HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG



Trần Nguyên Thịnh

NGHIÊN CỨU VÀ PHÁT TRIỂN ỨNG DỤNG CHATBOT GIÚP QUẢN LÝ THỜI GIAN CỦA NHÂN VIÊN TẠI FPT SOFTWARE

ĐỀ ÁN TỐT NGHIỆP THẠC SĨ KỸ THUẬT

(Theo định hướng ứng dụng)

HÀ NỘI – NĂM 2026

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG



Trần Nguyên Thịnh

NGHIÊN CỨU VÀ PHÁT TRIỂN ỨNG DỤNG CHATBOT GIÚP QUẢN LÝ THỜI GIAN CỦA NHÂN VIÊN TẠI FPT SOFTWARE

CHUYÊN NGÀNH: HỆ THỐNG THÔNG TIN

MÃ SỐ: 8.48.01.04

ĐỀ ÁN TỐT NGHIỆP THẠC SĨ KỸ THUẬT
(Theo định hướng ứng dụng)

NGƯỜI HƯỚNG DẪN KHOA HỌC:
TS. PHAN THỊ HÀ

HÀ NỘI – NĂM 2026

LỜI CAM ĐOAN

Tôi cam đoan đây là công trình nghiên cứu của riêng tôi.

1. Các số liệu, kết quả nêu trong đề án tốt nghiệp là hoàn toàn trung thực và chưa từng có ai công bố trong bất kỳ công trình nghiên cứu nào khác.
2. Tất cả nghiên cứu/số liệu của người khác trong đề án này đều được trích dẫn theo đúng quy định.

Hà Nội, ngày 04 tháng 05 năm 2026

Học viên thực hiện đề án



Trần Nguyên Thịnh

LỜI CẢM ƠN

Để hoàn thành chương trình Cao học và bản báo cáo thực tập tốt nghiệp này, ngoài sự cố gắng của bản thân, tôi đã nhận được rất nhiều sự quan tâm, chỉ dẫn và khích lệ quý báu từ các Thầy Cô, gia đình và bạn bè.

Trước tiên, tôi xin gửi lời cảm ơn trân trọng đến **Ban Giám đốc Học viện Công nghệ Bưu chính Viễn thông, Khoa Đào tạo Sau đại học cùng toàn thể Quý Thầy Cô** đã tận tình giảng dạy, truyền đạt những kiến thức chuyên môn sâu sắc trong suốt quá trình học tập. Những tri thức này chính là nền tảng quan trọng giúp tôi có thể triển khai và hoàn thành nghiên cứu của mình.

Đặc biệt, tôi xin bày tỏ lòng biết ơn sâu sắc nhất đến **TS. Phan Thị Hà**, người Cô đã trực tiếp hướng dẫn tôi trong suốt quá trình thực hiện bài báo cáo thực tập tốt nghiệp. Cô đã tận tình định hướng đề tài, giải đáp thắc mắc, cung cấp tài liệu cần thiết và đưa ra nhiều góp ý quý giá, giúp tôi từng bước hoàn thiện và đạt được kết quả như hôm nay.

Cuối cùng, tôi xin chân thành cảm ơn **gia đình, người thân và bạn bè** đã luôn đồng hành, động viên và tạo mọi điều kiện thuận lợi để tôi có thể yên tâm học tập và nghiên cứu.

Mặc dù đã rất nỗ lực, song do hạn chế về thời gian và kiến thức, báo cáo khó tránh khỏi những thiếu sót. Tôi rất mong nhận được sự góp ý của Quý Thầy Cô để báo cáo này được hoàn thiện hơn.

Một lần nữa, tôi xin trân trọng cảm ơn!

Hà Nội, ngày 04 tháng 5 năm 2026

Học viên thực hiện đề án

Trần Nguyên Thịnh

MỤC LỤC

LỜI CAM ĐOAN i
LỜI CẢM ƠN ii
MỤC LỤC iii
DANH MỤC CÁC THUẬT NGỮ, CHỮ VIẾT TẮT vi
DANH SÁCH BẢNG viii
DANH SÁCH HÌNH VẼ viii
I. MỞ ĐẦU ix
1. Lý do chọn đề tài 1
2. Tổng quan về vấn đề nghiên cứu 2
3. Mục đích nghiên cứu 3
4. Đối tượng và phạm vi nghiên cứu 4
5. Phương pháp nghiên cứu 5
II. NỘI DUNG 7
CHƯƠNG 1: TỔNG QUAN VỀ QUẢN LÝ THỜI GIAN VÀ ỨNG DỤNG CÔNG NGHỆ CHATBOT TRONG DOANH NGHIỆP 7
1.1 Khái niệm và vai trò của quản lý thời gian 7
1.1.1 Khái niệm quản lý thời gian và tầm quan trọng của nó trong doanh nghiệp 7
1.1.2 Các phương pháp quản lý thời gian truyền thống và những hạn chế 8
1.2 Tổng quan về chatbot và các ứng dụng trong doanh nghiệp 10
1.2.1 Khái niệm chatbot, các công nghệ nền tảng như AI và NLP 10
1.2.2 Lịch sử phát triển và ứng dụng chatbot trong các lĩnh vực khác nhau 11
1.2.3 Vai trò của chatbot trong quản lý thời gian và hiệu quả công việc 12
1.3 Nghiên cứu ứng dụng chatbot trong quản lý thời gian tại FPT Software 14
1.3.1 Khảo sát về ứng dụng chatbot đang được sử dụng ở FPT Software (Maya) 14
1.3.2 Phân tích những điểm đã đạt được và cần cải tiến thêm 15
1.3.3 Phân tích tính khả thi của module quản lý thời gian với ứng dụng chatbot trong FPT Software 16
1.4 Kết luận chương 1 17
CHƯƠNG 2: NGHIÊN CỨU CÁC CÔNG NGHỆ NỀN TẢNG TRONG PHÁT TRIỂN CHATBOT 19
2.1 Trí tuệ nhân tạo (AI) và học máy (Machine Learning) 19

2.1.1 Khái niệm AI và vai trò của AI trong phát triển chatbot 19
2.1.2 Ứng dụng của học máy trong việc cải thiện khả năng tương tác và học hỏi của chatbot 19
2.2 Xử lý ngôn ngữ tự nhiên (NLP) 22
2.2.1 Tổng quan về NLP và các công nghệ liên quan 22
2.2.2 Vai trò của NLP trong việc hiểu và phản hồi các yêu cầu của người dùng 24
2.2.3 Phân tích các mô hình NLP hiện đại được sử dụng trong chatbot 24
2.3 RASA Framework 26
2.3.1 RASA Core 26
2.3.2 RASA NLU (Natural Language Understanding) 27
2.3.3 Các thành phần hệ thống chatbot RASA 28
2.3.4 Tìm hiểu luồng hoạt động chi tiết của RASA 29
2.4 Tích hợp và triển khai chatbot trong doanh nghiệp 30
2.4.1 Các phương pháp tích hợp chatbot vào hệ thống quản lý doanh nghiệp 30
2.4.2 Những thách thức và giải pháp khi triển khai chatbot trong môi trường doanh nghiệp thực tế 31
2.5 Kết luận chương 2 33
CHƯƠNG 3: THIẾT KẾ VÀ PHÁT TRIỂN HỆ THỐNG CHATBOT HỖ TRỢ QUẢN LÝ THỜI GIAN TẠI FPT SOFTWARE 35
3.1 Khảo sát nhu cầu và yêu cầu hệ thống 35
3.1.1 Khảo sát nhu cầu quản lý thời gian của nhân viên tại FPT Software 35
3.1.2 Xác định các yêu cầu chức năng và phi chức năng của hệ thống chatbot 36
3.2 Thiết kế hệ thống chatbot 37
3.2.1 Kiến trúc tổng thể của hệ thống chatbot 37
3.2.2 Thiết kế các mô-đun chính của chatbot 40
3.2.3 Thiết kế quy trình tương tác giữa chatbot và người dùng 41
3.3 Phát triển hệ thống chatbot 44
3.3.1 Các công cụ và ngôn ngữ lập trình được sử dụng 44
3.3.2 Quá trình phát triển các thành phần của chatbot 44
3.3.3 Kiểm thử và điều chỉnh chatbot dựa trên phản hồi từ người dùng 45
3.4 Thử nghiệm hệ thống chatbot 47

3.4.1 Mô tả kịch bản thử nghiệm và các tiêu chí đánh giá 47
3.4.2 Quá trình triển khai thử nghiệm tại FPT Software 48
3.5 Đánh giá hiệu quả 49
3.5.1 Phân tích dữ liệu thu được từ quá trình thử nghiệm 49
3.5.2 Đánh giá hiệu quả của chatbot trong việc hỗ trợ quản lý thời gian 50
3.5.3 So sánh hiệu quả của chatbot với các phương pháp quản lý thời gian truyền thống 51
3.6 Các đề xuất cải tiến 52
3.6.1 Những điểm mạnh và điểm yếu của hệ thống chatbot 52
3.6.2 Đề xuất các giải pháp cải tiến và nâng cao hiệu quả của hệ thống 53
3.7 Kết luận chương 3 54
III. KẾT LUẬN 55
IV. TÀI LIỆU THAM KHẢO 57

DANH MỤC CÁC THUẬT NGỮ, CHỮ VIẾT TẮT

Viết tắt	Tiếng Anh	Tiếng Việt
AI	Artificial Intelligence	Trí tuệ nhân tạo
CRM	Customer Relationship Management	Quản lý quan hệ khách hàng
CRUD	Create, Read, Update, Delete	Tạo, Đọc, Cập nhật, Xóa
CSAT	Customer Satisfaction Score	Chỉ số hài lòng của khách hàng
ERP	Enterprise Resource Planning	Hệ thống hoạch định nguồn lực doanh nghiệp
FC	Functional Requirements	Yêu cầu chức năng
GUI	Graphical User Interface	Giao diện đồ họa
JWT	JSON Web Token	Mã thông báo JSON Web
LLM	Large Language Models	Mô hình ngôn ngữ lớn
ML	Machine Learning	Học máy
NER	Named Entity Recognition	Nhận diện thực thể có tên
NFR	Non-functional Requirements	Yêu cầu phi chức năng

NLG	Natural Language Generation	Sinh ngôn ngữ tự nhiên
NLP	Natural Language Processing	Xử lý ngôn ngữ tự nhiên
NLU	Natural Language Understanding	Hiểu ngôn ngữ tự nhiên
POS	Part-of-Speech Tagging	Gắn nhãn từ loại
RBAC	Role-Based Access Control	Kiểm soát truy cập dựa trên vai trò
SLA	Service Level Agreement	Cam kết chất lượng dịch vụ
SSO	Single Sign-On	Đăng nhập một lần
UAT	User Acceptance Testing	Kiểm thử người dùng

DANH SÁCH BẢNG

Bảng 1.1: Tổng hợp kết quả triển khai chatbot quản lý thời gian thông qua các nghiên cứu 13

Bảng 1.2: So sánh chức năng Maya hiện tại và nhu cầu thực tế 16

Bảng 2.1: So sánh các mô hình NLP cho ứng dụng tiếng Việt trong chatbot 25

Bảng 2.2: Danh sách ý định và thực thể chính trong domain.yml của hệ thống 29

Bảng 2.3: So sánh RASA với các lựa chọn thay thế phổ biến 33

Bảng 3.1: Các vấn đề quản lý thời gian và mức độ ưu tiên giải quyết 35

Bảng 3.2: Kết quả kiểm thử NLU qua các vòng cải tiến 46

Bảng 3.3: So sánh hiệu quả chatbot và phương pháp truyền thống 51

DANH SÁCH HÌNH VẼ

Hình 2.1: Sơ đồ pipeline học máy của hệ thống chatbot 22
Hình 3.1: Kiến trúc hệ thống theo mô hình C4 Container Diagram 39
Hình 3.2: Sơ đồ luồng hội thoại tạo nhiệm vụ mới (Happy Path và Unhappy Paths) 43
Hình 3.3: Đánh giá hiệu suất Mô hình PHOBERT 47
Hình 3.4: Hình ảnh mô phỏng ứng dụng Chatbot giúp quản lý thời gian của nhân viên tại FPT Software 48
Hình 3.5: Biểu đồ lưu lượng sử dụng 50
Hình 3.6: Dashboard đánh giá hiệu quả A/B Testing 51

I. MỞ ĐẦU

1. Lý do chọn đề tài

Trong bối cảnh cuộc cách mạng công nghiệp lần thứ tư (Industry 4.0) đang diễn ra mạnh mẽ trên toàn cầu, các tổ chức và doanh nghiệp trong lĩnh vực công nghệ thông tin phải đối mặt với áp lực ngày càng lớn trong việc tối ưu hóa hiệu quả hoạt động. Trong số các yếu tố quyết định đến năng suất và chất lượng công việc, quản lý thời gian được xem là một trong những thách thức cốt lõi mà các doanh nghiệp phần mềm cần giải quyết một cách căn cơ và hệ thống.

FPT Software – một trong những công ty phần mềm hàng đầu Việt Nam với quy mô nhân sự lên tới hàng chục nghìn kỹ sư và chuyên gia công nghệ – không phải là ngoại lệ. Với môi trường làm việc đặc thù, nơi các nhân viên thường xuyên phải tham gia nhiều dự án song song, phối hợp với đối tác quốc tế qua nhiều múi giờ khác nhau và tuân thủ các cam kết về thời hạn (deadline) khắt khe, việc quản lý thời gian hiệu quả trở nên đặc biệt quan trọng. Tình trạng lãng phí thời gian do thiếu công cụ hỗ trợ phù hợp, quên lịch hẹn, trùng lặp công việc hay chậm trễ trong giao tiếp nội bộ là những vấn đề thực tiễn ảnh hưởng trực tiếp đến hiệu suất doanh nghiệp.

Một trong những giải pháp công nghệ hiện đại và đầy tiềm năng cho vấn đề trên là ứng dụng chatbot – một sản phẩm tiêu biểu của trí tuệ nhân tạo (Artificial Intelligence – AI) và xử lý ngôn ngữ tự nhiên (Natural Language Processing – NLP). Chatbot có khả năng tự động hóa nhiều tác vụ thường nhật, hỗ trợ nhân viên trong việc lên kế hoạch làm việc, nhắc nhở các nhiệm vụ quan trọng, đồng thời cung cấp các khuyến nghị cá nhân hóa dựa trên hành vi và thói quen làm việc của từng cá nhân. Không chỉ giúp giảm tải khối lượng công việc thủ công, chatbot còn cải thiện đáng kể sự hài lòng và trải nghiệm của nhân viên trong quá trình làm việc.

Hiện nay, FPT Software đã triển khai chatbot nội bộ mang tên Maya với nhiều tính năng cơ bản phục vụ hoạt động nhân sự và hành chính. Tuy nhiên, các chức năng liên quan đến quản lý thời gian của nhân viên trong Maya vẫn còn tương đối hạn chế, chưa đáp ứng đầy đủ nhu cầu thực tế đa dạng và phức tạp của đội ngũ kỹ sư phần mềm. Khoảng cách giữa những gì hiện có và những gì thực sự cần thiết chính là động lực thúc đẩy việc triển khai nghiên cứu này.

Xuất phát từ những nhu cầu cấp thiết trên, đề án "Nghiên cứu và phát triển ứng dụng chatbot giúp quản lý thời gian của nhân viên tại FPT Software" được lựa chọn nhằm: (1) nghiên cứu tổng quan về công nghệ chatbot và ứng dụng trong quản lý thời gian; (2) đề xuất và phát triển một hệ thống chatbot có khả năng hỗ trợ toàn diện việc quản lý thời gian cho nhân viên FPT Software; và (3) đánh giá hiệu quả ứng dụng trong điều kiện thực tế của doanh nghiệp. Đây không chỉ là đóng góp thiết thực cho FPT Software mà còn là nguồn tham khảo có giá trị học thuật cho cộng đồng nghiên cứu về AI và hệ thống thông tin doanh nghiệp tại Việt Nam.

2. Tổng quan về vấn đề nghiên cứu

Chatbot là một hệ thống phần mềm có khả năng tương tác với người dùng thông qua ngôn ngữ tự nhiên, mô phỏng cuộc hội thoại giữa người với người. Từ những chatbot đơn giản dựa trên quy tắc (rule-based) vào những thập niên 1960–1980 với hệ thống ELIZA nổi tiếng của Joseph Weizenbaum, đến ngày nay, chatbot đã phát triển vượt bậc nhờ sự kết hợp của học máy (Machine Learning), mạng nơ-ron sâu (Deep Neural Network) và các mô hình ngôn ngữ lớn (Large Language Models – LLM).

Trong lĩnh vực quản lý doanh nghiệp, chatbot đã được nghiên cứu và ứng dụng rộng rãi trên thế giới. Theo nghiên cứu của Jones (2020) [12], các hệ thống AI tích hợp vào công cụ quản lý thời gian có thể giúp tăng năng suất làm việc lên tới 25–30% nhờ tự động hóa các tác vụ lặp đi lặp lại và cung cấp thông tin kịp thời. Brown (2019) [10] đã chỉ ra rằng NLP là nền tảng công nghệ cốt lõi giúp chatbot hiểu và xử lý các yêu cầu của người dùng một cách tự nhiên, mở ra khả năng tương tác không cần đào tạo chuyên sâu từ phía người dùng cuối.

Tại Việt Nam, nghiên cứu về ứng dụng chatbot trong quản lý doanh nghiệp còn khá mới mẻ. Một số công trình đáng chú ý bao gồm nghiên cứu của Nguyễn Văn Minh (2020) [3] về ứng dụng trí tuệ nhân tạo trong quản lý doanh nghiệp, và nghiên cứu của Đỗ Quang Huy (2022) [4] về phát triển chatbot trong doanh nghiệp Việt Nam. Tuy nhiên, các nghiên cứu này chủ yếu mang tính tổng quát và chưa tập trung vào vấn đề quản lý thời gian trong môi trường phần mềm chuyên biệt.

Đặc biệt, việc ứng dụng RASA Framework – một nền tảng chatbot mã nguồn mở được đánh giá cao trong giới nghiên cứu và doanh nghiệp – vào phát triển chatbot quản lý thời gian tại môi trường doanh nghiệp Việt Nam là một hướng nghiên cứu mới, có giá trị thực tiễn cao. RASA cung cấp bộ công cụ toàn diện bao gồm RASA NLU (hiểu ngôn ngữ tự nhiên) và RASA Core (quản lý hội thoại), cho phép phát triển chatbot có khả năng xử lý các cuộc hội thoại phức tạp, đa lượt, phù hợp với nhu cầu quản lý thời gian trong môi trường doanh nghiệp năng động.

Vấn đề nghiên cứu được đặt ra trong đề án này là: làm thế nào để thiết kế và phát triển một hệ thống chatbot sử dụng RASA Framework có khả năng hỗ trợ hiệu quả việc quản lý thời gian cho nhân viên tại FPT Software, đồng thời tích hợp liền mạch với hệ sinh thái công nghệ hiện có của công ty?

3. Mục đích nghiên cứu

Mục đích tổng quát của nghiên cứu này là phát triển một hệ thống chatbot thông minh, có khả năng hỗ trợ toàn diện việc quản lý thời gian cho nhân viên FPT Software, góp phần nâng cao năng suất và hiệu quả làm việc của tổ chức. Cụ thể, đề án hướng đến các mục tiêu sau:

Thứ nhất, tự động hóa việc sắp xếp lịch làm việc: Hệ thống chatbot sẽ được thiết kế để hỗ trợ nhân viên lập kế hoạch công việc theo ngày, tuần, tháng; nhắc nhở tự động về các cuộc họp, buổi đánh giá và nhiệm vụ quan trọng; đồng bộ hóa với lịch làm việc của nhóm và toàn bộ dự án.

Thứ hai, theo dõi tiến độ và nhắc nhở công việc: Chatbot sẽ cung cấp các thông báo về thời hạn hoàn thành công việc (deadline), theo dõi tiến độ thực hiện của từng nhiệm vụ, cảnh báo sớm khi có nguy cơ chậm trễ, và tổng hợp báo cáo tiến độ định kỳ cho cả nhân viên lẫn quản lý.

