**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**

**Ứng dụng chatbot hỗ trợ điều khiển các thiết bị IOT**

**VŨ TIẾN QUÂN**

quan.vt163397@sis.hust.edu.vn

**Ngành Công nghệ thông tin**

**Chuyên ngành Kỹ thuật máy tính**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | TS. Nguyễn Kim Khánh  Chữ ký của GVHD |
| **Bộ môn:** | Kỹ thuật máy tính |
| **Viện:** | Công nghệ thông tin và truyền thông |
| **HÀ NỘI, 6/2021** | |

**ĐỀ TÀI TỐT NGHIỆP**

**Thông tin sinh viên:**

Họ và tên: Vũ Tiến Quân Mã sinh viên: 20163397

Số điện thoại: 0363501053 Email: [quan.vt163397@sis.hust.edu.vn](mailto:quan.vt163397@sis.hust.edu.vn)

Lớp: CNTT1.02 – K61

Đồ án được thực hiện tại: Trường Đại học Bách khoa Hà Nội

**Mục tiêu đồ án:**

Xây dựng ứng dụng chatbot hỗ trợ điều khiển các thiết bị IOT

**Các mục chính trong đồ án:**

* Xây dựng app chatbot hỗ trợ điều khiển thiết bị IOT
* Xây dựng website quản trị viên
* Thiết kế công tắc bật tắt với module ESP32

**Lời cam đoan:**

Tôi – Vũ Tiến Quân xin cam đoan toàn bộ kết quả tôi đạt được trong quá trình làm đồ án dưới sự hỗ trợ của thầy giáo TS. Nguyễn Kim Khánh không sao chép bất kỳ một công trình nào khác

Giáo viên hướng dẫn

Ký và ghi rõ họ tên

**Lời cảm ơn**

Lời đầu tiên, em xin dành sự cảm ơn chân thành nhất với thầy giáo TS. Nguyễn Kim Khánh đã dành thời gian suốt 5 tháng để hỗ trợ em hoành thành đồ án. Nhờ sợ hỗ trợ tận tâm của thầy em đã có thêm nhiều kinh nghiệm và kiến thức quý báu. Em cũng xin cảm ơn các thầy các cô trong viện Cộng nghệ thông tin và Truyền thông đã giúp em trong suốt 5 năm học vừa qua để em có thể thêm những kiến thức nền tảng vững chắc.

Lời cuối cùng em xin cảm ơn đến các bạn bè, gia đình, những người bạn trong lớp CNTT1.02 đã luôn ở bên động viên, giúp đỡ em những lúc em khó khăn trong cuộc sống. Được học tập và làm việc dưới mái trường đại học Bách Khoa Hà nội là niềm vinh hạnh lớn lao của em.

Em xin chân thành cảm ơn!

**Tóm tắt nội dung đồ án**

Trong giai đoạn phát triển mạnh mẽ của nền công nghiệp 4.0 hiện nay. Chatbot là một công cụ hỗ trợ đắc lực cho các doanh nghiệp, công ty. Điển hình như các công cụ tự động trả lời tin nhắn như google assistant, siri, hay amazon alexa đều phát triển rất mạnh, ngay cả messeger của facebook hiện nay cũng hỗ trợ người dùng tạo chatbot doanh nghiệp của riêng mình. Không những vậy sự phát triển của IOT cũng đang có bước tiến mạnh mẽ. Điển nhình như mô hình nhà thông minh đang càng ngày càng nở rộ trên lãnh thổ Việt Nam. Dựa vào sự kết hợp đó, em đã phát triển một hệ thống bao gồm cả phần mềm và phần cứng kết hợp giữa chatbot và IOT.

Quá trình hoàn thành đồ án em đã sử dụng các công cụ như Visual Studio Code, Docker, Xcode, Android Studio, ESP32, …

Sinh viên thực hiện

Ký và ghi rõ họ tên

Kết quả mà em đã đạt được cơ bản đã đáp ứng được mục tiêu đề ra.Nhưng để phát triển mở rộng thực tế cần tối ưu hơn về phần giao diện ứng dụng và phần cứng esp32.

Dự định tương lai em sẽ cũng cố thêm về phần mạch để có thể giảm thiểu tối đa khối lượng và giá thành để có thể áp dụng vào cuộc sống.

Sinh viên thực hiện

Ký và ghi rõ họ tên

**MỤC LỤC**

[CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI 1](#_Toc74424006)

[1.1 Đặt vấn đề 1](#_Toc74424007)

[1.2 Định hướng giải pháp 1](#_Toc74424008)

[1.3 Bố cục đồ án 2](#_Toc74424009)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 3](#_Toc74424010)

[2.1 Trí tuệ nhân tạo 3](#_Toc74424011)

[2.1.1 Khái niệm 3](#_Toc74424012)

[2.1.2 Ứng dụng của trí tuệ nhân tạo 3](#_Toc74424013)

[2.2 Xử lý ngôn ngữ tự nhiên 3](#_Toc74424014)

[2.2.1 Xử lý ngôn ngữ tự nhiên là gì? 3](#_Toc74424015)

[2.2.2 Các bước xử lý 3](#_Toc74424016)

[2.3 Học máy 4](#_Toc74424017)

[2.3.1 Học máy là gì? 4](#_Toc74424018)

[2.3.2 Các giải thuật của học máy 4](#_Toc74424019)

[2.4 Học sâu 5](#_Toc74424020)

[2.4.1 Học sâu là gì? 5](#_Toc74424021)

[2.5 Chatbot 6](#_Toc74424022)

[2.5.1 Định nghĩa chatbot 6](#_Toc74424023)

[2.5.2 Phân loại chatbot 7](#_Toc74424024)

[2.6 Rasa 7](#_Toc74424025)

[2.6.1 Rasa là gì? 7](#_Toc74424026)

[2.6.2 Rasa NLU 8](#_Toc74424027)

[2.6.3 Rasa CORE 11](#_Toc74424028)

[2.7 Các công nghệ sử dụng trong phạm vi đồ án 12](#_Toc74424029)

[2.7.1 Firebase 12](#_Toc74424030)

[2.7.2 Reactjs 14](#_Toc74424031)

[2.7.3 React Native 14](#_Toc74424032)

[2.7.4 Docker 15](#_Toc74424033)

[2.7.5 ExpressJS 15](#_Toc74424034)

[2.7.6 PostgreSQL 16](#_Toc74424035)

[2.7.7 Socketio 16](#_Toc74424036)

[2.7.8 Module WIFI ESP32 16](#_Toc74424037)

[CHƯƠNG 3. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 17](#_Toc74424038)

[3.1 Mô tả tổng quan 17](#_Toc74424039)

[3.1.1 Yêu cầu hệ thống và các tác nhân 17](#_Toc74424040)

[3.1.2 Sơ đồ use case tổng quan 18](#_Toc74424041)

[3.1.3 Sơ đồ use case phân rã 18](#_Toc74424042)

[3.2 Đặc tả use case 21](#_Toc74424043)

[3.2.1 Đăng ký 21](#_Toc74424044)

[3.2.2 Đăng nhập 22](#_Toc74424045)

[3.2.3 Đăng xuất 22](#_Toc74424046)

[3.2.4 Quản lý thông tin cá nhân 23](#_Toc74424047)

[3.2.5 Đổi mật khẩu 24](#_Toc74424048)

[3.2.6 Chat 25](#_Toc74424049)

[3.2.7 Quản lý danh sách người dùng 26](#_Toc74424050)

[3.2.8 Quản lý danh sách thiết bị 28](#_Toc74424051)

[3.3 Biểu đồ tuần tự 31](#_Toc74424052)

[3.3.1 Biểu đồ tuần tự đăng ký 31](#_Toc74424053)

[3.3.2 Biểu đồ tuần tự đăng nhập 32](#_Toc74424054)

[3.3.3 Biểu đồ tuần tự đăng xuất 32](#_Toc74424055)

[3.3.4 Biểu đồ tuần tự chat 32](#_Toc74424056)

[3.3.5 Biểu đồ tuần tự quản lý thông tin cá nhân 33](#_Toc74424057)

[3.3.6 Đổi mật khẩu 33](#_Toc74424058)

[3.3.7 Biểu đồ tuần tự quản lý danh sách thiết bị của người dùng 34](#_Toc74424059)

[3.3.8 Biểu đồ tuần tự quản lý danh sách thiết bị của quản trị viên 35](#_Toc74424060)

[3.3.9 Biểu đồ tuần tự quản lý danh sách người dùng 36](#_Toc74424061)

[3.4 Biểu đồ lớp 37](#_Toc74424062)

[3.4.1 Biểu đồ lớp ca sử dụng đăng nhập 37](#_Toc74424063)

[3.4.2 Biểu đồ lớp ca sử dụng chat 38](#_Toc74424064)

[3.4.3 Biểu đồ lớp ca sử dụng quản lý thiết bị 38](#_Toc74424065)

[3.4.4 Biểu đồ lớp ca sử dụng quản lý người dùng 38](#_Toc74424066)

[3.5 Biểu đồ hoạt động 39](#_Toc74424067)

[3.5.1 Đăng ký 39](#_Toc74424068)

[3.5.2 Đăng nhập 39](#_Toc74424069)

[3.5.3 Chat 40](#_Toc74424070)

[3.5.4 Ghép nối thiết bị 40](#_Toc74424071)

[3.6 Biểu đồ component 40](#_Toc74424072)

[3.7 Thiết kế cơ sở dữ liệu 42](#_Toc74424073)

[3.8 Thiết kế chi tiết các bảng dữ liệu 42](#_Toc74424074)

[CHƯƠNG 4. TRIỂN KHAI 45](#_Toc74424075)

[4.1 Giao diện Chatbot 45](#_Toc74424076)

[4.2 Rasa 46](#_Toc74424077)

[4.2.1 Pipeline Components 47](#_Toc74424078)

[4.2.2 Đầu ra 50](#_Toc74424079)

[4.3 Mạch điều khiển thiết bị 51](#_Toc74424080)

[4.4 Giao diện quản trị viên 53](#_Toc74424081)

[4.5 Giao diện Firebase Realtime database 54](#_Toc74424082)

[CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 55](#_Toc74424083)

[5.1 Kết luận 55](#_Toc74424084)

[5.2 Hướng phát triển đồ án trong tương lai 55](#_Toc74424085)

[TÀI LIỆU THAM KHẢO 56](#_Toc74424086)

**DANH MỤC HÌNH ẢNH**

[Hình 2.1 Rasa component Lifecycle 8](#_Toc74520889)

[Hình 2.2 Cơ chế hoạt động của CountVectorsFeaturizer 9](#_Toc74520890)

[Hình 2.3 Bộ phân loại ý định và trích xuất thực thể 10](#_Toc74520891)

[Hình 3.1 Sơ đồ use case tổng quan 19](#_Toc74520892)

[Hình 3.2 Sơ đồ use case phân rã quản lý danh sách thiết bị (đối với quản trị viên) 20](#_Toc74520893)

[Hình 3.3 Sơ đồ use case phân rã quản lý danh sách thiết bị (đối với người dùng) 20](#_Toc74520894)

[Hình 3.4 Sơ đồ use case phân rã quản lý thông tin cá nhân 21](#_Toc74520895)

[Hình 3.5 Sơ đồ use case phân rã quản lý danh sách người dùng 21](#_Toc74520896)

[Hình 3.6 Sơ đồ use case phân rã đăng ký 21](#_Toc74520897)

[Hình 3.7 Biểu đồ tuần tự đăng ký 32](#_Toc74520898)

[Hình 3.8 Biểu đồ tuần tự đăng nhập 33](#_Toc74520899)

[Hình 3.9 Biểu đồ tuần tự đăng xuất 33](#_Toc74520900)

[Hình 3.10 Biểu đồ tuần tự chat 33](#_Toc74520901)

[Hình 3.11 Biểu đồ tuần tự quản lý thông tin cá nhân - Cập nhật 34](#_Toc74520902)

[Hình 3.12 Biểu đồ tuần tự quản lý thông tin cá nhân - Xoá tài khoản 34](#_Toc74520903)

[Hình 3.13 Biểu đồ tuần tự quản lý thông tin cá nhân - Đổi mật khẩu 34](#_Toc74520904)

[Hình 3.14 Biểu đồ tuần tự quản lý danh sách thiết bị của người dùng - Xem danh sách thiết bị 35](#_Toc74520905)

[Hình 3.15 Biểu đồ tuần tự quản lý danh sách thiết bị của người dùng - Thêm mới thiết bị 35](#_Toc74520906)

[Hình 3.16 Biểu đồ tuần tự quản lý danh sách thiết bị của người dùng - Tìm kiếm thiết bị 36](#_Toc74520907)

