sẽ trình bày các nội dung mà mình thấy tâm đắc nhất cùng với đóng góp nổi bật trong suốt quá trình làm ĐATN.

Cuối cùng là Chương 6, em sẽ trình bày kết luận về toàn bộ ĐATN, những điều làm được và chưa làm được, định hướng phát triển sản phẩm trong tương lai.

Sau đây em sẽ đi vào chi tiết từng phần của đồ án.

# ­Phân tích yêu cầu

Chương 2 này đưa ra khảo sát các phần mềm, công cụ, quy trình nghiệp vụ trong việc tạo chatbot trong những sản phầm phổ biến hiện nay. Từ đó đưa ra tổng quan chức năng của hệ thống phát triển và đi vào chi tiết làm rõ các chức năng.

## Khảo sát hiện trạng

### Chatbot

Với sự phát triển mạnh mẽ của các nền tảng mạng xã hội cũng nên tảng kinh doanh cũng đang dần chuyển hóa, các doanh nghiệp và tổ chức trên thế giới hiện nay ngày càng đầu tư nhiều hơn và quan tâm hơn về chatbot. Ở cấp độ cơ bản nhất, chatbot được xem như là một chương trình máy tính được lập trình đề mô phỏng cuộc hội thoại của con người thông qua phản hồi bằng giọng nói hoặc văn bản như Hình 1. Bot có thể tự động trả lời các câu hỏi đơn giản hoặc xử lý tình huống phức tạp dựa phụ thuộc vào thuật toán mà chúng được cài đặt. Một số ví dụ điển hình của chatbot như trợ lý ảo đến từ các tập đoàn công nghệ nổi tiếng trên thế giới, Google Assitant, Siri, Cortana và Alexa.

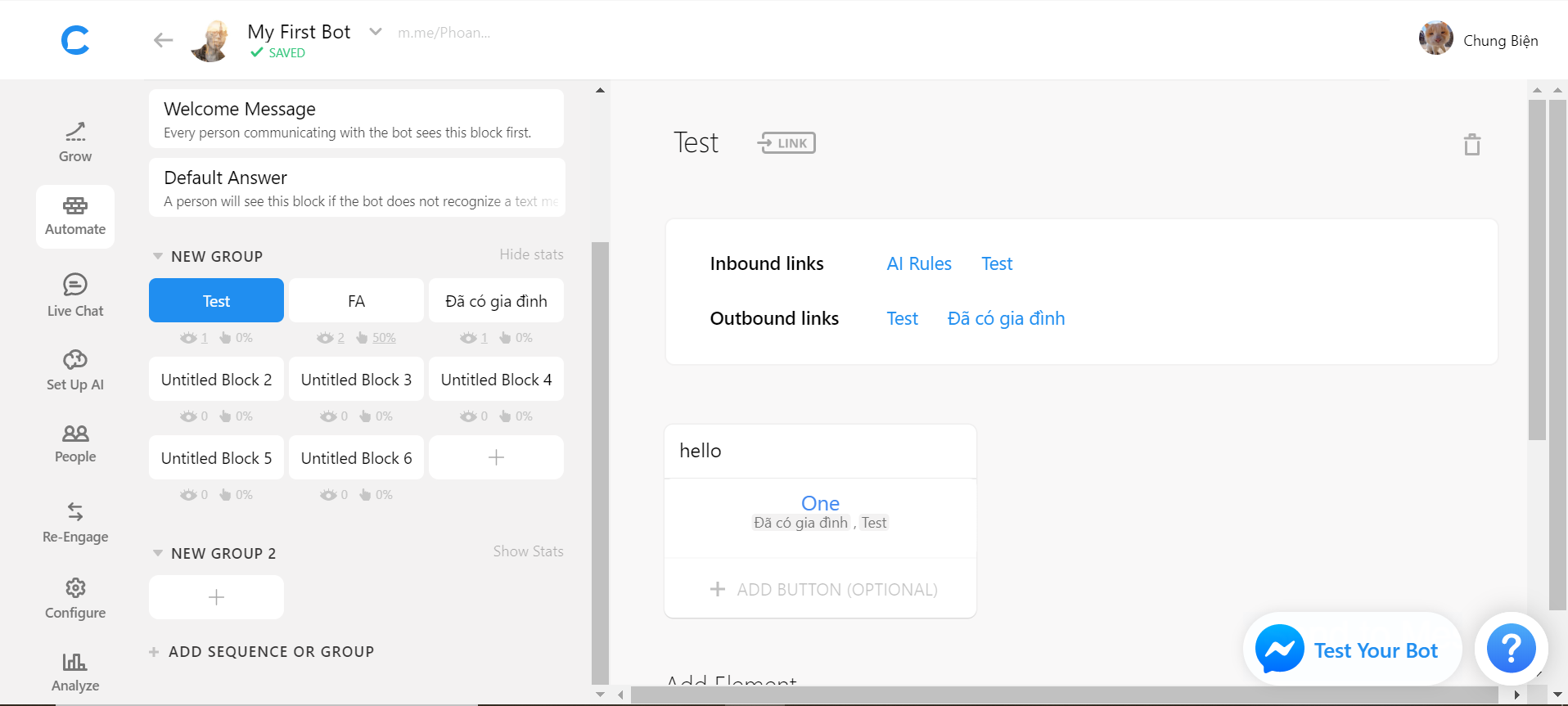
A close up of text on a white background

Description automatically generated

**Hình 1** Các hoạt động của chatbot

### Hệ thống Chatfuel

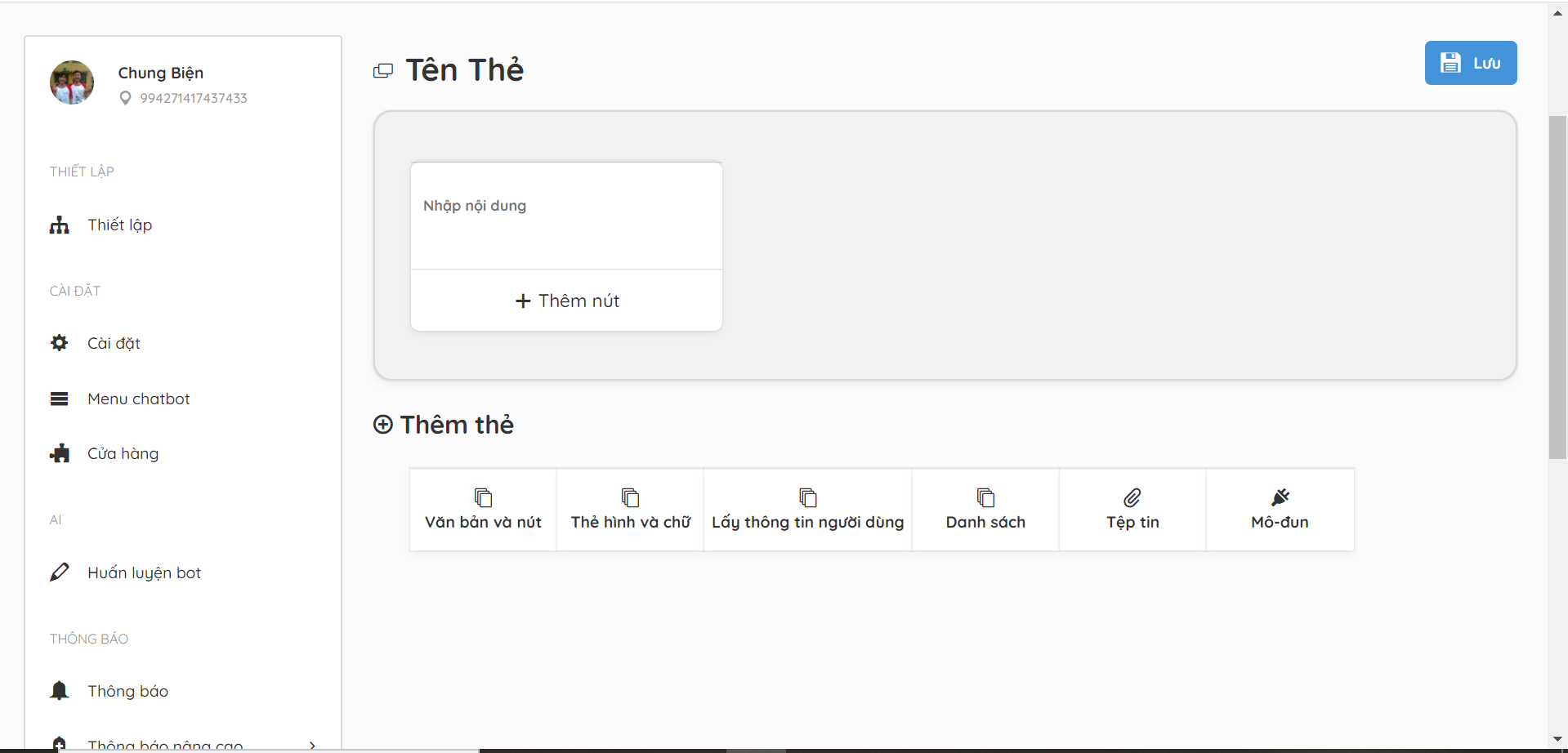
Chatfuel là một ứng dụng được tạo ra nhằm để hỗ trợ người dùng tạo các chatbot. Chatfuel tập trung hướng đến nền tàng Facebook Messeger. Chatfuel cho phép người dùng có thể tạo chuỗi các hành động dưới dạng các block – khối. Ngoài ra Chatfuel cũng hỗ trợ người quản lí xây dựng một kịch bản trả lời tin nhắn tự động với khách hàng dựa trên những dữ liệu dưới dạng các luật. Nhưng Chatfuel chỉ hỗ trợ trả lời đối với những tin nhắn đơn giản. Với những tin nhắn có độ phức tạp hoặc tin nhắn thỏa mãn nhiều luật thì Chatfuel vẫn chưa có thể trả lời.



**Hình 2** Giao diện Chatfuel

### Hệ thống Messnow

Messnow là nền tảng trực tuyến của Việt Nam. Tương tự như đối với Chatfuel thì Messnow cũng cho phép người dùng có thể tạo ra các hành động, quản lí các hành động dưới dạng thẻ. Không những tích hợp được trên Facebook, Messnow còn hỗ trợ thêm trên nền tảng nhắn tin trực tuyến của Việt Nam là Zalo. Không được như Chatfuel, Messnow cũng một giao diện và quy trình tạo dữ liệu, quản lí bao gồm rất nhiều bước, người dùng sẽ mất một khoảng thời gian để có thể làm quen và sử dụng 1 cách thành thạo. Messnow cũng cung cấp chức năng trả lời tin nhắn tự động dựa trên các kịch bản nhưng Messnow chỉ có thể phân tích đối với những tin nhắn ngắn. Messnow cũng không cho phép người quản trị viên tạo ra những tin nhắn cho nhiều người cùng 1 lúc.



**Hình 3** Giao diện Messnow

## Tổng quan chức năng

### Biểu đồ use case tổng quan

A close up of a map

Description automatically generated

**Hình 4** Biểu đồ use case tổng quan

Hệ thống bao gồm 3 tác nhân chính, đó là: Người dùng chưa đăng nhập, Quản trị viên và Ứng dụng bên thứ 3. Trong đó người dùng chưa đăng nhập có thể đăng nhập hoặc đăng kí 1 tài khoản mới để sử dụng hệ thống. Quản trị viên sẽ là tác nhân chính sử dụng các chức năng của hệ thống như: quản lí chuỗi hành động, quản lí luật, CRUD chatbot… Còn Ứng dụng bên thứ 3 là các nền tảng ở giữa kết nối giữa hệ thống và người dùng của ứng dụng đó (người nhắn tin).

### Biểu đồ use case phân dã Quản lí chuỗi hành động

A picture containing text, map

Description automatically generated

**Hình 5** Biểu đồ use case phân rã Quản lí chuỗi hành động

Hình 5 trình bày về phân rã use case Quản lí chuỗi hành động. Theo đó người quản trị viên có thể xem, sửa tên, xóa những chuỗi hành động mà chatbot đang có. Ngoài ra quản trị viên có thể xem và tủy chỉnh nội dung của mỗi chuỗi hành động bằng việc thêm sửa xóa các hành động con bên trong 1 chuỗi hành động.

### Biểu đồ use case phân dã Quản lí luật

Hình 6 trình bày về phân rã use case Quản lí luật. Theo đó người quản trị viên có thể xem, tìm kiếm luật dựa vào từ khóa có chứa trong chuỗi từ khóa của luật, xóa luật mà chatbot đang có. Mỗi 1 luật sẽ có cấu trúc gồm 1 chuỗi từ khóa và 1 danh sách các chuỗi hành động có thể trả về. Người dùng có thể tùy chỉnh thêm, sửa xóa từ khóa trong chuỗi từ khóa cũng như các chuỗi hành động có chứa trong luật đó.

A close up of a map

Description automatically generated

**Hình 6** Biểu đồ phân dã Use case Quản lí luật

### Biểu đồ use case phân dã nhắn tin

Hình 7 biểu diễn phân rã use case nhắn tin. Theo hình thì bên thứ 3 có thể nhận được các dạng tin nhắn là văn bản, hình ảnh, đường link url đến file âm thanh video hoặc thẻ chọn để người gửi cho người dùng các tùy chọn. Ngoài ra khi nhắn tin dưới dạng câu hỏi để thu thập thông tin từ người dùng. Dữ liệu trả lời sẽ được lưu lại trong hệ thống nhằm phục vụ cho những chức năng cần thiết.

A picture containing text, map

Description automatically generated

**Hình 7** Biển đồ phân rã Use case Nhắn tin

## Đặc tả chức năng

Bảng 1 dưới đây sẽ liệt kê các use case được sử dụng. Do kích thước báo cáo có hạn nên em sẽ đặc tả cụ thể 3 use case: Sửa nội dung chuỗi hành động, nhắn tin và gửi tin nhắn hàng cho nhiều người dùng. Các use case còn lại em sẽ trình bày chi tiết trong phần Phụ lục.