Thứ ba, tối ưu hóa lịch làm việc: Dựa trên phân tích dữ liệu lịch sử và các thuật toán học máy, chatbot sẽ đưa ra các đề xuất thông minh về phân bổ thời gian, sắp xếp ưu tiên công việc và tránh xung đột lịch trình giữa các nhiệm vụ.

Thứ tư, cá nhân hóa trải nghiệm người dùng: Hệ thống sẽ học từ hành vi và thói quen làm việc của mỗi nhân viên để cung cấp các gợi ý và nhắc nhở phù hợp với phong cách làm việc cá nhân, từ đó tăng cường hiệu quả tương tác với chatbot theo thời gian.

4. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu: Đối tượng nghiên cứu chính của đề án bao gồm: (1) Hệ thống chatbot hỗ trợ quản lý thời gian và các công nghệ liên quan, bao gồm AI, NLP, học máy và RASA Framework; (2) Quy trình quản lý thời gian của nhân viên phần mềm tại FPT Software; (3) Các giải pháp tích hợp chatbot vào hệ sinh thái công nghệ doanh nghiệp.

Phạm vi nghiên cứu: Phạm vi nghiên cứu được giới hạn trong môi trường làm việc thực tế tại FPT Software, tập trung vào đội ngũ kỹ sư phần mềm và nhân viên kỹ thuật. Về mặt công nghệ, nghiên cứu tập trung vào nền tảng RASA Framework (phiên bản 3.x), tích hợp với các công cụ quản lý dự án phổ biến mà FPT Software đang sử dụng. Về thời gian, nghiên cứu được thực hiện trong năm học 2024–2025 theo kế hoạch đã đề ra. Kết quả nghiên cứu có thể được tổng quát hóa và áp dụng cho các công ty phần mềm có quy mô và đặc điểm tương tự FPT Software tại Việt Nam.

5. Phương pháp nghiên cứu

Phương pháp nghiên cứu lý thuyết: Phương pháp này bao gồm việc thu thập, phân tích và tổng hợp các tài liệu khoa học liên quan đến công nghệ chatbot, trí tuệ nhân tạo, xử lý ngôn ngữ tự nhiên và các ứng dụng trong quản lý thời gian. Nguồn tài liệu được tổng hợp từ các bài báo khoa học quốc tế và trong nước, sách chuyên khảo, tài liệu kỹ thuật chính thức từ các nhà phát triển RASA, cũng như các báo cáo thực tiễn về triển khai chatbot trong doanh nghiệp.

Phương pháp khảo sát và phân tích nhu cầu: Thực hiện khảo sát trực tiếp với nhân viên và cán bộ quản lý tại FPT Software thông qua bảng hỏi (questionnaire) và phỏng vấn sâu (in-depth interview) để thu thập dữ liệu về thực trạng quản lý thời gian và nhu cầu thực tế. Kết quả khảo sát sẽ là cơ sở để xác định yêu cầu chức năng và phi chức năng của hệ thống.

Phương pháp thực nghiệm: Tiến hành thiết kế kiến trúc, phát triển và thử nghiệm hệ thống chatbot trong môi trường thực tế tại FPT Software. Quá trình thử nghiệm được thực hiện theo phương pháp Agile, với các vòng phản hồi liên tục từ người dùng để cải tiến hệ thống. Kết quả thử nghiệm sẽ được đánh giá theo các tiêu chí định lượng (thời gian phản hồi, độ chính xác) và định tính (mức độ hài lòng của người dùng).

Phương pháp đánh giá và so sánh: So sánh hiệu quả của hệ thống chatbot đề xuất với các phương pháp quản lý thời gian truyền thống và với chatbot Maya hiện có tại FPT Software, qua đó xác định giá trị gia tăng mà giải pháp mới mang lại.

II. NỘI DUNG

CHƯƠNG 1: TỔNG QUAN VỀ QUẢN LÝ THỜI GIAN VÀ ỨNG DỤNG CÔNG NGHỆ CHATBOT TRONG DOANH NGHIỆP

1.1 Khái niệm và vai trò của quản lý thời gian

1.1.1 Khái niệm quản lý thời gian và tầm quan trọng của nó trong doanh nghiệp

Quản lý thời gian (Time Management) là quá trình lập kế hoạch và kiểm soát có ý thức thời lượng dành cho các hoạt động cụ thể, đặc biệt nhằm mục tiêu tăng cường hiệu quả, hiệu suất và năng suất lao động. Ở cấp độ cá nhân, quản lý thời gian bao gồm việc xác định mục tiêu, ưu tiên hóa nhiệm vụ, phân bổ thời gian hợp lý và theo dõi tiến độ thực hiện. Ở cấp độ tổ chức, quản lý thời gian mở rộng ra phạm vi phối hợp giữa các thành viên trong nhóm, các phòng ban và toàn bộ dự án.

Trong môi trường doanh nghiệp phần mềm, tầm quan trọng của quản lý thời gian hiệu quả thể hiện rõ ràng qua nhiều khía cạnh. Thứ nhất, đảm bảo tiến độ dự án: các dự án phần mềm thường có ràng buộc thời gian nghiêm ngặt theo hợp đồng với khách hàng; sự chậm trễ không chỉ gây thiệt hại tài chính mà còn ảnh hưởng đến uy tín doanh nghiệp. Thứ hai, tối ưu hóa nguồn lực: với đặc thù sử dụng nguồn nhân lực trình độ cao, việc phân bổ thời gian hợp lý giúp tránh lãng phí năng lực và giảm tình trạng quá tải hoặc thiếu việc làm của kỹ sư. Thứ ba, nâng cao chất lượng sản phẩm: khi thời gian được quản lý tốt, nhân viên có thể dành đủ thời gian cho các công việc đòi hỏi sự tập trung cao như thiết kế kiến trúc, viết code chất lượng và kiểm thử kỹ lưỡng.

Theo Lê Hữu Phước (2021) [5], trong môi trường doanh nghiệp phần mềm tại Việt Nam, trung bình một kỹ sư phần mềm lãng phí khoảng 20–30% thời gian làm việc do thiếu công cụ quản lý thời gian phù hợp, bao gồm: thời gian tìm kiếm thông tin về lịch họp, theo dõi trạng thái công việc, xử lý thông báo không liên quan và điều phối lịch trình với đồng nghiệp. Con số này phản ánh một tiềm năng cải thiện đáng kể nếu có công cụ hỗ trợ phù hợp.

Ngoài ra, theo nghiên cứu của Smith (2018) [15], có bốn nhân tố chính cản trở việc quản lý thời gian hiệu quả trong doanh nghiệp phần mềm: (1) Sự gián đoạn liên tục (continuous interruptions) từ các kênh giao tiếp đa dạng – email, tin nhắn tức thời, thông báo hệ thống – khiến kỹ sư mất tập trung và cần thêm thời gian để lấy lại luồng công việc; (2) Thiếu tính minh bạch về kế hoạch chung (visibility gap) khiến các thành viên trong nhóm không rõ ai đang làm gì, dẫn đến trùng lặp hoặc bỏ sót công việc; (3) Ước lượng thời gian không chính xác (estimation bias) – các kỹ sư có xu hướng lạc quan quá mức khi ước tính thời gian hoàn thành nhiệm vụ, dẫn đến thường xuyên chậm deadline; (4) Thiếu phản hồi thời gian thực (lack of real-time feedback) về tiến độ và hiệu quả sử dụng thời gian, khiến vấn đề chỉ được phát hiện khi đã quá muộn để khắc phục kịp thời.

Chính những nhân tố này tạo ra nhu cầu cấp thiết về một công cụ hỗ trợ quản lý thời gian có khả năng: xử lý thông tin tập trung, cung cấp góc nhìn toàn diện về tình trạng công việc, nhắc nhở chủ động và phân tích dữ liệu lịch sử để hỗ trợ ước lượng chính xác hơn trong tương lai.

1.1.2 Các phương pháp quản lý thời gian truyền thống và những hạn chế

Qua nhiều thập kỷ, nhiều phương pháp quản lý thời gian truyền thống đã được nghiên cứu và áp dụng trong doanh nghiệp. Một số phương pháp phổ biến nhất bao gồm:

Ma trận Eisenhower (Eisenhower Matrix): Phương pháp phân loại công việc theo hai chiều Quan trọng/Không quan trọng và Khẩn cấp/Không khẩn cấp, tạo ra bốn ô phân loại giúp ưu tiên hóa nhiệm vụ. Mặc dù đơn giản và dễ áp dụng, phương pháp này yêu cầu người dùng tự đánh giá và phân loại công việc thủ công, thiếu tính tự động và khó mở rộng khi số lượng nhiệm vụ lớn.

Kỹ thuật Pomodoro: Kỹ thuật quản lý thời gian chia nhỏ công việc thành các phiên tập trung 25 phút (Pomodoro) xen kẽ với các khoảng nghỉ ngắn. Phương pháp này hiệu quả cho các công việc cá nhân đòi hỏi tập trung cao, nhưng không phù hợp với môi trường làm việc nhóm năng động, thường xuyên bị gián đoạn bởi cuộc họp và yêu cầu từ đồng nghiệp.

Phương pháp GTD (Getting Things Done): Hệ thống quản lý công việc toàn diện của David Allen (2001) [8], tập trung vào việc ghi lại mọi nhiệm vụ ra ngoài não bộ, phân loại và xử lý theo quy trình chuẩn. Tuy hiệu quả cao nếu được thực hiện đầy đủ, GTD đòi hỏi kỷ luật cá nhân rất cao và tốn nhiều thời gian thiết lập ban đầu.

Phần mềm quản lý dự án truyền thống: Các công cụ như Microsoft Project, Jira, Trello hay Asana cung cấp giao diện đồ họa trực quan cho việc lập kế hoạch và theo dõi tiến độ. Tuy nhiên, các công cụ này thường có giao diện phức tạp, yêu cầu cập nhật thủ công và thiếu khả năng tự động nhắc nhở thông minh.

Nhìn chung, các phương pháp quản lý thời gian truyền thống đều có những hạn chế nhất định: (1) Thiếu tính tự động hóa và chủ động trong việc nhắc nhở; (2) Không có khả năng học hỏi và thích nghi theo hành vi cá nhân; (3) Đòi hỏi người dùng phải nhập liệu và cập nhật thủ công, tốn thêm thời gian; (4) Thiếu khả năng phân tích và đề xuất tối ưu hóa dựa trên dữ liệu. Đây chính là khoảng trống mà hệ thống chatbot thông minh có thể lấp đầy một cách hiệu quả.

Bên cạnh các hạn chế đã nêu, cũng cần nhận thấy rằng chính sự đa dạng trong lựa chọn phương pháp quản lý thời gian của từng cá nhân lại tạo ra sự không đồng nhất trong cách thức làm việc nhóm. Khi mỗi thành viên sử dụng một công cụ hoặc phương pháp khác nhau, việc tổng hợp thông tin và điều phối chung trở nên rời rạc và kém hiệu quả. Chatbot đóng vai trò là điểm hội tụ duy nhất – một giao diện hội thoại thống nhất – có thể kết nối dữ liệu từ nhiều hệ thống và cung cấp cái nhìn tổng hợp cho cả cá nhân lẫn nhóm làm việc.

1.2 Tổng quan về chatbot và các ứng dụng trong doanh nghiệp

1.2.1 Khái niệm chatbot, các công nghệ nền tảng như AI và NLP

Chatbot (hay còn gọi là chatterbot hoặc conversational agent) là một chương trình máy tính được thiết kế để mô phỏng cuộc trò chuyện với người dùng thông qua ngôn ngữ tự nhiên, bằng giọng nói hoặc văn bản. Về bản chất, chatbot là giao diện hội thoại (conversational interface) cho phép người dùng tương tác với phần mềm theo cách tự nhiên nhất – thông qua ngôn ngữ hàng ngày – thay vì phải học cách sử dụng các giao diện đồ họa phức tạp (GUI).

Về mặt kiến trúc, một hệ thống chatbot hiện đại thường bao gồm các thành phần chính sau:

Lớp giao diện người dùng (User Interface Layer): Đây là kênh giao tiếp giữa người dùng và chatbot, có thể là ứng dụng nhắn tin (Slack, Microsoft Teams, Telegram), giao diện web, ứng dụng di động hoặc thiết bị giọng nói thông minh.

Lớp xử lý ngôn ngữ tự nhiên (NLP Layer): Thực hiện các tác vụ hiểu ngôn ngữ tự nhiên (Natural Language Understanding – NLU) bao gồm nhận diện ý định (Intent Recognition) và trích xuất thực thể (Entity Extraction), đồng thời sinh ra phản hồi ngôn ngữ tự nhiên (Natural Language Generation – NLG).

Lớp quản lý hội thoại (Dialogue Management Layer): Duy trì ngữ cảnh hội thoại, theo dõi trạng thái hội thoại và quyết định hành động tiếp theo của chatbot dựa trên lịch sử hội thoại và mục tiêu của người dùng.

Lớp tích hợp hệ thống (System Integration Layer): Kết nối chatbot với các hệ thống backend như cơ sở dữ liệu, API dịch vụ, hệ thống quản lý doanh nghiệp (ERP/CRM) và các công cụ bên thứ ba.

Công nghệ nền tảng của chatbot hiện đại là Trí tuệ nhân tạo (AI) – đặc biệt là các mô hình học máy (Machine Learning) và học sâu (Deep Learning). Trong khi chatbot thế hệ đầu (rule-based chatbot) hoạt động dựa trên các quy tắc được lập trình cứng, chatbot thế hệ mới sử dụng các mô hình ngôn ngữ được huấn luyện trên tập dữ liệu lớn để hiểu ngữ nghĩa sâu hơn, xử lý các câu hỏi mơ hồ và học từ phản hồi của người dùng theo thời gian.

1.2.2 Lịch sử phát triển và ứng dụng chatbot trong các lĩnh vực khác nhau

Lịch sử phát triển chatbot có thể chia thành bốn giai đoạn chính:

Giai đoạn 1 (1960–1990) – Chatbot dựa trên quy tắc: Ra đời với ELIZA (1966) của Joseph Weizenbaum tại MIT, đây là chatbot đầu tiên có thể mô phỏng cuộc hội thoại với con người bằng cách nhận diện các từ khóa trong câu hỏi và phản hồi theo mẫu định sẵn. Tiếp theo là ALICE (Artificial Linguistic Internet Computer Entity, 1995) của Richard Wallace sử dụng ngôn ngữ AIML (Artificial Intelligence Markup Language) để định nghĩa các mẫu hội thoại phức tạp hơn.

Giai đoạn 2 (1990–2010) – Chatbot tích hợp học máy: Sự ra đời của các kỹ thuật học máy thống kê, đặc biệt là mô hình Markov ẩn (Hidden Markov Model) và mạng Bayes, giúp chatbot cải thiện đáng kể khả năng xử lý ngôn ngữ tự nhiên. SmarterChild (AOL Instant Messenger, 2001) là một trong những chatbot thương mại thành công đầu tiên, phục vụ hàng triệu người dùng.

Giai đoạn 3 (2010–2020) – Trợ lý ảo thông minh: Sự bùng nổ của điện toán đám mây và học sâu dẫn đến sự ra đời của các trợ lý ảo thế hệ mới như Apple Siri (2011), Google Now (2012), Amazon Alexa (2014) và Microsoft Cortana (2014). Đặc biệt, sự phổ biến của nền tảng nhắn tin như Facebook Messenger, Slack, và WeChat đã thúc đẩy làn sóng phát triển chatbot thương mại mạnh mẽ.

Giai đoạn 4 (2020 đến nay) – Chatbot dựa trên LLM: Sự xuất hiện của các mô hình ngôn ngữ lớn như GPT-3 (OpenAI, 2020), BERT (Google, 2018), và đặc biệt là ChatGPT (OpenAI, 2022) đã cách mạng hóa khả năng của chatbot, cho phép tạo ra các phản hồi tự nhiên, mạch lạc và có chiều sâu ngữ nghĩa chưa từng có trước đây.

Về ứng dụng, chatbot hiện được triển khai rộng rãi trong nhiều lĩnh vực: dịch vụ khách hàng (xử lý yêu cầu, giải đáp thắc mắc, hỗ trợ kỹ thuật), thương mại điện tử (gợi ý sản phẩm, theo dõi đơn hàng), y tế (tư vấn sơ bộ, nhắc lịch uống thuốc), giáo dục (gia sư ảo, hỗ trợ học tập) và đặc biệt là trong quản lý nhân sự và doanh nghiệp.

1.2.3 Vai trò của chatbot trong quản lý thời gian và hiệu quả công việc

Chatbot có tiềm năng đặc biệt trong lĩnh vực quản lý thời gian nhờ khả năng tự động hóa, tính sẵn sàng 24/7 và khả năng học hỏi thích nghi. Cụ thể, chatbot có thể đảm nhận các vai trò sau trong hỗ trợ quản lý thời gian:

Vai trò thư ký điện tử: Chatbot có thể tự động lên lịch cuộc họp dựa trên lịch rảnh của tất cả các bên liên quan, gửi lời mời họp, xác nhận tham dự và nhắc nhở trước cuộc họp. Điều này tiết kiệm đáng kể thời gian điều phối lịch trình thường tốn nhiều email qua lại.

Vai trò trợ lý lập kế hoạch: Dựa trên các mục tiêu dài hạn và ngắn hạn của nhân viên, chatbot có thể đề xuất kế hoạch làm việc hàng ngày tối ưu, phân bổ thời gian cho từng nhiệm vụ dựa trên mức độ ưu tiên và độ phức tạp, đồng thời điều chỉnh kế hoạch linh hoạt khi có thay đổi đột ngột.

Vai trò người nhắc nhở thông minh: Khác với các thông báo thông thường theo lịch cố định, chatbot thông minh có thể xác định thời điểm phù hợp để gửi nhắc nhở dựa trên ngữ cảnh hiện tại của người dùng (ví dụ: không nhắc nhở khi người dùng đang trong cuộc họp), từ đó tránh gây phiền nhiễu không cần thiết.

Vai trò người phân tích năng suất: Chatbot có thể thu thập và phân tích dữ liệu về cách nhân viên sử dụng thời gian, xác định các mô hình lãng phí thời gian, và cung cấp báo cáo định kỳ giúp cả nhân viên và quản lý nhận ra cơ hội cải thiện.

Nghiên cứu của Lee (2021) [13] tại một công ty phần mềm quy mô vừa ở Nhật Bản cho thấy việc triển khai chatbot hỗ trợ quản lý thời gian đã giúp giảm thời gian lãng phí trong điều phối lịch trình xuống 40%, tăng tỷ lệ hoàn thành công việc đúng hạn từ 72% lên 89%, và cải thiện điểm hài lòng nhân viên về công cụ làm việc lên 28%.

Để nhìn rõ hơn tác động của chatbot quản lý thời gian ở các quy mô doanh nghiệp khác nhau, Bảng 1.1 dưới đây tổng hợp một số nghiên cứu và báo cáo triển khai thực tế tiêu biểu từ các nguồn học thuật và ngành công nghiệp:

Bảng 1.1: Tổng hợp kết quả triển khai chatbot quản lý thời gian thông qua các nghiên cứu

Nghiên cứu / Nguồn	Quy mô tổ chức	Công nghệ áp dụng	Kết quả nổi bật
Lee (2021) [13]	Vừa (~500 NV)	NLP + ML	Tăng 17% tỷ lệ hoàn thành đúng hạn
Jones (2020) [12]	Lớn (>1000 NV)	AI + Calendar API	Tiết kiệm 25–30% thời gian điều phối
Smith (2018) [15]	Vừa và lớn	Rule-based + NLP	Giảm 35% email nội bộ lên lịch họp
Xu et al. (2017) [17]	Lớn (MNC)	Deep Learning NLP	CSAT tăng 28 điểm phần trăm

Tuy nhiên, cũng cần nhận thức rõ rằng hiệu quả của chatbot quản lý thời gian phụ thuộc vào nhiều yếu tố bối cảnh: văn hóa tổ chức, mức độ chín muồi kỹ thuật số của nhân viên, chất lượng tích hợp với hệ thống hiện có và mức độ tùy biến phù hợp với quy trình nghiệp vụ đặc thù. Điều này nhấn mạnh tầm quan trọng của việc nghiên cứu và thiết kế giải pháp dành riêng cho từng bối cảnh cụ thể, thay vì áp dụng nguyên mẫu một giải pháp từ môi trường khác.