[Hình 3.17 Biểu đồ tuần tự quản lý danh sách thiết bị của người dùng - Xoá thiết bị 36](#_Toc74520908)

[Hình 3.18 Biểu đồ tuần tự quản lý danh sách thiết bị của quản trị viên - Tìm kiếm thiết bị 36](#_Toc74520909)

[Hình 3.19 Biểu đồ tuần tự quản lý danh sách thiết bị của quản trị viên - Xem danh sách thiết bị 37](#_Toc74520910)

[Hình 3.20 Biểu đồ tuần tự quản lý danh sách thiết bị của quản trị viên - Xoá thiết bị 37](#_Toc74520911)

[Hình 3.21 Biểu đồ tuần tự quản lý danh sách người dùng - Tìm kiếm người dùng 37](#_Toc74520912)

[Hình 3.22 Biểu đồ tuần tự quản lý danh sách người dùng - Cập nhật thông tin người dùng 38](#_Toc74520913)

[Hình 3.23 Biểu đồ tuần tự quản lý danh sách người dùng - Xoá người dùng 38](#_Toc74520914)

[Hình 3.24 Biểu đồ lớp ca sử dụng đăng nhập 38](#_Toc74520915)

[Hình 3.25 Biểu đồ lớp ca sử dụng chat 39](#_Toc74520916)

[Hình 3.26 Biểu đồ lớp ca sử dụng quản lý thiết bị 39](#_Toc74520917)

[Hình 3.27 Biểu đồ lớp ca sử dụng quản lý người dùng 39](#_Toc74520918)

[Hình 3.28 Biểu đồ hoạt động - Đăng ký 40](#_Toc74520919)

[Hình 3.29 Biểu đồ hoạt động - Đăng nhập 40](#_Toc74520920)

[Hình 3.30 Biểu đồ hoạt động - Chat 41](#_Toc74520921)

[Hình 3.31 Biểu đồ hoạt động - Ghép nối thiết bị 41](#_Toc74520922)

[Hình 3.32 Biểu đồ component 42](#_Toc74520923)

[Hình 3.33 Thiết kế cơ sở dữ liệu 43](#_Toc74520924)

[Hình 4.1 Màn hình đăng nhập và đăng ký 46](#_Toc74520925)

[Hình 4.2 Màn hình sửa mật khẩu và thông tin tài khoản 46](#_Toc74520926)

[Hình 4.3 Màn hình Chat và quản lý thiết bị 47](#_Toc74520927)

[Hình 4.4 Thông báo xoá thiết bị, người dùng 47](#_Toc74520928)

[Hình 4.5 Mạch điều khiển thiết bị 53](#_Toc74520929)

[Hình 4.6 Hình ảnh thiết bị 54](#_Toc74520930)

[Hình 4.7 Admin - danh sách người dùng 55](#_Toc74520931)

[Hình 4.8 Admin - danh sách thiết bị 55](#_Toc74520932)

[Hình 4.9 Admin - chỉnh sửa người dùng 56](#_Toc74520933)

[Hình 4.10 Admin - xoá người dùng 56](#_Toc74520934)

[Hình 4.11 Giao diện Firebase realtime database 56](#_Toc74520935)

**DANH MỤC BẢNG BIỂU**

[Bảng 2.1 Độ ưu tiên của các chính sách 18](#_Toc73918556)

[Bảng 2.2 Các công nghệ sử dụng 19](#_Toc73918557)

[Bảng 3.1 Đặc tả use case - Đăng ký 28](#_Toc73918558)

[Bảng 3.2 Đặc tả use case – Đăng nhập 29](#_Toc73918559)

[Bảng 3.3 Đặc tả use case – Đăng xuất 29](#_Toc73918560)

[Bảng 3.4 Đặc tả use case - Quản lý thông tin cá nhân (Sửa) 30](#_Toc73918561)

[Bảng 3.5 Đặc tả use case - Quản lý thông tin cá nhân (Xoá tài khoản) 31](#_Toc73918562)

[Bảng 3.6 Đặc tả use case - Quản lý thông tin cá nhân (Đổi mật khẩu) 31](#_Toc73918563)

[Bảng 3.7 Đặc tả use case - Chat 32](#_Toc73918564)

[Bảng 3.8 Quản lý danh sách người dùng - Tìm kiếm người dùng 33](#_Toc73918565)

[Bảng 3.9 Quản lý danh sách người dùng - Cập nhật thông tin người dùng 33](#_Toc73918566)

[Bảng 3.10 Quản lý danh sách người dùng - Xoá người dùng 34](#_Toc73918567)

[Bảng 3.11 Quản lý danh sách thiết bị - Tìm kiếm thiết bị 35](#_Toc73918568)

[Bảng 3.12 Quản lý danh sách thiết bị - Thêm mới thiết bị 36](#_Toc73918569)

[Bảng 3.13 Quản lý danh sách thiết bị Xoá thiết bị 37](#_Toc73918570)

[Bảng 3.14 Bảng Customer 49](#_Toc73918571)

[Bảng 3.15 Bảng Admin 50](#_Toc73918572)

[Bảng 3.16 Bảng Device 50](#_Toc73918573)

[Bảng 3.17 Bảng CustomerDevice 50](#_Toc73918574)

[Bảng 3.18 Bảng Message 50](#_Toc73918575)

[Bảng 4.1 Các loại thực thể tiêu biểu 53](#_Toc73918576)

[Bảng 4.2 Các ý định tiêu biểu 53](#_Toc73918577)

[Bảng 4.3 Các thành phần trong mạch 59](#_Toc73918578)

**DANH MỤC TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| IOT | Internet of Things |
| AI | Artificial Intelligence |
| NLP | Natural Language Processing |
| BERT | Bidirectional Encoder Representations from Transformers |
| NLU | Natural Language Understanding |
| UI | User Interface |
| API | Application Programming Interface |
| HTTP | HyperText Transfer Protocol |
| URL | Uniform Resource Locator |
| DOM | Document Object Model |
| UDP | User Datagram Protocol |

# GIỚI THIỆU ĐỀ TÀI

## Đặt vấn đề

Qua quá trình khảo sát trên mạng và thực tế. Em nhận nhấy các công ty công nghệ bây giờ tập trung rất nhiều vào mảng trả lời tự động như các ông lớn facebook, google, apple hay microsoft,… Minh chứng là rất nhiều các ứng dụng trả lời tự động như google assistant, siri, amazon alexa,… đều phát triển và có chỗ đứng vững chắc, được nhiều người sử dụng. Cùng với đó thì sự phát triển của IOT cũng đã có sự phát triển không hề nhỏ khi càng ngày các mô hình nhà thông minh ngày càng phát triển và các thiết bị IOT cũng được bày bán càng ngày càng nhiều ở siêu thị hay các trang thương mại điện tử.

Mặc dù cả 2 khía cạnh đều phát triển như vậy nhưng để kết hợp cả hai với nhau vừa để phát triển vừa phù hợp với thị trường người dân Việt Nam thì chỉ có google hoàn thành tốt khi ứng dụng google assistant đã hỗ trợ được ngôn ngữ Tiếng Việt, có thể trà lời tự động tin nhắn của người dùng, kết nối điều khiển các thiết bị của google mà còn kết nối được với các app điều khiển của Xiaomi hay Boardlink, Philips Hue,… thông qua đó có thể điều khiển được các thiết bị IOT. Với các ứng dụng của nhà sản xuất thiết bị IOT hầu như đơn thuần ứng dụng chỉ có thể theo dõi hoạt động của thiết bị và điều khiển thiết bị đơn thuần.

## Định hướng giải pháp

Hiện nay chatbot cũng đang là một hướng phát triển được các công ty khá ưa chuộng, tuy nhiên ở thị trường Việt Nam thì chatbot hỗ trợ tiếng việt còn khá là ít. Bởi vì tiếng Việt cũng là một ngôn ngữ khá là khó, mỗi vùng một cách nói khác nhau nên để phát triển một chatbot dành cho ngôn ngữ Việt Nam khá không hề dễ dàng. Bên cạnh đó là sự phát triển của mảng nhà thông minh cũng không hề kém. Vì thế mục tiêu hoàn thành một chatbot vừa có thể trò chuyện với con người vừa có thể điều khiển các thiết bị trong gia đình sẽ là một mục tiêu vô cùng thiết thực đối với con người đang sống trong một môi trường cực kì hiện đại như ngày hôm nay.

Từ những yêu cầu như vậy em đã xác định được các nghiệp vụ cơ bản:

* Đối với người sử dụng
  + Có thể trò chuyện với chatbot, hỏi những câu hỏi, đặt vấn đề và yêu cầu chatbot giải quyết ví dụ như hỏi thời tiết, tình hình dịch bệnh covid hiện nay, xem kết quả xổ số…. Có thể quét và thêm được thiết bị, xem trạng thái của thiết bị và điều khiển thiết bị thông qua chat trực tiếp với bot.
* Đối với quản trị viên
  + Có thể quản lí người dùng và thiết bị

Em cũng đã chọn ra các phương án mình sẽ dùng trong hệ thống này:

* Thư viện mã nguồn mở Rasa giúp em đào tạo các mô hình giúp trích xuất và xử lý văn bản khi người dùng gửi vào
* Ứng dụng mobile app bằng React Native hỗ trợ người dùng gửi tin nhắn bằng văn bản và giọng nói, kết nối và điều khiển thiết bị…
* Website quản trị viên được xây dựng bằng ReactJS hỗ trợ quản lí người dùng và thiết bị
* Backend được xây dựng bằng Postgresql giúp lưu trữ lịch sử hội thoại, thông tin người dùng, thiết bị
* Sử dụng Firebase realtime database để lưu và cập nhật thông tin, trạng thái của thiết bị
* Module ESP32 lấy thông tin và trạng thái thiết bị từ Firebase Realtime database để bật tắt thiết bị

## Bố cục đồ án

Đồ án của em bao gồm các chương chính

* Chương 1: Giới thiệu đề tài
* Chương 2: Phân tích và thiết kế hệ thống
* Chương 3: Cơ sở lí thuyết
* Chương 4: Triển khai
* Chương 5: kết luận và hướng phát triển

# CƠ SỞ LÝ THUYẾT

## Trí tuệ nhân tạo

### Khái niệm

Trí tuệ nhân tạo (AI) là một thuật ngữ đề cập đến việc hệ thống máy tính được lập trình để mô phỏng nhận thức và khả năng ra quyết định của con người.

Mạng thần kinh nhân tạo (Neural Networks) một ví dụ về trí tuệ nhân tạo. Mô hình này mô phỏng cách các tế bào thần kinh (nơ-ron) làm việc trong não người.

### Ứng dụng của trí tuệ nhân tạo

Những tiến bộ kĩ thuật dựa trên trí tuệ nhân tạo được ứng dụng trong việc phân tích dữ liệu phức tạp ở các lĩnh vực khác nhau.

Ban đầu, AI là một loại lập trình máy tính mô phỏng nền tảng kiến thức và khả năng phân tích của con người trong các bối cảnh cụ thể. Điều này thường được thực hiện thông qua việc sử dụng các qui tắc "nếu - thì".

Vào cuối những năm 1990, các mạng nhanh hơn và bộ xử lí mạnh hơn cho phép AI được triển khai trong lĩnh vực hậu cần (logistics), khai thác dữ liệu, phân tích tài chính, chẩn đoán y tế và các lĩnh vực khác.

## Xử lý ngôn ngữ tự nhiên

### Xử lý ngôn ngữ tự nhiên là gì?

Theo wikipedia xử lý ngôn ngữ tự nhiên (natural language processing - NLP) là một nhánh của trí tuệ nhân tạo tập trung vào các ứng dụng trên ngôn ngữ của con người. Trong trí tuệ nhân tạo thì xử lý ngôn ngữ tự nhiên là một trong những phần khó nhất vì nó liên quan đến việc phải hiểu ý nghĩa ngôn ngữ - công cụ hoàn hảo nhất của tư duy và giao tiếp

### Các bước xử lý

* Phân tích hình thái

Trong bước này từng từ sẽ được phân tích và các ký tự không phải chữ (như các dấu câu) sẽ được tách ra khỏi các từ. Trong tiếng Anh và nhiều ngôn ngữ khác, các từ được phân tách với nhau bằng dấu cách. Tuy nhiên trong tiếng Việt, dấu cách được dùng để phân tách các tiếng (âm tiết) chứ không phải từ. Cùng với các ngôn ngữ như tiếng Trung, tiếng Hàn, tiếng Nhật, phân tách từ trong tiếng Việt là một công việc không hề đơn giản.

* Phân tích cú pháp

Dãy các từ sẽ được biến đổi thành các cấu trúc thể hiện sự liên kết giữa các từ này. Sẽ có những dãy từ bị loại do vi phạm các luật văn phạm.