**Bảng 1** Danh sách các use case

|  |  |  |
| --- | --- | --- |
| Nhóm use case | Mã use case | Tên use case |
|  | UC001 | Đăng nhập |
|  | UC002 | Đăng kí |
|  | UC003 | Đăng xuất |
|  | UC004 | CRD chatbot |
|  | UC005 | CRUD nhóm chuỗi hành động |
|  | UC006 | Xem thông tin người dùng |
|  | UC007 | Nhắn tin cho nhiều người |
|  | UC008 | Nhắn tin trả lời |
| Quản lí chuỗi hành động | UC009 | CRUD chuỗi hành động |
|  | UC010 | Tìm kiếm chuỗi hành động |
|  | UC011 | Sửa nội dung chuỗi hành động |
|  | UC012 | CRUD hành động con |
| Quản lí luật | UC013 | CRUD luật |
|  | UC014 | Tìm kiếm luật |

### Đặc tả use case Sửa nội dung chuỗi hành động

**Bảng 2** Đặc tả use case “Sửa nội dung chuỗi hành động

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC0011 | Tên Use case | Sửa nội dung chuỗi hành động |
| Tác nhân | Quản trị viên | | |
| Tiền điều kiện | Người dùng đã đăng nhập vào hệ thống với vai trò là quản trị viên  Đã thực hiện tạo chatbot  Đã liên kết thành công chatbot với Official Account trên trang [https://oa.zalo.me/](https://oa.zalo.me/manage/oa)  Đã thực hiện tạo thành công chuỗi hành động | | |
| Luồng sự kiện chính  (Thành công) | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | |  | Quản trị viên | Ấn button tương ứng với các tùy chọn hành động con trên bảng điều khiển | |  | Hệ thống | Tạo ra hành động con tương ứng với giá trị các trường là rỗng | |  | Quản trị viên | Thực hiện nhập các giá trị cần thiết để tạo 1 hành động con | |  | Hệ thống | Lưu trữ các trường vào bộ nhớ tạm | |  | Quản trị viên | Thực hiện ấn button “Lưu” | |  | Hệ thống | Thực hiện lưu dữ liệu và thông báo thành công | | | |
| Luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1a. | Quản trị viên | Thực hiện sửa dữ liệu các hành động con đã có | | 6a. | Hệ thống | Hệ thống báo lỗi khi thất bại | | | |
| Hậu điều kiện | Không có | | |

\* Dữ liệu đầu vào của thông tin cấu hình gồm các trường dữ liệu sau:

**Bảng 3** Dữ liệu đầu ra use case “Sửa nội dung chuỗi hành động”

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **STT** | **Loại hành động con** | **Trường dữ liệu** | **Mô tả** | **Bắt buộc?** | **Điều kiện hợp lệ** | **Ví dụ** |
|  | Văn bản | Câu trả lời dưới dạng văn bản | Câu trả lời của hành động văn bản | Không |  | Sản phẩm là áo phông không cổ |
|  | Hình ảnh | Hình ảnh | Hình ảnh được up load lên của hành động hình ảnh | Có |  |  |
|  |  | Tiêu đề hình ảnh | Tiêu đề cho hình ảnh upload của loại hành động hình ảnh | Không |  | Long tee |
|  | Âm thanh | Đường dẫn đến tệp tin âm thanh | Đường dẫn âm thanh của loại hành động âm thanh | Không |  | https://www.dropbox.com/s/7on4swzwc50p12d/TeA-ft-PC-OfficialMV.mp3 |
|  | Video | Đường dẫn đến tệp tin video | Đường dẫn video của loại hành động video | Không |  | https://www.dropbox.com/s/7kpb0eayvshn0g1/saocungdc.mp4 |
|  | Thẻ chọn | Ảnh bìa | Ảnh upload lên cho loại hành động là thẻ chọn | Không |  |  |
|  |  | Tiêu đề | Tiêu đề chính của loại hành động là thẻ chọn | Không |  | Sự kiện giảm giá |
|  |  | Mô tả | Mô tả cho loại hành động là thẻ chọn | Không |  | Sự kiện giảm giá toàn bộ sản phẩm nhân dịp BF |
|  |  | Button truy cập URL | Button điều hướng người dùng đến trang link url | Có |  |  |
|  |  | Button SMS | Button điều hướng người dùng mở ứng dụng nhắn tin của thiết bị di động | Có |  |  |
|  |  | Button Phone | Button điều hướng người dùng mở ứng dụng điện thoại trong thiết bị di động | Có |  |  |
|  | Câu hỏi lấy dữ liệu | Câu hỏi | Câu hỏi cho loại hành động lấy dữ liệu | Không |  | Bạn chọn áo màu gì nhỉ? |
|  |  | Giá trị lưu | Giá trị lưu vào hệ thống với loại hành động câu hỏi lấy dữ liệu | Không |  | Color\_shirt |

\* Dữ liệu đầu ra:

**Bảng 4** Dữ liệu đầu ra use case “Sửa nội dung chuỗi hành động”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Trường dữ liệu** | **Mô tả** | **Dạng hiển thị** | **Ví dụ** |
|  | Chuỗi hành động | 1 chuỗi bao gồm các hành động con |  |  |

### Đặc tả use case Nhắn tin

**Bảng 5** Đặc tả use case “Nhắn tin”

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC008 | Tên Use case | Nhắn tin trả lời |
| Tác nhân | Ứng dụng bên thứ 3 | | |
| Tiền điều kiện | Đã thực hiện tạo chatbot  Đã liên kết thành công chatbot với Official Account trên trang <https://oa.zalo.me/>  Đã thực hiện tạo thành công chuỗi hành động  Đã thực hiện tạo các luật thành công | | |
| Luồng sự kiện chính  (Thành công) | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | |  | Ứng dụng bên thứ 3 | Gửi request có chứa tin nhắn của người dùng đến hệ thống | |  | Hệ thống | Thực hiện kiểm tra và lấy dữ liệu chatbot được nhận tin nhắn | |  | Hệ thống | Thực hiện phân tích tin nhắn và trả ra chuỗi hành động tiếp theo | |  | Hệ thống | Gửi tin nhắn đến người dùng | | | |
| Luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 2a. | Hệ thống | Báo lỗi đến hệ thống bên thứ 3. | | | |
| Hậu điều kiện | Không có | | |

### Đặc tả chức năng gửi tin nhắn cho nhiều người

**Bảng 6** Đặc tả chức năng gửi tin nhắn cho nhiều người

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC007 | Tên Use case | Nhắn tin cho nhiều người |
| Tác nhân | Quản trị viên | | |
| Tiền điều kiện | Đã thực hiện tạo chatbot  Đã liên kết thành công chatbot với Official Account trên trang <https://oa.zalo.me/>  Đã thực hiện tạo thành công chuỗi hành động  Đã có người sử dụng ứng dụng | | |
| Luồng sự kiện chính  (Thành công) | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | |  | Quản trị viên | Chọn danh sách người dùng cần gửi tin nhắn | |  | Quản trị viên | Thực hiện nhập nội dung văn bản cho tin nhắn | |  | Quản trị viên | Ấn nút gửi để thực hiện yêu cầu gửi tin nhắn cho danh sách người dùng | |  | Hệ thống | Gửi tin nhắn đến người dùng trong danh sách | |  | Hệ thống | Thống báo khi đã gửi thành công | | | |
| Luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1a. | Quản trị viên | Ấn nút xem chi tiết nội dung người dùng trên màn hình | | 2a. | Quản trị viên | Thực hiện chọn chuỗi hành động | | 5a. | Hệ thống | Thông báo lỗi | | | |
| Hậu điều kiện | Không có | | |

\* Dữ liệu đầu vào của tin nhắn dạng văn bản hoặc chuỗi hành động đã có trong hệ thống:

**Bảng 7** Dữ liệu đầu cho use case nhắn tin cho nhiều người

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Trường dữ liệu** | **Mô tả** | **Bắt buộc?** | **Điều kiện hợp lệ** | **Ví dụ** |
|  | Tin nhắn | Tin nhắn dạng văn bản | Có |  | Chào mừng BF shop có mở đợt sale khủng |
|  | Tin nhắn | Tin là 1 chuỗi các hành động đã được cài đặt sẵn trong hệ thống | Có |  | Wellcom Block |

## Yêu cầu phi chức năng

### Tính dễ dùng

Do người sử dụng hệ thống là những người không có chuyên môn về kĩ thuật nên hệ thống được xây dựng với cách tiếp cận đơn giản và trực quan. Các chức năng được thiết kế sao cho dễ thao tác. Ngoài ra, các lỗi có hướng dẫn cụ thể, thông báo bằng Tiếng Việt để người dùng biết vị trí lỗi và có thể thao tác để có thể khắc các lỗi.

### Tính bảo mật

Bảo mật đối với một website, một ứng dụng luôn là một trong những ưu tiên hàng đầu. Có nhiều dạng bảo mật như bảo mật thông tin người dùng, bảo mật dữ liệu,… Áp dụng vào hệ thống đã xây dựng, em sử dụng JWT (Json Web Token). Đây là là 1 tiêu chuẩn mở định nghĩa cách thức truyền tin an toàn giữa các thành viên bằng 1 đối tượng JSON. Mỗi người dùng sẽ được cung cấp một đoạn mã chứa những thông tin cơ bản của họ. Nếu người dùng hết hạn hoặc không có đoạn mã này hệ thống sẽ không cho phép người dùng thao tác với bất kì chức năng nào của hệ thống. Cùng với đó mật khẩu của người dùng cũng được mã hóa sử dụng kĩ thật MD5 trước khi lưu vào cơ sở dữ liệu, tránh để lộ thông tin người dùng.

### Tính dễ bảo trì

Ứng dụng đang trong quá trình xây dựng và phát triển và được thiết kế theo thành 2 phần riêng biệt là server và client. Ứng dụng cho phép khả năng sửa chữa, cập nhật sao cho phù hợp với những yêu cầu thực tế khi sử dụng. Các nâng cấp không hề ảnh hưởng đến sự hoạt động của hệ thống.

## Kết chương

Như vậy, trong chương 2, em đã đưa ra tổng quan các chức năng mà hệ thống cung cấp, đặc tả một số chức năng chính, một số yêu cầu phi chức năng của hệ thống. Tiếp theo ở Chương 3, em sẽ đưa ra các công nghệ sử dụng, ưu nhược điểm của các công nghệ và lý do tại sao lại chọn công nghệ đó để giải quyết vấn đề đã đặt ra.

# Công nghệ sử dụng

Từ những yêu cầu trong Chương 2, Chương 3 này sẽ giới thiệu về công nghệ sử dụng trong hệ thống quản lí và xây dựng chatbot. Hệ thống được xây dựng thành hai phần riêng biệt front-end và back-end. Phần fron-tend sử dụng ReactJs kết hợp Redux, Redux Saga và thư viện giao diện Material-UI. Phần back-end sử dụng Node.js cùng với MongoDB để lưu trữ dữ liệu và Multer để upload file.

## Front-end

### ReactJs

ReactJs (hay React) là một thư viện Javascript mã nguồn mở dùng để xây dựng giao diện người dùng. ReactJs được phát triển bởi Facebook và được Facebook triển khai lần đầu tiên cho ứng dụng Newfeeds vào năm 2011. React hỗ trợ việc xây dựng những thành phần (components) UI có tính tương tác cao, có trạng thái và có thể sử dụng lại được. Một số tính năng được coi là nổi trội khiến em lựa chọn ReactJs:

**JSX** là viết tắt của Javascript XML. Đây là một loại cú pháp mở rộng của JavaScript viết theo kiểu XML. JSX cho phép bạn viết các mã HTML và CSS trong JavaScript đồng thời nó tự động tối ưu hóa mã code khi biên dịch sang mã Javascript. Trong quá trình biên dịch các lỗi cũng sẽ được phát hiện ngay và thông báo cho lập trình viên giúp việc sửa chữa, gỡ lỗi trở nên thuận tiện và dễ dàng hơn.

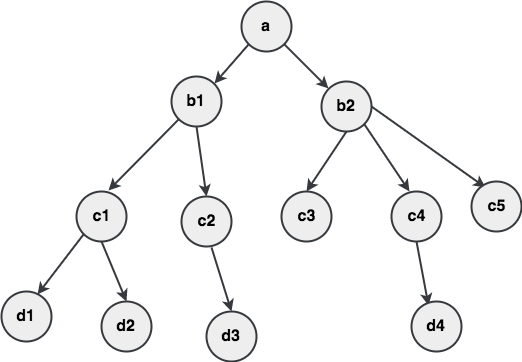
**Component:** ReactJs tổ chức, chia nhỏ các thành phần trong giao diện thành các component và trong các component có thể có các component con. Mỗi component luôn tồn tại 2 khái niệm là thuộc tính truyền vào (props) và trạng thái (state). Trong đó, state là trạng thái riêng của mỗi component, khi state thay đổi thì toàn bộ component đó sẽ được cập nhật lại (render). Props là thuộc tính mô tả những gì component sẽ render. Props chứa dữ liệu được truyền từ component cha xuống component con. Trong một component, ReactJs cho phép nhà phát triển kiếm soát sự thay đổi của state bên trong mỗi component còn props được quy định là giá trị bất biến.

**Virtual DOM:** Virtual DOM là một object Javascript, mỗi object chứa đầy đủ thông tin cần thiết để tạo ra một DOM, bất cứ khi nào có thay đổi xảy ra trong ứng dụng web sẽ tạo ra một bản sao của DOM thực. Mọi sự thay đổ sẽ được thực hiện trên bản sao này. Sau khi thực hiện cập nhật xong, nó kiểm tra sự khác biệt giữa DOM thực và bản sao. Sau đó, DOM thực sẽ chỉ cập nhật những thành phần đã thực sự thay đổi. Điều này làm cho ứng dụng nhanh hơn và không lãng phí bộ nhớ.

Ngoài ra ReactJs còn có 1 tính năng gọi là Luồng dữ liệu một chiều (One-way Data Binding). Tức là ReactJs sẽ thực hiện truyền dữ liệu lần lượt từ component tra xuống component con. Điều này sẽ giúp luồng dữ liệu được kiểm soát một cách tốt hơn. Nhưng cũng mang lại nhiều một số bất cập. Những điểm bất cập em sẽ nói rõ hơn trong phần 3.1.2.

### Redux

Trong phần trước em có nhắc đến 1 số hạn chế với One-way Data Binding. Dữ liệu được truyền từ component cha đến component con thông qua thuộc tính đầu vào (props). Khi muốn truyền ngược từ con sang cha, ta phải truyền thông qua một hàm callback từ con sang cha. Cứ lần lượt như vậy đến khi thay đổi được state ở component mong muốn.

Xét ví dụ: chúng ta có 1 ứng dụng mà các node như trong hình là tượng trưng cho một component được mô hình tree-node.

**Hình 8** Cấu trúc cây thư mục ứng dụng giả định

Giả sử nếu có 1 hành động (action) ở node d3 được kích hoạt và ta muốn thay đổi trạng thái (state) của node d4 và c3 thì luồng dữ liệu sẽ được truyền đi như sau:

Quy trình cập nhật state cho node d4: d3c2b1ab2c4d4

Quy trình cập nhật state cho node c3: d3c2b1ab2c3

Với những bài toán nhỏ thì việc cập nhật state qua lại giữa các componet hoặc các page có thể dễ dàng nhưng nếu ứng dụng lớn hơn có rất nhiều nhánh và node con thì việc thao tác update state qua lại giữa các component trở nên phức tạp hơn khiến cho flow của code cũng khó đọc và khó debug hơn.

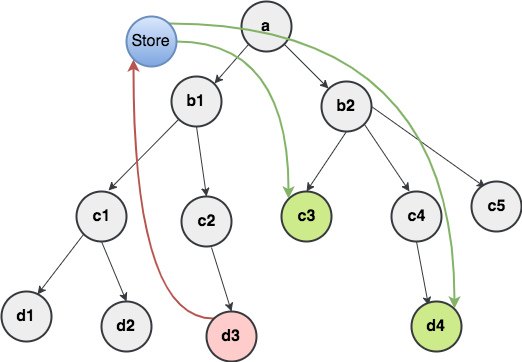
A close up of a clock

Description automatically generated

**Hình 9** Luồng dữ liệu khi sử dụng One-way data binding

Để giải quyết vấn đề trên thì công nghệ được áp dụng vào đó là Redux. Redux là một thư viện JavaScript mã nguồn mở để quản lý trạng thái ứng dụng. Thư viện này hoạt động rất đơn giản. Nó có 1 nơi gọi là store lưu trữ toàn bộ state toàn bộ ứng dụng. Mỗi component có thể truy xuất, thay đổi state được lưu trữ thông qua các action và hoàn toàn trực tiếp thay vì việc phải gọi thông qua props của các component trung gian.

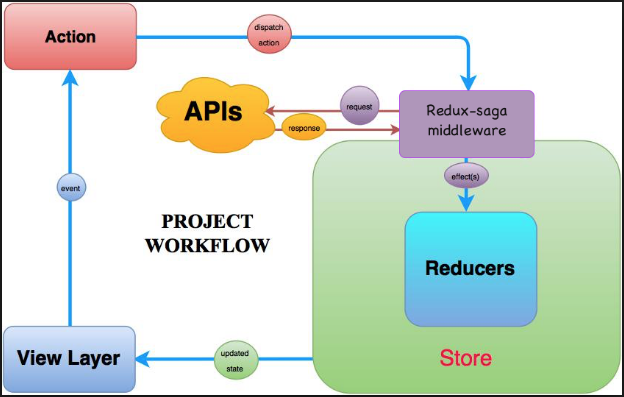
Áp dụng Redux vào ví dụ đã nêu ở trên:

  
**Hình 10** Luồng dữ liệu khi áp dụng Redux

### Redux Saga

Khi ta muốn thực hiện 1 chức năng và thay đổi state trên store của Redux thì ta cần thực hiện thông qua việc gọi (dispatch) các action. Trong thực tế một ứng dụng thường đòi hỏi có những thao tác xử lý cần thời gian để phản hồi (các thao tác bất đồng bộ lấy dữ liệu từ API hay các thao tác đọc ghi file hay đọc Cookie từ trình duyệt…). Các thao tác như vậy trong lập trình hàm gọi là **side effects.** Để có thể giải quyết được các side effect này ta cần thực hiện nó ở middeware.

Redux-Saga là một thư viện Redux middleware, giúp quản lý những side effect trong ứng dụng Redux trở nên đơn giản hơn. Bằng việc sử dụng tối đa tính năng Generator function, nó cho phép ta viết code bất đồng bộ nhìn giống như là code đồng bộ.



**Hình 11** Sơ đồ tổng quan nguyên lí hoạt động của Redux Saga

**Generator function** là một function có khả năng tạm ngưng thực thi trước khi hàm kết thúc, và có thể tiếp tục chạy ở 1 thời điểm khác.