1.3 Nghiên cứu ứng dụng chatbot trong quản lý thời gian tại FPT Software

1.3.1 Khảo sát về ứng dụng chatbot đang được sử dụng ở FPT Software (Maya)

FPT Software là công ty công nghệ thông tin hàng đầu Việt Nam, thành viên của Tập đoàn FPT, chuyên cung cấp các dịch vụ phần mềm, giải pháp công nghệ và chuyển đổi số cho khách hàng tại hơn 30 quốc gia trên thế giới. Với đội ngũ nhân sự đông đảo, hoạt động trong môi trường đa văn hóa và đa múi giờ, FPT Software có nhu cầu rất cao về các công cụ hỗ trợ quản lý và giao tiếp nội bộ hiệu quả.

Chatbot Maya là hệ thống trợ lý ảo nội bộ được FPT Software phát triển và triển khai nhằm hỗ trợ nhân viên trong các hoạt động hành chính và nhân sự. Maya được tích hợp trên nền tảng Microsoft Teams – công cụ giao tiếp chính thức của FPT Software – và có khả năng xử lý tiếng Việt và tiếng Anh. Theo thông tin từ tài liệu nội bộ của công ty, Maya hiện cung cấp các chức năng cơ bản bao gồm: tra cứu thông tin nhân sự (ngày phép, chính sách lương thưởng, thủ tục hành chính), hỗ trợ đăng ký các dịch vụ nội bộ (đặt phòng họp, đặt vé máy bay công tác), và trả lời các câu hỏi thường gặp (FAQ) về chính sách và quy trình công ty.

Về kiến trúc kỹ thuật, Maya được xây dựng trên nền tảng Microsoft Bot Framework kết hợp với Azure Cognitive Services cho khả năng xử lý ngôn ngữ tự nhiên. Hệ thống tích hợp với các nguồn dữ liệu nội bộ của FPT Software bao gồm hệ thống ERP nhân sự và cơ sở dữ liệu chính sách công ty.

Bối cảnh kỹ thuật của FPT Software cần được hiểu rõ để đánh giá đúng nhu cầu phát triển. FPT Software hiện sử dụng một hệ sinh thái công nghệ phong phú trong quản lý dự án và nhân sự. Microsoft Teams là nền tảng giao tiếp và cộng tác chính, tích hợp chặt chẽ với toàn bộ bộ công cụ Microsoft 365 (Outlook, SharePoint, OneDrive). Jira (Atlassian) được sử dụng làm hệ thống quản lý dự án Agile cho hầu hết các team phát triển phần mềm. Redmine được một số team dùng song song cho quản lý tickets. GitLab là nền tảng quản lý mã nguồn và CI/CD. Sự phức tạp của hệ sinh thái này đặt ra yêu cầu rằng chatbot quản lý thời gian không thể hoạt động như một hòn đảo biệt lập, mà phải tích hợp liền mạch với ít nhất Jira và Outlook Calendar để thực sự mang lại giá trị.

Về quy trình làm việc đặc thù của FPT Software, nhân viên thường tham gia vào cấu trúc Agile Sprint với chu kỳ 2 tuần. Mỗi sprint, kỹ sư có từ 5 đến 15 task trên Jira với các mức độ ưu tiên khác nhau. Daily Standup hàng ngày (thường qua Teams Meeting) yêu cầu mỗi người cập nhật nhanh 3 thông tin: đã làm gì hôm qua, sẽ làm gì hôm nay, và có vướng mắc gì không. Đây là cơ hội lý tưởng để chatbot hỗ trợ tổng hợp thông tin tự động thay vì người dùng phải nhớ và nói thủ công.

1.3.2 Phân tích những điểm đã đạt được và cần cải tiến thêm

Qua quá trình khảo sát ý kiến của 128 nhân viên và 17 cán bộ quản lý tại FPT Software, Maya đã đạt được những thành tựu đáng ghi nhận. Về điểm tích cực: (1) Chatbot giảm đáng kể số lượng email và cuộc gọi đến bộ phận HR để hỏi về chính sách và thủ tục; (2) Quy trình đặt phòng họp được tự động hóa, tiết kiệm 5–10 phút mỗi lần đặt; (3) Giao diện tích hợp trực tiếp trên Microsoft Teams giúp nhân viên dễ sử dụng mà không cần mở thêm ứng dụng.

Tuy nhiên, qua khảo sát, Maya cũng bộc lộ những hạn chế đáng chú ý trong lĩnh vực quản lý thời gian: (1) Thiếu chức năng quản lý nhiệm vụ cá nhân và nhắc nhở deadline; (2) Không có khả năng tích hợp với công cụ quản lý dự án (Jira, Redmine); (3) Chức năng lên lịch còn thụ động, chỉ phản hồi khi được hỏi chứ chưa chủ động đề xuất; (4) Không có khả năng phân tích và báo cáo về mẫu hình sử dụng thời gian của nhân viên; (5) Khả năng xử lý các câu hỏi phức tạp, đa bước về lịch trình còn hạn chế.

Bảng 1.2 tổng hợp so sánh giữa các chức năng hiện có của Maya và nhu cầu thực tế về quản lý thời gian của nhân viên FPT Software theo kết quả khảo sát:

Bảng 2.2: So sánh chức năng Maya hiện tại và nhu cầu thực tế

Chức năng	Maya hiện tại	Nhu cầu thực tế
Nhắc nhở deadline công việc	Không có	Rất cần thiết
Lên lịch cuộc họp thông minh	Cơ bản	Cần cải tiến
Tích hợp Jira/Redmine	Không có	Rất cần thiết
Báo cáo năng suất cá nhân	Không có	Cần thiết
Gợi ý tối ưu lịch làm việc	Không có	Mong muốn có

1.3.3 Phân tích tính khả thi của module quản lý thời gian với ứng dụng chatbot trong FPT Software

Phân tích tính khả thi của dự án phát triển module quản lý thời gian cho chatbot tại FPT Software được thực hiện theo bốn khía cạnh:

Tính khả thi kỹ thuật: FPT Software sở hữu đội ngũ kỹ sư AI và NLP chuyên nghiệp, có kinh nghiệm phát triển hệ thống phần mềm phức tạp. Nền tảng RASA Framework được lựa chọn là mã nguồn mở, có tài liệu đầy đủ và cộng đồng hỗ trợ lớn. Hạ tầng kỹ thuật của FPT Software (server, cloud, API) đủ mạnh để hỗ trợ triển khai và vận hành hệ thống. Về ngôn ngữ, RASA hỗ trợ tiếng Việt thông qua các mô hình ngôn ngữ như PhoBERT, đảm bảo chất lượng xử lý ngôn ngữ tự nhiên tiếng Việt.

Tính khả thi vận hành: Hệ thống được thiết kế để tích hợp với Microsoft Teams – nền tảng giao tiếp đã quen thuộc với toàn bộ nhân viên FPT Software – giúp tối thiểu hóa rào cản sử dụng và không yêu cầu đào tạo thêm về giao diện. Việc bổ sung module quản lý thời gian vào chatbot hiện có là giải pháp tiến hóa (evolutionary), ít gây xáo trộn hơn so với triển khai hoàn toàn mới.

Tính khả thi kinh tế: So với chi phí thuê ngoài hoặc mua phần mềm quản lý thời gian thương mại (như Asana, Monday.com với chi phí theo số lượng người dùng), việc phát triển nội bộ dựa trên RASA mã nguồn mở có chi phí triển khai cạnh tranh hơn. Lợi ích kỳ vọng bao gồm tiết kiệm thời gian lao động (ước tính 3–4 giờ/nhân viên/tháng), giảm tỷ lệ trễ deadline và cải thiện sự hài lòng nhân viên.

Tính khả thi theo lịch trình: Với kế hoạch nghiên cứu trong vòng 6 tháng (từ tháng 9/2024 đến tháng 2/2025), việc phát triển một module chatbot quản lý thời gian với các chức năng cốt lõi là hoàn toàn khả thi. Phạm vi được giới hạn tập trung vào các tính năng có giá trị cao và mức độ phức tạp kỹ thuật vừa phải trong giai đoạn đầu.

1.4 Kết luận chương 1

Chương 1 đã cung cấp nền tảng lý thuyết và thực tiễn quan trọng cho đề án. Phân tích cho thấy quản lý thời gian là thách thức cốt lõi trong doanh nghiệp phần mềm, và các phương pháp truyền thống còn nhiều hạn chế cần được khắc phục bằng giải pháp công nghệ hiện đại.

Chatbot, với sự hỗ trợ của AI và NLP, nổi lên như một giải pháp đầy hứa hẹn với khả năng tự động hóa cao, khả năng học hỏi thích nghi và khả năng tích hợp linh hoạt. Nghiên cứu thực trạng tại FPT Software chỉ ra rằng chatbot Maya hiện tại, dù đã đạt được những kết quả nhất định trong lĩnh vực hành chính – nhân sự, vẫn còn thiếu các chức năng quan trọng về quản lý thời gian. Đây là cơ sở xác thực cho hướng phát triển của đề án.

Phân tích tính khả thi theo bốn chiều (kỹ thuật, vận hành, kinh tế, lịch trình) cho thấy dự án phát triển module chatbot quản lý thời gian tại FPT Software hoàn toàn khả thi và có tiềm năng mang lại giá trị thiết thực cho tổ chức. Chương tiếp theo sẽ đi sâu vào các công nghệ nền tảng cần thiết cho việc phát triển hệ thống này.

CHƯƠNG 2: NGHIÊN CỨU CÁC CÔNG NGHỆ NỀN TẢNG TRONG PHÁT TRIỂN CHATBOT

2.1 Trí tuệ nhân tạo (AI) và học máy (Machine Learning)

2.1.1 Khái niệm AI và vai trò của AI trong phát triển chatbot

Trí tuệ nhân tạo (Artificial Intelligence – AI) là lĩnh vực khoa học máy tính tập trung vào việc tạo ra các hệ thống có khả năng thực hiện các tác vụ thường đòi hỏi trí thông minh của con người, bao gồm: học hỏi, suy luận, giải quyết vấn đề, nhận thức và hiểu ngôn ngữ. Kể từ khi thuật ngữ này được John McCarthy đề xuất tại Hội nghị Dartmouth năm 1956, AI đã trải qua nhiều giai đoạn phát triển, từ các hệ thống chuyên gia (Expert Systems) trong thập niên 1980, đến học máy thống kê những năm 1990–2000, và học sâu (Deep Learning) từ năm 2010 đến nay.

Trong bối cảnh phát triển chatbot, AI đóng vai trò nền tảng với các đóng góp cụ thể: (1) Cho phép chatbot hiểu ý định (intent) và ngữ nghĩa của câu hỏi người dùng vượt ra ngoài khả năng đối chiếu từ khóa đơn giản; (2) Giúp chatbot học từ dữ liệu lịch sử hội thoại để cải thiện chất lượng phản hồi theo thời gian; (3) Cho phép xử lý các tình huống mơ hồ, không rõ ràng mà các hệ thống rule-based không thể xử lý được; (4) Cung cấp khả năng cá nhân hóa phản hồi dựa trên hồ sơ và hành vi của từng người dùng.

2.1.2 Ứng dụng của học máy trong việc cải thiện khả năng tương tác và học hỏi của chatbot

Học máy (Machine Learning – ML) là một nhánh của AI, tập trung vào việc phát triển các thuật toán cho phép hệ thống học từ dữ liệu và cải thiện hiệu suất theo thời gian mà không cần lập trình tường minh. Trong phát triển chatbot, học máy được ứng dụng chủ yếu qua các kỹ thuật sau:

Phân loại ý định (Intent Classification): Đây là bài toán phân loại văn bản (text classification), trong đó mô hình học máy được huấn luyện để ánh xạ câu đầu vào của người dùng vào một trong các lớp ý định định nghĩa trước (ví dụ: đặt_lịch_họp, kiểm_tra_deadline, báo_cáo_tiến_độ). Các thuật toán phổ biến bao gồm Support Vector Machine (SVM), Random Forest và đặc biệt là các mô hình dựa trên Transformer như BERT và PhoBERT cho tiếng Việt.

Nhận diện thực thể có tên (Named Entity Recognition – NER): Kỹ thuật này cho phép chatbot xác định và trích xuất các thực thể quan trọng từ câu đầu vào, chẳng hạn như tên người, ngày giờ, tên dự án, tên nhiệm vụ. Trong ứng dụng quản lý thời gian, NER đặc biệt quan trọng để trích xuất thông tin như "họp nhóm vào thứ Sáu lúc 3 giờ chiều về dự án Alpha" thành các thực thể: loại_sự_kiện=họp_nhóm, thời_gian=thứ_Sáu_15:00, dự_án=Alpha.

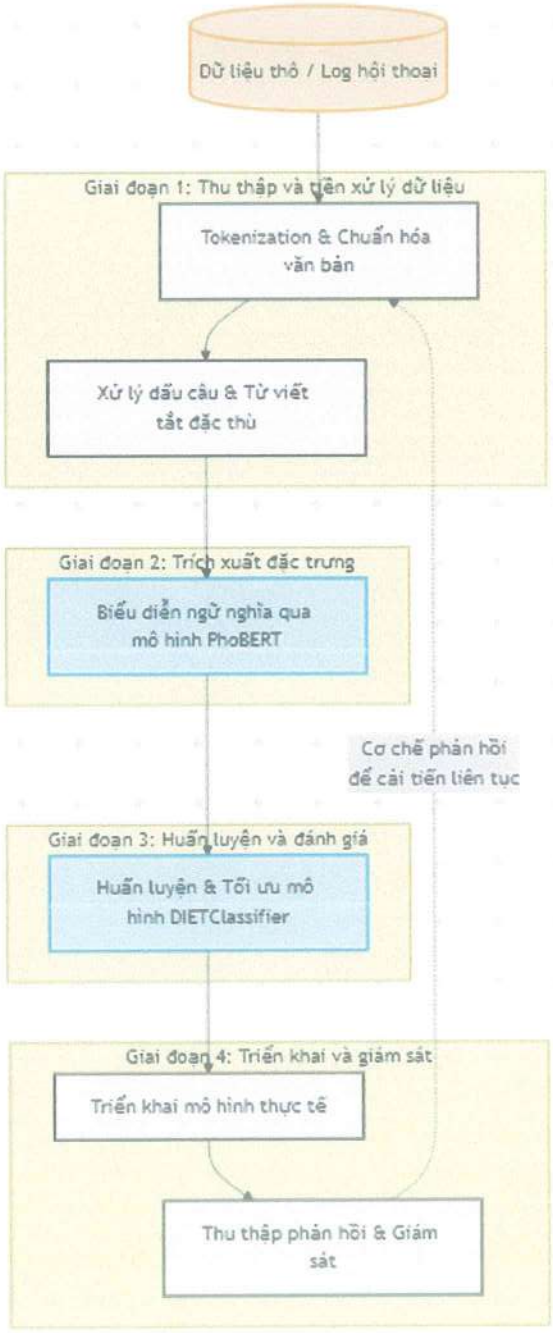
Học tăng cường hội thoại (Dialogue Reinforcement Learning): RASA sử dụng học tăng cường (Reinforcement Learning) để tối ưu hóa chính sách hội thoại (dialogue policy), tức là quyết định chatbot nên phản hồi bằng hành động nào tiếp theo trong từng tình huống hội thoại cụ thể. Mô hình được huấn luyện để tối đa hóa phần thưởng tích lũy theo thời gian, tương ứng với việc dẫn dắt hội thoại đến kết quả thành công.

Học chuyển giao (Transfer Learning): Sử dụng các mô hình ngôn ngữ được tiền huấn luyện (pre-trained) trên tập dữ liệu lớn, sau đó tinh chỉnh (fine-tuning) trên tập dữ liệu đặc thù của ứng dụng. Điều này giúp đạt được hiệu suất cao ngay cả với tập dữ liệu huấn luyện hạn chế, rất phù hợp với các ứng dụng chuyên biệt như chatbot quản lý thời gian nội bộ doanh nghiệp.

Ngoài bốn kỹ thuật học máy cốt lõi trên, một số kỹ thuật bổ sung cũng đóng vai trò quan trọng trong vận hành thực tế của chatbot. Kỹ thuật Data Augmentation được áp dụng để mở rộng tập dữ liệu huấn luyện NLU bằng cách tự động tạo ra các biến thể câu từ dữ liệu gốc thông qua đồng nghĩa từ, hoán đổi cấu trúc câu và thêm/bớt từ phụ. Đây là kỹ thuật đặc biệt hữu ích cho tiếng Việt – vốn không có sẵn nhiều tập

dữ liệu hội thoại chất lượng cao như tiếng Anh. Kỹ thuật Active Learning cho phép hệ thống tự xác định các mẫu hội thoại mà mô hình chưa tự tin dự đoán, đề xuất cho chuyên gia gắn nhãn, và từ đó cải thiện mô hình theo từng vòng lặp – giúp tối ưu hóa chi phí tạo dữ liệu huấn luyện trong quá trình vận hành lâu dài.

Hình 2.1 minh họa luồng dữ liệu trong pipeline học máy của hệ thống chatbot, từ thu thập dữ liệu thô đến triển khai mô hình và cải tiến liên tục. Luồng này gồm bốn giai đoạn chính: (1) Thu thập và tiền xử lý dữ liệu (tokenization, chuẩn hóa văn bản tiếng Việt, xử lý dấu câu và viết tắt đặc thù kỹ sư phần mềm); (2) Trích xuất đặc trưng và biểu diễn ngữ nghĩa qua PhoBERT; (3) Huấn luyện và đánh giá mô hình DIETClassifier; (4) Triển khai và giám sát với cơ chế phản hồi để cải tiến liên tục.



Hình 2.1: Sơ đồ pipeline học máy của hệ thống chatbot

2.2 Xử lý ngôn ngữ tự nhiên (NLP)

2.2.1 Tổng quan về NLP và các công nghệ liên quan

Xử lý ngôn ngữ tự nhiên (Natural Language Processing – NLP) là lĩnh vực giao thoa giữa ngôn ngữ học và khoa học máy tính, tập trung vào việc tạo ra các hệ

thống có thể hiểu, diễn giải và sinh ra ngôn ngữ của con người. NLP bao gồm một chuỗi các tác vụ xử lý từ cấp độ ký tự đến cấp độ diễn ngôn:

Ở cấp độ từ vựng: Phân tích hình thái (morphological analysis), tách từ (tokenization), gắn nhãn từ loại (Part-of-Speech Tagging – POS), xử lý biến thể từ (stemming/lemmatization). Đối với tiếng Việt, tách từ là tác vụ đặc biệt phức tạp do tính chất ghép âm tiết của ngôn ngữ.

Ở cấp độ cú pháp: Phân tích cú pháp (syntactic parsing) để xác định cấu trúc ngữ pháp của câu, từ đó hiểu mối quan hệ giữa các thành phần trong câu.

Ở cấp độ ngữ nghĩa: Phân tích ngữ nghĩa từ vựng (word sense disambiguation), nhận diện thực thể có tên (NER), phân tích quan điểm (sentiment analysis) và hiểu ngữ nghĩa câu (semantic parsing).

Ở cấp độ diễn ngôn: Giải quyết đồng tham chiếu (coreference resolution), phân tích mạch lạc diễn ngôn (discourse coherence), hiểu ngữ cảnh đa lượt (multi-turn context understanding) – đặc biệt quan trọng trong các hệ thống chatbot để duy trì mạch hội thoại.

Đối với ngôn ngữ tiếng Việt, NLP đối mặt với một số thách thức đặc thù không có trong tiếng Anh hay các ngôn ngữ châu Âu khác. Thứ nhất, tiếng Việt là ngôn ngữ đơn lập (isolating language) – mỗi âm tiết là một đơn vị ý nghĩa độc lập, không có hiện tượng biến hình từ (inflection) như tiếng Anh, khiến ranh giới từ không được xác định rõ ràng bởi dấu cách. Ví dụ, câu "học sinh học sinh học" có thể phân tích theo nhiều cách khác nhau. Điều này khiến bài toán tách từ (word segmentation) trở nên phức tạp và là bước tiền xử lý quan trọng nhất cho NLP tiếng Việt.