* Phân tích ngữ nghĩa

Thêm ngữ nghĩa vào các cấu trúc được tạo ra bởi bộ phân tích cú pháp.

* Tích hợp văn bản

Ngữ nghĩa của một câu riêng biệt có thể phụ thuộc vào những câu đứng trước, đồng thời nó cũng có thể ảnh hưởng đến các câu phía sau.

* Phân tích thực nghĩa

Cấu trúc thể hiện điều được phát ngôn sẽ được thông dịch lại để xác định nó thật sự có nghĩa là gì.

Tuy nhiên, ranh giới giữa 5 bước xử lý này cũng rất mong manh. Chúng có thể được tiến hành từng bước một, hoặc tiến hành cùng lúc - tùy thuộc vào giải thuật và ngữ cảnh cụ thể.

## Học máy

### Học máy là gì?

Học máy là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kĩ thuật cho phép các hệ thống "học" tự động từ dữ liệu để giải quyết những vấn đề cụ thể. Ví dụ như các máy có thể "học" cách phân loại thư điện tử xem có phải thư rác hay không và tự động xếp thư vào thư mục tương ứng. Học máy rất gần với suy diễn thống kê (statistical inference) tuy có khác nhau về thuật ngữ.

Học máy có liên quan lớn đến thống kê, vì cả hai lĩnh vực đều nghiên cứu việc phân tích dữ liệu, nhưng khác với thống kê, học máy tập trung vào sự phức tạp của các giải thuật trong việc thực thi tính toán. Vì thế một phần của học máy là nghiên cứu sự phát triển các giải thuật suy luận xấp xỉ mà có thể xử lý được.

Học máy có hiện nay được áp dụng rộng rãi bao gồm máy truy tìm dữ liệu,chẩn đoán y khoa, phát hiện thẻ tín dụng giả, phân tích thị trường chứng khoán, phân loại các chuỗi DNA, nhận dạng tiếng nói và chữ viết, dịch tự động, chơi trờ chơi và cử động robot.

### Các giải thuật của học máy

Các thuật toán học máy được phân loại theo kết quả mong muốn của thuật toán. Các loại thuật toán thường dùng bao gồm:

* Học có giám sát: trong đó, thuật toán tạo ra một hàm ánh xạ dữ liệu vào tới kết quả mong muốn. Một phát biểu chuẩn về một việc học có giám sát là bài toán phân loại: chương trình cần học một hàm ánh xạ một vector {\displaystyle [X\_{1},X\_{2},\ldots X\_{N}]}[X1, X2,… XN] tới một vài lớp bằng cách xem xét một số mẫu dữ liệu - kết quả của hàm đó.
* Học không giám sát: mô hình hóa một tập dữ liệu, không có sẵn các ví dụ đã được gắn nhãn.
* Học nửa giám sát: kết hợp các ví dụ có gắn nhãn và không gắn nhãn để sinh một hàm hoặc một bộ phân loại thích hợp.
* Học tăng cường: trong đó, thuật toán học một chính sách hành động tùy theo các quan sát về thế giới. Mỗi hành động đều có tác động tới môi trường, và môi trường cung cấp thông tin phản hồi để hướng dẫn cho thuật toán của quá trình học.
* Chuyển đổi: tương tự học có giám sát nhưng không xây dựng hàm một cách rõ ràng. Thay vì thế, cố gắng đoán kết quả mới dựa vào các dữ liệu huấn luyện, kết quả huấn luyện, và dữ liệu thử nghiệm có sẵn trong quá trình huấn luyện.
* Học cách học: trong đó thuật toán học thiên kiến quy nạp (dự đoán đầu ra đúng cho các tình huống chưa gặp phải trước đây) của chính mình, dựa theo các kinh nghiệm đã gặp.

## Học sâu

### Học sâu là gì?

Học sâu là một tập con của Học máy dựa trên một tập hợp các thuật toán để cố gắng mô hình dữ liệu trừu tượng hóa ở mức cao bằng cách sử dụng nhiều lớp xử lý với cấu trúc phức tạp.

Học sâu là một phần của một họ các phương pháp học máy rộng hơn dựa trên đại diện học của dữ liệu. Một quan sát (ví dụ như một hình ảnh) có thể được biểu diễn bằng nhiều cách như một vector của các giá trị cường độ cho mỗi điểm ảnh, hoặc một cách trừu tượng hơn như là một tập hợp các cạnh, các khu vực hình dạng cụ thể, vv. Một vài đại diện làm khiến việc học các nhiệm vụ dễ dàng hơn (ví dụ, nhận dạng khuôn mặt hoặc biểu hiện cảm xúc trên khuôn mặt) từ các ví dụ. Một trong những hứa hẹn của học sâu là thay thế các tính năng thủ công bằng các thuật toán hiệu quả đối với học không giám sát hoặc học nửa giám sát và tính năng phân cấp.

Các nghiên cứu trong lĩnh vực này cố gắng thực hiện các đại diện tốt hơn và tạo ra các mô hình để tìm hiểu các đại diện này từ dữ liệu không dán nhãn quy mô lớn. Một số đại diện được lấy cảm hứng bởi những tiến bộ trong khoa học thần kinh và được dựa trên các giải thích của mô hình xử lý và truyền thông thông tin trong một hệ thống thần kinh, chẳng hạn như mã hoá thần kinh để cố gắng để xác định các mối quan hệ giữa các kích thích khác nhau và các phản ứng liên quan đến thần kinh trong não.

Nhiều kiến trúc học sâu khác nhau như mạng nơ-ron sâu, mã mạng nơ-ron tích chập sâu, mạng niềm tin sâu và mạng nơ-ron tái phát đã được áp dụng cho các lĩnh vực như thị giác máy tính, tự động nhận dạng giọng nói, xử lý ngôn ngữ tự nhiên, nhận dạng âm thanh ngôn ngữ và tin sinh học, chúng đã được chứng minh là tạo ra các kết quả rất tốt đối với nhiều nhiệm vụ khác nhau.

## Chatbot

### Định nghĩa chatbot

Đầu tiên chúng ta sẽ tìm hiểu về Bot. Một Bot là một phần mềm thực hiện các nhiệm vụ, công việc con người yêu cầu một cách tự động. Hoặc Bot cũng có thể là một chương trình máy tính được thiết kế để giao tiếp với người dùng thông qua kết nối Internet.

Còn với Chat nghĩa là trò chuyện, giao tiếp qua lại giữa hai người. Chatbot chính là hệ thống các Bot ở trong trạng thái trực tuyến, trên các website hoặc trên các nền tảng, giao diện chat khác của social media để chat tự động với người dùng.

Chatbot được hiểu thông thường như một cái máy có thể đối thoại một cách tự nhiên với con người. Ví dụ: bất kỳ người dùng nào cũng có thể hỏi chatbot  một câu hỏi hoặc một câu lệnh bất kỳ và chatbot sẽ trả lời hoặc thực hiện một hoạt động phù hợp đáp lại người dùng.

Chatbot tương tác với chúng ta như một hệ thống trả lời tin nhắn nhanh chóng, tự động. Bằng cách xây dựng, giả lập các mô hình tương tác, kich bản tương tác như của con người sử dụng phương pháp trong Machine Learning,  hệ thống Chatbot có thể tự học, tự hiểu các câu hỏi, nhu cầu của người dùng, khách hàng và thực hiện, đưa ra các phản hồi sao cho phù hợp.

Chatbot sau khi được lập trình và huấn luyện nó sẽ tự động làm việc một cách độc lập như một con người. Chỉ những câu hỏi, tin nhắn của khách hàng đã được cấu trúc lại thành các câu, ý định ngắn gọn với ngôn ngữ tự nhiên (natural language) và thêm vào hệ thống kèm theo các kịch bản đối thoại tương ứng đã xây dựng trước đó thì Chatbot mới có khả năng đưa ra phản hồi.

Chatbot sẽ sử dụng database – cơ sở dữ liệu, nơi lưu trữ các câu hỏi, câu đối thoại đã được huấn luyện cho Chatbot – để phản hồi lại người dùng tại bất kỳ thời điểm nào. Trong trường hợp Chatbot không hiểu câu hỏi của người dùng, có thể do Chatbot chưa được huấn luyện kỹ thì Chatbot sẽ phản hồi sai thông tin, không phù hợp với mong muốn của người dùng. Tuy nhiên, mỗi Chatbot mặc dù làm việc độc lập nhưng vẫn được vận hành và quản lý bởi người xây dựng hệ thống. Do đó, Chatbot sẽ chuyển thông tin đến người quản lý khi không hiểu ý định của người dùng hay khách hàng.

Chatbot được huấn luyện và hoàn thiện trong thời gian dài sẽ tăng khả năng tự học, tự phát triển về phạm vi hiểu biết các ý đình của người dùng và đạt được độ chính xác, độ tin cậy cao trong các phản hồi đưa ra.

Chatbot thường trao đổi với người dùng qua hình thức tin nhắn hoặc âm thanh.

### Phân loại chatbot

Sau khi tìm hiểu thì em có phân loại Chatbot theo những tiêu chí sau:

* Chatbot trò chuyện theo kịch bản

Loại Chatbot này hoạt động thông qua dữ liệu được lập trình sẵn. Khi khách truy cập đặt câu hỏi, phần mềm sẽ đưa ra các tùy chọn liên quan. Họ sẽ nhấp chọn một mục tương ứng có thể đáp ứng mục đích tìm kiếm. Sau đó, Robot sẽ đưa ra câu trả lời liên quan nhất với thông tin người dùng vừa nhấp vào.

Tuy nhiên, đôi khi người dùng sẽ phải trả lời nhiều câu hỏi, nhấp nhiều tùy chọn mới có thể tìm được chính xác những gì mình cần. Yêu cầu của khách hàng thường sẽ được giải đáp khá chậm. Đối với một số câu hỏi không được lập trình sẵn, Robot có thể không trả lời được, hoặc giải đáp thiếu chính xác.

* Chatbot trò chuyện theo từ khóa

Loại này dùng **Machine Learning** để xử lý linh hoạt các truy vấn của người dùng. Những con Robot sẽ được huấn luyện để hiểu những từ, cụm từ liên quan đến câu hỏi nhất định. Nhờ đó, Robot sẽ có thể hiểu được mục đích của người dùng khi gặp phải những cụm từ này. Sau đó, nó sẽ trả về kết quả phù hợp. Ưu điểm của hình thức này là không đưa ra những tùy chọn rập khuôn như loại ở trên. Theo em cảm thấy thì loại Chatbot này là loại Chatbot khá là phổ biến ở nước ta hiện nay.

* Chatbot trò chuyện theo ngữ cảnh

Đây là loại Chatbot hoạt động nhờ sự kết hợp giữa Natural Language Processing – xử lý ngôn ngữ tự nhiên, AI – trí tuệ nhân tạo và Machine Learning – học máy. Nó hoạt động dựa trên việc ghi nhớ lại sở thích, bối cảnh của khách truy cập từ các cuộc trò chuyện trước đó. Điều này cho phép Chatbot đưa ra phản hồi phù hợp nhất với truy vấn của khách hàng.

Có nhiều nền tảng chatbot với NLP, chẳng hạn như:

* Rasa
* Microsoft Bot Framework
* Dialogflow
* Amazon Lex

Trong đồ án em xin sử dụng Rasa để phát triển chatbot.

## Rasa

### Rasa là gì?

Rasa là một công cụ Machine Learning mã nguồn mở dành cho các nhà phát triển và nhóm sản phẩm để mở rộng các bot ngoài việc trả lời các câu hỏi đơn giản.