### Material-UI

Material UI là dự án mã nguồn mở được MIT (the Massachusetts Institute of Technology) cấp phép, một thư viện các React Component đã được tích hợp thêm cả Google’s Material Design. Material UI không chỉ mang lại cho trang web một giao diện hoàn toàn mới với những button, textfield, toogle,… được thiết kế theo một phong cách tự do hơn với cách bố trí dạng lưới, các hiệu ứng chiều sâu như ánh sáng, đổ bóng, … mà còn mang lại trải nghiệm tốt cho lập trình viên, giúp việc lập trình trở nên nhanh hơn.

## Back-end

### NodeJs

NodeJs là một nền tảng mã nguồn mở được viết bởi Ryan Dahl vào năm 2009 dựa trên Javascript engine V8 của Google. Về bản chất NodeJs là Javascript. Trước khi NodeJs ra đời, Javascript chủ yếu được sử dụng để nhúng vào HTML sử dụng trên các website cho phép người dùng tương tác với trình duyệt. Với sự ra đời của NodeJs, lập trình viên có thể dùng chạy Javascript bên phía server để xây dựng được các ứng dụng một cách nhanh chóng và dễ dàng mở rộng. Một số ưu điểm nổi bật của NodeJs:

* Tạo được các ứng dụng có tốc độ xử lý nhanh với thời gian thực (Realtime)
* Xây dựng cho các sản phẩm có lượng truy cập lớn, cần mở rộng nhanh
* JSON APIs với cơ chế non-blocking I/O(Input/Output) và mô hình kết hợp với Javascript là sự lựa chọn tuyệt vời cho các dịch vụ Webs làm bằng JSON
* Cộng đồng nodejs rất lớn với nhiều package hoàn toàn miễn phí

**Non-blocking**: Trong javascript, hầu hết các lời gọi I/O đều là non-blocking. Nghĩa là khi có HTTP request, truy xuất dữ liệu trong DB hoặc đọc ghi vào bộ nhớ thì hệ thống sẽ không tạm dừng (blocking) các đoạn code tiếp theo (như các ngôn ngữ server khác PHP, Ryby,...) mà sẽ trao quyền thực thi những lời gọi I/O này cho hệ thống và thực thi những đoạn code tiếp theo, khi hệ thống đã thực thi xong những lời gọi hệ thống này thì hàm callback truyền vào sẽ tự động được gọi.

### ExpressJs

ExpressJs là một Framework nhỏ nhưng linh hoạt được xây dựng trên nền tảng của Nodejs.  Nó cung cấp các tính năng mạnh mẽ để phát triển web hoặc mobile. **Expressjs** hỗ trợ các method HTTP và middleware tạo ra API vô cùng mạnh mẽ và dễ sử dụng. Tổng hợp một số ưu điểm chính của Expressjs như sau:

* Thiết lập các lớp trung gian (middleware) để trả về các HTTP request.
* Định nghĩa rõ ràng routes cho phép sử dụng với các hành động khác nhau dựa trên phương thức HTTP và URL.
* Hỗ trợ mạnh về REST API

Từ những ưu điểm trên và những yêu cầu trong quá trình tìm hiểu ở chương 2 thì ExpressJs là sự lựa chọn hợp lí trong quá trình xây dựng và phát triển đề tài.

### MongoDB

MongoDB là một cơ sở dữ liệu mã nguồn mở và là cơ sở dữ liệu phi quan hệ NoSQL hàng đầu, được sử dụng để lưu các dữ liệu lớn. MongoDB được viết bằng C++. Khác với cách lưu trữ dữ liệu theo bảng, hàng trong cơ sở dữ liệu quan hệ (RDBMS), dữ liệu trong MongoDB được lưu dưới dạng các collection, document. Mỗi một collection được tổ chức dưới dạng một object JSON (trong MongoDB được gọi là BSON) bao gồm các cặp giá trị key-value mà không cần khóa chính, khóa ngoại hay dạng dữ liệu cụ thể. MongoDB hỗ trợ đa nền tảng, nó có thể chạy trên Windows, Linux và Mac... Nó hỗ trợ hầu hết các ngôn ngữ lập trình phổ biến như C#, Java, PHP, Javascript... và các môi trường phát triển khác nhau. Một số ưu điểm của MongoDB:

* Dữ liệu linh hoạt (có thể tuỳ chỉnh số lượng trường, không cố định như cơ sở dữ liệu quan hệ)
* Cấu trúc của một Document rất rõ ràng, dễ đọc và dể hiểu.
* Không cần sử dụng đến các lệnh Join phức tạp.
* Khả năng mở rộng dữ liệu dễ dàng, không cần phải quá bận tâm về kiểu dữ liệu, khóa chính, khóa ngoại như SQL.
* Lưu dữ liệu trên RAM, giúp truy xuất dữ liệu nhanh hơn.

**NoSQL:** là viết tắt của Not Only SQL hay Not SQL. NoSQL là một hệ thống quản lý phân phối (DMS) cho các mô hình dữ liệu phi quan hệ, có cấu trúc linh hoạt, API đơn giản và cũng dễ dàng mở rộng, phù hợp với các ứng dụng hiện đại với số lượng dữ liệu phân tán khổng lồ hay các web thời gian thực. Thuật ngữ này được giới thiệu lần đầu tiên năm 1998 và từ đó được gọi chung cho các “lightweight” cơ sở dữ liệu quan hệ mã nguồn mở nhưng không sử dụng truy vấn SQL. Đến năm 2009, NoSQL được giới thiệu lại trong một hội thảo về CSDL mã nguồn mở phân tán bởi Eric Evans, đã đánh dấu bước phát triển mới cho thế hệ cơ sở dữ liệu: phân tán (distributed) và phi quan hệ (non-relational).

### Rest API

REST (REpresentational State Transfer) được đưa ra vào năm 2000, trong luận văn tiến sĩ của Roy Thomas Fielding (đồng sáng lập giao thức HTTP). Nó là một dạng chuyển đổi cấu trúc dữ liệu, là một phong cách kiến ​​trúc cho việc thiết kế các ứng dụng có kết nối. Nó sử dụng HTTP đơn giản để tạo cho giao tiếp giữa các Phía client và server.

**API** (**A**pplication **P**rogramming **I**nterface) là giao diện lập trình ứng dụng giúp tạo ra các phương thức kết nối với các thư viện và ứng dụng khác nhau.

REST API là một tiêu chuẩn dùng trong việc thết kế các thiết kế API cho các ứng dụng web để quản lý các tài nguyên (resource). RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng (web, mobile…) khác nhau giao tiếp với nhau. Chức năng quan trọng nhất của REST là quy định cách sử dụng các HTTP method như:

* GET: để truy xuất dữ liệu
* POST: để tạo mới dữ liệu
* PUT: để cập nhật cả một dữ liệu
* PATCH: để cập nhật một thành phần, thuộc tính của dữ liệu
* DELETE: để xoá một dữ liệu

### Elasticsearch

**Elasticsearch** là một công cụ tìm kiếm (search engine) kế thừa từ nền tảng Apache Lucene. Nó cung cấp một bộ máy tìm kiếm dạng phân tán, có đầy đủ công cụ với một giao diện web HTTP có hỗ trợ dữ liệu [JSON](https://topdev.vn/blog/json-la-gi/). **Elasticsearch** được phát triển bằng Java và được phát hành dạng mã nguồn mở theo giấy phép Apache. **Elasticsearch** chạy trên server riêng và đồng thời giao tiếp thông qua RESTful do vậy nên nó không phụ thuộc vào client viết bằng gì hay hệ thống hiện tại của bạn viết bằng gì. Nên việc tích hợp nó vào hệ thống bạn là dễ dàng, bạn chỉ cần gửi request http. Ngoài ra, Elasticsearch cũng có thể coi là một document oriented database giống như MongoDB, nó chứa dữ liệu giống như một database và thực hiện tìm kiếm trên những dữ liệu đó. Người dùng cũng có thể sử dụng các chức năng thêm, sửa, xóa như 1 giống như MongoDB.

Đúng với định nghĩa là một công cụ tìm kiếm thì Elasticsearch thường được sử dụng như là một DB thứ 2 chạy song song với 1 DB chính chỉ phục vụ cho mục đích tìm kiếm. Tất cả dữ liệu được lưu vào Elasticsearch đều được đánh chỉ mục (Index), đây là lý do tại sao hiệu năng tìm kiếm của Elasticsearch rất cao. Nhưng khác với chỉ mục thông thường ES sẽ đánh chỉ mục đảo ngược (Inverted Index).

Inverted Index là kĩ thuật thay vì đánh index theo đơn vị document trong MongoDB thì Elasticsearch sẽ tiến hành đánh chỉ mục theo đơn vị tạm gọi là term. Cụ thể hơn, Inverted Index là một cấu trúc dữ liệu, nhằm mục đích map giữa **term,** và**các document chứa term đó.**

**Xét 1 ví dụ để thấy được cách đánh chỉ mục đảo ngược của Elasticsearch. Ta có 3 Document lần lượt là D1, D2, D3.**

**D1 = “This is first document”  
D2 = “This is second one”  
D3 = “one two”**

Inverted Index của 3 documents đó sẽ được lưu dưới dạng như sau:

**“this” => {D1, D2}  
“is” => {D1, D2}  
“first” => {D1}  
“document” => {D1}  
“second” => {D2}  
“one” => {D2, D3}  
“two” => {D3}**

**Sau đó Elasticsearch sẽ sử dụng 2 kĩ thuật đó là** N-Gram và Morphological Analysis để tính toán và đưa ra điểm số (score) cho các kết quả trả về.

**N-Gram** là kĩ thuật tokenize một chuỗi thành các chuỗi con, thông qua việc **chia đều** chuỗi đã có thành các chuỗi con đều nhau, có độ dài là nằm từ 1 đến 3.

**Morphological Analysis** là kỹ thuật xử lý ngôn ngữ tự nhiên (National Language Procesing). Đơn giản là kỹ thuât tách các chuỗi thành từ có nghĩa dựa theo ngôn ngữ.

Một số ưu điểm của Elasticsearch:

* Là phần mềm mã nguồn mở, hoàn toàn miễn phí, cộng đồng phát triển lớn
* Tốc độ nhanh: Elasticsearch có khả năng thực hiện những câu truy vấn phức tập một cách nhanh chóng và cũng có thể lưu lại hầu hết cấu trúc truy vấn vào bộ nhớ đệm để sử dụng cho việc filter kết quả…
* Hỗ trợ Full-text search: với các tính năng như tách từ, tách câu, tạo chỉ mục cho dữ liệu
* Hỗ trợ tìm kiếm mờ, tự động hoàn thành (autocomplete): giúp bạn có thể tìm ra kết quả kể cả khi bạn viết sai chính tả.

## Kết chương

Qua chương 3, em đã trình bày chi tiết những công nghệ mà em đã sử dụng trong quá trình làm ĐATN. Ứng dụng được xây dựng dựa trên ReactJs kết hợp với thư viện giao diện MaterialUI ở phía Client, Node.js cùng cơ sở dữ liệu MongoDB phía server. Trong chương tiếp theo em sẽ trình bày rõ hơn cách áp dụng những công nghệ, cùng kiến trúc trong quá trình xây dựng và phát triển ứng dụng.

# Phát triển và triển khai ứng dụng

Trong chương 3 em đã trình bày những công nghệ để áp dụng vào ĐATN. Chương 4 này sẽ trình bày chi tiết thiết kế kiến trúc cho tới chi tiết của công cụ sử dụng. Với phạm vi và mục tiêu của đề tài đã nêu ra, em sẽ đưa ra những kết quả đã đạt được trong quá trình làm đồ án và xây dựng một số trường hợp kiểm thử.

## Thiết kế kiến trúc

### Thiết kế tổng quan

A close up of a map

Description automatically generated

**Hình 12** Kiến trúc tổng quan

Thiết kế tổng qua của ứng dụng được biểu diện trong hình 12. Đây là kiến trúc được sử dụng phổ biến trong những năm gần đây. Một ưu điểm không thể không kể đến của kiến trúc này là sự tách biệt rõ ràng giữa front-end và back-end. Theo đó front-end và back-end chỉ giao tiếp với nhau thông qua API. Nhờ độc lập giữa 2 bên nên việc nâng cấp cũng như mở rộng của mỗi bên đều không ảnh hưởng đến bên còn lại. Front-end sử dụng thêm công nghệ mới nhằm tối ưu hóa UI/UX, nâng cao trải nghiệm người dùng những vấn không hề ảnh hưởng đến back-end. Tương tự đối với back-end, sự thay đổi trong việc áp dụng quy trình nghiệp vụ hay sự tối ưu những logic phức tạp vẫn đảm bảo được không ảnh hưởng đến front-end.

Như đã trình bày trong chương 3, phía front-end sử dụng ReactJs để xử lí phần UI của ứng dụng. Kết hợp với ReactJs là Redux và Redux-Saga để quản lí state. NodeJs, ExpressJs được sử dụng để xử lí logic và thao tác với cơ sở dữ liệu MongoDB và Elasticsearch. Hai thành phần giao tiếp với nhau thông qua REST API.

### Kiến trúc front-end

**A close up of a logo

Description automatically generated**

**Hình 13** Kiến trúc front-end

ReactJs là thư viện hướng theo cấu trúc component-base, các thành phần giao diện được tổ chức thành các component, trạng thái (state) của cả ứng dụng được Redux quản lí ở store thông qua reducer, khi có một sự kiện (event) của ứng dụng được ký hiệu là một hành động (Action), hay nói cách khác, tương tác của người dùng trên giao diện làm kích hoạt (dispatch) một action. Action có thể chỉ làm thay đổi trạng thái của ứng dụng đơn thuần hoặc action có thể gọi lên server để thực hiện 1 hành động. Với action có tương tác với server, sau khi API được gọi dữ liệu trả về sẽ được thông qua middleware (redux-saga). Từ những dữ liệu trả về từ server, middleware sẽ thay đổi state của ứng dụng. State sau khi đã được thay đổi sẽ kéo theo view thay đổi.

Cách tổ chức kiến trúc như trên cho công cụ giúp việc mở mở rộng, phát triển thêm các chức năng cho sản phẩm dễ dàng hơn, tái sử dụng được các component đã có, đồng thời cũng giúp việc quản lí mã nguồn được khoa học, rõ ràng, dễ hiểu.

### Kiến trúc back-end

A picture containing text, drawing

Description automatically generated

**Hình 14** Mô hình MVC

Đã là một lập trình viên thì không thể không biết đến MVC hay mô hình MVC. MVC là từ viết tắt của Model-View-Controller. Đây là một mẫu kiến ​​trúc, mô hình lập trình phổ biến được sử dụng để tạo cấu trúc cho nhiều trang web, ứng dụng tiên tiến. Trong đó View là phần hiển thị cho người dùng, cho phép người dùng tương tác với hệ thống. Nói một cách đơn giản, View chính là những gì người dùng có thể nhìn thấy và tương tác. Controller đảm nhận việc điều hướng các yêu cầu (request) từ người dùng và gọi đúng phương thức xử lý trong Model và cuối cùng trả kết quả lại cho View. Cuối cùng là Model, đây là nơi chứa nghiệp vụ logic, các phương thức xử lý dữ liệu, truy xuất dữ liệu từ database và trả ra cho Controller. Model độc lập với giao diện người dùng (View). Mục đích quan trọng nhất của Model là kết nối cơ sở dữ liệu.

Hình 12 phía trên đã chỉ rõ cách áp dụng mô hình MVC trong việc xây dựng ứng dụng. Với việc thiết kế API server thì phần View hiển thị không có. Do back-end sử dụng framework ExpressJs nên việc thiết kế kiến trúc back-end như trên cũng một phần lớn phải phụ thuộc kiến trúc của framework. Cụ thể back-end sẽ có thêm 2 thành phần là Middleware và Routes. Trong ExpressJs, Middleware là một phần rất quan trọng, nó giúp chúng ta có thể kiểm tra tính xác thực của người dùng, lọc và ngăn chặn các request xấu được client gửi đến cho server trong quá trình chạy của ứng dụng. Sau khi request được xác thực và kiểm tra không phải là 1 request xấu. Request sẽ được truyền đến Route. Route như 1 bộ phận định tuyến, nó sẽ quy định Controller sẽ thực hiện và xử lí ứng với request từ client. Ngoài ra, back-end còn có 1 thành phần là Services. Services là nơi chứa logic nghiệp vụ (Business Logic). Model trong mô hình MVC chính là “Services” và “Models” trong kiến trúc của ứng dụng. Việc tách nhỏ Model ra thành 2 phần, 1 phần là xử lí nghiệp vụ và 1 là phần thao tác với cơ sở dữ liệu điều này vừa đảm bảo nhiệm vụ chính của Model là kết nối với cơ sở dữ liệu, vừa có thể tái sử dụng mã nguồn, một Controller có thể gọi nhiều Service khác nhau.

