TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG ĐA NỀN TẢNG**

**ỨNG DỤNG CHAT GPT BẰNG REACT NATIVE**

*Người hướng dẫn*: **GV TRẦN ĐẠI NHÂN**

*Người thực hiện*: **NGUYỄN NGỌC HẬU – 52000659**

**VŨ TRƯỜNG GIANG - 52000656**

Lớp **: 20050261, 20050281**

Khoá  **: 24**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG ĐA NỀN TẢNG**

**ỨNG DỤNG CHAT GPT BẰNG REACT NATIVE**

*Người hướng dẫn*: **GV TRẦN ĐẠI NHÂN**

*Người thực hiện*: **NGUYỄN NGỌC HẬU – 52000659**

**VŨ TRƯỜNG GIANG - 52000656**

Lớp **: 20050261, 20050281**

Khoá  **: 24**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

LỜI CẢM ƠN

Em xin chân thành cảm ơn thầy Trần Đại Nhân đã dành thời gian xem xét và đánh giá báo cáo của em về chủ đề "Khảo sát chi tiết về React Native và ứng dụng". Em đặc biệt biết ơn sự hướng dẫn tận tình của thầy trong suốt quá trình thực hiện báo cáo. Nhờ sự hướng dẫn và hỗ trợ của thầy, em đã có thể hoàn thành báo cáo một cách tốt nhất.

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của GV Trần Đại Nhân. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 6 tháng 5 năm 2024*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Nguyễn Ngọc Hậu*

*Vũ Trường Giang*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Thiết kế kế và triển khai ứng dụng chat gpt bằng react native.

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc166935827)

[TÓM TẮT iv](#_Toc166935828)

[MỤC LỤC 5](#_Toc166935829)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 8](#_Toc166935830)

[CHƯƠNG 1 – GIỚI THIỆU ĐỀ TÀI 9](#_Toc166935831)

[1.1 Lịch sử phát triển: 9](#_Toc166935832)

[1.2 Cốt lõi react native: 9](#_Toc166935833)

[1.3 Hỗ trợ: 10](#_Toc166935834)

[1.4 Ứng dụng: 10](#_Toc166935835)

[1.5 Lợi ích và hạn chế của react native: 10](#_Toc166935836)

[1.6 So sánh react native và kivy: 11](#_Toc166935837)

[1.7 Kết luận: 13](#_Toc166935838)

[CHƯƠNG 2 – ỨNG DỤNG 14](#_Toc166935839)

[2.1 Giới thiệu app ChatBot: 14](#_Toc166935840)

[2.2 Đặc tả yêu cầu: 14](#_Toc166935841)

[2.3 Các use-case hệ thống: 15](#_Toc166935842)

[2.4 Actor người dùng: 16](#_Toc166935843)

[2.5 Đặc tả use-case: 17](#_Toc166935844)

[2.5.1 Use-case 1: 17](#_Toc166935845)

[2.5.2 Use-case 2: 18](#_Toc166935846)

[2.5.3 Use-case 3: 19](#_Toc166935847)

[2.5.4 Use-case 4: 20](#_Toc166935848)

[2.5.5 Use-case 5: 21](#_Toc166935849)

[2.5.6 Use-case 6: 22](#_Toc166935850)

[2.6 Công nghệ React được sử dụng: 23](#_Toc166935851)

[2.6.1 Welcome.js: 23](#_Toc166935852)

[2.6.2 Features.js: 24](#_Toc166935853)

[2.6.3 HomeScreen.js: 25](#_Toc166935854)

[2.6.4 index.js: 29](#_Toc166935855)

[2.6.5 api.js: 30](#_Toc166935856)

[2.7 Hình ảnh demo ứng dụng chatBot: 32](#_Toc166935857)

[2.7.1 Welcome.js: 32](#_Toc166935858)

[2.7.2 Features.js: 33](#_Toc166935859)

[2.7.3 HomeScreen.js: 34](#_Toc166935860)

[2.8 Kết luận và hiệu quả khi sử dụng ứng dụng: 35](#_Toc166935861)

[TÀI LIỆU THAM KHẢO 36](#_Toc166935862)

[PHỤ LỤC 37](#_Toc166935863)

**DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT**

**CÁC CHỮ VIẾT TẮT**

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

Hình 1: React-native

Hình 2: Actor người dùng

Hình 3: Code Welcome.js

Hình 4: Code Features.js

Hình 5: Code HomeScreen.js

Hình 6: Code index.js

Hình 7: Code api.js

**DANH MỤC BẢNG**

Bảng 1: Lợi ích và hạn chế của react native

Bảng 2: Bảng so sánh react native và kivy

Bảng 3: Các yêu cầu của hệ thống

Bảng 4: Các use-case của hệ thống

Bảng 5: Use-case Get started

Bảng 6: Use-case chọn botChat

Bảng 7: Use-case nhập văn bản

Bảng 8: Use-case gửi văn bản

Bảng 9: Use-case trả lời câu hỏi

Bảng 10: Use-case trở về trang chủ

# CHƯƠNG 1 – GIỚI THIỆU ĐỀ TÀI

## Lịch sử phát triển:

- React Native được Facebook phát triển và giới thiệu lần đầu tiên vào năm 2015. Nó nhanh chóng trở thành một framework phổ biến cho phát triển ứng dụng di động đa nền tảng, cho phép lập trình viên tạo ứng dụng iOS và Android bằng JavaScript.