Rasa bao gồm Rasa NLU và Rasa CORE

### Rasa NLU

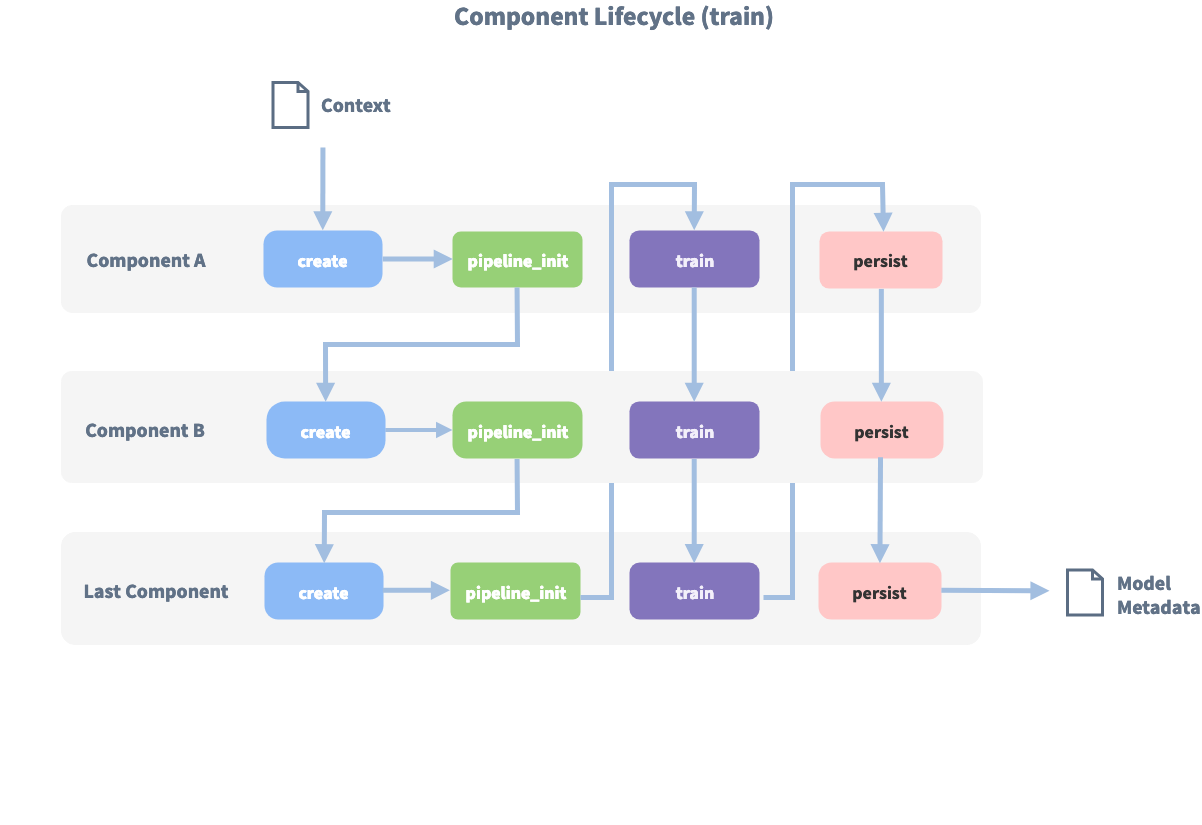
* Định nghĩa

Mô hình NLU được sử dụng để rút ra ý nghĩa từ việc nhập văn bản. Việc đào tạo mô hình NLU trên dữ liệu này cho phép mô hình đưa ra dự đoán về ý định và thực thể trong thông điệp người dùng mới, ngay cả khi thông báo không khớp với bất kỳ ví dụ nào mà mô hình đã thấy trước đây.

Training pipeline: Các mô hình NLU được tạo ra thông qua một đường ống đào tạo, còn được gọi là một đường ống xử lý. Quy trình đào tạo là một chuỗi các bước xử lý cho phép mô hình tìm hiểu các mẫu cơ bản của dữ liệu đào tạo.

Word embeddings: chuyển đổi từ thành vectơ hoặc biểu diễn số dày đặc dựa trên nhiều thứ nguyên. Các từ tương tự được biểu diễn bằng các vectơ tương tự, điều này cho phép kỹ thuật nắm bắt ý nghĩa của chúng. Nhúng từ được sử dụng bởi các thành phần của đường dẫn đào tạo để làm cho dữ liệu văn bản dễ hiểu đối với mô hình học máy.

Bất kể chọn đường ống nào, nó sẽ tuân theo cùng một trình tự cơ bản.



Hình 2.1 Rasa component Lifecycle

* Tokenizer

Tokenizers lấy một dòng văn bản và chia nó thành các phần nhỏ hơn hoặc các token, thường là các từ riêng lẻ. Tokenizer phải là một trong những bước đầu tiên trong quy trình xử lý vì nó chuẩn bị dữ liệu văn bản để sử dụng trong các bước tiếp theo. Tất cả các đường ống đào tạo cần phải bao gồm một Tokenizer.

Sau khi một mô hình đã được huấn luyện bằng cách sử dụng loạt thành phần này, nó sẽ có thể chấp nhận dữ liệu văn bản thô và đưa ra dự đoán về các ý định và thực thể mà văn bản chứa. Trong đề tài của em, em đã dùng tokenizer áp dụng Word Segmentation của thư viện underthesea để phân tách ý định của người dùng phù hợp với ngôn ngữ Việt Nam.

Đầu vào: Văn bản thô.

Đầu ra: tokens cho tin nhắn của người dùng, phản hồi (nếu có) và ý định (nếu được chỉ định). Kết quả của mô hình phân tích cung cấp dữ liệu cho việc trích xuất thực thể.

* Featurizer

Featurizer lấy các token, hoặc các từ riêng lẻ và mã hóa chúng dưới dạng vectơ, là các đại diện số của các từ dựa trên nhiều thuộc tính. Mô hình phân loại ý định lấy đầu ra featurizer và sử dụng nó để đưa ra dự đoán về ý định nào phù hợp với thông điệp của người dùng. Đầu ra của mô hình phân loại ý định được thể hiện trong đầu ra cuối cùng của mô hình NLU dưới dạng danh sách các dự đoán ý định, sắp xếp theo thứ tự ưu tiên. Được chia thành 2 mô hình: thưa và dày đặc và kết quả của mô hình phân tích cung cấp dữ liệu cho bộ phân loại ý định.

A picture containing diagram

Description automatically generated

Hình 2.2 Cơ chế hoạt động của **CountVectorsFeaturizer**

**CountVectorsFeaturizer** là mô hình trích xuất đặc trưng thưa cho thông điệp của người dùng. Mô hình bag-of-words bỏ qua thứ tự của các từ trong nội dung văn bản và thay vào đó tập trung vào số lần các từ xuất hiện trong văn bản, đếm tần suất các từ nhất định từ dữ liệu đào tạo của bạn xuất hiện trong một thông báo.

BytePairFeaturizer là mô hình trích xuất đặc trưng dày đặc với bộ hỗ trợ nhiều ngôn ngữ bao gồm cả Tiếng Việt với tham số thành phần là số chiều 25 và sử dụng kích thước từ vựng là 1000. Byte Pair Encoding là một kỹ thuật nén từ cơ bản giúp chúng ta index được toàn bộ các từ kể cả trường hợp từ mở (không xuất hiện trong từ điển) nhờ mã hóa các từ bằng chuỗi các từ phụ (subwords). Nguyên lý hoạt động của Byte Pair Encoding dựa trên phân tích trực quan rằng hầu hết các từ đều có thể phân tích thành các thành phần con. Phương pháp Byte Pair Encoding sẽ thống kê tần suất xuất hiện của các từ phụ cùng nhau và tìm cách gộp chúng lại nếu tần suất xuất hiện của chúng là lớn nhất. Cứ tiếp tục quá trình gộp từ phụ cho tới khi không tồn tại các subword để gộp nữa, ta sẽ thu được tập subwords cho toàn bộ bộ văn bản mà mọi từ đều có thể biểu diễn được thông qua subwords.

* Bộ phân loại ý định và trích xuất thực thể (DIETClassifier)

Diagram

Description automatically generated

Hình . Bộ phân loại ý định và trích xuất thực thể

Bộ phân loại ý định và trích xuất thực thể nhận dữ liệu đầu vào là các đặc trưng mà em đã sử dụng ở trên được sử dụng để phân loại ý định và trích xuất thực thể. DIETClassifier vừa có thể phân loại ý định hay trích xuất thực thể song song hoặc chúng ta có thể lựa chọn một trong hai tuỳ theo nhu cầu của người sử dụng.

DIETClassifier không những đào tạo nhanh hơn mà còn đào tạo song song với các mô hình pre-train. Nó hoạt động nhanh hơn BERT và cải tiến theo tình trạng của bộ dữ diệu đầu vào phức tạp. Nói qua BERT thì BERT là một mô hình ngôn ngữ (Language Model) được tạo ra bởi Google AI. BERT được coi như là đột phá lớn trong Machine Learning bởi vì khả năng ứng dụng của nó vào nhiều bài toán NLP khác nhau: Question Answering, Natural Language Inference,... với kết quả rất tốt.

BERT mở rộng khả năng của các phương pháp trước đây như Word2vec, fastText bằng cách tạo các biểu diễn theo ngữ cảnh dựa trên các từ trước và sau đó để dẫn đến một mô hình ngôn ngữ với ngữ nghĩa phong phú hơn. Cùng với việc dùng thêm **CRFEntityExtractor,** RegexEntityExtractor và EntitySynonymMapper giúp cho việc trích xuất thực thể một cách chính xác nhất.

* Nhận dạng đối tượng được đặt tên (Named Entity Recognition)

CRFEntityExtractor hoạt động bằng cách xây dựng một mô hình được gọi là Trường ngẫu nhiên có điều kiện để thực hiện nhận dạng đối tượng được đặt tên. Phương pháp này xác định các thực thể trong một câu bằng cách quan sát các đặc điểm văn bản của một từ đích cũng như các từ xung quanh nó trong câu. Các tính năng đó có thể bao gồm tiền tố hoặc hậu tố của từ đích, viết hoa, liệu từ có chứa chữ số hay không, …

* RegexEntityExtractor

RegexEntityExtractor hỗ trợ Trích xuất các thực thể bằng cách sử dụng bảng tra cứu hoặc regex được xác định trong dữ liệu đào tạo

* EntitySynonymMapper

EntitySynonymMapper giúp ánh xạ các thực thể cùng giá trị để trả về các thực thể hiện có mà quá trình trích xuất trước đó đã trích xuất ra

### Rasa CORE

* Định nghĩa:

Rasa CORE là phần quan trọng để bot xử lý thông tin và trả lại message phù hợp với những thực thể và ý định mà Rasa NLU đã trích xuất ra

Rasa CORE sử dụng Rule và Stories để dự đoán cho cuộc hội thoại.

Rule trong Rasa để xử lý những câu chuyện ngắn, để cho chung luôn đi theo một hướng, không như story sẽ vẽ các trường hợp mà người dùng có thể yêu cầu.

Stories trong Rasa là bản trình bày cuộc trò chuyện giữa người dùng và trợ lý AI, được chuyển đổi thành một định dạng cụ thể trong đó thông tin người dùng nhập vào được thể hiện dưới dạng ý định (và thực thể khi cần thiết), trong khi phản hồi và hành động của trợ lý được thể hiện dưới dạng tên hành động. Với Stories càng bao quát được cuộc hội thoại thì bot sẽ càng thông minh hơn

Rasa lựa chọn các policies để quyết định hành động cần thực hiện ở mỗi bước trong cuộc trò chuyện dựa vào máy học hoặc các Rule để chúng ta có thể sử dụng song song

Trong phạm vi đồ án của em các policies được sử dụng:

* MemoizationPolicy

MemoizationPolicy để lưu trữ lịch sử của hội thoại để kiểm tra xem cuộc hội thoại đó có khớp với kịch bản nào trong stories hay không từ đó đưa ra được đầu ra mong muốn của hành động kế tiếp

* TEDPolicy:

TEDPolicy là một kiến trúc đa tác vụ để dự đoán hành động tiếp theo và nhận dạng thực thể, cấu hình cho TEDPolicy một thông số quan trọng là các nhà phát triển có thể cài đặt “max\_history” lưu số lịch sử hội thoại trước đó để xem xét và phân tích quyết định cho hành động tiếp theo nào sẽ được thực hiện, “epochs” để xác định số lần mô hình sẽ được đào tạo

* RulePolicy là xử lý các phần của hội thoại tuân theo một hành vi cố định. Sử dụng cho những hội thoại ngắn, luôn đi theo một hướng.

Độ ưu tiên của Policy:

Bảng 2. Độ ưu tiên của các chính sách

|  |  |
| --- | --- |
| RulePolicy | 6 |
| MemoizationPolicy hoặc AugmentedMemoizationPolicy | 3 |
| TEDPolicy | 1 |

Với độ ưu tiên càng cao thì các policy đó sẽ được ưu tiên cao hơn được sử dụng khi kết quả của hai chính sách dự đoán với độ tin cậy ngang nhau

Rasa CORE sử dụng action (hành động) để gửi trả một tin nhắn lại gửi cho người dùng.

* Các loại hành động:

Một câu trả lời là một thông điệp trợ lý sẽ gửi lại cho người dùng. Đây là hành động bạn chúng ta sử dụng thường xuyên nhất, khi bạn muốn trợ lý gửi văn bản, hình ảnh, nút hoặc tương tự đến người dùng.

Một hành động tuỳ chỉnh (custom action) là một hành động có thể chạy bất kỳ mã nào mà chúng ta muốn. Điều này có thể được sử dụng để thực hiện cuộc gọi API hoặc để truy vấn cơ sở dữ liệu chẳng hạn.

Form là một loại hành động tùy chỉnh đặc biệt, được thiết kế để xử lý logic nghiệp vụ. Nếu bạn có bất kỳ thiết kế hội thoại nào mà bạn mong đợi trợ lý yêu cầu một tập hợp thông tin cụ thể, bạn nên sử dụng các biểu mẫu.