## Thiết kế chi tiết

### Thiết kế chi tiết 1 số use case chính

Do phạm vi báo cáo có giới hạn nên mục này em chỉ trình bày biểu đồ trình tự của 3 use case chính đã nêu ra ở phần 2.3 là: Sửa nội dung chuỗi hành động, nhắn tin, gửi tin nhắn cho nhiều người dùng.

**Hình 15** là biểu đồ trình tự Sửa nội dung chuỗi hành động. Chuỗi hành động (block) bao gồm nhiều các hành động con (element) như văn bản, hình ảnh, thẻ chọn, câu hỏi lưu giá trị và âm thanh, video dưới dạng đường dẫn.

**Hình 16** biểu diễn biều đồ trình tự thể hiện quá trình server xử lí trả ra tin nhắn phù hợp sau khi nhận được tin nhắn của bên thứ 3.

**Hình 17** biểu diễn biều đồ trình tự của use case gửi tin nhắn cho nhiều người.

A screenshot of a cell phone

Description automatically generated

**Hình 15** Biểu đồ trình tự Sửa nội dung chuỗi hành động

A close up of a map

Description automatically generated

**Hình 16** Biểu đồ trình tự Nhắn tin

A close up of a map

Description automatically generated

**Hình 17** Gửi tin nhắn cho nhiều người

### Thiết kế giao diện front-end

A close up of text on a white background

Description automatically generated

**Hình 18** Thiết kế thành phần giao diện

Giao diện của ứng dụng sẽ gồm các thành phần như trong hình 18. Ứng dụng bao gồm 8 trang chính có khả năng cung cấp các chức năng cần thiết cho 1 ứng dụng quản lí chuỗi hội thoại. SignIn và SignUp cho phép người quản trị viên có thể đăng nhập hoặc đăng kí để có thể bắt đầu sử dụng các chức năng chính bên trong. NotFound là trang trả ra mặc định khi người dùng nhập sai một đường dẫn mà ứng dụng không cung cấp. Ngay sau khi đăng nhập người dùng sẽ đến với sảnh chờ (Lobby - page). Tại đây, người dùng có thể thấy danh sách những chatbot (Bot - component) đang hoạt động, có thể thêm hoặc xóa bot. Trang chủ (Home - page) hiển thị những thông tin cơ bản và cần thiết của chatbot đó như tên, mô tả, danh sách người theo dõi số phiên,... Người dùng có thể quản lí chuỗi hành động (Block) trong trang DashBoardBot. Trang này bao gồm 1 danh sách các chuỗi hành động đã tạo (ListBlock - component) và nội dung cụ thể (ContentBlock - component) của chuỗi hành động đó. Nội dung chuỗi hành động có thể bao gồm nhiều dạng hành động con khác nhau như: văn bản, hình ảnh, âm thanh, video, thẻ chọn, câu hỏi lấy dữ liệu người dùng. Tiếp đến là trang quản lí các luật (Rule - page). Trang này sẽ hiển thị ra danh sách các luật (LineRule - component) với những từ khóa và chuỗi hành động được thiết lập. Người dùng cũng có thể thêm, sửa, xóa các luật. Cuối cùng là trang tổng hợp và phân tích dữ liệu người dùng (Analyze - page). Danh sách người dùng có sử dụng trực tiếp chatbot thông qua nền tảng thứ 3 sẽ hiển thị tại đây. UserModal component trong trang này đóng vai trò là nơi hiển thị thông tin chi tiết của người dùng. Ngoài ra trang này cũng cho phép người quản trị thực hiện gửi những tin nhắn nhiều người dùng. Theo hình 18 thì 4 trang bao gồm: Home, DashBoardBot, Analyze, Rule đều có SideBar và Topbar. Trong đó Topbar để hiển thị người đang làm việc, bot đang được chỉnh sửa và button điều hướng về Lobby. SideBar - thanh bên trái giao diện có vai trò như 1 thanh menu điều hướng giúp người dùng có thể chuyển đổi các qua lại các trang một cách linh hoạt.

### Thiết kế thành phần back-end

A close up of a map

Description automatically generated

**Hình 19** Thành phần back-end

Hình 18 biểu diễn chi tiết các thành phần của back-end đã nêu ở mục 4.1.3. Nhìn vào hình 18 Routes sẽ là nơi tiếp nhận những request của người dùng. Từ nội dung đường dẫn yêu cầu (endpoint), phương thức gửi yêu cầu có trong request đó, route sẽ quyết định Controller sẽ thực hiện xử lí. Controllers quản lý những HTTP request, bóc tách dữ liệu “thô” từ request sau đó sẽ phân chia nhiệm vụ và đưa ra quyết định Service nào thực hiện nó và trả về response tương ứng sau khi đã xử lí. Services chỉ nhận dữ liệu liên quan để thực hiện chức năng chuyên biệt, tương tác trực tiếp với Models và Service khác. Cuối cùng Services sẽ định nghĩa cấu trúc dữ liệu trả ra cho Controller.

### Thiết kế cơ sở dữ liệu

A close up of a map

Description automatically generated

**Hình 20** Biểu đồ thực thể liên kết của ứng dụng

Chi tiết các thuộc tính và vai trò của các thuộc tính trong từng thực thể được thể hiện trong bảng các bảng dưới đây.

**Bảng 8** Chi tiết thuộc tính của Account

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Note | Giải thích |
| \_id | ObjectId |  | Mã id tự sinh trong MongoDB |
| firstName | String | Bắt buộc | Họ |
| lastName | String | Bắt buộc | Tên |
| userEmail | String | Bắt buộc | Email |
| password | String | Bắt buộc  Mã hóa | Mật khẩu |

**Bảng 9** Chi tiết thuộc tính của Bot

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Note | Giải thích |
| \_id | ObjectId |  | Mã id tự sinh trong MongoDB |
| name | String | Bắt buộc | Tên OA |
| description | String |  | Mô tả OA |
| oa\_id | String | Bắt buộc | Mã OA |
| tokenApp | String | Bắt buộc | Token của OA |
| user\_id | ObjectId | Bắt buộc | Id của quản trị viên |
| avatar | String |  | Ảnh avatar của OA |
| cover | String |  | Ảnh bìa của OA |
| deleteFlag | Boolean | Giá trị mặc định là false | Cờ báo hiệu đã xóa |

**Bảng 10** Chi tiết thuộc tính của Group

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Note | Giải thích |
| \_id | ObjectId |  | Mã id tự sinh trong MongoDB |
| name | String | Bắt buộc | Tên nhóm chuỗi hành động |
| defaultGroup | Boolean |  | Cờ đánh dấu nhóm hành động mặc định |
| blocks | Array |  | Mảng các ObjectId của chuỗi hành động |
| bot\_id | ObjectId | Bắt buộc | Id chatbot |
| deleteFlag | Boolean | Giá trị mặc định là false | Cờ báo hiệu đã xóa |

**Bảng 11** Chi tiết thuộc tính của Block

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Note | Giải thích |
| \_id | ObjectId |  | Mã id tự sinh trong MongoDB |
| name | String | Bắt buộc | Tên nhóm hành động |
| elements | Array |  | Mảng chứa ObjectId của hành động con |
| group\_id | ObjectId | Bắt buộc | Id của nhóm chuỗi hành động |
| deleteFlag | Boolean | Giá trị mặc định là false | Cờ báo hiệu đã xóa |

**Bảng 12** Chi tiết thuộc tính của Element

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Note | Giải thích |
| \_id | ObjectId |  | Mã id tự sinh trong MongoDB |
| element\_type | String | Bắt buộc  Giá trị mặc định thuộc các giá trị sau đây: TEXT, IMAGE, AUDIO, VIDEO, LIST, DATA\_CUSTOM | Loại hành động con |
| attachment\_msg | Object |  | Nội dung media của hành động con |
| text\_msg | String |  | Nội dung văn bản của hành động con |
| attribute | String | Giá trị mặc định là null  Chỉ được sẽ giá trị khi loại của hành động con là DATA\_CUSTOM | Tên thuộc tính sẽ lưu |
| block\_id | ObjectId | Bắt buộc | Id của chuỗi hành động |
| deleteFlag | Boolean | Giá trị mặc định là false | Cờ báo hiệu đã xóa |

**Bảng 13** Chi tiết thuộc tính của Rule

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Note | Giải thích |
| \_id | ObjectId |  | Mã id tự sinh trong MongoDB |
| keyword | String | Bắt buộc | Từ khóa dùng để phân tích tin nhắn |
| blocks | Array |  | Một mảng các chuỗi hành động có thể trả về |
| name | String | Bắt buộc | Tên của luật |
| bot\_id | ObjectId | Bắt buộc | Id của chatbot |
| deleteFlag | Boolean | Giá trị mặc định là false | Cờ báo hiệu đã xóa |

**Bảng 14** Chi tiết thuộc tính của Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Note | Giải thích |
| \_id | ObjectId |  | Mã id tự sinh trong MongoDB |
| name | String | Bắt buộc | Tên của thuộc tính |
| value | String | Bắt buộc | Giá trị thực tế lấy được từ người dùng |
| user\_id | ObjectId | Bắt buộc | Id của người dùng trong hệ thống |
| user\_app\_id | String | Bắt buộc | Id của người dùng trong nền tảng thứ 3 |
| bot\_id | ObjectId | Bắt buộc | Id của chatbot |
| deleteFlag | Boolean | Giá trị mặc định là false | Cờ báo hiệu đã xóa |

**Bảng 15** Chi tiết thuộc tính của User

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Note | Giải thích |
| \_id | ObjectId |  | Mã id tự sinh trong MongoDB |
| name | String | Tên của người dùng lấy từ nền tảng bên thứ 3 | Tên người dùng |
| user\_app\_id | String | Bắt buộc | Id của người dùng trong nền tảng thứ 3 |
| avatar | String |  | Đường dẫn đến ảnh avatar của người dùng.  Đường dẫn lấy được từ nền tảng thứ 3 |
| bot\_id | ObjectId | Bắt buộc | Id của chatbot |
| current\_session | String |  | Đoạn mã server sinh để kiểm soát phiên (session) nhắn tin của người dùng |
| old\_session | Array |  | Một mảng những mã session cũ. |
| element\_id | ObjectId |  | Mã Id của hành động con |

**Bảng 16** Chi tiết thuộc tính của BadMessage

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Note | Giải thích |
| \_id | ObjectId |  | Mã id tự sinh trong MongoDB |
| user\_id | ObjectId | Bắt buộc | Id của người dùng trong hệ thống |
| user\_name | String |  | Tên người dùng |
| message\_id | String | Bắt buộc | Id của tin nhắn lấy được từ bên thứ 3 |
| event\_name | String | Bắt buộc | Tên của sự kiện người dùng gửi tin nhắn |
| deleteFlag | Boolean | Giá trị mặc định là false | Cờ báo hiệu đã xóa |

## Xây dựng ứng dụng

### Thư viện và công cụ sử dụng

Trong quá trình xây dựng công cụ quản lý và hành động, em có sử dụng một số công cụ để hỗ trợ như ở Bảng 17 bên dưới.

**Bảng 17** Danh sách thư viện và công cụ sử dụng

|  |  |  |
| --- | --- | --- |
| **Mục đích** | **Công cụ và thư viện** | **Địa chỉ URL** |
| IDE lập trình | Visual Studio Code 64 bit | https://code.visualstudio.com/ |
| Cơ sở dữ liệu lưu trữ | MongoDB | https://www.mongodb.com/ |
| Cơ sở dữ liệu tìm kiếm | Elasticsearch | https://www.elastic.co/ |
| Front-end framework | ReactJS | https://reactjs.org/ |
| Thư viện UI component | Material-UI | https://material-ui.com/ |
| Thư viện quản lí state | Redux | https://redux.js.org/ |
| Middleware ReactJs | Redux-Saga | https://redux-saga.js.org/ |
| Back-end framework | ExpressJs | https://expressjs.com/ |
| Kho mã nguồn | Github | https://github.com/ |

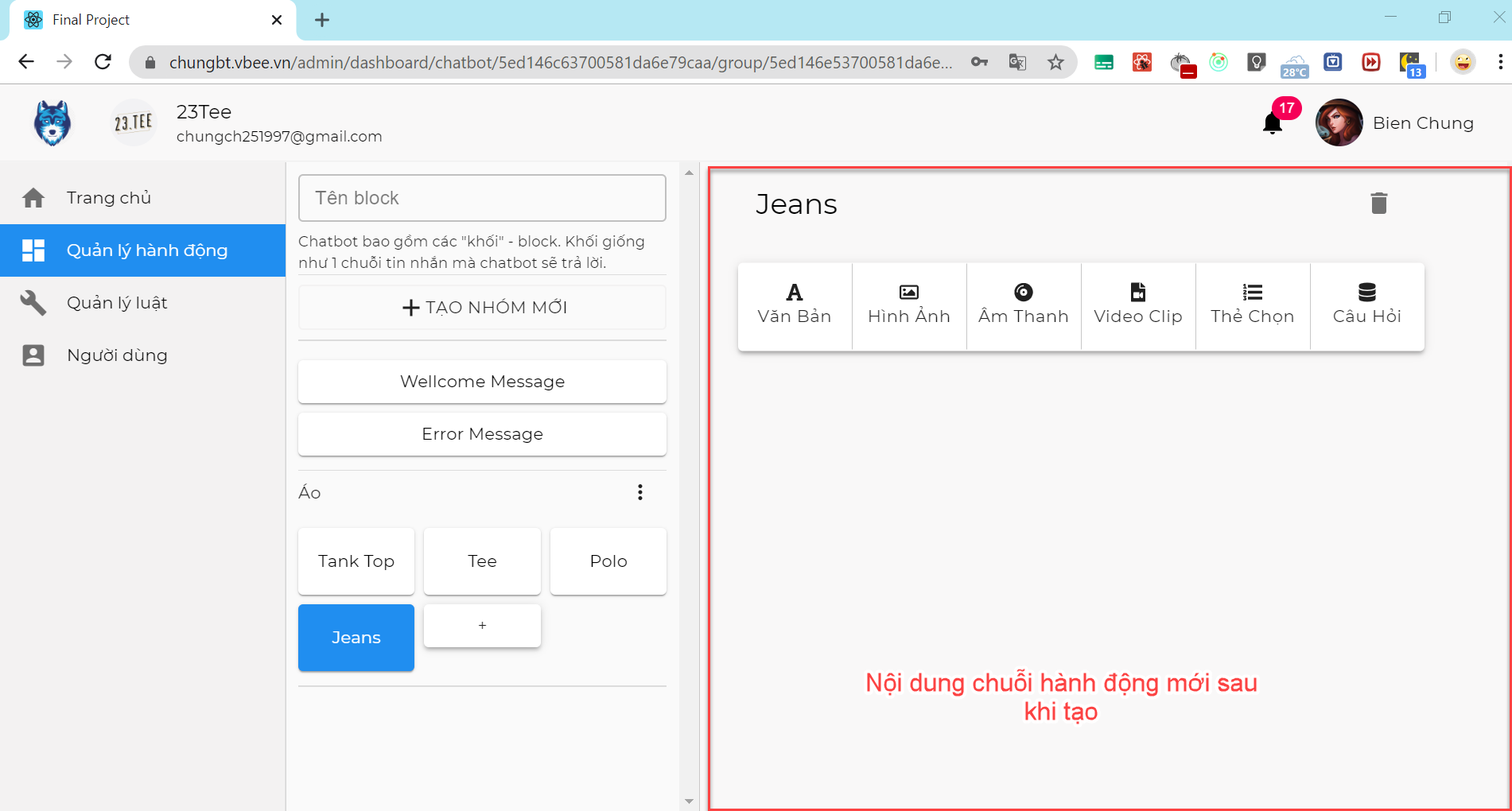
### Kết quả đạt được

Đồ án đã xây dựng được một công cụ trực quan giúp người quản lí doanh nghiệp có thể dễ dàng sử dụng với những chức năng chính: (i) quản lí chatbot, (ii) quản lí chuỗi hành động, (iii) quản lí luật (iv) xem thông tin người dùng. Ứng dụng hoàn thiện với hơn 37000 dòng code đối với cả front-end và back-end. Hiện tại người dùng có thể sử dụng ứng dụng theo đường dẫn https://chungbt.vbee.vn/.

