**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**NGUYỄN THỊ CẨM TIÊN**

**PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG ĐA NỀN TẢNG HỖ TRỢ TƯ VẤN VÀ ĐẶT LỊCH DỊCH VỤ LÀM ĐẸP TÍCH HỢP AI**

**NIÊN LUẬN**

**NGÀNH MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG DỮ LIỆU**

**MÃ SỐ 012345**

**NĂM 2024**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**NGUYỄN THỊ CẨM TIÊN**

**MÃ SỐ SINH VIÊN: B2004757**

**PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG ĐA NỀN TẢNG HỖ TRỢ TƯ VẤN VÀ ĐẶT LỊCH DỊCH VỤ LÀM ĐẸP TÍCH HỢP AI**

**NIÊN LUẬN**

**NGÀNH MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG DỮ LIỆU**

**MÃ SỐ 012345**

**NGƯỜI HƯỚNG DẪN**

**THS. TRANG THANH TRÍ**

**NĂM 2024**

**LỜI CẢM ƠN**

Để hoàn thành được đề tài niên luận này, em xin chân thành cảm ơn giáo viên hướng dẫn thầy Trang Thanh Trí đã tận tình chỉ dẫn và đưa ra nhiều lời khuyên bổ ích giúp em hoàn thành bài niên luận này một cách tốt nhất.

Em xin chân thành cảm ơn quý thầy cô khoa Mạng máy tính tạo cơ hội cho em được học tập, nghiên cứu và tích lũy kiến thức để hoàn thành đề tài niên luận này.

**MỤC LỤC**

[LỜI CẢM ƠN i](#_Toc17599)

[MỤC LỤC ii](#_Toc22035)

[TÓM TẮT iii](#_Toc4450)

[ABSTRACT iv](#_Toc30145)

[CHƯƠNG 1: GIỚI THIỆU 1](#_Toc6442)

[1.1 Đặt vấn đề 1](#_Toc17168)

[1.2 Mục tiêu đề tài 1](#_Toc14320)

[1.3 Đối tượng và phạm vi nghiên cứu 1](#_Toc13512)

[1.4 Bố cục của quyển báo cáo 1](#_Toc21065)

[CHƯƠNG 2: TỔNG QUAN TÀI LIỆU 3](#_Toc2644)

[2.1 Ngôn ngữ Dart 3](#_Toc2682)

[2.2 Flutter framework 3](#_Toc17252)

[2.4 Rasa framework 3](#_Toc5636)

[2.5 FastAPI framework 4](#_Toc27067)

[2.6 Hệ quản trị CSDL MySQL 4](#_Toc190)

[CHƯƠNG 3: PHƯƠNG PHÁP NGHIÊN CỨU 5](#_Toc26224)

[CHƯƠNG 4: KẾT QUẢ VÀ THẢO LUẬN 6](#_Toc20885)

[CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 7](#_Toc9766)

**TÓM TẮT**

Niên luận này tập trung vào việc nghiên cứu và phát triển ứng dụng di động tích hợp Trí tuệ Nhân tạo (AI) để cung cấp dịch vụ tư vấn chăm sóc sắc đẹp và chức năng cho phép đặt lịch hẹn. Mục tiêu của đề tài là cải thiện trải nghiệm người dùng trong lĩnh vực chăm sóc sắc đẹp thông qua sự tích hợp linh hoạt trí tuệ nhân tạo, giúp người dùng có thông tin chính xác và dịch vụ chăm sóc sắc đẹp phù hợp với nhu cầu cá nhân, đồng thời tối ưu hóa quy trình đặt lịch trực tuyến. Về phương pháp nghiên cứu, Trong nghiên cứu này, chúng tôi tiến hành thu thập dữ liệu đa dạng về các chủ đề có liên quan, nghiên cứu xây dựng mô hình trí tuệ nhân tạo và thiết kế quy trình đặt lịch trực tuyến linh hoạt và dễ sử dụng trên ứng dụng di động.

Sự khác biệt của ứng dụng so với những nghiên cứu trước đó đến từ việc tích hợp AI trả lời tự động để người dùng có thể có được câu trả lời cho những thắc mắc liên quan đến việc làm đẹp một cách nhanh chóng và chính xác, đem đến sự tiện lợi cho người dùng. Ứng dụng cũng hướng đến việc tích hợp các sản phẩm làm đẹp như một ứng dụng mua sắm trực tuyến và diễn đàn nơi chia sẻ những kinh nghiệm làm đẹp trong tương lai.