Hành động mặc định là các hành động được tích hợp vào trình quản lý đối thoại theo mặc định. Hầu hết trong số này được dự đoán tự động dựa trên các tình huống hội thoại nhất định

## Các công nghệ sử dụng trong phạm vi đồ án

Các công nghệ mà em đã sử dụng trọng phạm vi đồ án:

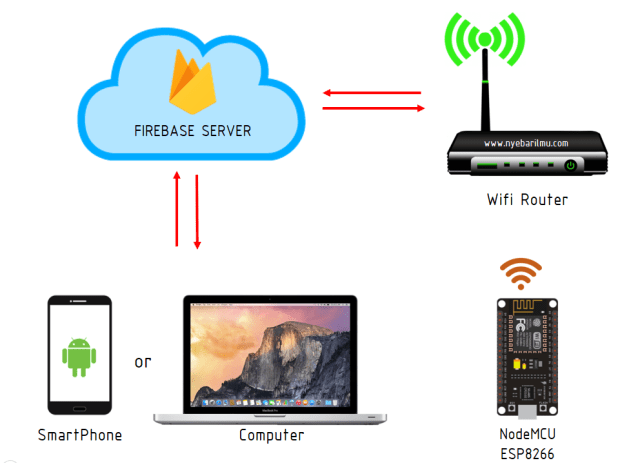
Bảng 2.2 Các công nghệ sử dụng

|  |  |
| --- | --- |
| Chatbot | Rasa |
| Giao diện người dùng | React native |
| Giao diện admin | Reactjs |
| IDE | Visual Studio Code |
| Module wifi | ESP32 |
| Realtime database | Firebase |
| Cơ sở dữ liệu | Postgresql |
| Kết nối chatbot với User App | Socketio |
| Setup môi trường cơ sở dữ liệu | Docker |

### Firebase

#### Định nghĩa

Firebase là một dịch vụ cơ sở dữ liệu thời gian thực được cung cấp bởi Google và hoạt động trên nền tảng đám mây. Nó giúp các lập trình phát triển nhanh các ứng dụng di động bằng cách đơn giản hóa các thao tác với cơ sở dữ liệu. Realtime Database của Firebase lưu trữ và đồng bộ dữ liệu người dùng thời gian thực. Các ứng dụng hỗ trợ tính năng này có thể lưu trữ và lấy dữ liệu từ máy chủ rất nhanh. Các dữ liệu được lưu trữ trong hệ thống cơ sở dữ liệu hỗ trợ NoSQL và được đặt trên nền tảng máy chủ đám mây. Dữ liệu được ghi và đọc với thời gian thấp nhất tính bằng mili giây. Nền tảng này hỗ trợ đồng bộ hóa dữ liệu của người dùng kể cả khi không có kết nối mạng. Tạo nên trải nghiệm xuyên suốt bất chấp tình trạng kết nối internet của người sử dụng.



Hình 2.4 Cơ chế của Firebase realtime database

#### Các khả năng chính của Realtime Database

* Realtime:

Firebase Realtime Database sử dụng đồng bộ dữ liệu mối khi dữ liệu có thay đổi, mọi thiết bị được kết nối sẽ nhận được thay đổi trong vài mili giây.

* Offline:

Khi người dùng ngoại tuyến, dữ liệu sẽ được lưu trên bộ nhớ cache của thiết bị và tự động đồng bộ khi bạn trực tuyến. Tất cả là tự động

* Accessible from Client Devices

Firebase Realtime Database có thể truy cập từ một thiết bị mobile hoặc trình duyệt web. Nó không cần một ứng dụng server nào cả. Bảo mật và xác thực dữ liệu có thể thông qua các Rule bảo mật của Firebase Realtime Database, các rule được thực thi khi dữ liệu được đọc hoặc ghi.

### Reactjs

#### Định nghĩa

React là một thư viện UI phát triển tại Facebook để 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.

#### Các khả năng chính

* Declarative

React giúp tạo UI tương tác đơn giản. Thiết kế các khung nhìn đơn giản cho từng trạng thái trong ứng dụng và React sẽ cập nhật và render đúng các thành phần phù hợp khi dữ liệu của bạn thay đổi.

* Component-Based

Xây dựng các component và quản lý các trạng thái của riêng chúng, sau đó kết hợp chúng để tạo các UI phức tạp.

Vì component logic được viết bằng JavaScript thay vì các mẫu, chúng ta có thể dễ dàng truyền dữ liệu phong phú qua ứng dụng của mình và tránh thao tác với DOM.

* Học một lần, Viết mọi nơi

Chúng ta có thể phát triển các tính năng mới trong React mà không cần viết lại mã hiện có.

React cũng có thể render trên máy chủ bằng Node và xây dựng ứng dụng di động bằng cách sử dụng React Native.

### React Native

#### Định nghĩa

Theo wikipedia: React Native là một framework mã nguồn mở được sáng tạo bởi Facebook. Nó được sử dụng để phát triển ứng dụng di động iOS và Android bằng các cho phép các nhà phát triển sử dụng React cùng với môi trường ứng dụng gốc (native).

Native App là các ứng dụng được phát triển và build trên chính những công cụ hỗ trợ mà nhà phát triển iOS và Android cung cấp cho người lập trình viên ở đây là XCode và Android.

Chúng được viết bằng ngôn ngữ mà nhà phát triển iOS và Android cung cấp ra iOS thì là Swift hiện tại, xưa là Objective C, còn Android thì hiện tại tồn tại song song là Java và Kotlin. Các ứng dụng Native được build dựa trên ngôn ngữ của chính hệ điều hành và sử dụng những tính năng sẵn có của hệ điều hành mà không phải thông qua bất cứ ứng dụng bên thứ ba hay engine nào khác để vận hành nên tốc độ là nhanh nhất.

Các nguyên tắc hoạt động của React Native gần như giống hệt với React ngoại trừ việc React Native không thao tác với DOM thông qua DOM ảo. Nó chạy một quá trình xử lý nền (phiên dịch Javascript viết bởi các nhà phát triển) trực tiếp trên thiết bị đầu cuối và giao tiếp với nền tảng gốc qua một cầu trung gian, bất đông bộ và theo đợt.

Các thành phần React bao bọc mã gốc và tương tác với API gốc qua mô hình UI khai báo và Javascript của React. Điều này giúp việc phát triển ứng dụng cho nhiều nền tảng nhanh hơn.

React Native không sử dụng HTML. Thay vào đó, nó sử dụng các thành phần khác từ luồng Javascript.

### Docker

#### Docker là gì?

Docker là nền tảng phần mềm cho phép chúng ta dựng, kiểm thử và triển khai ứng dụng một cách nhanh chóng. Docker đóng gói phần mềm vào các đơn vị tiêu chuẩn hóa được gọi là container có mọi thứ mà phần mềm cần để chạy, trong đó có thư viện, công cụ hệ thống, mã và thời gian chạy. Bằng cách sử dụng Docker, chúng ta có thể nhanh chóng triển khai và thay đổi quy mô ứng dụng vào bất kỳ môi trường nào.

#### Các thành phần chính của Docker

* Docker Engine: dùng để tạo ra Docker image và chạy Docker container.
* Docker Hub: dịch vụ lưu trữ giúp chứa các Docker image.
* Docker Machine: tạo ra các Docker engine trên máy chủ.
* Docker Compose: chạy ứng dụng bằng cách định nghĩa cấu hình các Docker container thông qua tệp cấu hình
* Docker image: một dạng tập hợp các tệp của ứng dụng, được tạo ra bởi Docker engine. Nội dung của các Docker image sẽ không bị thay đổi khi di chuyển. Docker image được dùng để chạy các Docker container.
* Docker container: một dạng runtime của các Docker image, dùng để làm môi trường chạy ứng dụng.

Trong phạm vi đồ án để đáp ứng nhu cầu cho việc setup môi trường PostgreSQL thì dùng Docker em có thể setup một container để chạy môi trường PostgreSQL dễ dàng và cách ly hoàn toàn với các môi trường khác. Nếu chúng ta không sử dụng thì chúng ta cũng có thể dễ dàng xoá bỏ image đó không ảnh hưởng đến hiệu suất của máy tính so với việc chúng ta cài đặt thông thường - làm máy tính chúng ta trở nên chậm chạp khi cài quá nhiều phần mềm.

### ExpressJS

Expressjs là một framework được xây dựng trên nền tảng của Nodejs. Nói sơ qua về Nodejs thì Nodejs là một nền tảng được xây dựng trên V8 JavaScript Engine – trình thông dịch thực thi mã JavaScript, giúp xây dựng các ứng dụng web một cách đơn giản và dễ dàng mở rộng. ExpressJS cung cấp các tính năng mạnh mẽ để phát triển web hoặc mobile. Expressjs hỗ trợ các giao thức HTTP (là các giao thức truyền tải siêu văn bản hoạt động trên mô hình máy chủ - máy khách) và midleware tạo ra API vô cùng mạnh mẽ và dễ sử dụng.

Express có những tính năng như:

* Cho phép thiết lập các lớp trung gian để trả về các HTTP request.
* Định nghĩa routing có thể được sử dụng với các hành động khác nhau dựa trên phương thức HTTP và URL.
* Cho phép trả về các trang HTML dựa vào các tham số truyền vào đến template.

### PostgreSQL

PostgreSQL là một hệ quản trị cơ sở dữ liệu quan hệ – đối tượng. PostgreSQL là mã nguồn mở miễn phí. Người dùng có thể tự do sử dụng, chỉnh sửa và phân bổ PostgreSQL theo nhiều hình thức khác nhau. So với nhiều hệ quản trị cơ sở dữ liệu khác, PostgreSQL không quá yêu cầu về công tác bảo trì bởi tính ổn định cao, có thể phát triển nhiều ứng dụng khác nhau với chi phí tương đối thấp. PostgreSQL cung cấp cho người dùng nhiều tính năng hiện đại, khả năng ổn định cao, tốc độ tuyệt vời, chính vì vậy mà PostgreSQL trở nên phổ biến

### Socketio

Socketio là một công cụ hỗ trợ các ứng dụng real time, nó yêu cầu tốc độ phản hồi  ngay lập tức như: chat, trực tiếp bóng đá,…

Việc sử dụng tương đối giống nhau ở cả client lẫn server nó bao gồm 3 phần chính:

* Khởi tạo kết nối
* Lắng nghe event
* Gửi event

Riêng server thì không cần phải khởi tạo kết nối mà chỉ lắng nghe các sự kiện từ client gửi lên

### Module WIFI ESP32

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with low confidence

Hình 2.5 Cấu tạo module wifi ESP32

**Esp32 hỗ trợ 2 giao tiếp không dây** là wifi và bluetooth và được tích hợp các loại cảm biến như cảm biến từ trường, đo nhiệt độ và cảm biến điện dung với các thông số:

* CPU: Xtensa Dual-Core LX6 microprocessor.
* Chạy hệ 32 bit
* Tốc độ xử lý 160MHZ up to 240 MHz
* Tốc độ xung nhịp đọc flash chip 40mhz --> 80mhz (tùy chỉnh khi lập trình)
* RAM: 520 KByte SRAM trong đó 8 KB RAM RTC tốc độ cao – 8 KB RAM RTC tốc độ thấp (dùng ở chế độ DeepSleep)
* Nhiệt độ hoạt động -40 + 85 độ C
* Điện áp hoạt động: 2.2-3.6V
* Số cổng GPIOs : 36
* Dòng điện tối đa qua mỗi chân: 40mA

# PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## Mô tả tổng quan

### Yêu cầu hệ thống và các tác nhân

Yêu cầu hệ thống

* Người sử dụng có thể vào và chat với bot, có khả năng thêm thiết bị và điều khiển bật tắt nhờ chatbot thông qua tin nhắn hoặc giọng nói
* Quản trị viên có thể quản lý thông tin của người sử dụng và thiết bị

Hệ thống chatbot bao gồm các tác nhân

* Người dùng
* Quản trị viên
* Thiết hỗ trợ điều khiển bật tắt

### Sơ đồ use case tổng quan

Diagram

Description automatically generated

Hình . Sơ đồ use case tổng quan

### Sơ đồ use case phân rã

* Sơ đồ use case phân rã quản lý danh sách thiết bị (đối với quản trị viên)

Diagram

Description automatically generated

Hình . Sơ đồ use case phân rã quản lý danh sách thiết bị (đối với quản trị viên)

* Sơ đồ use case phân rã quản lý danh sách thiết bị (đối với người dùng)

Diagram

Description automatically generated

Hình . Sơ đồ use case phân rã quản lý danh sách thiết bị (đối với người dùng)