### Minh hoạ các chức năng chính

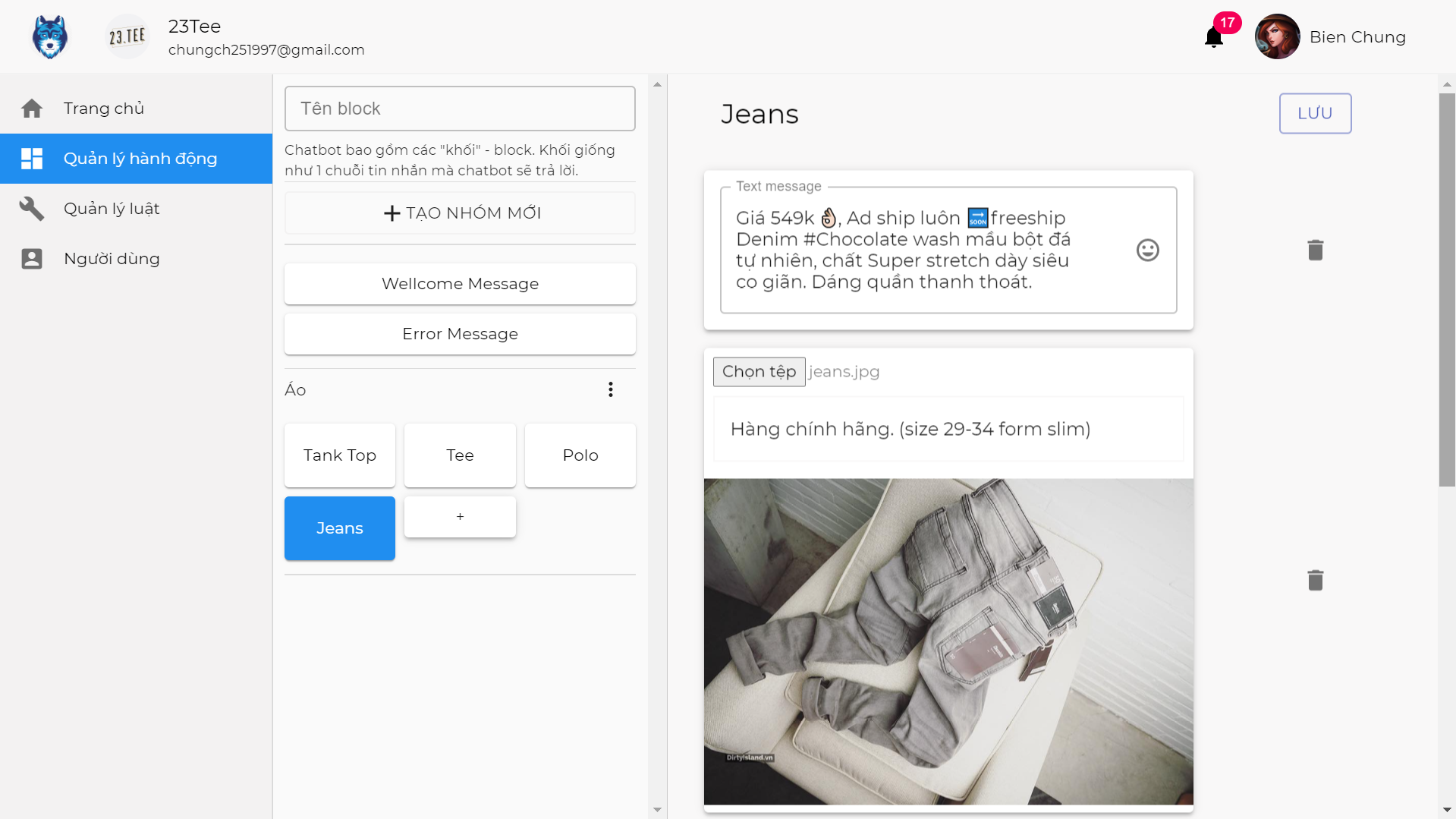
Trong phần này, em sẽ minh họa một số chức năng chính của ứng dụng đã xây dựng do phạm vi của báo cáo có giới hạn. Ba chức năng chính sẽ được trình bày là: (i) sửa nội dung chuỗi hành động, (ii) nhắn tin (sử dụng nền tảng thứ 3), (iii) nhắn tin cho nhiều người.

#### Sửa nội dung chuỗi hành động



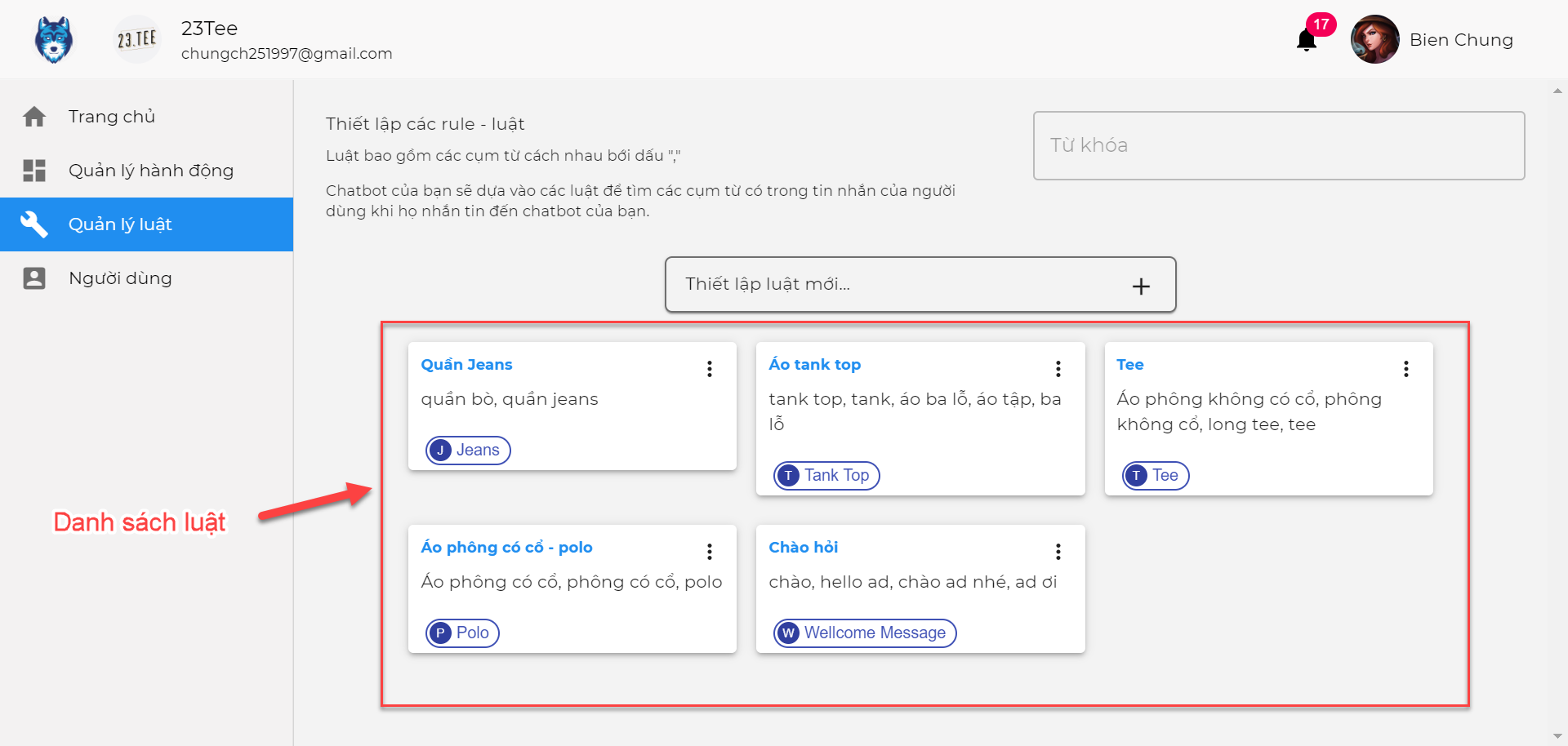
**Hình 21** Giao điện sau khi tạo mới 1 chuỗi hành động

Tại màn hình Quản lý hành động (DashBoardBot), người quản trị viên sau khi tạo mới 1 chuỗi hành động thì bên phải màn hình sẽ hiển thị nội dung của chuỗi hành động mới là rỗng. Người quản trị có thể thêm các chuỗi hành động con nhờ bảng chọn. Các loại hành động con mà ứng dụng cung cấp bao gồm: văn bản, hình ảnh, âm thanh, video, thẻ chọn, câu hỏi. Sau khi đã thêm các hành động con, người dùng cũng có thể sửa nội dung hoặc xóa những hành động con. Cuối cùng để kết thúc hành động Sửa nội dung chuỗi hành động, người dùng ấn nút “Lưu” để client yêu cầu server cập nhật lại nội dung cho chuỗi hành động mới.



**Hình 22** Giao điện sau khi sửa nội dung chuỗi hành động

#### Nhắn tin



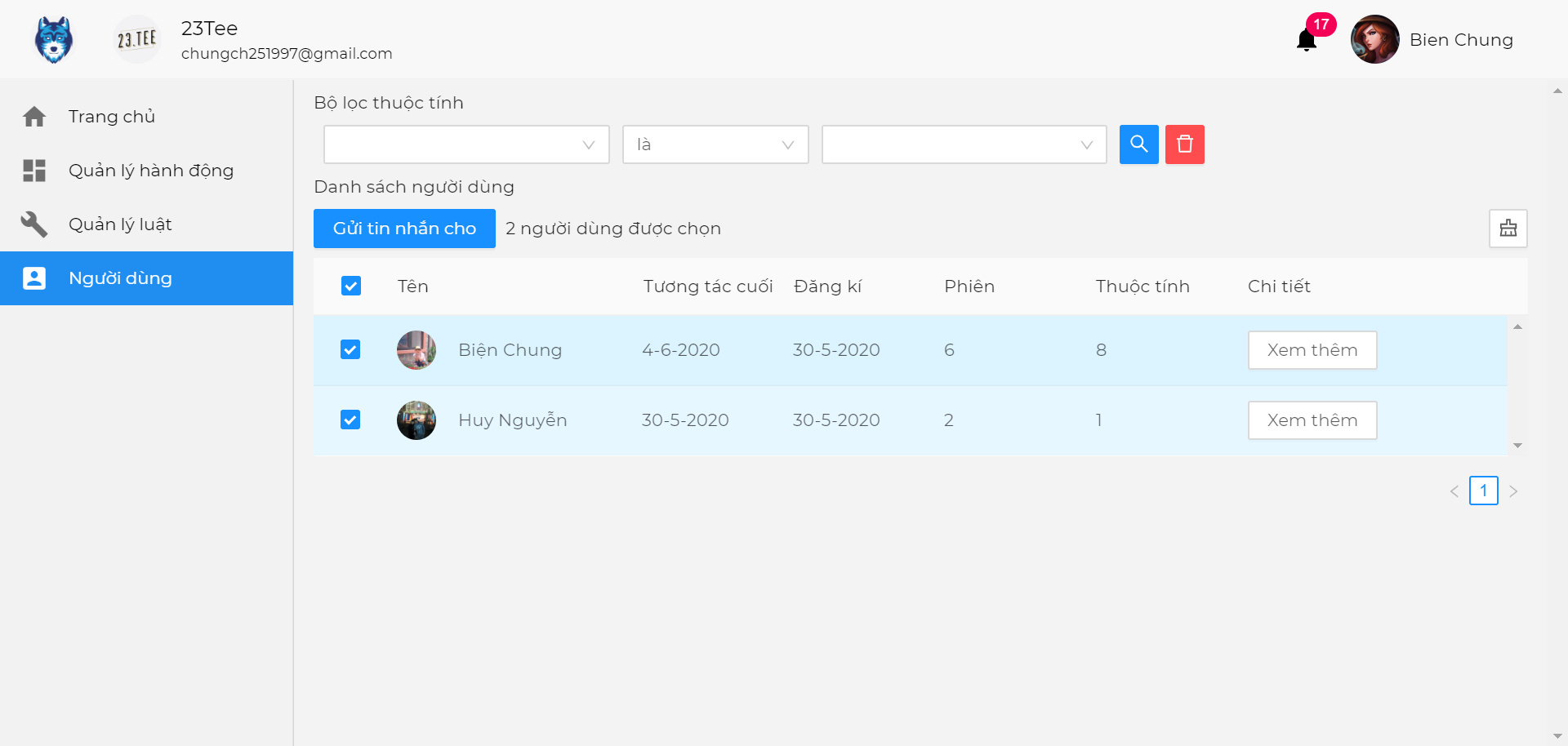
**Hình 23** Giao điện danh sách luật đã được thiết lập

Vì ứng dụng phân tích tin nhắn của người dùng dựa trên các luật nên cần thiết lập thành công các luật cho ứng dụng. Để kiểm tra người dùng có thể nhắn tin chứa từ khóa đã thiết lập cho cửa hàng đã liên kết với chatbot.



**Hình 24** Chuỗi hành động trả về khi nhắn tin

#### Gửi tin nhắn cho nhiều người

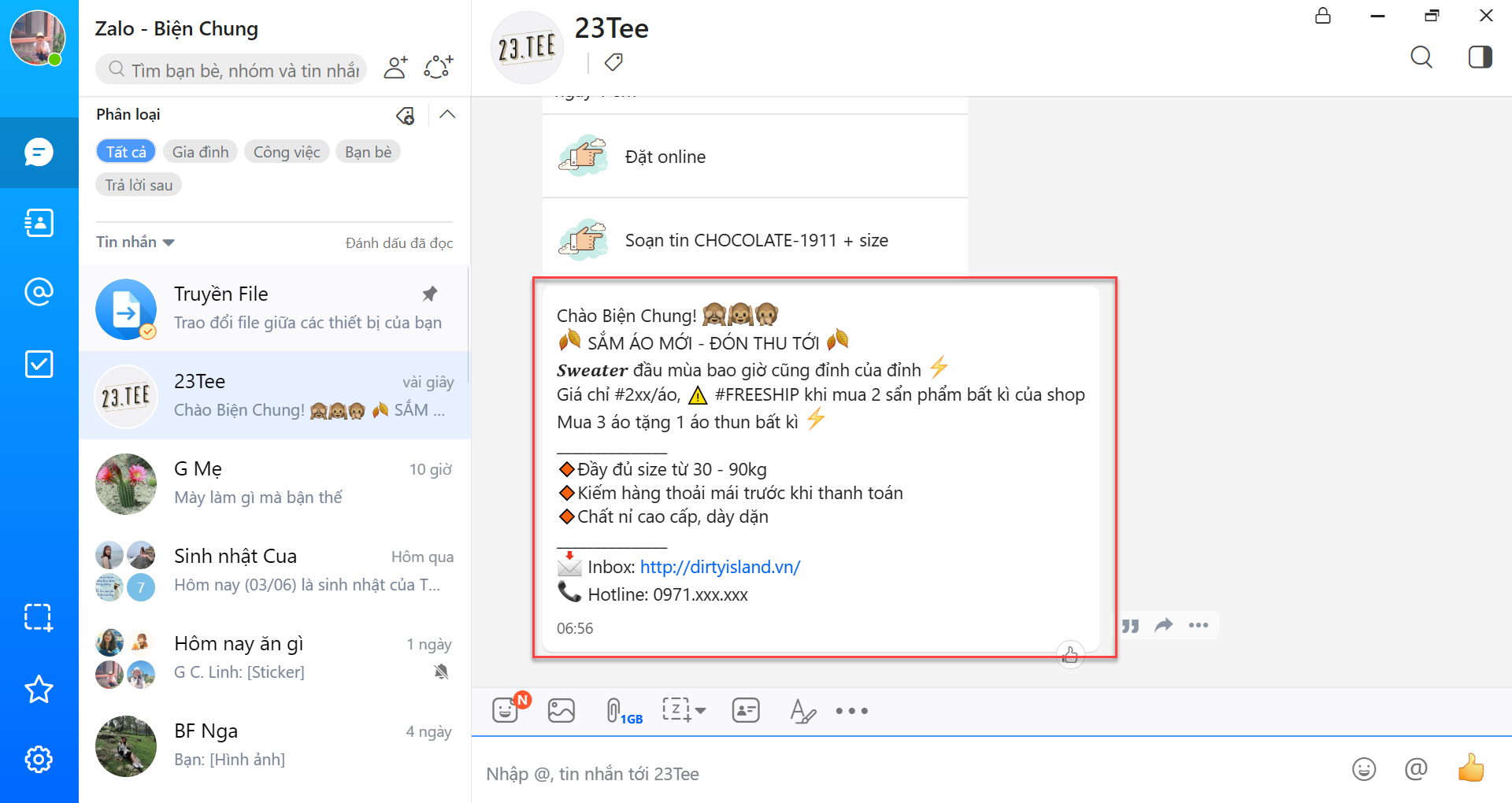


**Hình 25** Giao diện trang quản lí người dùng

Tại trang quản lí người dùng, người quản trị viên sẽ chọn danh sách những người muốn gửi tin nhắn. Sau đó ấn vào nút “Gửi tin nhắn cho” để bắt đầu thực hiện việc nhập tin nhắn. Người dùng có thể chọn loại tin nhắn muốn gửi. Đầu tiên người dùng có thể chọn tin nhắn dưới dạng văn bản. Sau đó có thể nhập nội dung muốn gửi vào textbox ở cửa sổ hiện ra.