- React Native là một framework mã nguồn mở để xây dựng các ứng dụng di động sử dụng JavaScript và React. Nó cho phép các nhà phát triển sử dụng cùng một cơ sở mã để xây dựng ứng dụng cho cả nền tảng iOS và Android, trong khi vẫn mang lại hiệu suất và trải nghiệm giống như bản gốc. Nó tận dụng sức mạnh của thư viện React của Facebook, giúp việc xây dựng giao diện người dùng năng động và tương tác dễ dàng hơn.



*Hình 1: React-native*

## Cốt lõi react native:

- **Sử dụng JavaScript:** React Native sử dụng JavaScript để viết giao diện người dùng, thay vì các ngôn ngữ native như Swift (iOS) hoặc Java (Android). Điều này giúp giảm thời gian phát triển và cho phép lập trình viên JavaScript dễ dàng chuyển sang phát triển ứng dụng di động.

- **Cấu trúc mô-đun:** React Native sử dụng một kiến trúc mô-đun, giúp chia nhỏ ứng dụng thành các phần nhỏ hơn, dễ quản lý hơn. Điều này giúp cải thiện khả năng bảo trì và khả năng tái sử dụng mã.

- **Cấu trúc React:** React Native dựa trên React, một thư viện JavaScript phổ biến cho việc xây dựng các giao diện người dùng. Điều này mang lại lợi ích của hiệu suất cao, khả năng tái sử dụng mã và khả năng bảo trì tốt.

## Hỗ trợ:

- Hệ điều hành: iOS và Android

- Ngôn ngữ lập trình: JavaScript, JSX

- Khung: React

- Công cụ: React Native CLI, Expo, NativeScript

- Thư viện: React Native Navigation, React Native Elements, React Native Gestures

## Ứng dụng:

- React Native là một lựa chọn cho việc phát triển ứng dụng di động đa nền tảng, cho phéptạo một ứng dụng duy nhất có thể chạy trên cả iOS và Android.

- React Native cung cấp một cách mạnh mẽ và linh hoạt để xây dựng các giao diện người dùng di động.

- React Native cho phép bạn chia sẻ mã giữa các nền tảng iOS và Android, giúp tiết kiệm thời gian và công sức phát triển.

- React Native có thể tạo ra ứng dụng có hiệu suất cao.

## Lợi ích và hạn chế của react native:

|  |  |
| --- | --- |
| **Lợi ích** | **Hạn chế** |
| **Phát triển nhanh chóng:** React Native cho phép bạn phát triển ứng dụng di động nhanh chóng hơn so với các framework native. | **Hiệu suất:** React Native có thể có hiệu suất thấp hơn so với các ứng dụng native được viết bằng Swift hoặc Java. |
| **Mã chia sẻ:** Bạn có thể chia sẻ mã giữa các nền tảng iOS và Android, giúp tiết kiệm thời gian và công sức phát triển. | **Tùy chỉnh:** React Native có thể ít tùy chỉnh hơn so với các framework native. |
| **Cộng đồng lớn:** React Native có một cộng đồng lớn và tích cực, cung cấp hỗ trợ và tài nguyên dồi dào. | **Vấn đề về bảo mật:** Bảo mật vẫn chưa được tối ưu với React Native. |
| **Dễ học:** React Native tương đối dễ học đối với các lập trình viên JavaScript. |  |
| **Tiết kiệm chi phí:** Việc sử dụng một codebase cho cả iOS và Android giúp giảm chi phí phát triển và bảo trì ứng dụng. |  |

*Bảng 1: Lợi ích và hạn chế của react native*

## So sánh react native và kivy:

- React Native và kivy là hai framework nổi tiếng để phát triển ứng dụng di động đa nền tảng. Cả 2 đều có ưu và nhược điểm riêng

**- Điểm giống nhau**:

+ **Phát triển đa nền tảng:** Cả hai framework đều cho phép bạn xây dựng một ứng dụng duy nhất có thể chạy trên cả iOS và Android, tiết kiệm thời gian và chi phí phát triển.

+ **Mã nguồn mở:** Cả React Native và Kivy đều là dự án mã nguồn mở, miễn phí và được cộng đồng hỗ trợ tích cực.

+ **Hiệu suất cao:** Cả hai framework đều có thể tạo ra các ứng dụng di động có hiệu suất cao và đáp ứng nhanh.