* Sơ đồ use case phân rã quản lý thông tin cá nhân

Diagram

Description automatically generated

Hình . Sơ đồ use case phân rã quản lý thông tin cá nhân

* Sơ đồ use case phân rã quản lý danh sách người dùng

Diagram

Description automatically generated

Hình . Sơ đồ use case phân rã quản lý danh sách người dùng

* Sơ đồ use case phân rã đăng ký

A picture containing text, toilet

Description automatically generated

Hình . Sơ đồ use case phân rã đăng ký

## Đặc tả use case

### Đăng ký

Bảng 3. Đặc tả use case - Đăng ký

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC003 | Tên Use case | Đăng ký |
| Tác nhân | Khách | | |
| Mô tả | Khách yêu cầu đăng ký thành viên mới | | |
| Sự kiện kích hoạt | Khách chọn chức năng đăng đăng ký | | |
| Tiền điều kiện | Không có | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Khách | Chọn chức năng đăng ký. | | 2. | Hệ thống | Hiển thị giao diện đăng ký. | | 3. | Khách | Nhập các trường thông tin cá nhân. | | 4. | Khách | Yêu cầu đăng ký. | | 5. | Hệ thống | Kiểm tra xem khách đã nhập các trường bắt buộc hay chưa. | | 6. | Hệ thống | Kiểm tra email có tồn tại trong hệ thống chưa. | | 7. | Hệ thống | Gửi mã xác thực cho khách qua email và chuyển hướng đến trang nhập mã xác thực. | | 8. | Khách | Nhập mã xác thực mà hệ thống đã gửi qua email. | | 9. | Hệ thống | Kiểm tra mã xác thực có khớp với cơ sở dữ liệu. | | 10. | Hệ thống | Lưu thông tin tài khoản và hiển thị giao diện home. | | | |
| Các luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 5a. | Hệ thống | Thông báo lỗi: Cần nhập các rường bắt buộc nếu người dùng nhập thiếu. | | 6a. | Hệ thống | Thông báo lỗi: Đăng ký thất bại nếu email đã tồn tại trong cơ sở dữ liệu. | | 9a. | Hệ thống | Thông báo lỗi: Mã xác minh không chính xác. | | | |
| Hậu điều kiện | - Khách sau khi đăng ký thành công được chuyển hướng đến màn hình home | | |

### Đăng nhập

Bảng 3. Đặc tả use case – Đăng nhập

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC001 | Tên Use case | Đăng nhập |
| Tác nhân | Người dùng | | |
| Mô tả | Khách sử dụng email và mật khẩu của mình để đăng nhập vào hệ thống | | |
| Sự kiện kích hoạt | Khách chọn chức năng đăng nhập | | |
| Tiền điều kiện | Người dùng đã đăng ký tài khoản thành công. | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Người dùng | Chọn chức năng đăng nhập. | | 2. | Hệ thống | Hiển thị giao diện đăng nhập. | | 3. | Người dùng | Nhập email và mật khẩu. | | 4. | Người dùng | Yêu cầu đăng nhập. | | 5. | Hệ thống | Kiểm tra xem khách đã nhập các trường đúng yêu cầu chưa. | | 6. | Hệ thống | Kiểm tra email và mật khẩu | | 7. | Hệ thống | Hiển thị giao diện home | | | |
| Các luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 5a. | Hệ thống | Thông báo lỗi: Cần nhập các rường bắt buộc nếu người dùng nhập thiếu. | | 6a. | Hệ thống | Thông báo lỗi: Đăng nhập thất bại nếu không tìm thấy email và mật khẩu trong hệ thống | | | |
| Hậu điều kiện | - Người dùng sau khi đăng nhập thành công được chuyển hướng đến màn hình home. | | |

### Đăng xuất

Bảng 3. Đặc tả use case – Đăng xuất

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC002 | Tên Use case | Đăng xuất |
| Tác nhân | Người dùng, Quản trị viên | | |
| Mô tả | Người dùng yêu cầu đăng xuất | | |
| Sự kiện kích hoạt | Người dùng bấm nút đăng xuất | | |
| Tiền điều kiện | Người dùng đã đăng nhập vào hệ thống | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Người dùng | Chọn nút đăng xuất | | 2. | Hệ thống | Hệ thống huỷ bỏ phiên đăng nhập của người dùng. | | 3. | Hệ thống | Chuyển đến màn hình đăng nhập. | | | |
| Các luồng sự kiện thay thế | - Không có | | |
| Hậu điều kiện | - Khách sau khi đăng xuất thành công được chuyển hướng đến màn hình đăng nhập.  - Thông báo lỗi nếu đăng nhập thất bại. | | |

### Quản lý thông tin cá nhân

#### Sửa

Bảng 3. Đặc tả use case - Quản lý thông tin cá nhân (Sửa)

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC004-1 | Tên Use case | Quản lý thông tin cá nhân – Sửa |
| Tác nhân | Người dùng | | |
| Mô tả | Người dùng muốn yêu cầu thay đổi thông tin cá nhân. | | |
| Sự kiện kích hoạt | Người dùng nhấp vào nút sửa. | | |
| Tiền điều kiện | Người dùng đã đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Người dùng | Yêu cầu xem thông tin cá nhân | | 2. | Hệ thống | Hiển thị giao diện thông tin cá nhân của người dùng. | | 3. | Người dùng | Yêu cầu chỉnh sửa thông tin cá nhân. | | 4. | Hệ thống | Hiển thị giao diện chỉnh sửa. | | 5. | Người dùng | Nhập thông tin cá nhân mới. | | 5. | Hệ thống | Kiểm tra các trường đã chính xác hay chưa. | | 6. | Hệ thống | Hiển thị thông báo cập nhật thông tin thành công. | | | |
| Các luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 5a. | Hệ thống | Thông báo lỗi: Cần nhập các trường bắt buộc nếu người dùng nhập thiếu hoặc nhập không chính xác. | | | |
| Hậu điều kiện | - Hệ thống hiện thông báo khi người dùng sau khi cập nhật thông tin thành công.  - Thông báo lỗi nếu đăng ký tài khoản thất bại. | | |

#### Xoá tài khoản

Bảng 3. Đặc tả use case - Quản lý thông tin cá nhân (Xoá tài khoản)

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC004-2 | Tên Use case | Quản lý thông tin cá nhân – Xoá tài khoản |
| Tác nhân | Người dùng | | |
| Mô tả | Người dùng muốn xoá tài khoản khỏi hệ thống. | | |
| Sự kiện kích hoạt | Người dùng nhấp vào nút xoá. | | |
| Tiền điều kiện | Người dùng đã đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Người dùng | Yêu cầu xem thông tin cá nhân | | 2. | Hệ thống | Hiển thị giao diện thông tin cá nhân của người dùng. | | 3. | Người dùng | Yêu cầu xoá tài khoản. | | 5. | Hệ thống | Hiện thông báo hỏi người dùng có chắc chắn xoá tài khoản. | | 6. | Người dùng | Nhấn vào yêu cầu đồng ý xoá tài khoản. | | 6. | Hệ thống | Hiển thị thông báo thành công và chuyển về trang đăng nhập. | | | |
| Các luồng sự kiện thay thế | - Không có | | |
| Hậu điều kiện | - Hệ thống hiện thông báo xoá tài khoản thành công và chuyển hướng về màn hình đăng nhập.  - Hệ thống thông báo lỗi nếu xoá tài khoản thất bại. | | |

### Đổi mật khẩu

Bảng 3. Đặc tả use case - Quản lý thông tin cá nhân (Đổi mật khẩu)

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC005 | Tên Use case | Đổi mật khẩu |
| Tác nhân | Người dùng | | |
| Mô tả | Người dùng muốn đổi mật khẩu. | | |
| Sự kiện kích hoạt | Người dùng chuyển hướng đến màn hình đổi mật khẩu. | | |
| Tiền điều kiện | Người dùng đã đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Người dùng | Yêu cầu đổi mật khẩu. | | 2. | Hệ thống | Hiển thị giao diện nhập mật khẩu mới. | | 3. | Người dùng | Yêu cầu đổi mật khẩu. | | 5. | Hệ thống | Kiểm tra mật khẩu. | | 6. | Hệ thống | Hiển thị thông báo đổi mật khẩu thành công. | | | |
| Các luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 5a. | Hệ thống | Thông báo lỗi: Mật khẩu cũ không chính xác. | | 5b. | Hệ thống | Thông báo lỗi: Mật khẩu mới và xác nhận mật khẩu mới không trùng khớp. | | | |
| Hậu điều kiện | - Hệ thống hiện thông báo thay đổi mật khẩu thành công.  - Thông báo lỗi nếu thay đổi mật khẩu thất bại. | | |

### Chat

Bảng 3. Đặc tả use case - Chat

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC006 | Tên Use case | Chat |
| Tác nhân | Người dùng | | |
| Mô tả | Người dùng chat với bot | | |
| Sự kiện kích hoạt | Người dùng nhập tin nhắn và nhấn gửi. | | |
| Tiền điều kiện | Người dùng đã đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Người dùng | Nhập tin nhắn và gửi. | | 2. | Hệ thống | Xử lý tin nhắn người dùng và trả lời tin nhắn phù hợp với tin nhắn người dùng. | | | |
| Các luồng sự kiện thay thế | - Không có | | |
| Hậu điều kiện | - Không có | | |

### Quản lý danh sách người dùng

#### Tìm kiếm người dùng

Bảng 3. Quản lý danh sách người dùng - Tìm kiếm người dùng

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC007-1 | Tên Use case | Tìm kiếm người dùng |
| Tác nhân | Quản trị viên | | |
| Mô tả | Quản trị viên tìm kiếm nhân viên và hiển thị danh sách tìm được | | |
| Sự kiện kích hoạt | Quản trị viên chọn chức năng tìm kiếm nhân viên | | |
| Tiền điều kiện | Đăng nhập thành công với vai trò Quản trị viên | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Quản trị viên | Chọn chức năng tìm kiếm người dùng. | | 2. | Hệ thống | Lấy danh sách người dùng và hiển thị ô tìm kiếm. | | 3. | Quản trị viên | Nhập dữ liệu vào ô tìm kiếm. | | 4. | Quản trị viên | Yêu cầu tìm kiếm. | | 5. | Hệ thống | Kiểm tra xem quản trị viên đã nhập vào ô tìm kiếm chưa. | | 6. | Hệ thống | Tìm và lấy về thông tin những người dùng thỏa mãn yêu cầu tìm kiếm. | | 7. | Hệ thống | Hiện thị danh sách những người dùng thoả mãn yêu cầu tìm kiếm. | | | |
| Các luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 7a. | Hệ thống | Thông báo: Không tìm thấy người dùng. | | | |
| Hậu điều kiện | - Trả về thành công danh sách người dùng  - Thông báo không tìm thấy dữ liệu nếu danh sách rỗng | | |

#### Cập nhật thông tin người dùng

Bảng 3. Quản lý danh sách người dùng - Cập nhật thông tin người dùng

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC007-2 | Tên Use case | Cập nhật thông tin người dùng. |
| Tác nhân | Quản trị viên | | |
| Mô tả | Quản trị viên nhân cập nhật thông tin của người dùng. | | |
| Sự kiện kích hoạt | Quản trị viên chọn vào nguời dùng và nhấn cập nhật. | | |
| Tiền điều kiện | Đăng nhập thành công với vai trò Quản trị viên | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Quản trị viên | Yêu cầu chức năng cập nhật người dùng. | | 2. | Hệ thống | Lấy danh sách người dùng. | | 3. | Quản trị viên | Nhấn vào người dùng và chọn cập nhật. | | 4. | Hệ thống | Hiển thị thông tin chi tiết của người dùng. | | 5. | Quản trị viên | Nhập thông tin cập nhật. | | 6. | Hệ thống | Kiểm tra thông tin cập nhật của người dùng. | | 7. | Hệ thống | Hiển thị thông báo cập nhật người dùng thành công. | | | |
| Các luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 6a. | Hệ thống | Thông báo: Hiển thị thông báo nếu quản trị viện nhập thông tin cập nhật lỗi. | | 7a. | Hệ thống | Thông báo: Hiển thị thông báo thất bại nếu cập nhật thất bại. | | | |
| Hậu điều kiện | - Trả về thông báo cập nhật thành công.  - Thông báo lỗi nếu cập nhật thất bại. | | |