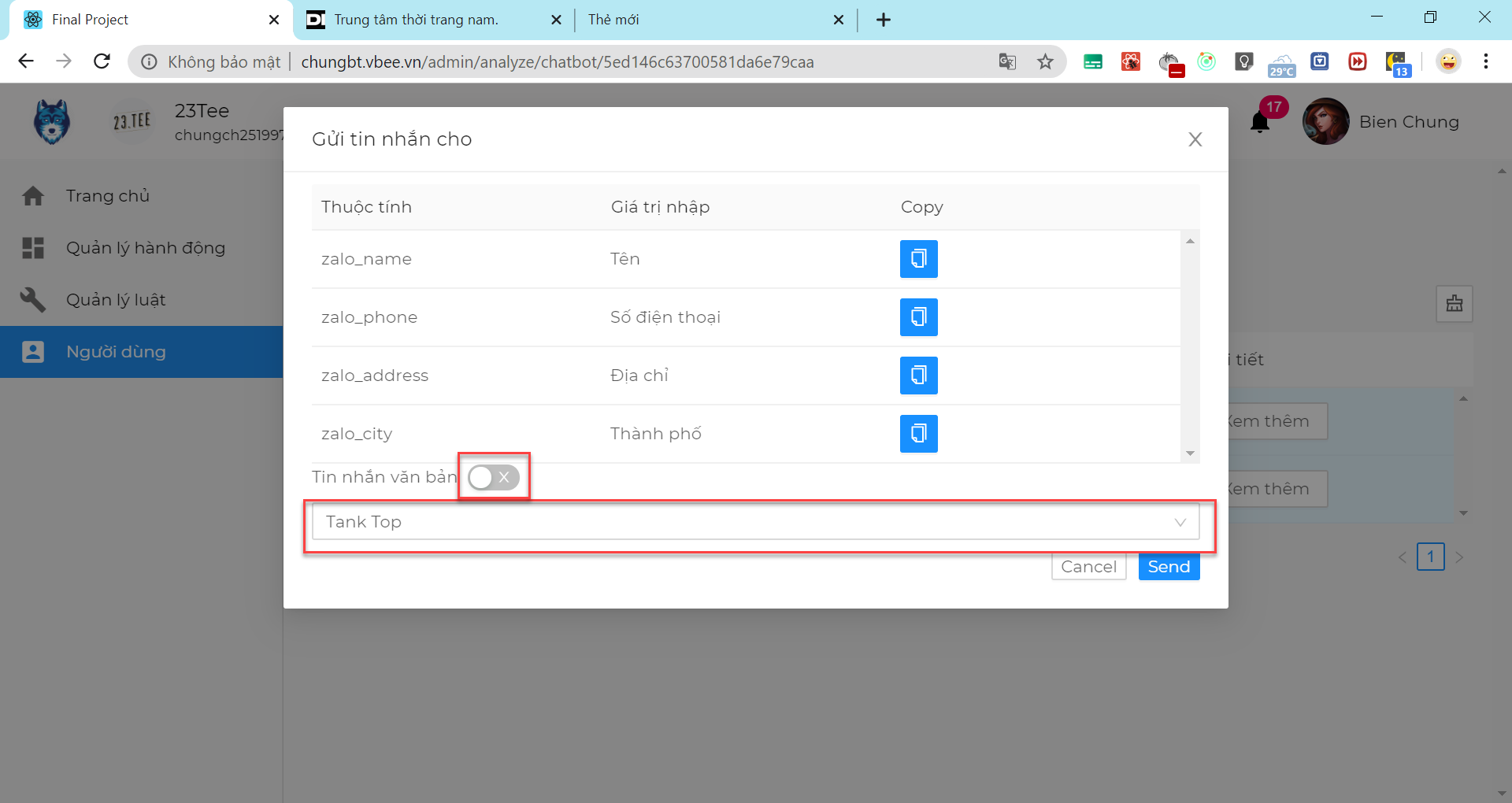


**Hình 26** Giao diện gửi tin nhắn dưới dạng văn bản thông thường



**Hình 27** Kết quả khi sau khi gửi tin nhắn dạng văn bản

Nếu không muốn gửi tin nhắn dạng văn bản người dùng cũng có thể chuyển đổi qua lại thông qua nút công tắc ở ngay phía trên textbox nhập văn bản. Sau đó người dùng có thể chọn bất kì 1 chuỗi hành động đã được thiết định trước đó và ấn gửi. Những người trong danh sách nhận được tin nhắn sẽ nhận được chuỗi hành động đó.



**Hình 28** Giao diện gửi tin nhắn cho nhiều người dưới dạng chuỗi hành động



**Hình 29** Kết quả khi sau khi gửi tin nhắn dạng chuỗi hành động

## Kiểm thử

Phần này sẽ sử dụng kĩ thuật kiểm thử hộp đen cho những chức năng quan trọng nhất của công cụ như: (i) sửa nội dung chuỗi hội thoại, (ii) nhắn tin, (iii) gửi tin nhắn cho nhiều người.

**Bảng 18** Kiểm thử chức năng Sửa nội dung chuỗi hành động

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Mô tả | Quy trình test | Kết quả mong muốn | Kết quả thực tế | Trạng thái |
| 1 | Sửa nội dung chuỗi hành động  (Thêm hành động con) | Vào trang Quản lí hành động | Thông báo “Lưu thành công”  Thêm thành công hành động con vào chuỗi hành động | Thông báo “Lưu thành công”  Thêm thành công hành động con vào chuỗi hành động | Pass |
| Thực hiện chọn chuỗi hành động muốn sửa |
| Thực hiện thêm hành động con |
| Điền thông tin cho hành động con mới tạo |
| Ấn nút “Lưu” |
| 2 | Sửa nội dung chuỗi hành động  (Xóa hành động con) | Vào trang Quản lí hành động | Thông báo “Xóa thành công”  Xóa thành công hành động con trong chuỗi hành động | Thông báo “Xóa thành công”  Xóa thành công hành động con trong chuỗi hành động | Pass |
| Thực hiện chọn chuỗi hành động muốn sửa |
| Thực hiện xóa hành động con |
| Ấn nút “Xác nhận xóa” |
| 3 | Sửa nội dung chuỗi hành động  (Sửa nội dung hành động con) | Vào trang Quản lí hành động | Thông báo “Lưu thành công”  Nội dung của hành động con được cập nhật | Thông báo “Lưu thành công”  Nội dung của hành động con được cập nhật | Pass |
| Thực hiện chọn chuỗi hành động muốn sửa |
| Sửa thông tin của hành động con |
| Ấn nút “Lưu” |
| 4 | Sửa nội dung chuỗi hành động  (Sửa nội dung hành động con, nhưng không ấn lưu) | Vào trang Quản lí hành động | Nội dung chuỗi hành động vẫn được giữ nguyên | Nội dung chuỗi hành động vẫn được giữ nguyên | Pass |
| Thực hiện chọn chuỗi hành động muốn sửa |
| Thực hiện sửa nội dung chuỗi hành động |
| Ấn chọn chuỗi hành động khác |
| 5 | Sửa nội dung chuỗi hành động  (Sửa nội dung hành động con với dạng là thẻ chọn) | Vào trang Quản lí hành động | Đóng cửa sổ  Thêm thành công mới 1 lựa chọn vào trong hành động con đó | Đóng cửa sổ  Thêm mới 1 lựa chọn vào trong hành động con đó | Pass |
| Thực hiện chọn chuỗi hành động muốn sửa |
| Thực hiện sửa hành động con có dạng là thẻ chọn |
| Chọn “Thêm mới lựa chọn” |
| Nhập đầy đủ thông tin |
| Ấn “Thêm” |
| 6 | Sửa nội dung chuỗi hành động  (Sửa nội dung hành động con với dạng là thẻ chọn) | Vào trang Quản lí hành động | Cừa sổ giữ nguyên  Thông báo “Bạn cần nhập đầy đủ thông tin” | Cừa sổ giữ nguyên  Thông báo “Bạn cần nhập đầy đủ thông tin” | Pass |
| Thực hiện chọn chuỗi hành động muốn sửa |
| Thực hiện sửa hành động con có dạng là thẻ chọn |
| Chọn “Thêm mới lựa chọn” |
| Nhập thiếu thông tin |
| Ấn “Thêm” |
| 7 | Sửa nội dung chuỗi hành động  (Sửa nội dung hành động con với dạng là thẻ chọn) | Vào trang Quản lí hành động | Đóng cửa sổ  Thêm thành công mới lựa chọn url vào trong hành động con đó | Đóng cửa sổ  Thêm thành công mới lựa chọn url vào trong hành động con đó | Pass |
| Thực hiện chọn chuỗi hành động muốn sửa |
| Thực hiện sửa hành động con có dạng là thẻ chọn |
| Chọn “Thêm mới lựa chọn” |
| Chọn loại button là đường dẫn URL |
| Nhập đường dẫn thực tế (đúng) |
| Ấn “Thêm” |
| 8 | Sửa nội dung chuỗi hành động  (Sửa nội dung hành động con với dạng là thẻ chọn) | Vào trang Quản lí hành động | Cửa sổ giữ nguyên  Thông báo “Đường dẫn URL không đúng” | Cửa sổ giữ nguyên  Thông báo “Đường dẫn URL không đúng” | Pass |
| Thực hiện chọn chuỗi hành động muốn sửa |
| Thực hiện sửa hành động con có dạng là thẻ chọn |
| Chọn “Thêm mới lựa chọn” |
| Chọn loại button là đường dẫn URL |
| Nhập đường dẫn không đúng định dạng là 1 url. |
| Ấn “Thêm” |
| 9 | Sửa nội dung chuỗi hành động  (Sửa nội dung hành động con với dạng là thẻ chọn) | Vào trang Quản lí hành động | Đóng cửa sổ  Thêm thành công mới lựa chọn sms hoặc số điện thoại vào trong hành động con đó | Đóng cửa sổ  Thêm thành công mới lựa chọn sms hoặc số điện thoại vào trong hành động con đó | Pass |
| Thực hiện chọn chuỗi hành động muốn sửa |
| Thực hiện sửa hành động con có dạng là thẻ chọn |
| Chọn “Thêm mới lựa chọn” |
| Chọn loại button là SMS hoặc Số điện thoại |
| Nhập số điện thoại thực tế |
| Ấn “Thêm” |
| 10 | Sửa nội dung chuỗi hành động  (Sửa nội dung hành động con với dạng là thẻ chọn) | Vào trang Quản lí hành động | Cửa sổ giữ nguyên  Thông báo “Số điện thoại không hợp lệ” | Cửa sổ giữ nguyên  Thông báo “Số điện thoại không hợp lệ” | Pass |
| Thực hiện chọn chuỗi hành động muốn sửa |
| Thực hiện sửa hành động con có dạng là thẻ chọn |
| Chọn “Thêm mới lựa chọn” |
| Chọn loại button là SMS hoặc Số điện thoại |
| Nhập sai đố điện thoại |
| Ấn “Thêm” |

**Bảng 19** Kiểm thử chức năng Nhắn tin

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Mô tả | Quy trình test | Kết quả mong muốn | Kết quả thực tế | Trạng thái |
| 1 | Nhắn tin | Người dùng nhắn tin cho chatbot thông qua nền tảng thứ 3 | Chatbot trả lời đúng chuỗi hành động đã thiết định trong luật | Chatbot trả lời đúng chuỗi hành động đã thiết định trong luật | Pass |
| Tin nhắn chứa từ khóa của 1 luật |
| 2 | Nhắn tin | Người dùng nhắn tin cho chatbot thông qua nền tảng thứ 3 | Chatbot trả lời bằng chuỗi hành động có điểm cao nhất sau khi dùng Elasticsearch | Chatbot trả lời bằng chuỗi hành động có điểm cao nhất sau khi dùng Elasticsearch | Pass |
| Tin nhắn chứa từ khóa của nhiều luật (2) |
| Thực hiện chọn chuỗi hành động muốn sửa |
| 3 | Nhắn tin | Người dùng nhắn tin cho chatbot thông qua nền tảng thứ 3 | Chatbot trả lời bằng chuỗi hành động có điểm cao nhất sau khi dùng Elasticsearch | Chatbot trả lời bằng chuỗi hành động có điểm cao nhất sau khi dùng Elasticsearch | Pass |
| Tin nhắn không chứa từ khóa |
| Sử dụng Elasticsearch tìm kiếm và có trả ra danh sách luật thỏa mãn |
| 4 | Nhắn tin | Người dùng nhắn tin cho chatbot thông qua nền tảng thứ 3 | Trả ra chuỗi hành động thông báo lỗi mặc định | Trả ra chuỗi hành động thông báo lỗi mặc định | Pass |
| Tin nhắn không chứa từ khóa |
| Sử dụng Elasticsearch có trả ra danh sách luật là rỗng |

**Bảng 20** Kiểm thử chức năng Gửi tin nhắn cho nhiều người

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Mô tả | Quy trình test | Kết quả mong muốn | Kết quả thực tế | Trạng thái |
| 1 | Gửi tin nhắn cho nhiều người  (Gửi tin nhắn dạng văn bản) | Vào màn hình Người dùng | Đóng cửa sổ gửi tin nhắn  Tin nhắn được gửi  Thông báo “Gửi thành công” | Đóng cửa sổ  Tin nhắn được gửi  Thông báo “Gửi thành công” | Pass |
| Chọn danh sách người dùng muốn nhắn tin |
| Nhập nội dung tin nhắn dưới dạng văn bản |
| Ấn “Gửi” |
| 2 | Gửi tin nhắn cho nhiều người  (Gửi tin nhắn dạng văn bản) | Vào màn hình Người dùng | Giữ nguyên cửa sổ gửi tin nhắn  Thông báo “Bạn cần điền nội dung cho tin nhắn” | Giữ nguyên cửa sổ gửi tin nhắn  Thông báo “Bạn cần điền nội dung cho tin nhắn” | Pass |
| Chọn danh sách người dùng muốn nhắn tin |
| Không nhập tin nhắn |
| Ấn “Gửi” |
| 3 | Gửi tin nhắn cho nhiều người  (Gửi tin nhắn dạng chuỗi hành động) | Vào màn hình Người dùng | Đóng cửa sổ gửi tin nhắn  Tin nhắn được gửi đến người dùng  Thông báo “Gửi thành công” | Đóng cửa sổ gửi tin nhắn  Tin nhắn được gửi đến người dùng  Thông báo “Gửi thành công” | Pass |
| Chọn danh sách người dùng muốn nhắn tin |
| Ấn công tắc chuyển sang chế độ chọn chuỗi hành động |
| Chọn chuỗi hành động muốn gửi |
| Ấn “Gửi” |
| 4 | Gửi tin nhắn cho nhiều người  (Gửi tin nhắn dạng chuỗi hành động) | Vào màn hình Người dùng | Giữ nguyên cửa sổ gửi tin nhắn  Thông báo “Bạn cần chọn chuỗi hành động muốn gửi” | Giữ nguyên cửa sổ gửi tin nhắn  Thông báo “Bạn cần chọn chuỗi hành động muốn gửi” | Pass |
| Chọn danh sách người dùng muốn nhắn tin |
| Ấn công tắc chuyển sang chế độ chọn chuỗi hành động |
| Không chọn chuỗi hành động |
| Ấn “Gửi” |

## Triển khai

Hiện tại ứng dụng đã được triển khai trên server với tên miền là <https://chungbt.vbee.vn/>. Cụ thể thông số liên quan đến cấu hình của ứng dụng sau khi được triển khai lên server thể hiện ở bảng 21 dưới đây:

**Bảng 21** Thông số cấu hình của ứng dụng và server triển khai

|  |  |
| --- | --- |
| Tên | Thông số |
| Front-End | 593MB |
| Back-End | 163MB |
| CPU Server | Intel® Xeon® Processor E5-2690 v4 (35M Cache, 2.60 GHz) |
| URL | https://chungbt.vbee.vn/ |
| URL API | https://api-chungbt.vbee.vn/ |

## Kết chương

Trong chương 4 này, em đã làm rõ cách áp dụng cơ sở lí thuyết và công nghệ đã đề cập trong chương 3 để thiết kế kiến trúc cho ứng dụng. Chương 5 kế tiếp em sẽ trình bày những giải pháp và những đóng góp nổi bật của bản thân trong quá trình làm ĐATN.