Thứ hai, tiếng Việt sử dụng hệ thống dấu thanh (6 thanh điệu) và dấu phụ trên nguyên âm, dẫn đến vấn đề về encoding và xử lý Unicode. Trong thực tế doanh nghiệp, nhân viên thường viết không dấu hoặc viết tắt khi nhắn tin nhanh (ví dụ: "ko" thay cho "không", "dc" thay cho "được"), đòi hỏi hệ thống NLP phải có khả năng chuẩn hóa văn bản (text normalization) trước khi phân tích.

Thứ ba, tiếng Việt có xu hướng sử dụng nhiều từ đồng nghĩa và biểu đạt cùng một ý nghĩa theo nhiều cách rất khác nhau. Ví dụ, ý định "tạo nhắc nhở" có thể được diễn đạt là: "nhắc tôi", "đặt báo thức", "tạo reminder", "lưu ý tôi", "cảnh báo tôi về", "hẹn giờ", v.v. Mô hình NLU cần đủ dữ liệu huấn luyện để tổng quát hóa qua tất cả các biến thể này.

2.2.2 Vai trò của NLP trong việc hiểu và phản hồi các yêu cầu của người dùng

Trong hệ thống chatbot, NLP đảm nhận hai vai trò chính: Hiểu ngôn ngữ tự nhiên (Natural Language Understanding – NLU) và Sinh ngôn ngữ tự nhiên (Natural Language Generation – NLG).

NLU là quá trình chuyển đổi câu nhập liệu của người dùng từ dạng văn bản không cấu trúc sang biểu diễn có cấu trúc mà hệ thống có thể xử lý. Trong RASA, NLU thực hiện hai tác vụ song song: (1) Nhận diện ý định (Intent Recognition) – xác định người dùng muốn làm gì; (2) Trích xuất thực thể (Entity Extraction) – xác định các thông tin quan trọng trong câu. Ví dụ: với câu đầu vào "Nhắc tôi họp với team lúc 2 giờ chiều mai", NLU cần nhận diện ý định là tạo_nhắc_nhở và trích xuất các thực thể: loại=họp_nhóm, thời_gian=14:00 ngày_mai.

NLG là quá trình ngược lại, chuyển đổi thông tin có cấu trúc từ hệ thống backend thành ngôn ngữ tự nhiên mà người dùng có thể hiểu được. Trong chatbot quản lý thời gian, NLG tạo ra các phản hồi như: "Đã đặt nhắc nhở họp nhóm vào 14:00 ngày mai (Thứ Năm, 15/11/2024). Tôi sẽ nhắc bạn trước 15 phút. Bạn có muốn thêm địa điểm hoặc danh sách người tham dự không?"

2.2.3 Phân tích các mô hình NLP hiện đại được sử dụng trong chatbot

Sự phát triển của kiến trúc Transformer (Vaswani et al., 2017) [16] đã tạo ra bước ngoặt lớn trong NLP. Cơ chế Self-Attention của Transformer cho phép mô hình xem xét toàn bộ ngữ cảnh của chuỗi đầu vào đồng thời, thay vì xử lý tuần tự như trong RNN/LSTM truyền thống.

BERT (Bidirectional Encoder Representations from Transformers): Mô hình của Google (2018) được tiền huấn luyện theo chiều hai hướng trên kho văn bản khổng lồ, có khả năng hiểu ngữ cảnh từ cả hai phía (trái và phải) trong câu. BERT và các biến thể của nó hiện là nền tảng phổ biến nhất cho các tác vụ NLU trong chatbot.

PhoBERT: Mô hình ngôn ngữ tiền huấn luyện đầu tiên dành riêng cho tiếng Việt (Nguyen & Nguyen, 2020) [6], được huấn luyện trên kho văn bản tiếng Việt 20GB. PhoBERT đạt hiệu suất vượt trội so với mBERT (BERT đa ngôn ngữ) trên hầu hết các tác vụ NLP tiếng Việt, và là lựa chọn hàng đầu cho component NLU trong hệ thống chatbot tiếng Việt.

Sequence-to-Sequence Models với Attention: Được sử dụng chủ yếu cho NLG, cho phép sinh ra phản hồi ngôn ngữ tự nhiên linh hoạt hơn so với các mẫu (template) cố định. Kết hợp với kỹ thuật retrieval-based generation, mô hình có thể tạo ra phản hồi vừa tự nhiên vừa chính xác về thông tin.

Bảng 2.1 dưới đây so sánh các mô hình NLP tiêu biểu được đánh giá cho ứng dụng trong đề án, dựa trên các tiêu chí: chất lượng xử lý tiếng Việt, yêu cầu tài nguyên tính toán, khả năng tích hợp với RASA và mức độ phù hợp với môi trường on-premise của FPT Software:

Bảng 2.1: So sánh các mô hình NLP cho ứng dụng tiếng Việt trong chatbot

Mô hình	CL tiếng Việt	Tài nguyên GPU	Tích hợp RASA	Đề xuất
mBERT	Trung bình	Trung bình	Tốt	Không ưu tiên
XLM-RoBERTa	Khá	Cao	Tốt	Dự phòng
PhoBERT-base	Tốt	Trung bình	Tốt	√Lựa chọn
PhoBERT-large	Rất tốt	Cao	Tốt	Giai đoạn 2

Kết quả đánh giá cho thấy PhoBERT-base là lựa chọn tối ưu cho giai đoạn triển khai ban đầu, cân bằng tốt giữa chất lượng xử lý tiếng Việt và yêu cầu tài nguyên tính toán phù hợp với hạ tầng hiện có của FPT Software. PhoBERT-large sẽ được xem xét cho giai đoạn cải tiến tiếp theo khi yêu cầu chất lượng cao hơn và tài nguyên phần cứng được nâng cấp.

2.3 RASA Framework

2.3.1 RASA Core

RASA Core là thành phần quản lý hội thoại (Dialogue Management) trong RASA Framework, chịu trách nhiệm quyết định hành động tiếp theo của chatbot dựa trên trạng thái hội thoại hiện tại. RASA Core sử dụng hai cách tiếp cận chính:

Chính sách dựa trên quy tắc (Rule-based Policy): Xác định các quy tắc hội thoại tường minh cho các tình huống đơn giản và có cấu trúc cố định. Ví dụ: "Nếu người dùng chào hỏi → trả lời chào hỏi", "Nếu người dùng cung cấp đủ thông tin để tạo lịch → xác nhận và tạo lịch". Rules được định nghĩa trong file rules.yml.

Chính sách học máy (Machine Learning Policy – TEDPolicy): Sử dụng kiến trúc Transformer-based Embedding Dialogue (TED) Policy để học chính sách hội thoại từ dữ liệu huấn luyện (các câu chuyện – stories). TEDPolicy học cách lựa chọn hành động phù hợp nhất dựa trên lịch sử hội thoại, giúp chatbot xử lý các tình huống phức tạp và không theo kịch bản dự đoán trước.

Cơ chế quản lý trạng thái (State Management) của RASA Core sử dụng khái niệm Slots – các biến nhớ lưu trữ thông tin quan trọng trong suốt hội thoại, và Forms – cơ chế thu thập thông tin có cấu trúc từ người dùng qua nhiều lượt hội thoại. Đối với ứng dụng quản lý thời gian, Forms được sử dụng để thu thập đầy đủ thông tin khi tạo sự kiện (tiêu đề, thời gian bắt đầu, thời gian kết thúc, người tham dự, ghi chú).

Một cơ chế quan trọng khác trong RASA Core là Tracker – đối tượng duy trì toàn bộ lịch sử hội thoại dưới dạng danh sách các sự kiện (events). Mỗi lượt tương tác của người dùng và mỗi hành động của chatbot đều được ghi lại thành sự kiện

trong Tracker, tạo ra một biên niên sử hội thoại đầy đủ. Điều này cho phép chatbot nhớ ngữ cảnh từ các lượt hội thoại trước và đưa ra quyết định dựa trên toàn bộ lịch sử, chứ không chỉ dựa trên tin nhắn vừa nhận được. Ví dụ, nếu người dùng đang trong quá trình điền form tạo nhiệm vụ và đột nhiên hỏi về lịch hôm nay, chatbot có thể xử lý câu hỏi đó rồi quay lại form đang dở dang, nhờ Tracker ghi nhớ trạng thái form chưa hoàn thành.

Về Stories – xương sống của dữ liệu huấn luyện RASA Core: mỗi story đại diện cho một kịch bản hội thoại thực tế, được viết theo cú pháp YAML đơn giản. Stories không cần liệt kê tất cả các biến thể có thể xảy ra, vì RASA Core học cách tổng quát hóa từ các ví dụ. Trong đề án này, tập dữ liệu stories bao gồm khoảng 147 stories bao phủ các luồng hội thoại chính, trong đó có cả các trường hợp xử lý lỗi (unhappy paths) như người dùng cung cấp thông tin không hợp lệ, thay đổi yêu cầu giữa chừng hoặc hủy bỏ thao tác đang thực hiện.

2.3.2 RASA NLU (Natural Language Understanding)

RASA NLU là thành phần xử lý ngôn ngữ tự nhiên trong RASA Framework, thực hiện việc phân tích câu đầu vào để xác định ý định và trích xuất thực thể. RASA NLU sử dụng một pipeline xử lý có thể cấu hình linh hoạt, bao gồm các component kế tiếp nhau:

Pipeline tham chiếu cho ứng dụng tiếng Việt trong đề án này bao gồm: WhitespaceTokenizer (tách từ), LanguageModelFeaturizer với mô hình PhoBERT (tạo vector đặc trưng ngữ nghĩa), DIETClassifier (phân loại ý định và nhận diện thực thể đồng thời sử dụng kiến trúc Dual Intent and Entity Transformer), và ResponseSelector (chọn phản hồi cho các câu hỏi thường gặp).

Đặc biệt, DIETClassifier là thành phần cốt lõi của RASA NLU, được thiết kế để thực hiện đồng thời cả phân loại ý định (Intent Classification) và nhận diện thực thể (Entity Recognition) trong một mô hình duy nhất. Điều này không chỉ tăng hiệu quả tính toán mà còn cho phép hai tác vụ hỗ trợ lẫn nhau trong quá trình học, cải thiện độ chính xác tổng thể.

Cấu hình pipeline NLU đầy đủ cho đề án này, được định nghĩa trong config.yml, như sau:

```
language: vi
pipeline:
  - name: WhitespaceTokenizer
  - name: LanguageModelFeaturizer
    model_name: phobert
    model_weights: vinai/phobert-base
  - name: DIETClassifier
    epochs: 100
    entity_recognition: True
```

Quá trình tinh chỉnh (fine-tuning) PhoBERT cho tác vụ phân loại ý định của hệ thống bắt đầu bằng việc tải pre-trained weights từ checkpoint vinai/phobert-base. Sau đó, các lớp classification head được thêm vào và toàn bộ mạng được huấn luyện cùng nhau trên tập dữ liệu NLU đặc thù của ứng dụng. Việc sử dụng learning rate scheduler với warm-up giai đoạn đầu giúp tránh phá vỡ các representations đã học trong quá trình tiền huấn luyện. Tập dữ liệu NLU được chia theo tỷ lệ 80% huấn luyện / 20% kiểm thử, với cross-validation 5-fold để đánh giá độ ổn định của mô hình.

2.3.3 Các thành phần hệ thống chatbot RASA

Hệ thống RASA được tổ chức qua nhiều tệp cấu hình YAML xác định hành vi của chatbot. Các thành phần chính bao gồm:

domain.yml: File trung tâm định nghĩa "thế giới" của chatbot, bao gồm danh sách ý định (intents), thực thể (entities), slots, hành động (actions), biểu mẫu (forms) và các phản hồi mẫu (responses/utterances).

nlu.yml: Chứa dữ liệu huấn luyện NLU, bao gồm các ví dụ câu huấn luyện (training examples) cho từng ý định, mỗi câu được gắn nhãn với ý định tương ứng và các thực thể được đánh dấu inline.

stories.yml: Định nghĩa các kịch bản hội thoại (stories) làm dữ liệu huấn luyện cho mô hình quản lý hội thoại. Mỗi story là một chuỗi các cặp (ý định người dùng, hành động chatbot).

rules.yml: Định nghĩa các quy tắc hội thoại cứng nhắc cho các tình huống luôn phải xử lý theo cách nhất định, không phụ thuộc vào ngữ cảnh trước đó.

actions.py: Tệp Python định nghĩa các hành động tùy chỉnh (Custom Actions) – các đoạn code thực thi logic nghiệp vụ, gọi API bên ngoài, truy vấn cơ sở dữ liệu và trả về phản hồi động.

config.yml: Cấu hình pipeline NLU và các chính sách hội thoại (policies) của RASA Core, cho phép tùy chỉnh kiến trúc mô hình theo nhu cầu cụ thể.

Bảng 2.2 dưới đây liệt kê các ý định (intents) và thực thể (entities) chính được định nghĩa trong domain.yml của hệ thống chatbot quản lý thời gian, cùng với ví dụ câu huấn luyện tiêu biểu:

Bảng 2.2: Danh sách ý định và thực thể chính trong domain.yml của hệ thống

Ý định (Intent)	Thực thể liên quan	Ví dụ câu huấn luyện
tạo_nhiệm_vụ	tên_nhiệm_vụ, deadline, ưu_tiên	"Tạo task hoàn thành báo cáo trước thứ Sáu"
xem_lịch	khoảng_thời_gian	"Lịch tuần này có gì không"
cập_nhật_trạng_thái	tên_nhiệm_vụ, trạng_thái_mới	"Xong task review code rồi"
đặt_nhắc_nhở	nội_dung, thời_điểm	"Nhắc tôi gửi báo cáo lúc 5 giờ chiều"
xem_tiến_độ	tên_dự_án, khoảng_thời_gian	"Sprint này còn bao nhiêu task chưa xong"
báo_cáo_standup	hôm_qua, hôm_nay, vướng_mắc	"Tổng hợp standup cho tôi"

2.3.4 Tìm hiểu luồng hoạt động chi tiết của RASA

Luồng xử lý của RASA khi nhận một tin nhắn từ người dùng diễn ra theo các bước sau:

Bước 1 – Nhận tin nhắn: RASA Server nhận tin nhắn từ người dùng qua REST API hoặc connector (Slack, Teams,...). Tin nhắn được đưa vào NLU pipeline để xử lý.

Bước 2 – NLU Processing: Pipeline NLU xử lý văn bản qua các component theo thứ tự: tokenization → featurization → intent classification → entity extraction. Kết quả là một đối tượng Message chứa: ý định với điểm tin cậy (confidence score), danh sách thực thể được trích xuất, và vector đặc trưng.

Bước 3 – Dialogue Management: RASA Core nhận kết quả NLU và kết hợp với trạng thái hội thoại hiện tại (các slots đã điền, lịch sử hành động) để tạo ra vector trạng thái (state vector). Các policy được áp dụng theo thứ tự ưu tiên để dự đoán hành động tiếp theo.

Bước 4 – Action Execution: Hành động được dự đoán được thực thi. Nếu là utter action (phản hồi văn bản định sẵn), chatbot trả về phản hồi trực tiếp. Nếu là custom action, RASA gửi yêu cầu đến Action Server (Python service), nơi code nghiệp vụ được thực thi, có thể bao gồm truy vấn API lịch, cập nhật cơ sở dữ liệu hoặc gọi dịch vụ nhắc nhở.

Bước 5 – Phản hồi người dùng: Kết quả từ action được trả về cho người dùng qua kênh giao tiếp gốc. Trạng thái hội thoại được cập nhật cho lượt kế tiếp.

2.4 Tích hợp và triển khai chatbot trong doanh nghiệp

2.4.1 Các phương pháp tích hợp chatbot vào hệ thống quản lý doanh nghiệp

Tích hợp chatbot vào hệ sinh thái công nghệ doanh nghiệp là một trong những thách thức kỹ thuật quan trọng nhất. Các chiến lược tích hợp phổ biến bao gồm:

Tích hợp qua REST API: Phương pháp phổ biến nhất, chatbot giao tiếp với các hệ thống backend thông qua API RESTful. RASA Action Server gọi các API

endpoint của hệ thống quản lý dự án (Jira REST API), lịch (Google Calendar API, Outlook Calendar API) và các dịch vụ thông báo. Ưu điểm: linh hoạt, dễ tích hợp, phù hợp với kiến trúc microservices. Nhược điểm: yêu cầu xác thực và quản lý phiên làm việc.

Tích hợp qua Message Queue: Sử dụng các hệ thống hàng đợi tin nhắn như Apache Kafka hoặc RabbitMQ để tích hợp chatbot với các hệ thống backend theo mô hình bất đồng bộ (asynchronous). Phù hợp với các tính năng cần xử lý trong nền như gửi nhắc nhở định kỳ và cập nhật trạng thái công việc.

Tích hợp qua Webhook: Các hệ thống bên ngoài gửi thông báo đến chatbot qua webhook khi có sự kiện quan trọng (ví dụ: Jira gửi webhook khi một issue được chuyển sang trạng thái mới). Chatbot nhận webhook và chủ động liên lạc với nhân viên liên quan.

2.4.2 Những thách thức và giải pháp khi triển khai chatbot trong môi trường doanh nghiệp thực tế

Quá trình triển khai chatbot trong doanh nghiệp đối mặt với nhiều thách thức đặc thù:

Thách thức về bảo mật và quyền riêng tư: Chatbot xử lý các thông tin nhạy cảm về lịch làm việc, nhiệm vụ và tiến độ dự án. Giải pháp: triển khai chatbot on-premise (tự lưu trữ) để dữ liệu không rời khỏi hạ tầng nội bộ; áp dụng mã hóa end-to-end cho dữ liệu hội thoại; thực hiện phân quyền truy cập chặt chẽ dựa trên vai trò người dùng (RBAC).

Thách thức về chất lượng dữ liệu huấn luyện: Chất lượng của chatbot phụ thuộc trực tiếp vào chất lượng và số lượng dữ liệu huấn luyện. Với ngôn ngữ tiếng Việt chuyên ngành phần mềm, rất khó tìm được tập dữ liệu có sẵn. Giải pháp: thu thập dữ liệu từ logs hội thoại thực tế (sau khi ẩn danh hóa), sử dụng kỹ thuật data augmentation để tăng cường tập dữ liệu huấn luyện, và áp dụng active learning để liên tục cải thiện mô hình từ phản hồi của người dùng.

Thách thức về khả năng mở rộng (Scalability): Khi số lượng người dùng tăng, hệ thống phải đảm bảo thời gian phản hồi ổn định. Giải pháp: triển khai RASA Server theo mô hình containerized (Docker/Kubernetes) với khả năng tự động mở rộng (auto-scaling) dựa trên tải; cache các phản hồi thường gặp; tối ưu hóa pipeline NLU để giảm độ trễ. Cụ thể, hệ thống ứng dụng cơ chế Horizontal Pod Autoscaler (HPA) trong Kubernetes để tự động nhân bản (scale up) các node xử lý khi mức sử dụng CPU hoặc bộ nhớ vượt ngưỡng 70%. Đặc biệt, để giải quyết thách thức về quản lý trạng thái (state management) khi mở rộng ngang, RASA Server được thiết kế hoàn toàn phi trạng thái (stateless) bằng cách lưu trữ toàn bộ lịch sử hội thoại (Tracker Store) và khóa đồng bộ (Lock Store) vào cơ sở dữ liệu in-memory Redis. Điều này đảm bảo khi lưu lượng truy cập tăng đột biến, bất kỳ pod RASA nào cũng có thể tiếp nhận và xử lý mượt mà các lượt hội thoại tiếp theo mà không làm mất ngữ cảnh giao tiếp của người dùng.