#### Xoá người dùng

Bảng 3. Quản lý danh sách người dùng - Xoá người dùng

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC007-3 | Tên Use case | Xoá người dùng |
| Tác nhân | Quản trị viên | | |
| Mô tả | Quản trị viên yêu cầu xoá người dùng. | | |
| Sự kiện kích hoạt | Quản trị viên nhấn chọn người dùng và nhấn xoá. | | |
| Tiền điều kiện | Đăng nhập thành công với vai trò Quản trị viên | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Quản trị viên | Yêu cầu xoá người dùng. | | 2. | Hệ thống | Lấy danh sách người dùng. | | 3. | Quản trị viên | Nhấn vào người dùng và chọn xoá. | | 4. | Hệ thống | Hiển thị thông báo hỏi có muốn xoá người dùng không. | | 5. | Quản trị viên | Nhấn nút đồng ý xoá. | | 6. | Hệ thống | Xoá người dùng. | | 7. | Hệ thống | Hiển thị thông báo xoá người dùng thành công. | | | |
| Các luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 7a. | Hệ thống | Thông báo: Hiển thị thông báo nếu xoá người dùng thất bại. | | | |
| Hậu điều kiện | - Trả về thông báo xoá thành công và cập nhật lại danh sách người dùng.  - Thông báo lỗi nếu xoá người dùng thất bại. | | |

### Quản lý danh sách thiết bị

#### Tìm kiếm thiết bị

Bảng 3. Quản lý danh sách thiết bị - Tìm kiếm thiết bị

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC008-1 | Tên Use case | Tìm kiếm thiết bị |
| Tác nhân | Quản trị viên, Người dùng | | |
| Mô tả | Quản trị viên, người dùng tìm kiếm thiết bị và hiển thị danh sách tìm kiếm. | | |
| Sự kiện kích hoạt | Quản trị viên, người dùng chọn chức năng tìm kiếm thiết bị. | | |
| Tiền điều kiện | - Quản trị viên đăng nhập thành công.  - Người dùng đăng nhập thành công. | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Quản trị viên  Người dùng | Chọn chức năng tìm kiếm thiết bị. | | 2. | Quản trị viên  Người dùng | Nhập dữ liệu vào ô tìm kiếm. | | 3. | Quản trị viên  Người dùng | Yêu cầu tìm kiếm. | | 4. | Hệ thống | Kiểm tra xem quản trị viên (người dùng) đã nhập vào ô tìm kiếm chưa. | | 5. | Hệ thống | Tìm và lấy về thông tin những thiết bị thỏa mãn yêu cầu tìm kiếm. | | 6. | Hệ thống | Hiện thị danh sách những thiết bị thoả mãn yêu cầu tìm kiếm. | | | |
| Các luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 6a. | Hệ thống | Thông báo: Không tìm thấy thiết bị. | | | |
| Hậu điều kiện | - Trả về thành công danh sách thiết bị  - Thông báo không tìm thấy dữ liệu nếu danh sách rỗng | | |

#### Thêm mới thiết bị

Bảng 3. Quản lý danh sách thiết bị - Thêm mới thiết bị

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC008-2 | Tên Use case | Thêm mới thiết bị |
| Tác nhân | Người dùng | | |
| Mô tả | Người dùng thêm mới thiết bị | | |
| Sự kiện kích hoạt | Người dùng chọn chức năng thêm mới thiết bị. | | |
| Tiền điều kiện | Người dùng đã đăng nhập thành công. | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Người dùng | Chọn chức năng thêm thiết bị. | | 2. | Hệ thống | Quét danh sách thiết bị. | | 3. | Hệ thống | Hiển thị danh sach thiết bị quét được. | | 4. | Người dùng | Chọn thiết bị kết nối. | | 5. | Người dùng | Nhập thông tin để cấu hình thiết bị. | | 6. | Hệ thống | Kết nối thiết bị. | | 7. | Hệ thống | Trả kết quả kết nối thiết bị thành công. | | | |
| Các luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 7a. | Hệ thống | Thông báo: Kết nối thiết bị thất bại. | | | |
| Hậu điều kiện | - Trả về thông báo kết nối thiết bị thành công  - Thông báo kết nối thiết bị thất bại nếu không thể kết nối thiết bị | | |

#### Xoá thiết bị

Bảng 3. Quản lý danh sách thiết bị Xoá thiết bị

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC008-3 | Tên Use case | Xoá thiết bị |
| Tác nhân | Quản trị viên, người dùng | | |
| Mô tả | Quản trị viên (hoặc người dùng) yêu cầu xoá người thiết bị. | | |
| Sự kiện kích hoạt | Quản trị viên (hoặc người dùng) nhấn chọn thiết bị và nhấn xoá. | | |
| Tiền điều kiện | Quản trị viên hoặc người dùng đã đăng nhập thành công | | |
| Luồng sự kiện chính | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1. | Quản trị viên  Người dùng | Yêu cầu xoá thiết bị. | | 2. | Hệ thống | Lấy danh sách thiết bị. | | 3. | Quản trị viên  Người dùng | Nhấn vào thiết bị và chọn xoá. | | 4. | Hệ thống | Hiển thị thông báo hỏi có chắc chắn xoá thiết bị không. | | 5. | Quản trị viên  Người dùng | Nhấn nút đồng ý xoá. | | 6. | Hệ thống | Xoá thiết bị. | | 7. | Hệ thống | Hiển thị thông báo xoá thiết bị thành công. | | | |
| Các luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 7a. | Hệ thống | Thông báo: Hiển thị thông báo nếu xoá thiết bị thất bại. | | | |
| Hậu điều kiện | - Trả về thông báo xoá thành công và cập nhật lại danh sách thiết bị.  - Thông báo lỗi nếu xoá thiết bị thất bại. | | |

## Biểu đồ tuần tự

### Biểu đồ tuần tự đăng ký

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự đăng ký

### Biểu đồ tuần tự đăng nhập

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự đăng nhập

### Biểu đồ tuần tự đăng xuất

Diagram

Description automatically generated with low confidence

Hình . Biểu đồ tuần tự đăng xuất

### Biểu đồ tuần tự chat

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự chat

### Biểu đồ tuần tự quản lý thông tin cá nhân

#### Cập nhật

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự quản lý thông tin cá nhân - Cập nhật

#### Xoá tài khoản

Diagram

Description automatically generated with low confidence

Hình . Biểu đồ tuần tự quản lý thông tin cá nhân - Xoá tài khoản

### Đổi mật khẩu

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự quản lý thông tin cá nhân - Đổi mật khẩu

### Biểu đồ tuần tự quản lý danh sách thiết bị của người dùng

#### Xem danh sách thiết bị

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự quản lý danh sách thiết bị của người dùng - Xem danh sách thiết bị

#### Thêm mới thiết bị

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự quản lý danh sách thiết bị của người dùng - Thêm mới thiết bị

#### Tìm kiếm thiết bị

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự quản lý danh sách thiết bị của người dùng - Tìm kiếm thiết bị

#### Xoá thiết bị

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự quản lý danh sách thiết bị của người dùng - Xoá thiết bị

### Biểu đồ tuần tự quản lý danh sách thiết bị của quản trị viên

#### Tìm kiếm thiết bị

A picture containing diagram

Description automatically generated

Hình . Biểu đồ tuần tự quản lý danh sách thiết bị của quản trị viên - Tìm kiếm thiết bị

#### Xem danh sách thiết bị

Calendar

Description automatically generated with medium confidence

Hình . Biểu đồ tuần tự quản lý danh sách thiết bị của quản trị viên - Xem danh sách thiết bị

#### Xoá thiết bị

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự quản lý danh sách thiết bị của quản trị viên - Xoá thiết bị

### Biểu đồ tuần tự quản lý danh sách người dùng

#### Tìm kiếm người dùng

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự quản lý danh sách người dùng - Tìm kiếm người dùng

#### Cập nhật thông tin người dùng

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự quản lý danh sách người dùng - Cập nhật thông tin người dùng

#### Xoá người dùng

Diagram

Description automatically generated with medium confidence

Hình . Biểu đồ tuần tự quản lý danh sách người dùng - Xoá người dùng

## Biểu đồ lớp

### Biểu đồ lớp ca sử dụng đăng nhập

Diagram

Description automatically generated

Hình . Biểu đồ lớp ca sử dụng đăng nhập

### Biểu đồ lớp ca sử dụng chat

**Diagram

Description automatically generated**

Hình . Biểu đồ lớp ca sử dụng chat

### Biểu đồ lớp ca sử dụng quản lý thiết bị

**Diagram

Description automatically generated**

Hình . Biểu đồ lớp ca sử dụng quản lý thiết bị

### Biểu đồ lớp ca sử dụng quản lý người dùng

Diagram

Description automatically generated

Hình . Biểu đồ lớp ca sử dụng quản lý người dùng

## Biểu đồ hoạt động

### Đăng ký

Diagram

Description automatically generated

Hình . Biểu đồ hoạt động - Đăng ký

### Đăng nhập

Diagram

Description automatically generated

Hình . Biểu đồ hoạt động - Đăng nhập

### Chat

Diagram

Description automatically generated

Hình . Biểu đồ hoạt động - Chat

### Ghép nối thiết bị

Diagram

Description automatically generated

Hình . Biểu đồ hoạt động - Ghép nối thiết bị

## Biểu đồ component

Diagram

Description automatically generated

Hình . Biểu đồ component

Chia hệ thống thành các SubSystem như sau:

* Administrator Web: Phần website phục vụ cho quản trị viên.
* Customer App: Phần app phục vụ cho người dùng.
* Business ChatBot: Phần tương tác với người dùng và truy cập cơ sở dữ liệu Chatbot.
* ChatBot Server: Server chứa thông tin cơ sở dữ liệu Chatbot.
* Business Database: Phần tương tác với quản trị viên và nhận thông tin xử lý từ Business Chatbot cùng với đó truy cập cơ sở dữ liệu.
* Database Server: Phần chứa thông tin cơ sở dữ liệu người dùng.
* ThinkBoard Server: Phần chứa thông tin của module wifi.
* Module wifi: Phần xử lý tín hiệu được gửi từ ThinkBoard Server.

## Thiết kế cơ sở dữ liệu

Diagram

Description automatically generated

Hình . Thiết kế cơ sở dữ liệu

## Thiết kế chi tiết các bảng dữ liệu

* Bảng Customer

Bảng 3. Bảng Customer

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Ý nghĩa | Bắt buộc |
| id | VARCHAR(32) | Khóa của bảng | T |
| email | VARCHAR(50) | Email của người dùng | T |
| sex | ENUM('male', 'female') | Giới tính của người dùng | T |
| firstName | VARCHAR(50) | Tên của người dùng | T |
| lastName | VARCHAR(50) | Họ của người dùng | T |
| avatar | LONGTEXT | Hình ảnh của người dùng | T |
| password | VARCHAR(50) | Password của người dùng | T |
| otpSecret | VARCHAR(32) | Mã OTP | T |
| isActive | TINYINT | Trạng thái hoạt động | T |
| deleted | TINYINT | Trạng thái xoá | T |
| createAt | DATETIME | Ngày tạo | T |
| updateAt | DATETIME | Ngày cập nhật | T |

* Bảng Admin

Bảng 3. Bảng Admin

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Ý nghĩa | Bắt buộc |
| id | VARCHAR(32) | Khóa của bảng | T |
| email | VARCHAR(50) | Email của người dùng | T |
| password | VARCHAR(50) | Password của người dùng | T |
| createAt | DATETIME | Ngày tạo | T |
| updateAt | DATETIME | Ngày cập nhật | T |

* Bảng Device

Bảng 3. Bảng Device

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Ý nghĩa | Bắt buộc |
| id | VARCHAR(32) | Khóa của bảng | T |
| name | VARCHAR(32) | Tên của device | T |
| status | TINYINT | Trạng thái của device | T |
| deleted | TINYINT | Trạng thái xoá | T |
| createAt | DATETIME | Ngày tạo | T |
| updateAt | DATETIME | Ngày cập nhật | T |

* Bảng customerDevices

Bảng 3.17 Bảng CustomerDevice

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Ý nghĩa | Bắt buộc |
| id | VARCHAR(32) | Khóa của bảng | T |
| customerId | VARCHAR(32) | Id của customer | T |
| deviceId | VARCHAR(32) | Id của device | T |
| deleted | TINYINT | Trạng thái xoá | T |
| createAt | DATETIME | Ngày tạo | T |
| updateAt | DATETIME | Ngày cập nhật | T |

* Bảng Message

Bảng 3. Bảng Message

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Ý nghĩa | Bắt buộc |
| id | VARCHAR(32) | Khóa của bảng | T |
| customerId | VARCHAR(32) | Id của người dùng | T |
| isBot | TINYINT | Kiểm tra xem tin nhắn là của người dùng hay phản hồi của bot | T |
| message | LONGTEXT | Message của người dùng (hoặc Bot) | T |
| deleted | TINYINT | Trạng thái xoá | T |
| createAt | DATETIME | Ngày tạo | T |
| updateAt | DATETIME | Ngày cập nhật | T |