# Các giải pháp và đóng góp nổi bật

Chương 4 em đã nói chi tiết cách phát triển và triển khai ứng dụng. Ứng dụng quản lí hành động ra đời với mục đích tảng tạo chatbot đáp ứng nhu cầu con người trong quá trình giao tiếp và trao đổi thông tin. Ngoài những chức năng mà bất cứ nền tảng nào cũng có, ứng dụng còn bao gồm những ý tưởng để cải tiến sản phẩm. Trong chương 5 em sẽ trình bày cụ thể về những giải pháp cũng như đóng góp nổi bật của bản thân em trong quá trình xây dựng và hoàn thành đồ án này.

## Sử dụng Elasticsearch và chỉ mục để cải thiện kết quả trả về

### Chỉ mục trong MongoDB

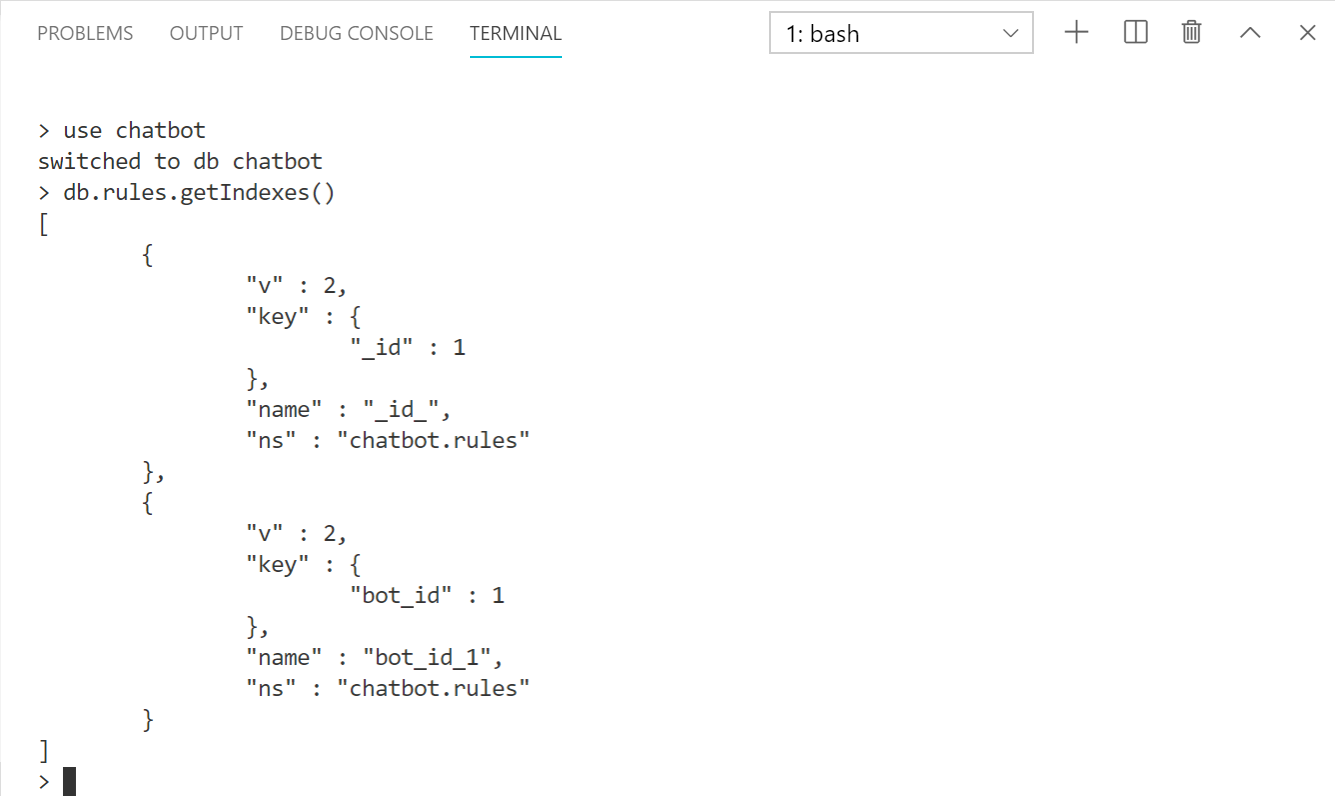
Một trong những chức năng chính của hệ thống là có thể trả lời người dùng dựa trên những luật đã được thiết lập sẵn. Từ tin nhắn người dùng ứng dụng bên thứ 3 sẽ chuyển tin nhắn đó đến ứng dụng và xử lí, truy xuất luật thỏa mãn và gửi lại người dùng chuỗi hành động tương ứng. Trong thực tế, khi sử dụng ứng dụng với lượng người dùng lớn thì số lượng tin nhắn gửi về trong 1 khoảng thời gian cũng tỉ lệ thuận với số người dùng. Vấn đề đặt ra ở đây là với số lượng yêu cầu truy xuất lớn như vậy thì làm sao để tăng tốc độ việc truy xuất dữ liệu từ database. Một kĩ thuật mà em sử dụng nhằm giải quyết vấn đề này đó là sử dụng chỉ mục (Indexes).

A picture containing clock

Description automatically generated

**Hình 30** Ví dụ về đánh chỉ mục trong MongoDB

Chỉ mục trong Database cũng giống như mục lục của một cuốn sách. Thay vì tìm từng trang của cuốn sách Database tạo một mục lục, nó chỉ việc tìm nội dung của cuốn sách qua mục lục đó. Qua đó giúp cho câu lệnh truy vấn nhanh hơn. Một câu truy vấn không có index được gọi là “table scan”. Nghĩa là Database phải xem qua toàn bộ các Document có trong Collection đó để tìm được kết quả truy vấn. Đối với các collection lớn, câu truy vấn dạng này sẽ rất chậm. Thay vào đó nếu sử dụng Indexes nó hỗ trợ việc thực thi những câu lệnh truy vấn một cách hiệu quả.



**Hình 31** Chỉ mục trong Rule Collection

Khi tìm kiếm luật thì danh sách chúng ta cần duyệt qua có thể được thu hẹp lại là số luật của chatbot đó. Hình 29 cho biết số chỉ mục đánh trong bảng Rule. Ngoài chỉ mục mặc định của MongoDB là \_id thì chỉ mục cần đánh là bot\_id.

### Sử dụng Elastic Search

Những nền tảng tạo chatbot sử dụng các luật hiện nay về cơ bản sẽ có nguyên lí hoạt động như sau: lấy tin nhắn từ nền tảng thứ 3, kiểm tra độ trùng khớp với từ khóa của luật, trả về hành động. Nhưng trong thực tế với mức độ linh hoạt của mỗi ngôn ngữ sẽ dẫn đến 1 số vấn đề đặt ra đối với các xử lí trên: (i) tin nhắn không chứa từ khóa, (ii) tin nhắn chưa nhiều từ khóa của nhiều luật, (iii) tin nhắn sai lỗi chính tả. Người quản trị không thể lường trước hết mọi từ khóa hoặc mọi tin nhắn mà người dùng có thể gửi đến cho chatbot. Cách giải quyết mà các nền tảng hiện tại đang dùng để xử lí trường hợp này đó là trả ra chuỗi hành động lỗi mặc định của chatbot đó. Với những ưu điểm như hỗ trợ Full-text search, tìm kiếm mờ thì em đề xuât sử dụng Elasticsearch cải tiến cách xử lí trên.

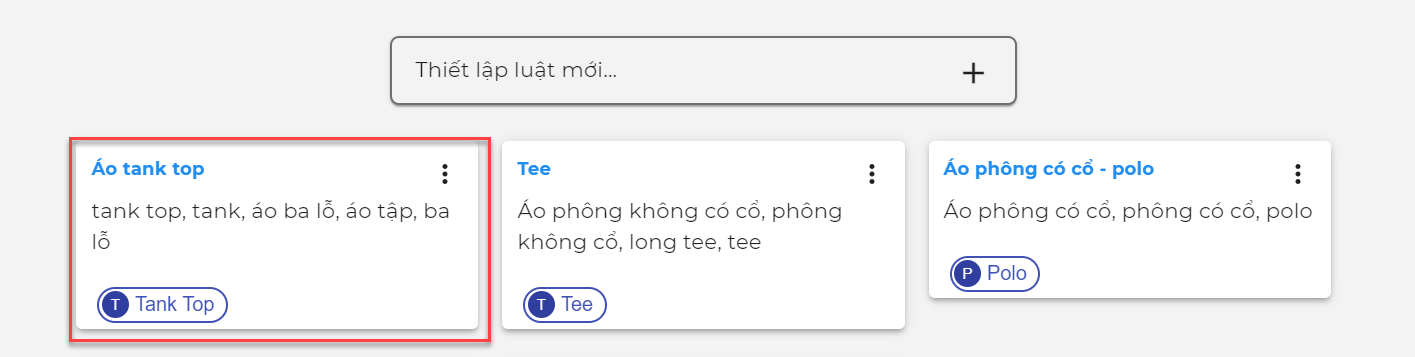
Hình 30 dưới đây trình bày quá trình xử lí từ khi nhận được tin nhắn của người sử dụng đến khi trả về hành động cụ thể.

A close up of a map

Description automatically generated

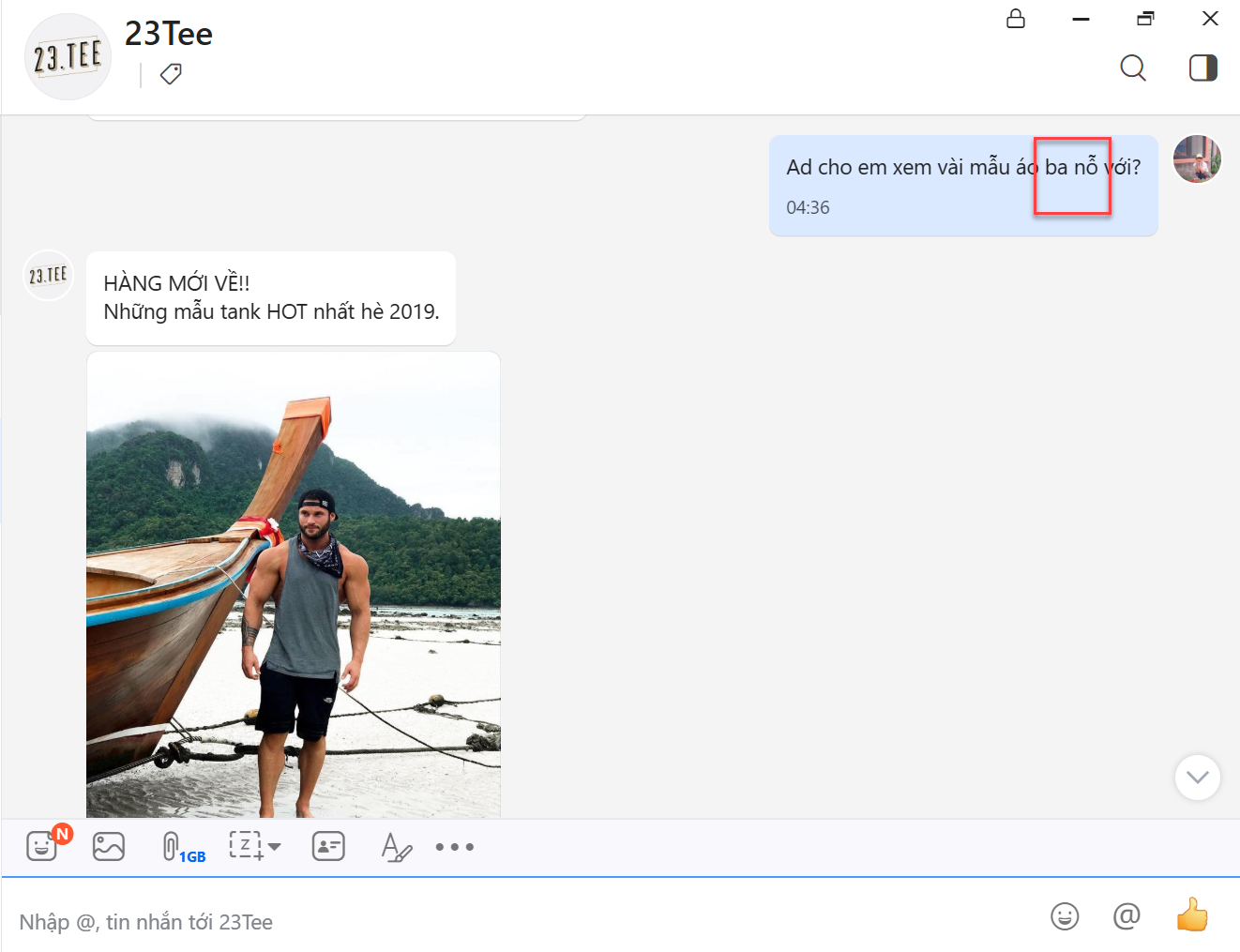
**Hình 32** Lưu đồ biểu diễn quá trình xử lí khi tích hợp Elasticsearch

Đầu tiên server sẽ kiểm trong tin nhắn có chứa từ khóa thỏa mãn trong server hay không. Sau khi quá trình kiểm tra hoàn tất, đối với trường hợp thỏa mãn 1 luật thì server sẽ trả ra ngay chuỗi hành động ứng với luật đó. Tiếp đến với tin nhắn chưa nhiều hơn 1 luật, server sẽ sử dụng Elasticsearch để kiểm tra xem luật nào có điểm (score) lớn nhất. Sau đó sẽ trả trả về chuỗi hành động tương ứng. Cuối cùng đối với trường hợp không thỏa mãn bất cứ luật nào, server tiếp tục sử dụng Elasticsearch để tìm kiếm luật có điểm số cao nhất. Nếu như ngay cả Elasticsearch cũng trả ra không có luật nào thỏa mãn lúc này server sẽ trả về chuỗi hành động lỗi mặc định.



**Hình 33** Danh sách các luật đã thiết lập

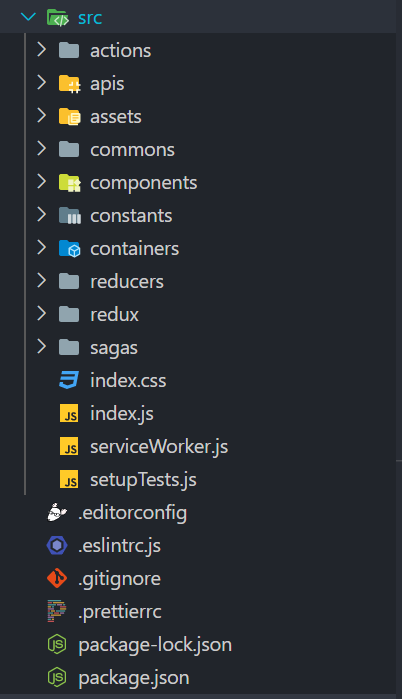
Để có thể so sánh được với những chatbot sử dụng những công nghệ AI hay ML là điều không thể. Nhưng so với những ứng dụng hay nền tàng hiện tại như Chatfuel hay MessNow thì việc đưa Elasticsearch là 1 cái tiến hơn so với cách làm thông thường. Cách làm này giúp tăng mức độ hiệu quả khi chatbot.