**ABSTRACT**

This dissertation focuses on researching and developing a mobile application that integrates Artificial Intelligence (AI) to provide beauty care advisory services and features appointment scheduling capabilities. The aim of the project is to enhance user experience in the beauty care domain through flexible integration of artificial intelligence, enabling users to access accurate information and beauty care services tailored to their individual needs, while optimizing the online appointment booking process. In terms of research methodology, we collect diverse data on relevant topics for this study, conduct research to build artificial intelligence models, and design a flexible and user-friendly online appointment booking process within the mobile application.

The application differentiates itself from previous research by incorporating AI-driven automatic responses, allowing users to quickly and accurately receive answers to beauty-related queries, providing convenience for users. Additionally, the application aims to integrate beauty products as an online shopping feature and includes a forum for sharing beauty experiences in the future.

**CHƯƠNG 1: GIỚI THIỆU**

1.1 Đặt vấn đề

Trong thời đại công nghệ ngày nay, việc phát triển ứng dụng di động đã trở thành một xu hướng không thể phủ nhận, mang lại lợi ích to lớn cho cả người sử dụng và doanh nghiệp. Sự tiện lợi, linh hoạt và khả năng tương tác cao của các ứng dụng di động đã làm thay đổi cách chúng ta tương tác với thế giới xung quanh. Thông qua việc sử dụng các nền tảng di động, chúng ta có khả năng kết nối mọi người và dịch vụ một cách nhanh chóng và hiệu quả, từ việc đặt lịch hẹn, mua sắm, đến giáo dục và giải trí.

Ứng dụng di động không chỉ là công cụ đơn giản mà còn là một kênh giao tiếp mạnh mẽ, giúp doanh nghiệp tăng cường tương tác và tạo ra một trải nghiệm người dùng không giới hạn. Việc tích hợp các công nghệ mới, như trí tuệ nhân tạo càng làm tăng cường khả năng làm giàu nội dung và tương tác, mở ra những khía cạnh mới trong sáng tạo và đổi mới.

Ứng dụng di động ngày càng đa dạng và phong phú với hàng nghìn ứng dụng rải rác trên nhiều lĩnh vực như ứng dụng mua bán trực tuyến, ứng dụng hỗ trở học tập, ứng dụng mạng xã hội,...Tuy nhiên những ứng dụng để tương tác với các sản phẩm dịch vụ vẫn chưa được quan tâm và phát triển mạnh mẽ. Do đó khách hàng vẫn gặp nhiều khó khăn trong quá trình sử dụng các dịch vụ tại cửa hàng. Khách hàng đôi khi phải chờ đợi quá lâu, nhân viên phải làm việc quá tải. Những vấn đề này đều gây ảnh hưởng đến chất lượng phục vụ và mức độ hài lòng của khác hàng. Nhằm mục đích góp phần cải thiện vấn đề trên, chúng tôi đã chọn đề tài “Phát triển ứng dụng di động đa nền tảng hỗ trợ tư vấn và đặt lịch dịch vụ làm đẹp tích hợp AI”.

1.2 Mục tiêu đề tài

Mục tiêu của đề tài này là xây dựng ứng dụng cho phép người dùng đặt lịch với cửa hàng làm đẹp và giải đáp thắc mắc liên quan đến chăm sóc khách hàng bằng AI. Những mục tiêu chính của đề tài:

- Nghiên cứu và xây dựng ứng dụng đa nền tảng

- Xây dựng chức năng đặt lịch hẹn

- Ứng dụng AI vào chăm sóc khách hàng

1.3 Đối tượng và phạm vi nghiên cứu

Trong đề tài này chúng tôi tập trung nghiên cứu xây dựng một ứng dụng đặt lịch hẹn dịch vụ làm đẹp có tích hợp AI chạy đa nền tảng, đặc biệt là trên các thiết bị di động.

Phạm vi nghiên cứu này sẽ tập trung vào việc phát triển một ứng dụng kết hợp giữa tính năng đặt lịch và khả năng tương tác của trí tuệ nhân tạo để tạo ra một trải nghiệm đặt lịch hẹn dịch vụ làm đẹp nhanh chóng và hiệu quả đem lại sử thoải mái cho người dùng.