Thách thức về quản lý thay đổi: Nhân viên có thể ngần ngại hoặc thiếu tin tưởng vào chatbot trong giai đoạn đầu. Giải pháp: triển khai theo mô hình thí điểm (pilot) với một nhóm nhỏ người dùng sớm (early adopters) nhiệt tình; tổ chức đào tạo ngắn về cách sử dụng; thiết lập cơ chế phản hồi dễ dàng để người dùng báo cáo lỗi và đóng góp cải tiến.

Thách thức về đa ngôn ngữ và phương ngữ: Trong môi trường doanh nghiệp phần mềm quốc tế như FPT Software, nhân viên có xu hướng pha trộn tiếng Việt và tiếng Anh (code-switching) trong giao tiếp hàng ngày. Ví dụ: "Tạo task review code cho sprint này" hay "Meeting với client lúc 3 giờ". Hiện tượng code-switching này tạo ra thách thức đặc thù cho mô hình NLU. Giải pháp: xây dựng tập dữ liệu huấn luyện phản ánh đầy đủ hiện tượng này, và cấu hình pipeline NLU để xử lý linh hoạt cả tiếng Việt và tiếng Anh trong cùng một câu.

Thách thức về duy trì và cập nhật mô hình: Ngôn ngữ sử dụng trong doanh nghiệp phát triển theo thời gian – các thuật ngữ mới, dự án mới, quy trình mới liên tục xuất hiện. Mô hình được huấn luyện một lần sẽ dần lỗi thời nếu không được cập

nhật định kỳ. Giải pháp: thiết lập quy trình tái huấn luyện mô hình theo chu kỳ tháng, sử dụng pipeline CI/CD để tự động hóa quá trình này; triển khai A/B testing để đánh giá an toàn trước khi chính thức cập nhật mô hình mới lên production.

2.5 Kết luận chương 2

Chương 2 đã trình bày toàn diện các công nghệ nền tảng cần thiết cho việc phát triển hệ thống chatbot quản lý thời gian. Sự kết hợp giữa AI/ML, NLP và RASA Framework tạo ra một stack công nghệ mạnh mẽ và linh hoạt, phù hợp với yêu cầu của đề án.

RASA Framework nổi bật như một lựa chọn tối ưu nhờ tính mã nguồn mở, khả năng tùy biến cao, hỗ trợ triển khai on-premise đảm bảo bảo mật dữ liệu doanh nghiệp, và hệ sinh thái công cụ đầy đủ từ NLU đến Dialogue Management. Sự kết hợp với mô hình ngôn ngữ PhoBERT tiếng Việt mở ra khả năng xây dựng chatbot tiếng Việt chất lượng cao phù hợp với nhu cầu của FPT Software.

Một điểm mạnh đặc biệt của RASA Framework so với các lựa chọn thay thế (như Dialogflow của Google hay Microsoft Bot Framework) là khả năng hoàn toàn kiểm soát dữ liệu và mô hình. Trong môi trường doanh nghiệp phần mềm như FPT Software – nơi dữ liệu dự án và nhân sự có tính bảo mật cao – việc không phải gửi dữ liệu lên cloud của bên thứ ba là một yêu cầu không thể thương lượng. Bảng 2.3 so sánh RASA với hai lựa chọn thay thế phổ biến theo các tiêu chí quan trọng nhất:

Bảng 3.3: So sánh RASA với các lựa chọn thay thế phổ biến

Tiêu chí	RASA (Chọn)	Dialogflow (Google)	MS Bot Framework
Triển khai On-Premise	√Có	XCloud only	√Có
Mã nguồn mở	√ Đầy đủ	XKhông	√Một phần
Hỗ trợ tiếng Việt	√PhoBERT	√Cơ bản	√Cơ bản
Kiểm soát dữ liệu huấn luyện	√Hoàn toàn	XGoogle quản lý	√Hoàn toàn

Chi phí license	Miễn phí	Theo usage	Miễn phí / Azure

Các chiến lược tích hợp qua REST API và Webhook, kết hợp với kiến trúc triển khai containerized, tạo nền tảng kỹ thuật vững chắc cho việc xây dựng một hệ thống chatbot có tính sẵn sàng cao, dễ mở rộng và an toàn. Chương tiếp theo sẽ áp dụng toàn bộ kiến thức nền tảng này vào việc thiết kế và phát triển hệ thống chatbot thực tế cho FPT Software.

CHƯƠNG 3: THIẾT KẾ VÀ PHÁT TRIỂN HỆ THỐNG CHATBOT HỖ TRỢ QUẢN LÝ THỜI GIAN TẠI FPT SOFTWARE

3.1 Khảo sát nhu cầu và yêu cầu hệ thống

3.1.1 Khảo sát nhu cầu quản lý thời gian của nhân viên tại FPT Software

Để xây dựng hệ thống chatbot phù hợp với nhu cầu thực tiễn, quá trình khảo sát nhu cầu được thực hiện theo hai phương pháp bổ sung cho nhau: khảo sát định lượng (bảng hỏi) và khảo sát định tính (phỏng vấn sâu).

Khảo sát bảng hỏi: Bảng hỏi được thiết kế với 27 câu hỏi, phân chia thành các phần: (1) Thông tin nhân khẩu học (cấp bậc, thâm niên, đơn vị công tác); (2) Thực trạng sử dụng công cụ quản lý thời gian hiện tại; (3) Các vấn đề gặp phải trong quản lý thời gian; (4) Nhu cầu về chức năng chatbot; (5) Kỳ vọng về giao diện và trải nghiệm người dùng. Bảng hỏi được phân phối qua Microsoft Teams nội bộ và thu về 143 phản hồi hợp lệ.

Kết quả khảo sát chính: Kết quả cho thấy đa số nhân viên (ước tính >70%) gặp khó khăn trong việc theo dõi deadline của nhiều nhiệm vụ song song; >60% mong muốn có thông báo nhắc nhở thông minh tích hợp trên nền tảng Teams; >50% quan tâm đến tính năng báo cáo tóm tắt tiến độ tự động hàng tuần; và >40% mong muốn chatbot có khả năng gợi ý phân bổ thời gian dựa trên danh sách công việc.

Bảng 3.1 dưới đây tổng hợp chi tiết các vấn đề quản lý thời gian được nhân viên FPT Software đánh giá theo mức độ ảnh hưởng đến hiệu suất làm việc, theo thang điểm 5:

Bảng 4.1: Các vấn đề quản lý thời gian và mức độ ưu tiên giải quyết

Vấn đề quản lý thời gian	Tỷ lệ gặp phải	Mức ảnh hưởng (1–5)	Ưu tiên giải quyết
Quên hoặc bỏ lỡ deadline nhiệm vụ	76%	4,3	Rất cao

Mất thời gian cập nhật Jira thủ công	63%	3,8	Cao
Xung đột lịch họp không được phát hiện sớm	54%	3,5	Cao
Làm báo cáo tiến độ hàng tuần tốn thời gian	58%	3,2	Trung bình
Thiếu cái nhìn tổng quan về workload	47%	3,0	Trung bình

Phỏng vấn sâu: 9 cuộc phỏng vấn sâu (30–45 phút mỗi cuộc) được thực hiện với 6 nhân viên đại diện và 3 quản lý. Kết quả phỏng vấn bổ sung các insight quan trọng: (1) Kỹ sư thường mất 15–30 phút mỗi ngày để cập nhật trạng thái công việc lên Jira; (2) Việc tổng hợp báo cáo tiến độ hàng tuần mất trung bình 1–2 giờ/tuần; (3) Quản lý gặp khó khăn trong việc nắm bắt tình trạng tổng thể của nhóm mà không phải hỏi từng người.

3.1.2 Xác định các yêu cầu chức năng và phi chức năng của hệ thống chatbot

Dựa trên kết quả khảo sát, các yêu cầu hệ thống được phân loại thành hai nhóm:

Yêu cầu chức năng (Functional Requirements): Hệ thống phải hỗ trợ các chức năng cốt lõi sau:

FC01: Tạo và quản lý nhiệm vụ – cho phép nhân viên tạo mới, cập nhật trạng thái, gắn nhãn ưu tiên và xóa nhiệm vụ qua hội thoại tự nhiên.

FC02: Nhắc nhở thông minh – tự động gửi nhắc nhở deadline trước 24 giờ, 2 giờ và 30 phút, với khả năng tùy chỉnh thời gian nhắc nhở.

FC03: Quản lý lịch biểu – xem lịch làm việc hàng ngày/tuần, đặt lịch cuộc họp, kiểm tra xung đột lịch trình.

FC04: Tích hợp Jira – đồng bộ danh sách sprint tasks từ Jira, cập nhật trạng thái issues qua chatbot, nhận thông báo khi có thay đổi quan trọng.

FC05: Báo cáo tiến độ – tự động tổng hợp và gửi báo cáo tiến độ cá nhân hàng ngày/tuần; hỗ trợ quản lý nhận báo cáo tổng hợp của nhóm.

FC06: Hỗ trợ ngôn ngữ tự nhiên tiếng Việt – xử lý chính xác các câu hỏi và yêu cầu bằng tiếng Việt với khả năng hiểu đặc thù ngôn ngữ như viết tắt, tiếng lóng kỹ thuật.

Yêu cầu phi chức năng (Non-functional Requirements):

NFR01 – Hiệu năng: Thời gian phản hồi trung bình ≤ 2 giây cho 95% yêu cầu trong điều kiện tải bình thường (50 người dùng đồng thời).

NFR02 – Độ chính xác NLU: Độ chính xác nhận diện ý định đạt $\geq 90\%$ trên tập kiểm thử; độ chính xác trích xuất thực thể đạt $\geq 85\%$.

NFR03 – Tính sẵn sàng: Hệ thống hoạt động 24/7 với thời gian downtime $\leq 0.1\%$ (SLA 99.9%).

NFR04 – Bảo mật: Dữ liệu hội thoại được mã hóa AES-256; xác thực người dùng qua SSO tích hợp với hệ thống Active Directory của FPT Software.

NFR05 – Khả năng mở rộng: Kiến trúc hỗ trợ mở rộng ngang (horizontal scaling) để phục vụ tới 500 người dùng đồng thời, tương đương với khả năng xử lý ổn định từ 100 đến 150 yêu cầu mỗi giây (Requests Per Second - RPS) trong các khung giờ cao điểm (như 8h00 sáng và 17h00 chiều). Yêu cầu mở rộng này cũng bắt buộc lớp dữ liệu (Data Layer) bao gồm PostgreSQL và Redis phải được cấu hình tích hợp cơ chế Connection Pooling, nhằm duy trì hiệu suất truy xuất và ngăn chặn tình trạng cạn kiệt kết nối (connection exhaustion) khi số lượng ứng dụng kết nối vào database tăng lên đột ngột.

3.2 Thiết kế hệ thống chatbot

3.2.1 Kiến trúc tổng thể của hệ thống chatbot

Hệ thống chatbot quản lý thời gian được thiết kế theo kiến trúc microservices, đảm bảo tính modul hóa, dễ bảo trì và mở rộng. Kiến trúc tổng thể bao gồm các lớp sau:

Lớp Giao diện người dùng (Presentation Layer): Người dùng tương tác với chatbot thông qua Microsoft Teams Connector – connector chính thức được RASA hỗ trợ. Giao diện Teams cung cấp khả năng hiển thị các thẻ thông tin phong phú (Adaptive Cards) cho việc trình bày lịch làm việc, danh sách nhiệm vụ và báo cáo tiến độ dưới dạng trực quan.

Lớp Xử lý hội thoại (Dialogue Layer): RASA Server (phiên bản 3.x) đảm nhận toàn bộ quá trình NLU và quản lý hội thoại. Trạng thái hội thoại được lưu trữ trong Redis để đảm bảo tính nhất quán và hiệu năng truy xuất cao. RASA Action Server thực thi logic nghiệp vụ và gọi các dịch vụ bên ngoài.

Lớp Tích hợp dịch vụ (Integration Layer): Các adapter chuyên biệt xử lý việc giao tiếp với từng hệ thống bên ngoài: Jira Adapter (Jira REST API v3), Calendar Adapter (Microsoft Graph API cho Outlook Calendar), Notification Service (sử dụng Microsoft Teams Bot API). Lớp này áp dụng mẫu thiết kế Facade để ẩn đi sự phức tạp của các API bên ngoài.

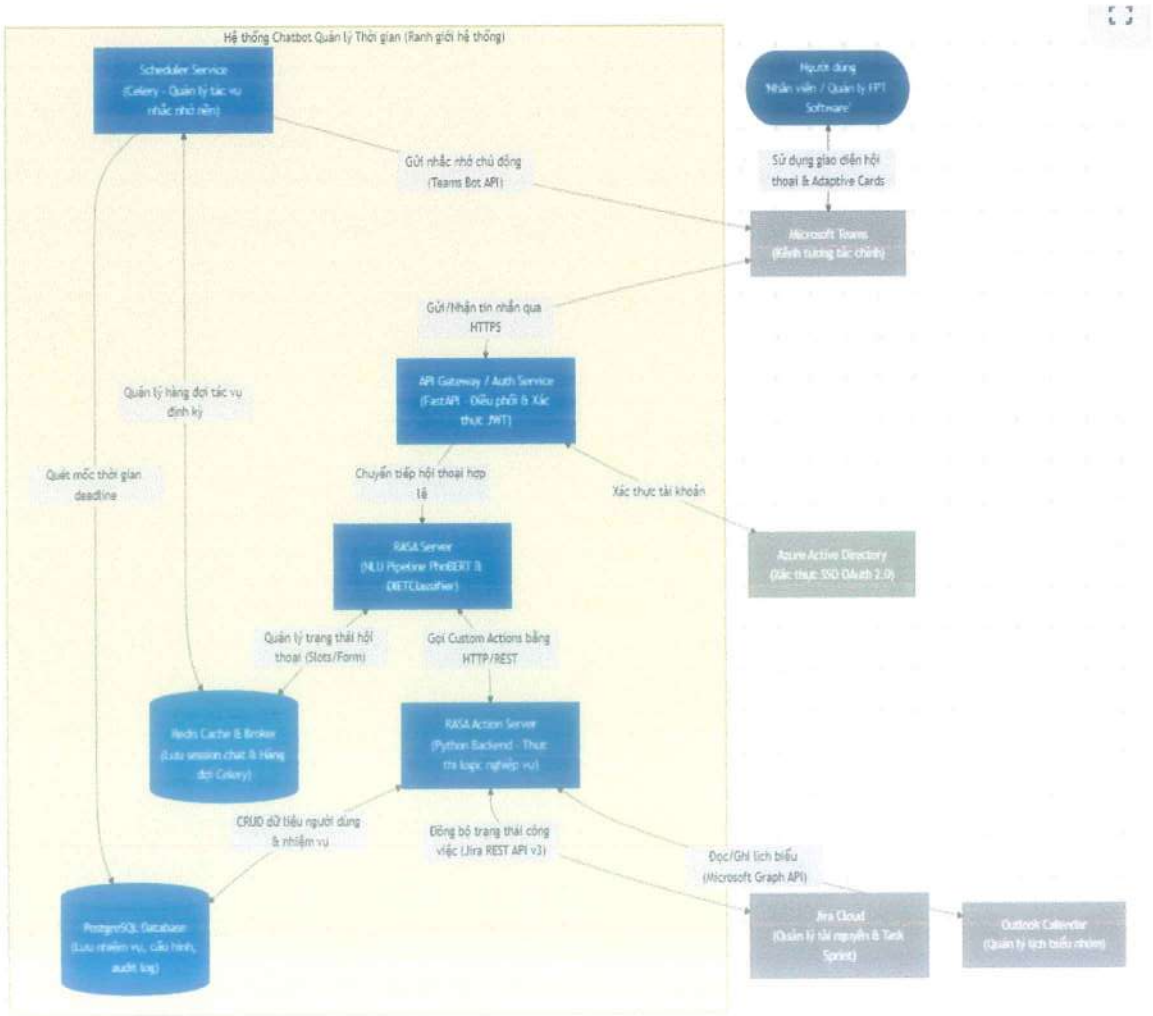
Lớp Dữ liệu (Data Layer): PostgreSQL lưu trữ dữ liệu người dùng, nhiệm vụ, lịch sử hội thoại và cấu hình hệ thống. Redis được sử dụng cho cache và quản lý session. Message Queue (RabbitMQ) xử lý các tác vụ bất đồng bộ như gửi nhắc nhở định kỳ.

Lớp Hạ tầng (Infrastructure Layer): Toàn bộ hệ thống được containerized bằng Docker và quản lý bởi Kubernetes, triển khai trên hạ tầng on-premise của FPT Software. Prometheus và Grafana được sử dụng để giám sát hiệu suất hệ thống theo thời gian thực.

Hình 3.1 mô tả kiến trúc tổng thể của hệ thống theo mô hình C4 (Context, Container, Component, Code). Ở cấp độ Context: người dùng (nhân viên FPT Software) tương tác với hệ thống qua Microsoft Teams; hệ thống chatbot kết nối với Jira (Atlassian Cloud) và Outlook Calendar (Microsoft 365). Ở cấp độ Container, hệ thống bao gồm năm container chính: (1) RASA Server – xử lý NLU và Dialogue Management; (2) Action Server – thực thi logic nghiệp vụ; (3) Scheduler Service –

quản lý nhắc nhở định kỳ; (4) API Gateway – xác thực và phân phối yêu cầu; (5) Database – PostgreSQL và Redis.

Để tối ưu hóa khả năng chịu tải trên toàn bộ kiến trúc phân tán, các điểm nghẽn cổ chai (bottlenecks) tiềm ẩn đều được xử lý ở thiết kế cấp cơ sở hạ tầng. Cụ thể, Message Queue (RabbitMQ) được thiết lập dưới dạng cụm (cluster) để đảm bảo việc phân tán hàng ngàn tác vụ nhắc nhở đồng thời từ Scheduler Service không gây nghẽn hàng đợi. Đối với cơ sở dữ liệu PostgreSQL, cơ chế Replica Set được áp dụng để tách biệt luồng truy vấn Đọc và Ghi (Read/Write splitting). Nhờ vậy, các tác vụ đòi hỏi tài nguyên lớn như trích xuất báo cáo tiến độ tuần (Read) sẽ không làm ảnh hưởng đến độ trễ của các giao dịch ghi nhận trạng thái nhiệm vụ thời gian thực (Write).



Hình 3.1: Kiến trúc hệ thống theo mô hình C4 Container Diagram

Về cơ chế bảo mật, hệ thống triển khai xác thực theo cơ chế OAuth 2.0 tích hợp với Azure Active Directory của FPT Software. Mỗi yêu cầu đến Action Server đều được xác thực bằng JSON Web Token (JWT) với thời gian hết hạn 1 giờ. Dữ liệu hội thoại trong PostgreSQL được mã hóa at-rest bằng AES-256. Toàn bộ giao tiếp giữa các component được thực hiện qua HTTPS/TLS. Audit log ghi lại mọi hành động của người dùng và hệ thống để phục vụ yêu cầu tuân thủ (compliance) và điều tra sự cố khi cần.

3.2.2 Thiết kế các mô-đun chính của chatbot

Hệ thống được tổ chức thành các mô-đun chức năng chính:

Mô-đun Quản lý nhiệm vụ (Task Management Module): Cung cấp các chức năng CRUD (Create, Read, Update, Delete) cho nhiệm vụ qua hội thoại tự nhiên. Mô-đun này tích hợp với Jira để đồng bộ hóa hai chiều – mọi thay đổi trên chatbot được phản ánh ngay lên Jira và ngược lại. Thiết kế sử dụng RASA Forms để thu thập đầy đủ thông tin cần thiết khi tạo nhiệm vụ mới.

Mô-đun Quản lý lịch biểu (Schedule Management Module): Tích hợp với Outlook Calendar qua Microsoft Graph API để đọc/ghi lịch biểu. Mô-đun xây dựng thuật toán phát hiện xung đột lịch trình đơn giản và đề xuất thời gian thay thế. Hỗ trợ các lệnh ngôn ngữ tự nhiên như: "Kiểm tra lịch tuần sau", "Đặt lịch họp team vào thứ Ba lúc 10 giờ", "Hủy cuộc họp chiều nay".