**Hình 34** Kết quả trả về với trường hợp sai chính tả

## Xây dựng cấu trúc thư mục rõ ràng

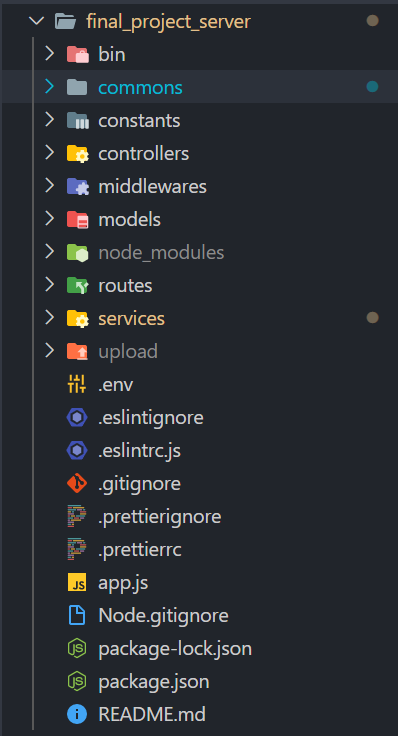
Khi đi thực tập tại những công ty lớn, em luôn đặt ra câu hỏi tại sao họ luôn có 1 danh sách và quy định chặt chẽ về các quy tắt đặt tên file, cấu trúc thư mục. Đến khi bắt tay vào làm ĐATN em mới nhận ra giá trị của việc cấu trúc thư mục. Sau đây là cấu trúc thư mục front-end:



**Hình 35** Cấu trúc thư mục front-end

* **action**: thư mục chứa những hành động sẽ được dispatch khi người dùng tương tác với UI.
* **apis:** thư mục chứa danh sách các hàm để giao tiếp với server.
* **assets**: thư mục chứa các tài nguyên như fonts, icons, ảnh.
* **commons**: thư mục chứa những hàm dùng chung.
* **components**: thư mục components chứa tất các các components liên quan tới UI của dự án.
* **constants**: thư mục chứa những hằng số (constant)
* **reducers**: thư mục chứa các hàm nhằm mục đích thay đổi trạng thái của ứng dụng
* **redux**: thư mục chứa cấu hình redux và redux dev tool
* **sagas**: thư mục chưa các file đóng vai trò là middlware

Đối với server thì commons và constanst có ý nghĩa giống với front-end. Những thư mục như: controllers, middleware, models, routes, services được ánh xạ từ thiết kế back-end. Ngoài ra còn có:



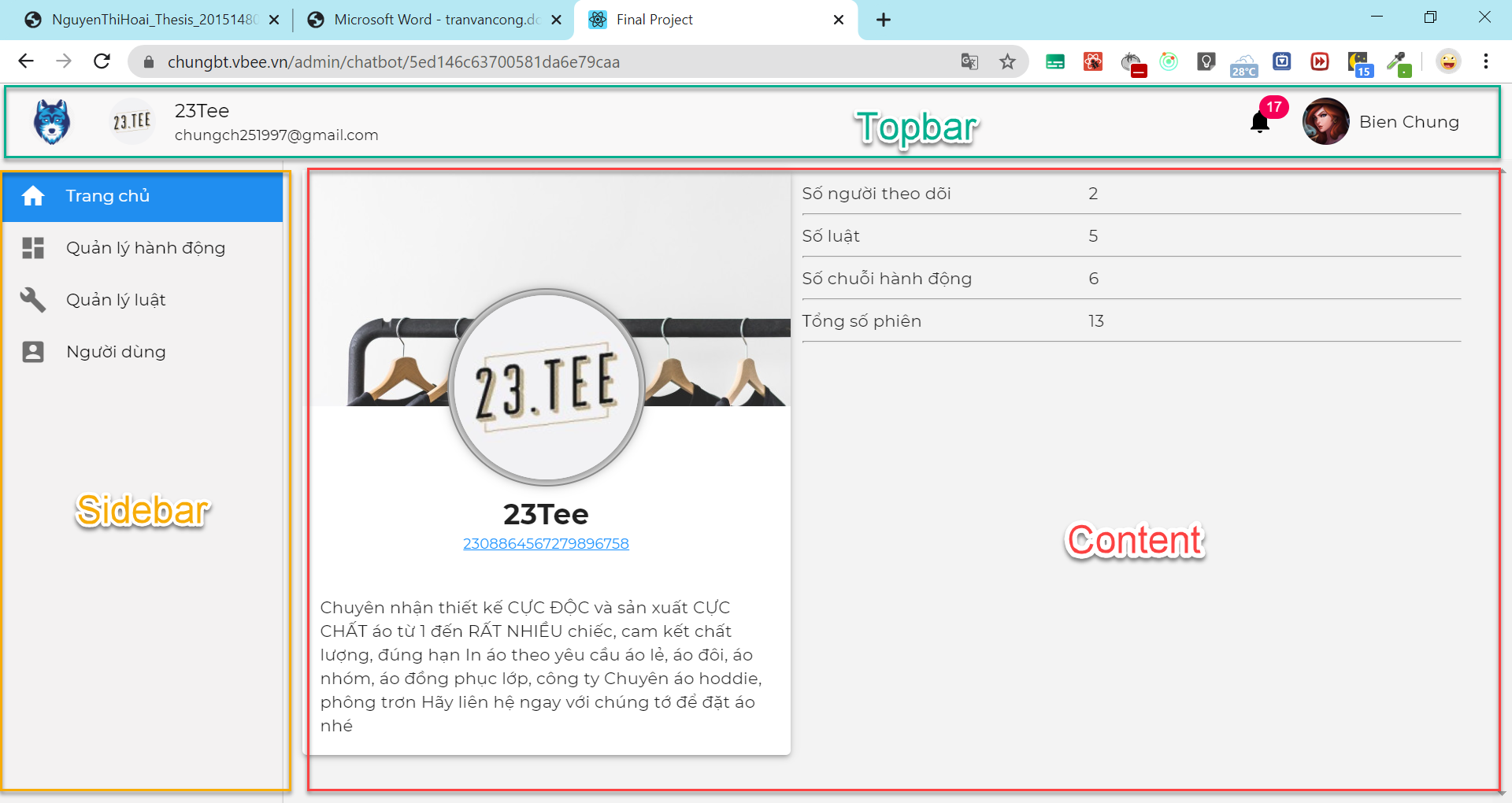
**Hình 36** Cấu trúc thư mục back-end

* **bin**: thư mục chứa file config ban đầu của dự án
* **upload**: chứa file được tải lên từ client

Với một cấu trúc tốt thời gian hoàn thành dự án sẽ nhanh hơn. Sau này trong quá trình phát triển phần mềm, ứng dụng sẽ trở nên lớn và phức tạp hơn thì quản lí sẽ càng ngày càng khó khăn. Một cấu trúc thư mục tốt sẽ giúp lập trình viên thay vì quản lí 1 mớ hỗn độn mà sẽ tập trung vào quản lí chức năng, tiết kiệm thời gian phát triển, làm việc nhóm cũng trở nên hiểu quả hơn.

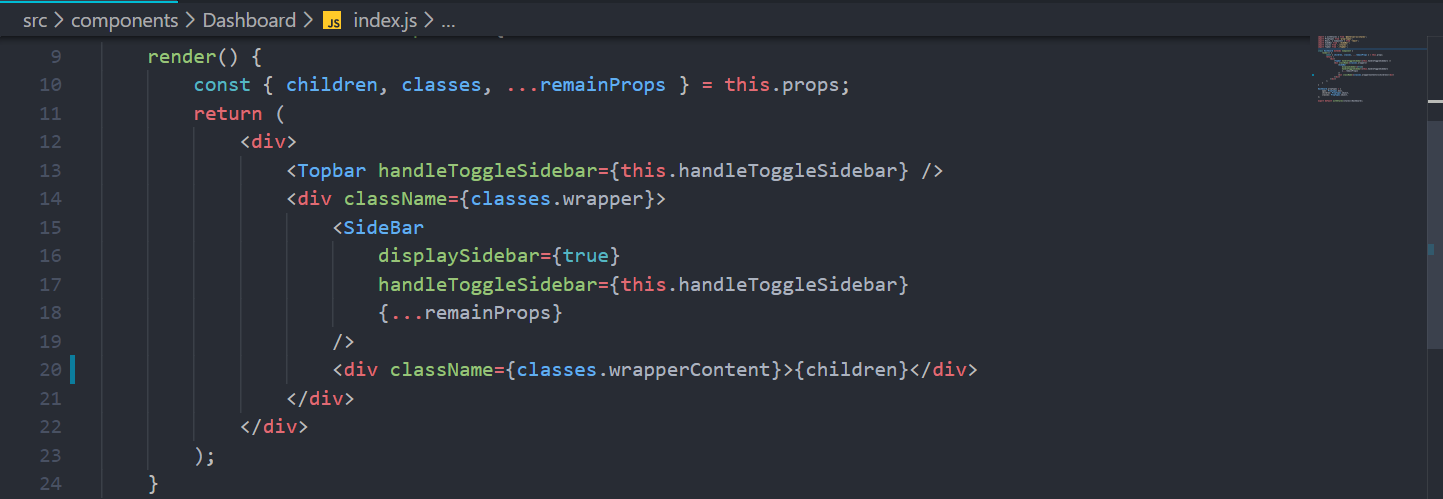
## Thiết kế thành phần có tính tái sử dụng cao và dễ mở rộng

Trong thời gian thực hiện ĐATN em đã có nhiều thay đổi trong cách tư duy cũng như phong cách code. Trước kia thi tham gia các project của các môn học thì việc tái sử dụng code với em gần như là không có. Với suy nghĩ “Code chạy được là được”, chương trình hay những ứng dụng viết ra có rất nhiều dòng code bị lặp, thừa. Nhưng với ReactJs, “component-based” em có một cái nhìn hoàn toàn khác về ý nghĩa của việc tái sử dụng code, cách sắp xếp tổ chức cấu trúc code để dễ dàng đọc và nâng cấp.



**Hình 37** Giao diện trang chủ của ứng dụng

Việc tái sử dụng các component dễ dàng thấy nhất đó là Sidebar và Topbar. Hình 18 có chỉ ra có đến 4 trang là: trang chủ (Home), quản lí hành động (DashBoardBot), quản lí luật (Rule), người dùng (Analyze) tái sử dụng 2 component trên. Hình 36 dưới dây sẽ chỉ ra rõ các tải sử dụng code trong trường hợp này.



**Hình 38** Cách tổ chức code trong file Dashboard

Từ hình có thể thấy, children chính là nội dung (content) của 4 trang đã nêu trên.

# Kết luận

Chương này sẽ nói về kết luận cuối cùng của nội dung đồ án và hướng phát triển sau toàn bộ quá trình làm ĐATN

## Kết luận

Trên đây em đã trình bày về ứng dụng quản lí chuỗi hành động sử dụng luật. Đồng thời cũng đã nêu ra những kỹ thuật, kiến trúc cũng như công nghệ đã áp dụng trong quá trình xây dựng và phát triển. Ứng dụng ra đời là kết quả của quá trình phân tích, so sánh với những sản phẩm tương đương đang có trên hiện nay. Từ đó phân tích, thiết kế để đưa ra những cải tiến mang tính thiết thực nhằm hoàn thiện hơn sản phẩm của mình.

So sánh với những ứng dụng hiện nay, ngoài những tính năng chính như quản lí chatbot, hay quản lí chuỗi hành động, trả lời người dùng thì ứng dụng có đã có thêm nhiều cải tiến. Đầu tiên phải kể đến ứng dụng có một giao diện trực quan dễ sử dụng. Khi thế giới đã có quá nhiều sản phẩm hay ứng dụng có những chức năng giống nhau thì việc quan tâm đến trải nghiệm người dùng sẽ tạo nên sự khác biệt và cũng để lại thiện cảm với người dùng hơn. Từ đó, tính thực tiễn của ứng dụng cũng tăng lên. Tiếp đến là cải tiến chức năng phân tích tin nhắn người của người dùng. Việc tích hợp Elasticsearch vào hệ thống đã cải thiện đáng kể chất lượng tin nhắn mà chatbot trả về, giúp người dùng có những trải nghiệm khi nhắn tin hơn hẳn so với những ứng dụng tương tự trên thị trường hiện nay. Nhưng để đưa được ứng dụng đến với người dùng thì còn phụ thuộc vào nhiều yếu tố. Sau khi triển khai trên máy chủ thực tế, ứng dụng hoạt động còn chưa ổn định, đôi khi vẫn có xuất hiện các lỗi, nhưng em sẽ luôn cố gắng khắc phục một cách kịp thời nhất.

Thực hiện một công việc gì đó hay cụ thể hơn là xây dựng ĐATN là cả 1 quá trình đi giải quyết những vấn đề, khó khăn cả chủ quan lẫn khách quan. Thu lượm thành quả, em đã biết cách tổ chức code, cây thư mục đảm bảo khả năng mở rộng cho công cụ. Học hỏi, áp dụng những công nghệ mới vào sản phẩm của mình. Ngoài những kĩ năng chuyên môn là những kĩ năng mềm cần thiết cho một lập trình viên như làm việc nhóm, giải quyết vấn đề, quản lí thời gian. Thêm vào đó, thông qua những buổi gặp trao đổi với cô hướng dẫn, em có thêm những trải nghiệm về quá trình xây dựng và phát triển một ứng dụng từ khi lên ý tưởng, lấy yêu cầu người dùng đến khi nhận được đánh giá của người dùng để sửa chữa nâng cấp. Tất cả là những hành trang quan trọng trên con đường trở thành một lập trình viên chuyên nghiệp.

## Hướng phát triển

Ứng dụng đang trong quá trình phát triển. Mặc dù đã có những chức năng cơ bản để có thể sử dụng nhưng vẫn có thể cải tiến và phát triển hơn nữa. Hiện nay nền tảng bên thứ 3 đước hỗ trợ là Zalo, trong tương lai em sẽ mở rộng trên các nền tảng lớn hơn như Facebook, Viber… Ngoài ra, loại bỏ những trang đăng nhập, đăng kí mà thay vào đó là đăng nhập qua Facebook, Google, Zalo… Không còn phải ghi nhớ tài khoản mật khẩu, hay điền những thông tin và gửi lên server nữa, việc cần làm chỉ đơn giản là vài cú nhấn chuột, đó sẽ là một điểm khiến người dùng có trải nghiệm tốt hơn. Tuy ứng dụng đã có những mục thống kê những thông tin của chatbot như số phiên chat, số người theo dõi,… nhưng để có thể trực quan hơn những con số, em sẽ tích hợp những biểu đồ tròn hoặc biểu đồ đường. Về mở rộng, ứng dụng hướng đến người quản trị viên của doanh nghiệp, một hướng phát triển khác là ứng dụng sẽ thêm một tác nhân đó là người quản trị hệ thống. Người quản trị hệ thống sẽ có giao diện riêng. Lúc này việc của họ không phải là quản lí chuỗi hành động mà là quán lí những người quản trị viên doanh nghiệp cũng như chatbot của họ.

Trên đây là những hướng phát triển do em đề xuất. Ứng dụng vẫn đang và sẽ tiếp trục phát triển trong tương lại. Em luôn sẵn sàng lắng nghe những góp ý, đánh giá từ người dùng để có thể hoàn thiện sản phẩm cũng như hướng phát triển tốt nhất cho ứng dụng.

# Tài liệu tham khảo

1. <https://vi.wikipedia.org/>
2. <https://aws.amazon.com/vi/nosql/>
3. [https://wiki.matbao.net/chatbot-la-gi-ung-dung-thuc-te-cua-chatbot-trong-kinh- doanh/](https://wiki.matbao.net/chatbot-la-gi-ung-dung-thuc-te-cua-chatbot-trong-kinh-%20doanh/)
4. <https://developers.zalo.me/>
5. ReactJs, <https://reactjs.org/>, lần cuối truy cập 8/6/2020
6. Redux, <https://redux.js.org/>, lần cuối truy cập 8/6/2020
7. Redux Saga, <https://redux-saga.js.org/>, lần cuối truy cập 8/6/2020
8. Express.js, https://expressjs.com/, lần cuối truy cập 8/6/2020
9. MongoDB, https://www.mongodb.com/, lần cuối truy cập 8/6/2020
10. Material UI, https://material-ui.com/, lần cuối truy cập 8/6/2020
11. Elasticsearch, <https://www.elastic.co/>, lần cuối truy cập 8/6/2020