1.4 Nội dung nghiên cứu

- Lập kế hoạch thự hiện đề tài bao gồm việc lên ý tưởng, thiết kế giao diện cho ứng dụng, tìm kiếm thống tin liên quan.

- Tìm hiểu về các framework có liên quan như Firebase, FastAPI, Flutter,..

- Nghiên cứu và xây dựng CSDL cho ứng dụng

-Tiến hành xây dựng và kiểm thử các chức năng

- Đánh giá kết quả và định hướng phát triển cho ứng dụng.

1.5 Bố cục của quyển báo cáo

Nội dung của quyển báo cáo được tổ chức thành 5 chương như sau:

- Chương 1 (Giới thiệu): chương này giới thiệu tổng quan về đề tài, mục tiêu, đối tượng và phạm vi nghiên cứu.

- Chương 2 (Tổng quan tài liệu): trình bày các công nghệ trong xây dựng ứng dụng di động, khái niệm liên quan đến trí tuệ nhân tạo, công nghệ xây dựng chat bot đơn giản.

- Chương 3 (Xây dựng ứng dụng): mô tả chi tiết phương pháp, kỹ thuật xử dụng để xây dựng ứng dụng.

- Chương 4 (Kết quả và thảo luận): trình bày kết quả thu được, đánh giá và so sánh.

- Chương 5 (Kết luận và hướng phát triển): kết luận chung cho đề tài, những hạn chế chưa giải quyết và hướng phát triển trong tương lai.

**CHƯƠNG 2: TỔNG QUAN TÀI LIỆU**

2.1 Ngôn ngữ Dart

Trong lĩnh vực lập trình, Dart là đứa con tâm huyết từ Lars Bak và Kasper Lund, ra đời dưới sự quản lý của Google. Bắt đầu từ lần ra mắt tại hội nghị GOTO tại Đan Mạch vào ngày 10 tháng 12 tháng 10 năm 2011. Một cột mốc quan trọng được ghi dấu bằng việc phát hành phiên bản Dart 1.0 vào ngày 14 tháng 11 năm 2013.



Với mục tiêu chính là hỗ trợ việc phát triển ứng dụng di động, web và server. Dart là một ngôn ngữ lập trình đa nền tảng, có thể sử dụng để phát triển ứng dụng trên nhiều nền tảng khác nhau, bao gồm iOS, Android, web và server. Dưới đây là mộ số đặc điểm nổi bậc của ngôn ngữ Dart:

- Hiệu suất cao: Dart sử dụng Dart VM để chạy mã nguồn trực tiếp, giúp tối ưu hóa hiệu suất. Ngoài ra, khả năng biên dịch Ahead-of-Time (AOT) giúp giảm độ trễ khi ứng dụng khởi chạy.

- Giao diện người dùng đẹp mắt: được sử dụng chủ yếu trong framework Flutter, Dart giúp xây dựng giao diện người dùng đẹp mắt và linh hoạt. Flutter sử dụng ngôn ngữ lập trình Dart để tạo ra các ứng dụng với trải nghiệm người dùng mượt mà và đồng nhất trên nhiều nền tảng.

- Kiểu dữ liệu chặt chẽ (Strong Typing): Dart hỗ trợ kiểu dữ liệu chặt chẽ, giúp phát hiện và tránh được nhiều lỗi phổ biến khi phát triển. Điều này giúp tăng tính ổn định và tin cậy của mã nguồn.

- Hỗ trợ lập trình bất đồng bộ: Dart hỗ trợ lập trình bất đồng bộ thông qua Future và Stream, giúp xử lý các tác vụ I/O mà không gây chặn chương trình.

2.2 Flutter framework

Flutter là một framework giao diện người dùng mã nguồn mở miễn phí được tạo bởi [Google](https://en.wikipedia.org/wiki/Flutter_(software)?ref=200lab.io" \t "https://200lab.io/blog/flutter-la-gi/_blank)và được phát hành vào tháng 5 năm 2017. Flutter dành cho việc xây dựng ứng dụng di động, web, và desktop nó nổi tiếng với sự nhanh chóng, hiệu quả cũng như giao diện đẹp mắt trong việc phát triển ứng dụng di động. Flutter cho phép phát triển ứng dụng đa nền tảng, bao gồm iOS, Android, web và desktop. Với mã nguồn duy nhất có thể được sử dụng để triển khai trên nhiều hệ điều hành khác nhau.