# TRIỂN KHAI

## Giao diện Chatbot

* Màn hình đăng nhập và đăng ký

Graphical user interface, application

Description automatically generatedGraphical user interface, text, application

Description automatically generated

Hình . Màn hình đăng nhập và đăng ký

* Màn hình đổi mật khẩu và sửa thông tin tài khoản

Graphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated

Hình . Màn hình sửa mật khẩu và thông tin tài khoản

* Màn hình chat và quản lý thiết bị

Graphical user interface, text, application, chat or text message

Description automatically generatedA picture containing graphical user interface

Description automatically generated

Hình . Màn hình Chat và quản lý thiết bị

* Thông báo xoá thiết bị, người dùng

Graphical user interface, text, application, chat or text message

Description automatically generatedGraphical user interface, application

Description automatically generated

Hình .4 Thông báo xoá thiết bị, người dùng

Cơ chế hoạt động:

* Người dùng sau khi đăng ký và nhập mã verification code tương ứng thống qua email có thể đăng nhập và hệ thống.
* Sau khi đăng nhập thì người dùng có thể chat trực tiếp với bot thông qua socketio. Người dùng cũng có tuỳ chọn chỉnh sửa thông tin cá nhân để cập nhật thông tin cá nhân tương ứng. Người dùng có thể chat trực tiếp qua âm thanh hoặc nhập trực tiếp vào ô chatbox để tương tác với bot.
* Về phần thiết bị người dùng có thể thêm thiết bị trực tiếp trong màn hình thiết bị. Để thêm thiết bị thì chúng ta reset thiết bị để đưa thiết bị về chế độ Access Point. Sau khi người dùng nhập tên wifi và mật khẩu chính xác thì sẽ gửi một gói tin UDP đến Access Point. Module ESP32 nhận gói tin và thiết lập kết nối Internet. Sau khi kết nối thành công thì ESP32 chuyển về chế độ Station và kết nối với mạng đã được cài đặt thực hiện đồng bộ dữ liệu từ Firebase realtime database để bật tắt. Người dùng sau khi kết nối thành công sẽ điều khiển được thiết bị thông qua cập nhật giá trị vào cơ sở dữ liệu firebase realtime database

## Rasa

* Các loại thực thể trong phạm vi đồ án:

Bảng 4.1 Các loại thực thể tiêu biểu

|  |  |
| --- | --- |
| enword | handsome, love, good,… |
| country | Việt Nam, Thái Lan,… |
| status | bật, tắt |
| device | Bóng đèn, công tắc, ổ cắm,… |
| city | Hà Nội, Nghệ An, Bắc Giang,… |

* Các ý định tiêu biểu mà em sử dụng:

Bảng 4.2 Các ý định tiêu biểu

|  |  |
| --- | --- |
| look\_up\_english | Tra cứu từ tiếng anh |
| ask\_covid\_all | Tra cứu tình hình diễn biến dịch covid trên toàn thế giới |
| ask\_covid\_by\_country | Tra cứu tình hình diễn biến covid ở mỗi nước |
| ask\_devices\_list | Kiểm tra danh sách thiết bị |
| ask\_control\_device | Điều khiển thiết bị |
| ask\_weather | Hỏi thông tin thời tiết |
| ask\_lottery | Hỏi kết quả xổ số |

Và còn một số ý định khác như chào hỏi, cảm ơn, hỏi tuổi, hát, ….

Với mỗi ý định, rasa sẽ trích xuất thông tin của thực thể, dựa vào ý định và thực thể để quyết định câu hỏi trả về cho người dùng. Mỗi ý định đều có một action tương ứng, action đấy có thể tự custom có thể lấy dữ liệu từ bên thứ ba, hoặc xử lý logic để trả thông tin về cho người dùng.

Ví dụ về việc sử dụng custom action cho ý định ask\_covid\_all và ask\_covid\_by\_country, em sử dụng api COVID-19 data của rapidapi để tìm kiếm thông tin dịch bệnh trên toàn thế giới, hay ý định ask\_devices\_list em lấy thông tin thiết bị từ cơ sở dữ liệu để gửi ra cho người dùng

### Pipeline Components

Về thành phần đường ống đào tạo được em sử dụng

* VietnameseTokenizer

Tokenizer với nhiệm vụ tách từ để tạo ra các token, do ngôn ngữ tiếng việt rất phong phú, bao gồm các cụm từ, tên riêng nên em đã sử dụng thư viện python underthesea khá phổ biến và hỗ trợ rất tốt việc tách từ tiếng Việt thành các token. Ví dụ như trong dữ liệu đào tạo, em tách câu đầu vào của người dùng thành các token như sau.

Đầu vào: “cho tôi xem thời tiết ở Hà Nội”

Đầu ra: “[cho] [tôi] [xem] [thời tiết] [ở] [Hà Nội]”

* **CountVectorsFeaturizer**

**CountVectorsFeaturizer tập trung vào số lần xuất hiện trong văn bản không quan tâm đến vị trí của từ đó trong văn bản, đếm tần suất của các từ trong dữ liệu đào tạo để từ đó làm đầu vào cho bộ phân loại ý định.**

Ví dụ với đầu vào của 2 văn bản là:

“nhà bạn ở đâu” => [1, 1, 1, 0, 0 ]

“địa chỉ nhà bạn chỗ nào” => [1, 1, 0, 1, 1]

[“nhà”, “bạn”, “ở đâu”, “địa chỉ”, “chỗ nào”]

* BytePairFeaturizer

**Giống như CountVectorsFeaturizer,** BytePairFeaturizer cũng trích xuất vector đặc trưng nhưng hỗ trợ nhiều ngôn ngữ trong đó có tiếng Việt và bộ từ vựng em cài đặt là 2000 và số chiều là 50 trả về vector đặc trưng dày đặc làm đầu vào cho pha sau

* Bộ phân loại ý định và trích xuất thực thể.

Trong đồ án em sử dụng bộ phân loại ý định và trích xuất thực thể DIETClassifier cùng với bộ trích xuất thực thể CRFEntityExtractor, RegexEntityExtractor lấy kết quả của 3 pha trước đó làm đầu vào và đầu ra. Bộ phân loại ý định yêu cầu vector đặc trưng trích xuất thực thể yêu cầu token. Ví dụ về phân loại ý định và trích xuất thực thể:

* + Độ tin cậy khi gửi tin nhắn với nội dung: “tình hình dịch bệnh thế giới”

|  |
| --- |
| {  "text": "tình hình dịch bệnh thế giới",  "intent": {  "name": "ask\_covid\_all",  "confidence": 0.8365494608879089  },  "entities": [],  "intent\_ranking": [  {  "name": "ask\_covid\_all",  "confidence": 0.8365494608879089  },  {  "name": "ask\_covid\_by\_country",  "confidence": 0.05365508422255516  },  {  "name": "deny",  "confidence": 0.024683210998773575  },  …  ]  } |

* + Độ tin cậy khi gửi tin nhắn với nội dung: “thời tiết ở Hà Nội”

|  |
| --- |
| {  "text": "thời tiết ở Hà Nội",  "intent": {  "name": "ask\_weather",  "confidence": 0.9829024076461792  },  "entities": [  {  "entity": "city",  "confidence\_entity": 0.9128881927976337,  "value": "Hà Nội",  "extractor": "CRFEntityExtractor"  },  {  "entity": "city",  "value": "Hà Nội",  "extractor": "RegexEntityExtractor"  },  {  "entity": "city",  "confidence\_entity": 0.9975904226303101,  "value": "Hà Nội",  "extractor": "DIETClassifier"  }  ],  "intent\_ranking": [  {  "name": "ask\_weather",  "confidence": 0.9829024076461792  },  {  "name": "introduce",  "confidence": 0.002703056437894702  },  …  ]  } |

### Đầu ra

Rasa sử dụng rules và stories để làm trách nhiệm xử lý đầu ra cho tin nhắn văn bản của người dùng. Ngoài ra, Rasa còn sử dụng các slot để lưu các giá trị của thực thể nhằm cung cấp cho phần xử lý thông tin đầu ra như gọi các action, trả kết quả về cho người dùng.

Ví dụ về stories Rasa

|  |
| --- |
| - story: ask\_information path  steps:  - intent: what\_name  - action: utter\_what\_name  - intent: how\_old  - action: utter\_how\_old  - intent: gender  - action: utter\_gender  - intent: address  - action: utter\_address  - story: control device  steps:  - intent: ask\_control\_device  - action: action\_control\_device |

Với càng nhiều số lượng kịch bản thì càng tăng độ thông minh cho bot và dự đoán càng tốt các khả năng xảy ra từ đó đưa ra các câu trả lời có độ chính xác cao.

## Mạch điều khiển thiết bị

Diagram, schematic

Description automatically generated

Hình . Mạch điều khiển thiết bị

* Các thành phần chính em sử dụng:

Bảng 4. Các thành phần trong mạch

|  |  |
| --- | --- |
| ESP32 | Module Wifi |
| ASM1117 | IC ổn áp giúp hạ dòng 5v xuống 3,3v |
| Opto PC817 | Cách ly dòng 5v và dòng 3,3v |
| Transistor NPN | Kích hoạt chế độ bật tắt |
| Rơle | Công tắc bật tắt đèn, cách ly dòng xoay chiều và dòng 1 chiều |
| Điot | Tránh thất thoát dòng điện làm hỏng mạch |

* Hình ảnh thiết bị

A picture containing cable, connector

Description automatically generated

Hình . Hình ảnh thiết bị

* Cơ chế hoạt động

Sau khi nạp code cho thiết bị, em cung cấp dòng đầu vào 5V cho rơle thông qua jack cắm DC và một phần khác dùng IC ổn áp ASM1777 chuyển dòng từ 5V xuống 3,3V để cung cấp cho module ESP32.

Chân GPIO 4 và GPIO 22 em sử dụng làm chân đầu ra của ESP32 Mạch sẽ nhận dữ liệu realtime từ Firebase realtime database và cập nhật trạng thái vào GPIO mà em đã cài đặt thông qua đó điều khiển bật (tắt) rơle. Ngoài ra trên thiết bị của em bao gồm 3 nút bấm, 1 nút bấm RESET, 1 nút nạp code và 1 nút để điều khiển thiết bị. trạng thái của thiết bị đều được lưu trên Firebase realtime database và sẽ được điều khiển thông qua ứng dụng chat và cũng có thể điều khiển trực tiếp thông qua nút bấm.

## Giao diện quản trị viên

Graphical user interface, application, Teams

Description automatically generated

Hình 4. Admin - danh sách người dùng

Graphical user interface, application

Description automatically generated

Hình . Admin - danh sách thiết bị

Graphical user interface, application

Description automatically generated

Hình . Admin - chỉnh sửa người dùng

Graphical user interface, application, Teams

Description automatically generated

Hình . Admin - xoá người dùng

## Giao diện Firebase Realtime database

Graphical user interface, text, application, email

Description automatically generated

Hình 4.11 Giao diện Firebase realtime database

Dữ liệu từ Firebase realtime database có thể cập nhật được từ app người dùng, website của quản trị viên và Rasa

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

Những kết quả em đã đạt được trong quá trình phát triển đồ án:

* Xây dựng một ứng dụng chatbot chạy trên hệ điều hành Android và Ios trả lời được các tin nhắn cơ bản và có khả năng điều khiển thiết bị
* Tìm hiểu cơ bản về thư viện mã nguồn mở Rasa để xây dựng chatbot
* Thiết kế mạch cơ bản hỗ trợ điều khiển bật tắt dựa vào module wifi ESP32
* Hỗ trợ smartconfig lên ESP32 trên ứng dụng điện thoại

Những hạn chế mà em còn gặp phải

* Số lượng dữ liệu đào tạo còn ít, bot chưa được thông minh lắm
* Mạch thiết kế khá cồng kềnh, chưa tích hợp module giảm áp khiến mạch phải dùng tới 2 ổ cắm
* Chưa thể điều khiển 2 công tắc cùng một lúc, chỉ có thể điều khiển được 1 trên mỗi mạch

## Hướng phát triển đồ án trong tương lai

Hướng phát triển để hệ thống có khả năng tích hợp vào cuộc sống

* Thêm dữ liệu đào tạo để bot có thể làm được nhiều khả năng hơn
* Giảm chi phí mạch để phù hợp với túi tiền và nhu cầu người dùng
* Tìm tòi và phát triển để chatbot có thể điều khiển được nhiều công tắc trên cũng một mạch

# TÀI LIỆU THAM KHẢO

[1] <https://rasa.com/docs/>

[2] <https://github.com/mobizt/Firebase-ESP32>

[3] <https://firebase.google.com/>

[4]<https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>

[5] <https://reactjs.org/>

[6] <https://reactnative.dev/>

[7] <https://www.docker.com/>