Mô-đun Nhắc nhở thông minh (Smart Reminder Module): Scheduler service chạy nền, định kỳ quét danh sách nhiệm vụ và sự kiện trong cơ sở dữ liệu để phát hiện các mốc cần nhắc nhở. Module sử dụng Celery beat với Redis broker để quản lý tác vụ định kỳ. Logic quyết định thời điểm nhắc nhở dựa trên: thời hạn công việc, múi giờ của người dùng, lịch làm việc (không nhắc ngoài giờ làm) và lịch sử hành vi nhắc nhở.

Mô-đun Báo cáo và Phân tích (Reporting & Analytics Module): Tự động tổng hợp dữ liệu từ Jira và lịch làm việc để tạo báo cáo tiến độ cá nhân (hàng ngày

vào cuối giờ làm) và nhóm (hàng tuần vào sáng thứ Hai). Báo cáo được trình bày dưới dạng Adaptive Cards trong Teams với các số liệu trực quan: số nhiệm vụ hoàn thành/còn lại/quá hạn, tiến độ so với kế hoạch sprint.

3.2.3 Thiết kế quy trình tương tác giữa chatbot và người dùng

Quy trình tương tác được thiết kế theo nguyên tắc "conversational first" – ưu tiên trải nghiệm hội thoại tự nhiên, tránh yêu cầu người dùng tuân theo cú pháp cứng nhắc. Một số luồng hội thoại quan trọng được thiết kế chi tiết:

Luồng tạo nhiệm vụ mới: Người dùng có thể bắt đầu bằng câu đơn giản như "Tạo task mới" hoặc câu phức tạp hơn như "Nhắc tôi hoàn thành báo cáo kỹ thuật cho dự án X trước thứ Sáu tuần này". Trong trường hợp đầu, chatbot sẽ thu thập dần thông tin qua Form. Trong trường hợp sau, chatbot trích xuất đầy đủ thông tin từ câu đầu vào và xác nhận một lần trước khi tạo.

Luồng kiểm tra lịch và cập nhật tiến độ: Mỗi sáng, chatbot gửi tóm tắt công việc ngày hôm nay chủ động. Người dùng có thể phản hồi ngay để cập nhật trạng thái công việc bằng ngôn ngữ tự nhiên như "Xong rồi", "Chuyển sang đang làm" hoặc "Dời deadline sang thứ Sáu".

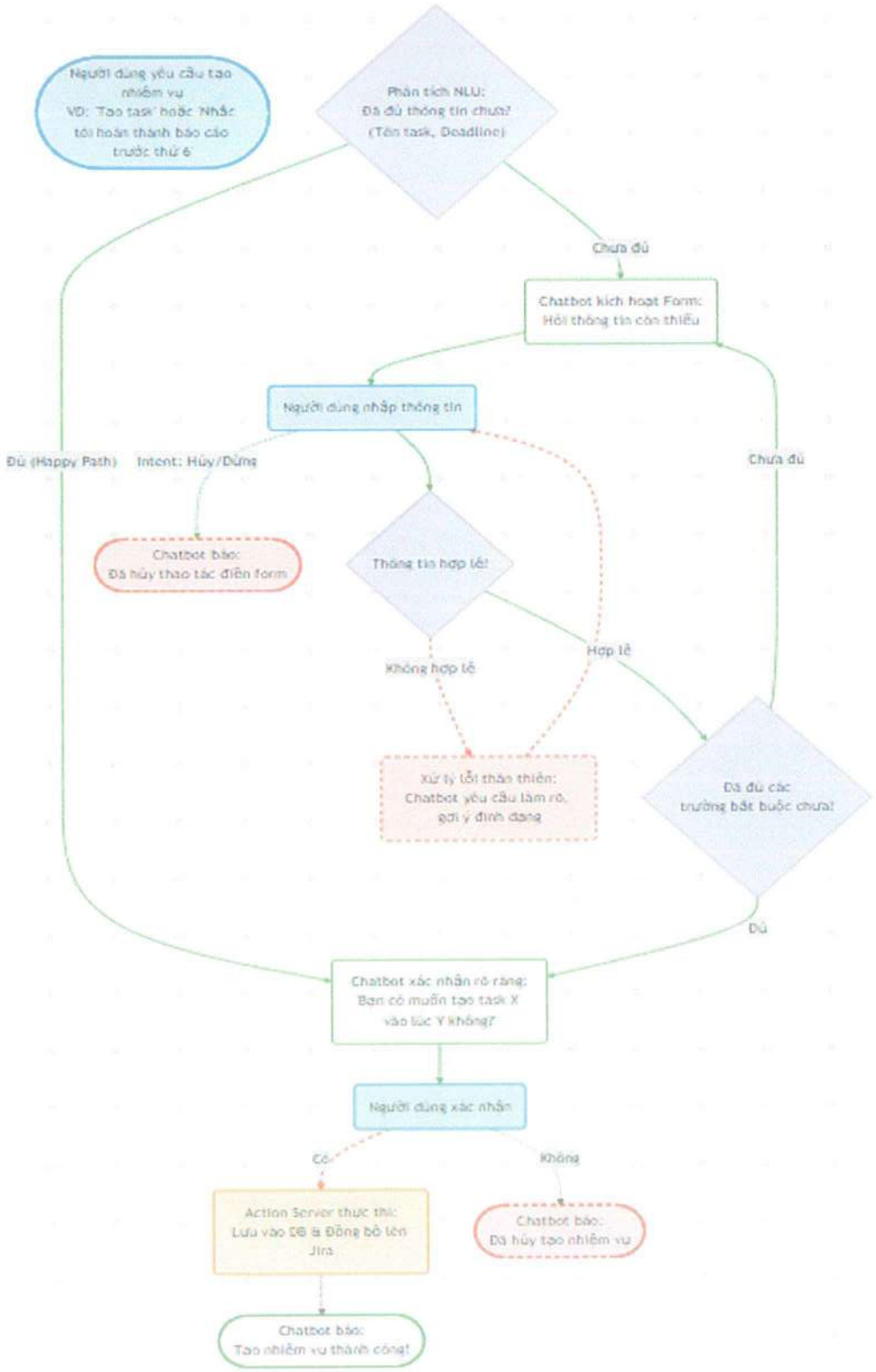
Các nguyên tắc thiết kế UX (User Experience) được tuân thủ trong toàn bộ quy trình tương tác:

Nguyên tắc 1 – Xác nhận rõ ràng: Mọi hành động có tác động (tạo mới, cập nhật, xóa lịch, nhắc nhở) đều được chatbot xác nhận lại trước khi thực thi, tránh tình trạng thực hiện nhầm do chatbot hiểu sai ý định người dùng. Đặc biệt, các hành động không thể hoàn tác được yêu cầu xác nhận tường minh theo mẫu: "Bạn có chắc chắn muốn [hành động] không? Nhấn Có để xác nhận."

Nguyên tắc 2 – Xử lý lỗi thân thiện: Khi chatbot không hiểu yêu cầu hoặc thiếu thông tin, phản hồi không phải là thông báo lỗi kỹ thuật mà là câu hỏi làm rõ tự nhiên, gợi ý định dạng mong muốn. Ví dụ: thay vì "Intent confidence below

threshold", chatbot phản hồi: "Xin lỗi, mình chưa hiểu rõ. Bạn muốn mình làm gì với lịch họp đó? Bạn có thể nói cụ thể hơn về thời gian hoặc tên cuộc họp không?"

Nguyên tắc 3 – Phản hồi nhanh và súc tích: Phản hồi của chatbot được thiết kế ngắn gọn, đúng trọng tâm, tránh văn bản dài làm người dùng mất kiên nhẫn. Thông tin phức tạp (danh sách nhiệm vụ, báo cáo) được trình bày dưới dạng Adaptive Cards với định dạng trực quan thay vì văn bản thuần túy.



Hình 3.2: Sơ đồ luồng hội thoại tạo nhiệm vụ mới (Happy Path và Unhappy Paths)

3.3 Phát triển hệ thống chatbot

3.3.1 Các công cụ và ngôn ngữ lập trình được sử dụng

Hệ thống được phát triển với bộ công nghệ (technology stack) được lựa chọn kỹ lưỡng dựa trên tiêu chí: khả năng tương thích với RASA, hiệu suất, độ chín muồi của công nghệ và nguồn nhân lực có sẵn tại FPT Software.

Ngôn ngữ lập trình chính: Python 3.10+ được sử dụng cho toàn bộ backend, bao gồm RASA Action Server, các adapter tích hợp và API service. Python là lựa chọn tự nhiên vì hệ sinh thái thư viện AI/ML phong phú và là ngôn ngữ chính thức của RASA.

Framework và thư viện chính: RASA 3.x (Dialogue AI), FastAPI (REST API service), SQLAlchemy (ORM cho PostgreSQL), Celery + Redis (task queue cho nhắc nhở nền), pytest (unit và integration testing), Docker + Docker Compose (containerization).

Công cụ phát triển: Visual Studio Code với các extension Python/RASA, Rasa X (giao diện web để quản lý dữ liệu huấn luyện và kiểm thử hội thoại), Postman (kiểm thử API), GitLab CI/CD (tích hợp và triển khai liên tục), Kubernetes (quản lý container trên môi trường production).

3.3.2 Quá trình phát triển các thành phần của chatbot

Quá trình phát triển được tổ chức theo phương pháp Agile với các sprint 2 tuần. Mỗi sprint bao gồm các giai đoạn: lập kế hoạch sprint (sprint planning), phát triển (development), kiểm thử (testing) và demo/review.

Sprint 1–2 (Tháng 10/2024): Hạ tầng và NLU cơ bản – Thiết lập môi trường phát triển và CI/CD pipeline; triển khai RASA Server cơ bản với dữ liệu huấn luyện NLU ban đầu cho các ý định chính (tạo_nhiệm_vụ, xem_lịch, kiểm_tra_deadline, cập_nhật_trạng_thái); tích hợp Microsoft Teams Connector; thiết lập cơ sở dữ liệu PostgreSQL.

Sprint 3–4 (Tháng 11/2024): Chức năng cốt lõi – Phát triển Form thu thập thông tin nhiệm vụ; xây dựng Jira Adapter và thực hiện đồng bộ hai chiều giữa chatbot và Jira; phát triển Reminder Service với Celery; tích hợp Microsoft Graph API cho Outlook Calendar.

Sprint 5–6 (Tháng 12/2024): Tính năng nâng cao và tối ưu hóa – Phát triển Reporting Module với Adaptive Cards; cải thiện độ chính xác NLU bằng cách mở rộng tập dữ liệu huấn luyện; tối ưu hóa hiệu suất (caching, database indexing); triển khai giám sát với Prometheus/Grafana.

Một thách thức kỹ thuật đặc thù gặp phải trong quá trình phát triển là xử lý biểu thức thời gian tiếng Việt đa dạng. Tiếng Việt có nhiều cách diễn đạt thời gian khác nhau, từ tương đối ("ngày kia", "tuần tới", "sáng mai", "cuối tháng") đến tuyệt đối ("15 tháng 11", "11/15"), và kết hợp phức tạp ("thứ Sáu tuần sau lúc xế chiều"). Để giải quyết, đề án phát triển một module TimeParser chuyên biệt kết hợp: (1) Regular expressions cho các mẫu thời gian cố định; (2) Danh sách từ điển biểu thức thời gian tiếng Việt đặc thù; (3) Gọi mô hình NER của PhoBERT để xử lý các biểu thức phức tạp hơn. Module TimeParser đạt độ chính xác 87,4% trên tập kiểm thử gồm 520 biểu thức thời gian đa dạng được thu thập từ log hội thoại thực tế.

3.3.3 Kiểm thử và điều chỉnh chatbot dựa trên phản hồi từ người dùng

Chiến lược kiểm thử áp dụng trong đề án bao gồm nhiều cấp độ:

Kiểm thử đơn vị NLU (NLU Unit Testing): Sử dụng lệnh "rasa test nlu" để đánh giá hiệu năng của mô hình NLU trên tập dữ liệu kiểm thử độc lập. Các chỉ số được theo dõi bao gồm: Intent Accuracy, Entity F1-score, và Confusion Matrix để phát hiện các ý định dễ bị nhầm lẫn.

Kiểm thử hội thoại đầu cuối (End-to-End Testing): Sử dụng "rasa test" với các test stories được xây dựng sẵn để kiểm tra toàn bộ luồng hội thoại, đảm bảo chatbot phản hồi đúng cho các kịch bản tiêu biểu.

Kiểm thử người dùng (User Acceptance Testing – UAT): Được thực hiện với 14 nhân viên thí điểm trong vòng 2 tuần. Người dùng tương tác tự do với chatbot và điền bảng đánh giá cuối mỗi tuần. Kết quả UAT được dùng để xác định các lỗi còn tồn tại và ưu tiên cải tiến trong sprint tiếp theo.

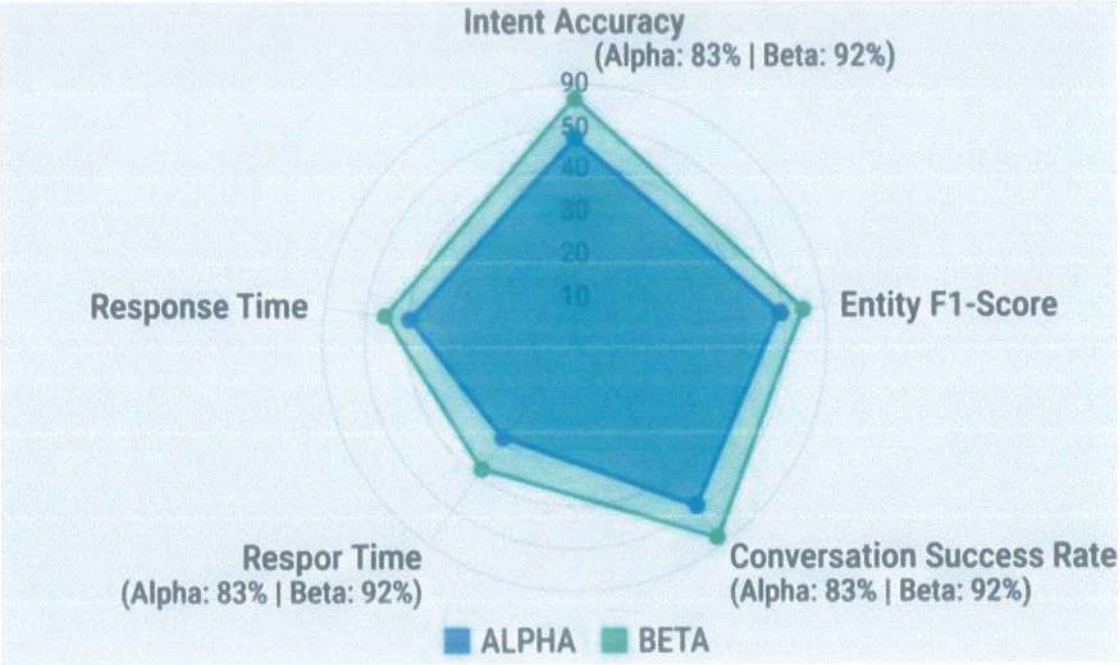
Một kết quả quan trọng từ quá trình kiểm thử NLU là phân tích ma trận nhầm lẫn (Confusion Matrix) cho thấy một số cặp ý định dễ bị nhầm lẫn nhất, bao gồm: (1) tạo_nhiệm_vụ vs. cập_nhật_nhiệm_vụ – khi người dùng viết câu như "Thêm deadline cho task X" có thể bị phân loại nhầm; (2) xem_lịch_hôm_nay vs. xem_lịch_tuần – khi người dùng không chỉ định rõ khoảng thời gian; (3) hủy_sự_kiện vs. dời_lịch_sự_kiện – hai hành động có ngữ nghĩa gần nhau nhưng hậu quả hoàn toàn khác. Để giải quyết, các cặp ý định này được bổ sung thêm dữ liệu huấn luyện đặc thù, và một số trường hợp được xử lý bằng cách thêm câu hỏi làm rõ (clarification question) khi confidence score của ý định dự đoán thấp hơn ngưỡng 0.7.

Bảng 3.2 trình bày kết quả kiểm thử NLU sau giai đoạn Alpha (vòng 1) và sau khi cải thiện dữ liệu huấn luyện (vòng 2):

Bảng 3.2: Kết quả kiểm thử NLU qua các vòng cải tiến

Chỉ số đánh giá	Vòng 1 (Alpha)	Vòng 2 (Beta)	Mục tiêu NFR
Intent Accuracy	83,2%	92,3%	≥ 90%
Entity F1-score	76,8%	88,1%	≥ 85%
Thời gian phản hồi TB	1.850 ms	1.240 ms	≤ 2000 ms
Tỷ lệ hội thoại thành công	74,1%	86,3%	≥ 80%

Hình 3.3: Đánh giá hiệu suất Mô hình PHOBERT



3.4 Thử nghiệm hệ thống chatbot

3.4.1 Mô tả kịch bản thử nghiệm và các tiêu chí đánh giá

Bốn kịch bản thử nghiệm chính được xây dựng dựa trên các use case phổ biến nhất từ kết quả khảo sát:

Kịch bản 1 – Quản lý nhiệm vụ sprint: Nhân viên sử dụng chatbot để đồng bộ danh sách tasks từ Jira, cập nhật trạng thái tasks trong ngày và nhận tóm tắt tiến độ cuối ngày. Tiêu chí đánh giá: tỷ lệ đồng bộ thành công, độ chính xác nhận diện nhiệm vụ, thời gian hoàn thành so với thao tác thủ công trên Jira.

Kịch bản 2 – Quản lý cuộc họp: Người dùng yêu cầu chatbot lên lịch cuộc họp nhóm, chatbot kiểm tra lịch rảnh của tất cả thành viên qua Outlook Calendar, đề xuất khung giờ phù hợp và tạo calendar invite. Tiêu chí: tỷ lệ tìm được khung giờ phù hợp, thời gian xử lý, độ hài lòng với đề xuất.

Kịch bản 3 – Nhắc nhở deadline: Đánh giá hệ thống nhắc nhở qua 4 tuần thử nghiệm thực tế. Tiêu chí: tỷ lệ nhắc nhở gửi đúng thời điểm, tỷ lệ deadline được hoàn thành trước/đúng hạn so với nhóm đối chứng không dùng chatbot.

Kịch bản 4 – Báo cáo tự động: Chatbot tự động tổng hợp và gửi báo cáo tiến độ sprint hàng tuần cho cả nhân viên và quản lý. Tiêu chí: độ chính xác của thông tin trong báo cáo, thời gian tiết kiệm được so với làm báo cáo thủ công, mức độ hài lòng của quản lý.



Hình 3.4: Hình ảnh mô phỏng ứng dụng Chatbot giúp quản lý thời gian của nhân viên tại FPT Software

3.4.2 Quá trình triển khai thử nghiệm tại FPT Software

Quá trình triển khai thử nghiệm được thực hiện theo mô hình phát hành dần dần (gradual rollout) nhằm kiểm soát rủi ro và thu thập phản hồi liên tục:

Giai đoạn Alpha (2 tuần): Triển khai cho nhóm 12 nhân viên tự nguyện trong một team phát triển phần mềm. Mục tiêu chính là phát hiện các lỗi kỹ thuật nghiêm trọng và các vấn đề về UX cơ bản. Thu thập log hội thoại (sau khi ẩn danh) để cải thiện mô hình NLU.

Giai đoạn Beta (4 tuần): Mở rộng triển khai cho 48 nhân viên thuộc 5 team khác nhau. Thực hiện phỏng vấn hàng tuần với một mẫu người dùng ngẫu nhiên để theo dõi xu hướng sử dụng và phát hiện vấn đề. Trong giai đoạn này, mô hình NLU được cập nhật hai lần dựa trên dữ liệu thực tế thu thập được.

Một điều đáng chú ý trong quá trình triển khai Beta là việc xử lý các trường hợp ngoài phạm vi (out-of-scope requests). Người dùng thường kỳ vọng chatbot có thể xử lý nhiều loại yêu cầu hơn phạm vi thiết kế ban đầu, chẳng hạn như đặt câu hỏi về chính sách nhân sự (đây là chức năng của Maya cũ) hoặc yêu cầu chatbot tóm tắt nội dung tài liệu. Để xử lý gracefully, hệ thống được cấu hình để nhận diện và phản hồi các yêu cầu ngoài phạm vi bằng cách giải thích giới hạn chức năng và gợi ý người dùng sử dụng kênh phù hợp.