Fluter bao gồm 2 thành phần quan trọng:

- SDK: một tập hợp các công cụ sẽ giúp bạn phát triển các ứng dụng của mình. Nó bao gồm các công cụ để biên dịch mã nguồn của bạn thành các mã máy native.

- Framework UI Library based on widgets: tập hợp các phần tử giao diện người dùng có thể tái sử dụng (Button, Text Inputs, Slider, ...) giúp bạn có thể cá nhân hóa ứng dụng bạn tạo theo nhu cầu cá nhân của mình.

Những ưu điểm nổi bật của Flutter:

- Cung cấp những giao diện người dùng đẹp mắt, hoạt động phong phú, scroll mượt mà và khả năng tự nhận thức nền tảng của các widget built – in

- Flutter có khả năng thể hiện cùng một UI trên nhiều nền tảng.

- Chức nănghot reload cho phép bạn thấy những thay đổi trong mã nguồn của bạn ngay lập tức trên màn hình ứng dụng mà không cần phải rebuild hoặc restart ứng dụng, giúp tăng tốc quá trình phát triển.

- Flutter sử dụng ngôn ngữ lập trình Dart và sử dụng công nghệ rendering riêng gọi là "Skia" để vẽ giao diện người dùng, giúp ứng dụng có hiệu suất cao và mượt mà.

- Flutter có một cộng đồng lớn và năng động, điều này có nghĩa là có nhiều tài nguyên, thư viện và công cụ hỗ trợ để bạn có thể phát triển ứng dụng của mình một cách nhanh chóng và dễ dàng.

2.4 Rasa framework

2.4.1 Tổng quan về Rasa



Rasa là một framework mã nguồn mở giúp các nhà phát triển xây dựng nên Chatbot hội thoại. Rasa còn có những thành phần cơ bản cho việc xây dựng nên Chatbot hội thoại như Xử lý ngôn ngữ tự nhiên (Natural Language Understanding) và Quản lý hội thoại (Dialogue management). Trong đó:

- Xử lý ngôn ngữ tự nhiên (NLU): Đây là một module đóng vai trò xử lý đầu vào của cuộc hội thoại. NLU sẽ xác định ý định của người dùng muốn nói sau đó trích xuất và chọn ra các thông tin chính.

- Quản lý hội thoại: Sau khi hiểu được câu nói của người dùng, module Dialogue management sẽ quyết định hành động tiếp theo trong hội thoại dựa trên các thông tin chính đã được trích xuất thông các chính sách mà nhà phát triển thêm vào.

2.4.2 Quy trình hoạt động

- Bắt đầu bằng việc thu thập dữ liệu và xây dựng mô hình NLU sử dụng Rasa NLU, sau đó huấn luyện mô hình với dữ liệu để chatbot có thể hiểu các yêu cầu từ người dùng.

- Sử dụng Rasa Core để định nghĩa các luồng trò chuyện và quyết định hành động tiếp theo của chatbot trong mỗi tình huống.

- Kết hợp mô hình NLU và Core vào một chatbot hoàn chỉnh và kiểm tra tính tương tác và hiệu suất của chatbot.Triển khai chatbot trên nền tảng mong muốn và kiểm tra tính năng và hiệu suất của nó trong môi trường thực tế.

Quy trình này có thểt được lập đi lập lại nhiều lần để cải thiện và tối ưu hiệu suất của chatbot.

2.5 FastAPI framework

FastAPI là một framework còn khá mới cho ngôn ngữ Python mà tiền thân của nó là OpenAPI được phát triển bởi Starlette và Pydantic. Chất lượng của framework này được thể hiện qua câu slogan “FastAPI framework, high performance, easy to learn, fast to code, ready for production”.



FastAPI có các đặc điểm nổi bật để cạnh tranh với các framework khác như:  
- Hiệu năng rất cao dù so sánh với NodeJS hay Go.

- Code dễ hơn và nhanh hơn, tốc độ code các features tăng khoảng 200 đến 300 %.

- Giảm số bugs của developper đến 40%.

- Hỗ trợ code dễ hơn với tự động gợi ý, debug cũng cần ít thời gian hơn.

- Giảm thiểu hóa việc lặp code. Các tham số truyền vào có nhiều tính năng.