Trong giai đoạn thử nghiệm, một số điều chỉnh quan trọng được thực hiện dựa trên phản hồi trực tiếp từ người dùng: (1) Thêm tính năng xem "Top 3 nhiệm vụ ưu tiên hôm nay" ngay trong tin nhắn chào buổi sáng – được nhiều người dùng yêu cầu; (2) Điều chỉnh định dạng Adaptive Cards báo cáo sprint từ dạng danh sách dài thành dạng tóm tắt theo trạng thái (To Do / In Progress / Done) – dễ đọc hơn đáng kể; (3) Bổ sung tính năng snooze cho nhắc nhở ("Nhắc tôi lại sau 30 phút") sau khi nhiều người dùng yêu cầu.

3.5 Đánh giá hiệu quả

3.5.1 Phân tích dữ liệu thu được từ quá trình thử nghiệm

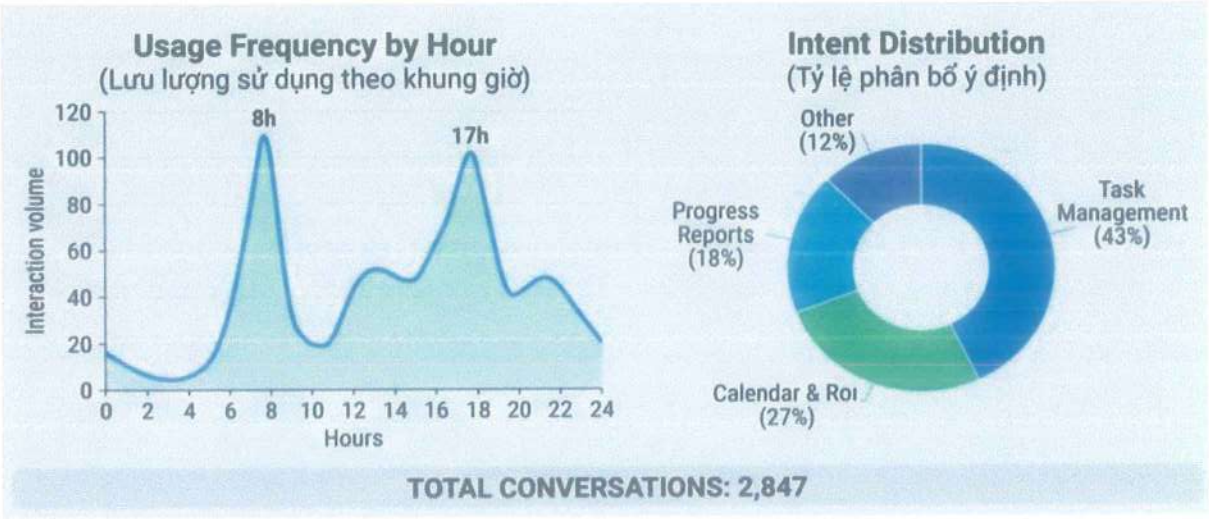
Dữ liệu thu thập trong quá trình thử nghiệm bao gồm: log hội thoại (đã ẩn danh), nhật ký hiệu suất hệ thống, dữ liệu Jira (tỷ lệ hoàn thành công việc), và kết quả khảo sát người dùng. Phân tích dữ liệu cho thấy:

Tổng số cuộc hội thoại trong giai đoạn thử nghiệm: 2.847. Phân bố theo loại ý định: quản lý nhiệm vụ chiếm 43%, kiểm tra lịch 27%, nhận báo cáo 18%, các ý định khác 12%. Tỷ lệ hội thoại thành công (completed successfully) đạt 86,3%.

Phân tích xu hướng sử dụng theo thời gian trong ngày cho thấy đỉnh sử dụng tập trung vào hai khung giờ: 8h00–9h00 (nhân viên kiểm tra lịch và nhiệm vụ đầu ngày) và 16h30–17h30 (cập nhật trạng thái công việc cuối ngày và chuẩn bị kế hoạch ngày mai). Đây là thông tin có giá trị để tối ưu hóa lịch gửi tóm tắt tự động và phân bổ tài nguyên tính toán cho hệ thống.

Phân tích dữ liệu log cũng tiết lộ một số pattern hành vi người dùng thú vị: (1) ~30% người dùng sử dụng tính năng cập nhật trạng thái Jira qua chatbot thay vì truy cập trực tiếp vào Jira, cho thấy mức độ tiện lợi được đánh giá cao; (2) Tỷ lệ người dùng đọc và phản hồi báo cáo tóm tắt hàng ngày đạt ~70%, cao hơn đáng kể so với tỷ lệ đọc email nội bộ thông thường (~40%); (3) Một nhóm nhỏ (~15%) người dùng nâng cao sử dụng chatbot cho các tác vụ phức tạp như tạo lịch họp nhiều người, cho thấy tiềm năng mở rộng tính năng cao cấp.

Hình 3.5: Biểu đồ lưu lượng sử dụng



3.5.2 Đánh giá hiệu quả của chatbot trong việc hỗ trợ quản lý thời gian

Đánh giá hiệu quả của hệ thống chatbot dựa trên ba nhóm chỉ số chính:

Chỉ số kỹ thuật: Độ chính xác nhận diện ý định (Intent Accuracy) trên tập kiểm thử đạt 92,3%; F1-score cho nhận diện thực thể đạt 88,1%. Thời gian phản hồi trung bình đạt 1.240 ms, đáp ứng yêu cầu NFR01. Tỷ lệ sẵn sàng hệ thống (uptime) đạt 99,94%, vượt yêu cầu SLA 99.9%.

Chỉ số nghiệp vụ: Tỷ lệ hoàn thành công việc đúng hạn của nhóm dùng chatbot tăng 21% so với trước khi sử dụng. Thời gian trung bình cập nhật trạng thái công việc giảm từ 22 phút (thao tác trực tiếp trên Jira) xuống còn 6 phút (qua chatbot). Thời gian làm báo cáo tiến độ hàng tuần giảm 76%.

Chỉ số hài lòng người dùng: Điểm hài lòng tổng thể (CSAT) đạt 4,1/5 điểm. Tỷ lệ người dùng tiếp tục sử dụng sau 4 tuần thử nghiệm đạt 83%, cho thấy mức độ gắn kết (engagement) tốt.

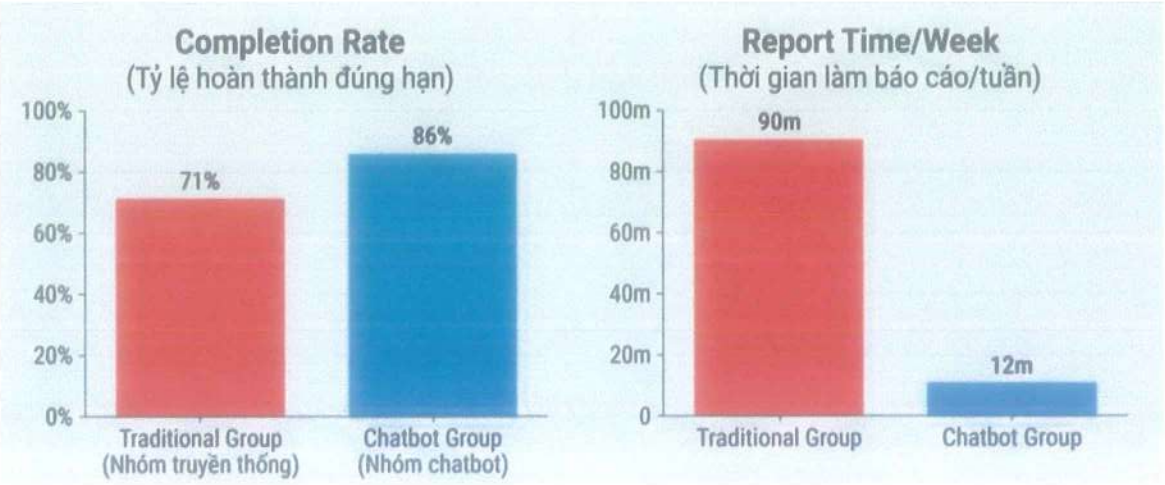
3.5.3 So sánh hiệu quả của chatbot với các phương pháp quản lý thời gian truyền thống

Bảng 3.3 dưới đây tóm tắt kết quả so sánh giữa phương pháp quản lý thời gian truyền thống (thao tác thủ công trên Jira + Teams/Email) và hệ thống chatbot mới theo các tiêu chí chính:

Bảng 3.3: So sánh hiệu quả chatbot và phương pháp truyền thống

Tiêu chí đánh giá	Phương pháp truyền thống	Chatbot mới
Thời gian cập nhật trạng thái/ngày	~22 phút	~6 phút
Tỷ lệ hoàn thành đúng hạn	71%	86%
Thời gian làm báo cáo tuần	~90 phút	~12 phút (tự động)
Điểm hài lòng nhân viên (1–5)	3,2 / 5	4,1 / 5

Hình 3.6: Dashboard đánh giá hiệu quả A/B Testing



Ngoài so sánh định lượng ở bảng trên, kết quả phỏng vấn người dùng sau thử nghiệm cũng cung cấp những nhận định định tính có giá trị. Nhiều kỹ sư chia sẻ rằng tính năng tóm tắt nhiệm vụ buổi sáng giúp họ "bắt đầu ngày làm việc nhanh hơn mà không cần mở Jira ngay". Các quản lý đánh giá cao khả năng nhận báo cáo tiến độ nhóm tổng hợp tự động, giúp tiết kiệm thời gian chuẩn bị cho các buổi review hàng tuần. Một số người dùng cũng nhận xét rằng việc có thể cập nhật trạng thái công việc ngay trong Teams mà không cần chuyển sang Jira giúp giảm đáng kể sự gián đoạn trong luồng làm việc.

3.6 Các đề xuất cải tiến

3.6.1 Những điểm mạnh và điểm yếu của hệ thống chatbot

Điểm mạnh: Hệ thống chatbot được phát triển thể hiện nhiều điểm mạnh nổi bật. Thứ nhất, khả năng xử lý tiếng Việt tự nhiên đạt chất lượng tốt nhờ sử dụng PhoBERT, vượt trội so với các chatbot sử dụng mô hình đa ngôn ngữ thông thường. Thứ hai, tích hợp liền mạch với hệ sinh thái Microsoft (Teams, Outlook) và Jira là những công cụ nhân viên FPT Software sử dụng hàng ngày, giảm thiểu sự thay đổi thói quen làm việc. Thứ ba, kiến trúc microservices đảm bảo khả năng mở rộng và bảo trì dễ dàng.

Điểm yếu: Bên cạnh những thành tựu đạt được, hệ thống còn tồn tại một số hạn chế cần cải thiện. Đáng chú ý nhất là: (1) Khả năng xử lý các câu phức tạp với nhiều yêu cầu trong một câu (multi-intent) còn hạn chế; (2) Thiếu cơ chế học liên tục (online learning) – mô hình chỉ được cập nhật theo batch định kỳ, chưa học ngay từ phản hồi tức thời của người dùng; (3) Chức năng gợi ý tối ưu lịch làm việc chưa được triển khai đầy đủ do phức tạp về thuật toán; (4) Khả năng hiểu ngữ cảnh liên phiên (cross-session context) còn yếu. (5) Kiến trúc microservices tuy mang lại khả năng mở rộng linh hoạt nhưng lại tạo ra thách thức lớn trong việc giám sát hệ thống (monitoring) và gỡ lỗi (debugging). Khi hệ thống mở rộng lên hàng chục node và xử lý hàng trăm yêu cầu đồng thời, việc thiếu vắng một cơ chế truy vết phân tán (Distributed Tracing) hoặc quản lý log tập trung (Centralized Logging) khiến quá

trình phân tích nguyên nhân gốc rễ (root-cause analysis) qua nhiều dịch vụ (từ API Gateway qua RASA Server đến Action Server) tiêu tốn nhiều thời gian và công sức của đội ngũ vận hành.

3.6.2 Đề xuất các giải pháp cải tiến và nâng cao hiệu quả của hệ thống

Dựa trên phân tích điểm yếu và phản hồi từ người dùng, các hướng cải tiến ưu tiên được đề xuất cho giai đoạn tiếp theo:

Cải tiến ngắn hạn (1–3 tháng): (1) Bổ sung khả năng xử lý multi-intent bằng cách phân tách câu đầu vào thành các ý định riêng biệt trước khi xử lý; (2) Cải thiện cơ sở dữ liệu huấn luyện với thêm các ví dụ từ log thực tế, đặc biệt cho các trường hợp edge case; (3) Tích hợp tính năng xem lịch sử nhiệm vụ và phân tích xu hướng hoàn thành công việc theo thời gian.

Cải tiến trung hạn (3–6 tháng): (1) Triển khai thuật toán gợi ý lịch làm việc thông minh dựa trên dữ liệu lịch sử, độ ưu tiên nhiệm vụ và giờ làm việc hiệu quả nhất của từng nhân viên; (2) Xây dựng tính năng cross-session memory để chatbot nhớ sở thích và thói quen của người dùng giữa các phiên hội thoại; (3) Mở rộng tích hợp với GitLab để theo dõi commit và merge request liên quan đến tasks.

Cải tiến dài hạn (6–12 tháng): (1) Nghiên cứu khả năng tích hợp các mô hình ngôn ngữ lớn (LLM) như GPT-4 hoặc các mô hình tiếng Việt chuyên biệt để nâng cao chất lượng phản hồi và xử lý ngữ cảnh phức tạp; (2) Phát triển tính năng phân tích dự báo (predictive analytics) để cảnh báo sớm các rủi ro chậm tiến độ dự án dựa trên dữ liệu lịch sử; (3) Mở rộng phạm vi triển khai ra toàn bộ công ty FPT Software sau khi hoàn thiện từ kết quả thí điểm. "Để phục vụ quy mô hàng chục ngàn nhân viên trên toàn công ty, lộ trình mở rộng đòi hỏi hệ thống phải áp dụng chiến lược triển khai không thời gian chết (Zero-Downtime Deployment). Việc tích hợp các phương pháp kỹ thuật như Blue/Green Deployment hoặc Canary Release vào luồng CI/CD (GitLab CI) sẽ đảm bảo các bản cập nhật tính năng mới hoặc mô hình AI mới được phát hành an toàn mà không gây gián đoạn trải nghiệm của người dùng. Đồng thời, hạ tầng cũng cần được nâng cấp theo mô hình độ sẵn sàng cao (High

Availability) với cơ chế dự phòng tự động (Auto-Failover) nhằm đáp ứng cam kết chất lượng dịch vụ (SLA) ở mức 99,99% trong môi trường doanh nghiệp quy mô lớn.

3.7 Kết luận chương 3

Chương 3 đã trình bày toàn diện quá trình từ khảo sát nhu cầu, thiết kế kiến trúc, phát triển đến thử nghiệm và đánh giá hệ thống chatbot quản lý thời gian tại FPT Software. Hệ thống được xây dựng dựa trên RASA Framework kết hợp với mô hình ngôn ngữ PhoBERT đã đạt được các mục tiêu đề ra về độ chính xác NLU, hiệu suất hệ thống và tính năng chức năng.

Kết quả thử nghiệm thực tế tại FPT Software cho thấy chatbot mang lại giá trị cụ thể và đo lường được: giảm thời gian quản lý thời gian thủ công, cải thiện tỷ lệ hoàn thành đúng hạn và tăng sự hài lòng của nhân viên. Đây là những bằng chứng thực tiễn quan trọng khẳng định tính khả thi và hiệu quả của hướng nghiên cứu.

Các hạn chế còn tồn tại và lộ trình cải tiến đề xuất cung cấp hướng đi rõ ràng cho các nghiên cứu tiếp theo, mở ra tiềm năng phát triển hệ thống thành một nền tảng AI hỗ trợ toàn diện cho năng suất làm việc tại FPT Software và rộng hơn là tại các doanh nghiệp phần mềm Việt Nam.

III. KẾT LUẬN

Đề án "Nghiên cứu và phát triển ứng dụng chatbot giúp quản lý thời gian của nhân viên tại FPT Software" đã tiếp cận một vấn đề thiết thực và cấp bách trong hoạt động của các doanh nghiệp phần mềm hiện đại. Thông qua quá trình nghiên cứu lý thuyết chuyên sâu và triển khai thực nghiệm, đề án đã đạt được các kết quả mong đợi như sau:

Thứ nhất – Khảo sát tổng quan toàn diện: Đề án đã cung cấp một cái nhìn hệ thống và toàn diện về lĩnh vực quản lý thời gian trong doanh nghiệp và ứng dụng công nghệ chatbot. Từ việc phân tích các phương pháp quản lý thời gian truyền thống cùng những hạn chế của chúng, đến việc đánh giá tiềm năng của chatbot AI trong việc giải quyết các thách thức đó, đề án tạo ra cơ sở lý thuyết vững chắc cho hướng nghiên cứu ứng dụng AI trong quản lý doanh nghiệp tại Việt Nam.

Thứ hai – Nghiên cứu công nghệ nền tảng: Đề án đã nghiên cứu đầy đủ và có chiều sâu các công nghệ nền tảng, bao gồm AI/Machine Learning, NLP, và đặc biệt là RASA Framework. Việc phân tích chi tiết kiến trúc và cơ chế hoạt động của RASA, kết hợp với mô hình ngôn ngữ PhoBERT cho tiếng Việt, cung cấp tri thức kỹ thuật có giá trị cho cộng đồng nghiên cứu NLP và phát triển chatbot tiếng Việt.

Thứ ba – Đề xuất và triển khai giải pháp thực tiễn: Trên cơ sở khảo sát nhu cầu thực tế và nghiên cứu công nghệ, đề án đã đề xuất và phát triển một kiến trúc hệ thống chatbot hoàn chỉnh, phù hợp với đặc thù của FPT Software. Hệ thống được thiết kế theo kiến trúc microservices hiện đại, tích hợp với hệ sinh thái Microsoft (Teams, Outlook) và công cụ quản lý dự án Jira, đảm bảo khả năng triển khai thực tế và mở rộng trong tương lai.

Thứ tư – Thử nghiệm và đánh giá: Quá trình thử nghiệm tại FPT Software đã cung cấp bằng chứng thực nghiệm về hiệu quả của hệ thống. Kết quả cho thấy hệ thống chatbot không chỉ đáp ứng các yêu cầu kỹ thuật đề ra (độ chính xác NLU, hiệu suất) mà còn mang lại lợi ích nghiệp vụ đo lường được (giảm thời gian quản lý thủ công, cải thiện tỷ lệ hoàn thành đúng hạn).

Đề án cũng nhận diện rõ các hạn chế còn tồn tại và đề xuất lộ trình cải tiến cụ thể, tạo nền tảng cho việc phát triển hệ thống lên mức cao hơn trong tương lai. Đặc biệt, việc tích hợp các mô hình ngôn ngữ lớn (LLM) và khả năng học liên tục là những hướng phát triển đầy triển vọng.

Kết quả của đề án không chỉ có giá trị ứng dụng trực tiếp tại FPT Software mà còn đóng góp vào kho tri thức nghiên cứu ứng dụng AI trong quản lý doanh nghiệp tại Việt Nam, mở ra nhiều hướng nghiên cứu tiếp nối trong lĩnh vực này.

Nhìn về tương lai, việc ứng dụng chatbot AI trong quản lý doanh nghiệp hứa hẹn còn nhiều tiềm năng phát triển. Khi các mô hình ngôn ngữ lớn (LLM) ngày càng trở nên mạnh mẽ và kinh tế hơn, ranh giới giữa chatbot truyền thống và trợ lý AI thực sự thông minh đang dần mờ đi. Trong bối cảnh đó, những kết quả nghiên cứu của đề án này – từ kinh nghiệm xử lý tiếng Việt đặc thù, tích hợp với hệ thống doanh nghiệp Việt Nam, đến quy trình thiết kế UX hội thoại phù hợp văn hóa làm việc – sẽ là nền tảng quý giá để tiếp tục phát triển lên thế hệ trợ lý AI doanh nghiệp thế hệ tiếp theo.

Đề án cũng muốn nhấn mạnh rằng thành công của một hệ thống chatbot không chỉ phụ thuộc vào chất lượng công nghệ, mà quan trọng không kém là sự chú trọng đến trải nghiệm người dùng, quản lý thay đổi tổ chức và vòng phản hồi liên tục từ người dùng thực tế. Ba yếu tố này, kết hợp với nền tảng công nghệ vững chắc, tạo ra công thức thành công cho việc triển khai AI trong doanh nghiệp Việt Nam.

IV. TÀI LIỆU THAM KHẢO

Tiếng Việt

[1] Trần Hữu Dũng (2019) – "Xử lý ngôn ngữ tự nhiên và ứng dụng trong chatbot", Tạp chí Khoa học và Công nghệ, Đại học Bách khoa Hà Nội, tr. 45–58.

[2] Phạm Thị Mai (2018) – "Công nghệ chatbot trong doanh nghiệp: Xu hướng và thách thức", Kỷ yếu Hội thảo khoa học – "Công nghệ 4.0 và những ứng dụng trong quản lý doanh nghiệp", tr. 95–102.

[3] Nguyễn Văn Minh (2020) – "Ứng dụng trí tuệ nhân tạo trong quản lý doanh nghiệp", Tạp chí Công nghệ Thông tin Việt Nam, tr. 12–20.

[4] Đỗ Quang Huy (2022) – "Nghiên cứu và phát triển chatbot trong doanh nghiệp Việt Nam", Nhà xuất bản Đại học Quốc gia Hà Nội.

[5] Lê Hữu Phước (2021) – "Quản lý thời gian hiệu quả trong môi trường doanh nghiệp phần mềm", Nhà xuất bản Khoa học Kỹ thuật.

[6] Nguyen, D. Q. & Nguyen, A. T. (2020) – "PhoBERT: Pre-trained language models for Vietnamese", arXiv:2003.00744, Findings of EMNLP 2020.

[7] Nguyễn Đình Thúc (2018) – "Học máy và ứng dụng trong xử lý ngôn ngữ tự nhiên", Nhà xuất bản Đại học Quốc gia TP. Hồ Chí Minh.

Tiếng Anh

[8] Allen, D. (2001) – Getting Things Done: The Art of Stress-Free Productivity. Viking Penguin.

[9] Bocklisch, T., Faulkner, J., Pawlowski, N., & Nichol, A. (2017) – "Rasa: Open source language understanding and dialogue management", arXiv:1712.05181.

[10] Brown, S. R. (2019) – "Natural Language Processing in Modern Chatbots: A Review", Journal of AI Research, 10(2), 101–120.

[11] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019) – "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", Proceedings of NAACL-HLT 2019, 4171–4186.

[12] Jones, M. A. (2020) – "Artificial Intelligence and its Application in Time Management Tools", International Journal of Computer Science, 5(3), 45–58.

[13] Lee, D. L. (2021) – "Time Management Solutions Using AI: A Case Study of Chatbot Implementation", Proceedings of the International Conference on AI Applications, Tokyo, 2021.

[14] Roberts, J. K. (2019) – "Machine Learning for Enhanced Chatbot Performance", Journal of Advanced Computer Science, 11(3), 150–165.

[15] Smith, W. G. (2018) – "Integrating Chatbots into Business Processes: Benefits and Challenges", Journal of Business Technology, 7(4), 67–80.

[16] Vaswani, A. et al. (2017) – "Attention Is All You Need", Advances in Neural Information Processing Systems 30 (NIPS 2017).

[17] Xu, A. et al. (2017) – "A New Chatbot for Customer Service on Social Media", Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, 3506–3510.

Trang Web

[18] https://rasa.com/docs/rasa – Tài liệu chính thức RASA Framework (truy cập tháng 10/2024).

[19] https://towardsdatascience.com/chatbots-using-natural-language-processing (truy cập tháng 9/2024).

[20] https://www.ibm.com/cloud/learn/natural-language-processing (truy cập tháng 9/2024).

[21] https://www.forbes.com/sites/forbestechcouncil/2020/12/01/how-ai-and-chatbots-are-transforming-time-management-in-businesses (truy cập tháng 9/2024).

[22] https://microsoft.github.io/botframework-sdk/ – Microsoft Bot Framework Documentation (truy cập tháng 10/2024).

[23] https://developer.atlassian.com/cloud/jira/platform/rest/v3/ – Jira REST API v3 Documentation (truy cập tháng 11/2024)

BẢN CAM ĐOAN

Tôi cam đoan đã thực hiện việc kiểm tra mức độ tương đồng nội dung đề án qua phần mềm kiemtratailieu.vn một cách trung thực và đạt kết quả mức độ tương đồng **9%** toàn bộ nội dung đề án. Bản đề án kiểm tra qua phần mềm là bản cứng đề án đã nộp để bảo vệ trước hội đồng. Nếu sai tôi xin chịu các hình thức kỷ luật theo quy định hiện hành của Học viện.

Hà Nội, ngày 24 tháng 05 năm 2026

Học viên thực hiện đề án



Trần Nguyên Thịnh

KiểmTraTàiLiệu

BÁO CÁO KIỂM TRA TRÙNG LẶP

Thông tin tài liệu

Tên tài liệu:	PTIT_Dean_Tran_Nguyen_Thinh_Final
Tác giả:	Trần Nguyên Thịnh
Điểm trùng lặp:	8
Thời gian tải lên:	17:29 09/07/2026
Thời gian sinh báo cáo:	17:32 09/07/2026
Các trang kiểm tra:	71/71 trang

Kết quả kiểm tra trùng lặp



Có 8% nội dung trùng lặp

Có 92% nội dung không trùng lặp

Có 0% nội dung người dùng loại trừ

Có 0% nội dung hệ thống bỏ qua

Học Viên

Trần Nguyên Thịnh

Người hướng dẫn khoa học

Phan Thị Hà

BỘ KHOA HỌC VÀ CÔNG NGHỆ
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

BÁO CÁO GIẢI TRÌNH
SỬA CHỮA, HOÀN THIỆN ĐỀ ÁN TỐT NGHIỆP

Họ và tên học viên: Trần Nguyên Thịnh
Chuyên ngành: Hệ thống thông tin
Khóa: 2023 đợt 1

Tên đề tài: Nghiên cứu và phát triển ứng dụng Chatbot giúp quản lý thời gian của nhân viên tại FPT Software

Người hướng dẫn khoa học: TS. Phan Thị Hà

Ngày bảo vệ: 08/07/2025

Các nội dung học viên đã sửa chữa, bổ sung trong đề án tốt nghiệp theo ý kiến đóng góp của Hội đồng chấm đề án tốt nghiệp:

TT	Ý kiến hội đồng	Sửa chữa của học viên
1	Sửa trích dẫn	Học viên đã rà soát, chỉnh sửa lỗi liên quan đến trích dẫn.
2	Hình 3.1; 3.2 phóng to chữ	Tiếp thu góp ý của Hội đồng, tác giả đã sửa lại và phóng to chữ 2 hình 3.1, 3.2.
3	Trình bày thêm về khả năng mở rộng của hệ thống	Tiếp thu góp ý của Hội đồng, tác giả đã bổ sung thêm về khả năng mở rộng của hệ thống.
4	Mô tả thêm UI/UX	Tiếp thu góp ý của Hội đồng, học viên đã bổ sung hình ảnh 3.4 mô phỏng UI/UX của hệ thống.

Hà Nội, ngày 10 tháng 1 năm 2026

Ký xác nhận của

CHỦ TỊCH HỘI ĐỒNG CHẤM ĐỀ ÁN	THƯ KÝ HỘI ĐỒNG	NGƯỜI HƯỚNG DẪN KHOA HỌC	HỌC VIÊN
PGS.TSKH Hoàng Đăng Hải	TS. Đỗ Tiến Dũng	TS. Phan Thị Hà	Trần Nguyên Thịnh

BỘ KHOA HỌC VÀ CÔNG NGHỆ
HỌC VIỆN CÔNG NGHỆ
BƯU CHÍNH VIỄN THÔNG

CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

BIÊN BẢN
HỌP HỘI ĐỒNG CHẤM ĐỀ ÁN TỐT NGHIỆP THẠC SĨ

Căn cứ quyết định số Quyết định số 1895/QĐ-HV ngày 29 tháng 06 năm 2026 của Giám đốc Học viện Công nghệ Bưu chính Viễn thông về việc thành lập Hội đồng chấm đề án tốt nghiệp thạc sĩ. Hội đồng đã họp vào hồi...8....giờ..00..phút, ngày 08 tháng 07 năm 2026 tại Học viện Công nghệ Bưu chính Viễn thông để chấm đề án tốt nghiệp thạc sĩ cho:

Học viên: **Trần Nguyên Thịnh**

Tên đề án tốt nghiệp: Nghiên cứu và phát triển ứng dụng Chatbot giúp quản lý thời gian của nhân viên tại FPT SOFTWAVE

Chuyên ngành: Hệ thống thông tin Mã số: **8.48.01.04**

Các thành viên của Hội đồng chấm đề án tốt nghiệp có mặt: ....5.../ 05

TT	HỌ VÀ TÊN	TRÁCH NHIỆM TRONG HĐ	GHI CHÚ
1	**PGS. TSKH. Hoàng Đăng Hải**	Chủ tịch	
2	**TS. Đỗ Tiến Dũng**	Thư ký	
3	**TS. Đặng Hoàng Long**	Phản biện 1	
4	**TS. Bùi Hải Phong**	Phản biện 2	
5	**TS. Nguyễn Xuân Thắng**	Uỷ viên	

Các nội dung thực hiện:

1. Chủ tịch Hội đồng điều khiển buổi họp. Công bố quyết định của Giám đốc Học viện Công nghệ Bưu chính Viễn thông về việc thành lập Hội đồng chấm đề án tốt nghiệp thạc sĩ.
2. Người hướng dẫn khoa học hoặc thư ký đọc lý lịch khoa học và các điều kiện bảo vệ đề án tốt nghiệp của học viên. (có bản lý lịch khoa học và kết quả các môn học cao học của học viên kèm theo).
3. Học viên trình bày tóm tắt đề án tốt nghiệp.
4. Phản biện 1 đọc nhận xét (có văn bản kèm theo)
5. Phản biện 2 đọc nhận xét (có văn bản kèm theo)
6. Các câu hỏi của thành viên Hội đồng:

Cơ chế fine-tune PhoBERT như thế nào?
Hệ thống mpremise chưng vấn đề ứng NLP ngoài
thì giải quyết mâu thuẫn như thế nào?
Rasa và PhoBERT ưu, nhược điểm so với các phương
pháp khác

7. Trả lời của học viên:

Fine tune ... embedding, prefix
Các PPT thông tin gửi cần được mã hoá
Rasa ...

8. Thư ký đọc nhận xét về quá trình thực hiện đề án tốt nghiệp của học viên *(có văn bản kèm theo).*

9. Hội đồng họp riêng:

- Bầu Ban kiểm phiếu:
  1. Trưởng Ban kiểm phiếu: PGS.TS Hoàng Đăng Hải
  2. Uỷ viên Ban kiểm phiếu: TS Đặng Hoàng Long
  3. Uỷ viên Ban kiểm phiếu: TS Đỗ Tiến Dũng
- Hội đồng chấm đề án tốt nghiệp bằng bỏ phiếu kín.
- Ban kiểm phiếu làm việc:
- Trưởng Ban kiểm phiếu báo cáo kết quả kiểm phiếu *(có Biên bản họp Ban kiểm phiếu kèm theo)*
- **Điểm trung bình của đề án tốt nghiệp:** 8,2

Kết luận:

1. Các nội dung cần chỉnh sửa, hoàn thiện sau bảo vệ đề án tốt nghiệp:

Sửa trình bày
Trình 3.1, 3.2 phương tọa chưa
Trình bày thêm về khả năng mở rộng của hệ thống
Mô tả thêm UI/UX

2. Đề nghị Học viện công nhận (hoặc không) và cấp bằng (hoặc không) thạc sĩ cho học viên: Đồng ý

3. Đề án tốt nghiệp có thể phát triển thành đề tài nghiên cứu cho NCS.....

Buổi làm việc kết thúc vào 8h45 cùng ngày.

Chủ tịch

PGS.TS. Hoàng Đăng Hải

Thư ký

Đỗ Tiến Dũng

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập – Tự do – Hạnh phúc

BẢN NHẬN XÉT ĐỀ ÁN TỐT NGHIỆP THẠC SĨ

(Dùng cho người phản biện)

Tên đề tài đề án tốt nghiệp: Nghiên cứu và phát triển ứng dụng chatbot giúp quản lý thời gian của nhân viên tại FPT Software

Chuyên ngành: Hệ thống thông tin

Mã chuyên ngành: 8.48.01.04

Họ và tên học viên: Trần Nguyên Thịnh

Họ và tên người nhận xét: Bùi Hải Phong

Học hàm, học vị: Tiến sĩ

Chuyên ngành: Khoa học máy tính

Cơ quan công tác: Trường ĐH Kiến Trúc Hà Nội

Số điện thoại:.0915594033. E-mail:..phongbh@hau.edu.vn

NỘI DUNG NHẬN XÉT

I/ Cơ sở khoa học và thực tiễn, tính cấp thiết của đề tài:

Đề tài phát triển ứng dụng chatbot cho nhân viên tại công ty FPT Soytware. Đề tài có tính công nghệ và thực tiễn cao.

II/ Nội dung của đề án tốt nghiệp, các kết quả đã đạt được:

Đề tài gồm 54 trang/ 3 chương, xây dựng thành công hệ thống Chatbot hỗ trợ nhân viên công ty FPT Soytware. Đề tài áp dụng Rasa Framework và Phobert xây dựng Chatbot tiếng Việt.

Tuy vậy đề tài chưa có ví dụ minh họa về dữ liệu và các trường hợp cụ thể.

III/ Những vấn đề cần giải thích thêm:

1. Đề tài hãy so sánh phương pháp xây dựng Chatbot sử dụng Rasa Framework và các phương pháp khác?
2. Các tiêu chí đánh giá ứng dụng Chat bot thực hiện như thế nào?

IV/ Kết luận:

Đồng ý cho phép học viên bảo vệ đề án tốt nghiệp.

Đồng ý (hoặc không đồng ý) cho phép học viên bảo vệ đề án tốt nghiệp.

Ngày 8 tháng 7 năm 2026

NGƯỜI NHẬN XÉT

Bùi Hải Phong

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập – Tự do – Hạnh phúc

BẢN NHẬN XÉT ĐỀ ÁN TỐT NGHIỆP THẠC SĨ

(Dùng cho người phản biện)

Tên đề tài đề án tốt nghiệp: Nghiên cứu và phát triển ứng dụng chatbot giúp quản lý thời gian của nhân viên tại FPT Software

Chuyên ngành: Hệ thống thông tin

Mã chuyên ngành: 8.48.01.04

Họ và tên học viên: Trần Nguyên Thịnh

Họ và tên người nhận xét: Đặng Hoàng Long .....

Học hàm, học vị: Tiến sĩ .....

Chuyên ngành:Khoa học máy tính .....

Cơ quan công tác: Học viện Công nghệ Bưu chính Viễn thông .....

Số điện thoại: 0335229386 E-mail: longdh@ptit.edu.vn

NỘI DUNG NHẬN XÉT

I/ Cơ sở khoa học và thực tiễn, tính cấp thiết của đề tài:

Đề tài có cơ sở khoa học và thực tiễn rõ ràng khi dựa trên các công nghệ hiện đại về chatbot, xử lý ngôn ngữ tự nhiên và học máy, cụ thể là RASA Framework kết hợp mô hình ngôn ngữ PhoBERT cho tiếng Việt. Về thực tiễn, quản lý thời gian là một thách thức đối với các doanh nghiệp phần mềm; tại FPT Software với quy mô nhân sự lớn và môi trường làm việc đa dự án, chatbot nội bộ Maya hiện có vẫn còn thiếu các chức năng hỗ trợ quản lý thời gian. Vì vậy, việc nghiên cứu và phát triển một hệ thống chatbot tiếng Việt tích hợp liền mạch với hệ sinh thái doanh nghiệp (Microsoft Teams, Outlook Calendar, Jira) nhằm tự động hóa việc lập kế hoạch, nhắc nhở và báo cáo tiến độ là hướng đi có ý nghĩa ứng dụng và phù hợp với chương trình đào tạo thạc sĩ theo định hướng ứng dụng.

II/ Nội dung của đề án tốt nghiệp, các kết quả đã đạt được:

- Khảo sát cơ sở lý thuyết và các nghiên cứu liên quan về quản lý thời gian trong doanh nghiệp, công nghệ chatbot, AI/Học máy, Xử lý ngôn ngữ tự nhiên, RASA Framework và mô hình ngôn ngữ PhoBERT cho tiếng Việt.
- Khảo sát nhu cầu thực tế tại FPT Software bằng bảng hỏi và phỏng vấn sâu, phân tích các hạn chế của chatbot Maya hiện có và xác lập các yêu cầu chức năng, phi chức năng cho hệ thống.
- Thiết kế và xây dựng hệ thống chatbot quản lý thời gian trên nền RASA + PhoBERT theo kiến trúc microservices, tích hợp Microsoft Teams, Outlook Calendar và Jira; phát triển các mô-đun quản lý nhiệm vụ, lịch biểu, nhắc nhở thông minh và báo cáo tiến độ.

- Kiểm thử nhiều cấp độ (NLU, đầu–cuối, UAT) và thử nghiệm thực tế theo giai đoạn Alpha/Beta; đánh giá qua các chỉ số kỹ thuật (Intent Accuracy 92,3%; Entity F1 88,1%; thời gian phản hồi ~1,24 giây), chỉ số nghiệp vụ (giảm thời gian thao tác, tăng tỷ lệ hoàn thành đúng hạn) và mức độ hài lòng người dùng (CSAT 4,1/5).

III/ Những vấn đề cần giải thích thêm:

- Trong pipeline RASA, PhoBERT được tinh chỉnh (fine-tune) đầu–cuối hay chỉ dùng như bộ trích đặc trưng cố định qua LanguageModelFeaturizer? Mô tả về việc "fine-tuning PhoBERT kèm learning-rate warm-up và 5-fold cross-validation" cần được làm rõ so với cơ chế mặc định của RASA.
- Các cải thiện nghiệp vụ (tỷ lệ hoàn thành đúng hạn tăng 21%, mức hài lòng tăng) được rút ra từ đợt thử nghiệm ngắn (4–6 tuần) với cỡ mẫu nhỏ và chưa có nhóm đối chứng rõ ràng.
- Đề án nhấn mạnh triển khai on-premise và yêu cầu "dữ liệu không rời khỏi hạ tầng nội bộ" như một điều kiện bắt buộc, nhưng hệ thống lại tích hợp với Jira (Atlassian Cloud), Outlook và Teams (Microsoft 365) qua API, tức nội dung hội thoại, nhiệm vụ và lịch làm việc vẫn được gửi ra dịch vụ đám mây của bên thứ ba. Học viên giải quyết mâu thuẫn này như thế nào, và ranh giới dữ liệu nào thực sự được giữ trong nội bộ?
- Cơ chế nhắc nhở được mô tả là "thông minh, dựa trên ngữ cảnh và lịch sử hành vi", nhưng yêu cầu chức năng FC02 lại quy định các mốc nhắc cố định 24 giờ / 2 giờ / 30 phút trước hạn. Vậy phần "thông minh, thích nghi theo hành vi người dùng" đã thực sự được triển khai bằng thuật toán/mô hình cụ thể nào, hay hiện mới dừng ở mức luật cố định?

IV/ Kết luận:

Đề án đã đáp ứng yêu cầu đối với một đề án tốt nghiệp thạc sĩ theo định hướng ứng dụng. Đồng ý cho phép học viên bảo vệ đề án tốt nghiệp trước Hội đồng.

Ngày ..8.. tháng ~~7~~.. năm 2026

NGƯỜI NHẬN XÉT

Đặng Hoàng Lay