Nhược điểm:

- Fast API là một framework còn khá mới và đang phát triển nên cộng đồng hỗ trợ và tài liệu còn hạn chế.

2.6 Hệ quản trị CSDL MySQL

MySQL là một hệ thống quản lý cơ sở dữ liệu quan hệ mã nguồn mở dựa trên ngôn ngữ truy vấn có cấu trúc ( SQL) được phát triển, phân phối và hỗ trợ bởi tập đoàn Oracle. MySQL được yêu thích và sử dụng rộng rãi trên toàn thế giới do có thể chạy trên hầu hết tất cả các nền tảng, bao gồm cả Linux , UNIX và Windows. MySQL thường được kết hợp với các ứng dụng web.



MySQL sử dụng ngôn ngữ SQL để truy cập, truy vấn và thao tác dữ liệu trong cơ sở dữ liệu. Dữ liệu được tổ chức theo dạng bảng và quan hệ giữa các bảng được quản lý thông qua các ràng buộc khóa ngoại. Cung cấp các tính năng để thêm, sửa đổi, xóa và truy vấn dữ liệu từ cơ sở dữ liệu. Hỗ trợ các tính năng bảo mật như quản lý quyền truy cập và mã hóa dữ liệu để bảo vệ dữ liệu khỏi sự truy cập trái phép.

2.7 Mô hình MVC

2.7.1 Lịch sử hình thành

MVC được tiến sĩ Trygve Reenskaug đưa vào [ngôn ngữ lập trình](https://vietnix.vn/ngon-ngu-lap-trinh/) Smalltalk-76 khi ông đến trung tâm Nghiên cứu Xerox Palo Alto (PARC) vào giữa năm 1970. Sau đó, việc triển khai trở nên phổ biến trong các phiên bản khác của Small- Talk. Năm 1988, các bài báo “The Journal of Object Technology” – JOT mang lại bước tranh toàn cảnh về MVC mang đến sự hiệu quả tốt nhất.

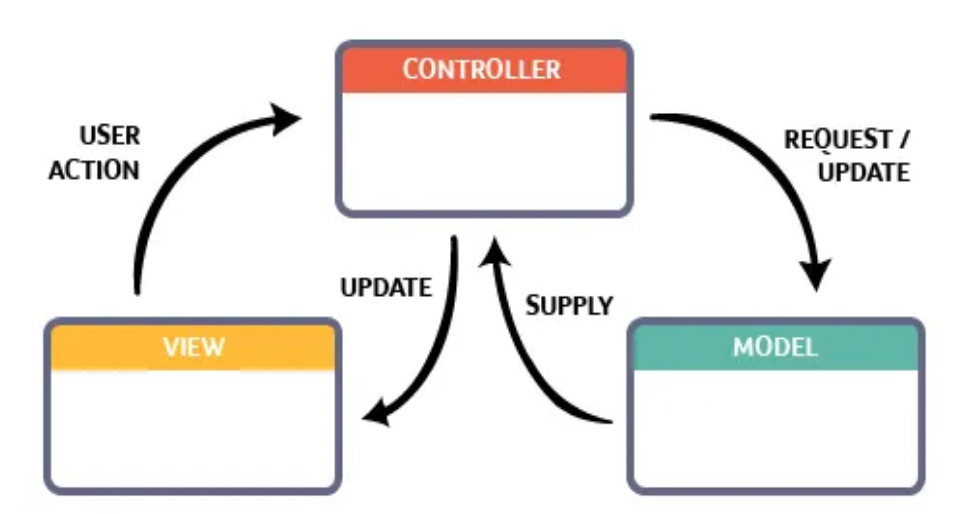
2.7.2 Tổng quan về mô hình MVC

MVC là viết tắt của cụm từ “Model-View-Controller”. Đây là mô hình thiết kế được sử dụng trong kỹ thuật phần mềm. MVC chia thành ba phần được kết nối với nhau và mỗi thành phần đều có một nhiệm vụ riêng của nó và độc lập với các thành phần khác. 3 thành phần bao gồm:

- Model (dữ liệu): Quản lí xử lí các dữ liệu.

- View (giao diện): Nơi hiển thị dữ liệu người dùng

- Controller (bộ điều khiển): điều khiển sự tương tác giữa 2 thành phần model và view



2.7.3 Các thành phần trong mô hình MVC

- View: Đây là phần giao diện (theme) dành cho người sử dụng. View là phương tiện hiển thị các đối tượng trong một ứng dụng. Chẳng hạn như hiển thị một cửa sổ, nút hay văn bản trong một cửa sổ khác. Nó bao gồm bất cứ thứ gì mà người dùng có thể nhìn thấy được.

- Model: Là bộ phận có chức năng lưu trữ toàn bộ dữ liệu của ứng dụng và là cầu nối giữa 2 thành phần bên dưới là View và Controller. Một model là dữ liệu được sử dụng bởi chương trình. Đây có thể là cơ sở dữ liệu, hoặc file XML bình thường hay một đối tượng đơn giản. Chẳng hạn như biểu tượng hay là một nhân vật trong game.

- Controller: Là bộ phận có nhiệm vụ xử lý các yêu cầu người dùng đưa đến thông qua View. Một controller bao gồm cả Model lẫn View. Nó nhận input và thực hiện các update tương ứng.

2.7.4 Luồng xử lý trong MVC

Yêu cầu được gửi từ trình duyệt web hoặc ứng dụng đến máy chủ. Router nhận yêu cầu từ client và xác định controller và action tương ứng với yêu cầu. Controller được gọi dựa trên thông tin được xác định bởi router. Controller tiếp nhận yêu cầu và thực hiện các logic kinh doanh liên quan, trong quá trình này controller có thể tương tác với model để truy vấn và cập nhật dữ liệu. Sau khi thực hiện xong quá trình xử lý Controller chuyển dữ liệu hoặc kết quả cho view. View được gọi và sử dụng dữ liệu được cung cấp để tạo ra giao diện người dùng .Khi view hoàn thành việc tạo ra giao diện, nó được trả về cho client dưới dạng HTML hoặc dữ liệu được trả về tùy thuộc vào loại ứng dụng.

2.7.5 Ưu điểm của MVC

- Tổ chức rõ ràng: Mô hình MVC phân chia ứng dụng thành ba phần chính: Model, View và Controller, giúp cho mã nguồn được tổ chức một cách logic và dễ hiểu. Điều này giúp cho việc phát triển và bảo trì mã nguồn dễ dàng hơn.

- Tính tái sử dụng cao: Với việc phân chia ứng dụng thành các phần nhỏ và có logic riêng biệt, việc tái sử dụng mã nguồn trở nên dễ dàng hơn. Controller và Model có thể được sử dụng lại trong nhiều phần của ứng dụng, giúp giảm thiểu việc viết mã lặp lại.

- Phát triển song song: Với mô hình MVC, các nhóm phát triển có thể làm việc độc lập với nhau trên các phần khác nhau của ứng dụng mà không cần phải lo lắng về sự xung đột. Controller, Model và View đều là các thành phần độc lập, cho phép phát triển song song một cách hiệu quả.

- Dễ bảo trì: Do mã nguồn được tổ chức rõ ràng và phân chia thành các phần nhỏ, việc bảo trì và debug trở nên dễ dàng hơn. Khi có sự thay đổi hoặc cập nhật, chỉ cần tập trung vào một phần cụ thể mà không ảnh hưởng đến các phần khác.

2.7.6 Nhược điểm của MVC

- Khả năng phức tạp: Mặc dù MVC phân chia ứng dụng thành các thành phần riêng biệt, nhưng đôi khi có thể dẫn đến sự phức tạp khi cần phải quản lý quá nhiều controllers, models và views, đặc biệt trong các ứng dụng lớn.

- Quản lý trạng thái: Trong một số trường hợp, việc quản lý trạng thái ứng dụng có thể trở nên khó khăn trong mô hình MVC. Đặc biệt là khi có sự phức tạp hoặc nhiều thông tin trạng thái cần phải được duy trì và chia sẻ giữa các thành phần khác nhau.

- Phức tạp khi điều hướng: Trong các ứng dụng lớn, việc quản lý và điều hướng từ Router có thể trở nên phức tạp và khó bảo trì, đặc biệt là khi cần xử lý các tình huống đặc biệt hoặc quản lý nhiều định tuyến khác nhau.

**CHƯƠNG 3: PHƯƠNG PHÁP NGHIÊN CỨU**

**CHƯƠNG 4: KẾT QUẢ VÀ THẢO LUẬN**

**CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**