+ **Dễ học:** Cả React Native và Kivy đều sử dụng các ngôn ngữ lập trình phổ biến, dễ học và có nhiều tài liệu hướng dẫn.

**- Điểm khác nhau:**

|  |  |  |
| --- | --- | --- |
|  | **React Native** | **Kivy** |
| **Ngôn ngữ lập trình** | Javascript | Python |
| **Cấu trúc** | Module | File python |
| **UI** | Native | Widget tùy chỉnh |
| **Hiệu suất** | Cao | Vừa |
| **Khả năng tùy chỉnh(Chỉnh sửa)** | Hạn chế | Cao |
| **Cộng đồng** | Lớn và hoạt động mạnh mẽ | Nhỏ |
| **Tài liệu** | Phong phú | Ít và khó hiểu |
| **Lựu chọn tốt nhất** | Phát triển ứng dụng di động nhanh chóng, dễ dàng với giao diện người dùng trực quan | Phát triển ứng dụng di động phức tạp, đòi hỏi hiệu suất cao và khả năng tùy chỉnh cao |

*Bảng 2: Bảng so sánh react native và kivy*

## Kết luận:

- React Native là một framework phát triển ứng dụng di động đa nền tảng mạnh mẽ, cho phép viết mã một lần và chạy trên cả iOS và Android, giúp tiết kiệm thời gian và chi phí phát triển. Với hiệu suất đủ tốt cho hầu hết các ứng dụng thông thường, React Native cung cấp tính năng Hot Reloading và có một cộng đồng hỗ trợ mạnh mẽ, cùng với nhiều thư viện và công cụ phát triển phong phú. Tuy nhiên, nó vẫn kém hơn về hiệu suất so với ứng dụng native thuần túy và đôi khi gặp khó khăn trong việc gỡ lỗi và testing. Mặc dù vậy, React Native vẫn là một lựa chọn hàng đầu cho việc phát triển ứng dụng di động nhờ tính linh hoạt và khả năng tái sử dụng mã nguồn cao.

# CHƯƠNG 2 – ỨNG DỤNG

## Giới thiệu app ChatBot:

- App ChatBot là một chương trình máy tính mô phỏng cuộc trò chuyện với con người bằng cách sử dụng trí tuệ nhân tạo (AI) và xử lý ngôn ngữ tự nhiên (NLP).

- Lý do chọn App chatBot làm đề tài là vì react native có thể dễ dàng sử dụng api của GPT-3.5 và hiệu quả rất tốt khi tương tác với người dùng.

## Đặc tả yêu cầu:

|  |  |
| --- | --- |
| **ĐẶC TẢ YÊU CẦU** | |
| Loại yêu cầu | Chi tiết |
| Yêu cầu chức năng | - Chọn chatBot  - Nói chuyện với chatBot  - Nhập văn bản để cho chatBot hiểu và thực thi. |
| Yêu cầu phi chức năng | - Có khả năng duy trì hệ thống.  - Hiệu suất hoạt động tốt.  - Khả năng sử dụng. |
| Yêu cầu lưu trữ | - Có thể sao lưu và phục hồi dữ liệu. |

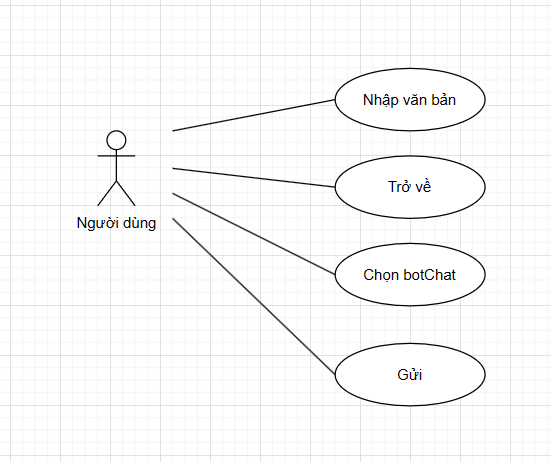
*Bảng 3: Các yêu cầu của hệ thống*

## Các use-case hệ thống:

|  |  |  |
| --- | --- | --- |
| **ID** | **Tên Use Case** | **Mô tả** |
| **UC1** | Get Started | - Khi người dùng ấn vào sẽ chuyển đến trang lựa chọn botchat |
| **UC2** | Chọn botChat | - Người dùng sẽ chọn botChat mà mình muốn dùng |
| **UC3** | Nhập văn bản | - Người dùng sẽ nhập văn bản cho botchat trả lời |
| **UC4** | Gửi văn bản | - Botchat nhận văn bản và xử lý câu hỏi của người dùng |
| **UC5** | Trả lời câu hỏi | - botChat trả lời câu hỏi của người dùng |
| **UC6** | Trở về trang chính | - Người dùng thoát ra trang chính |

*Bảng 4: Các use-case của hệ thống*

## Actor người dùng:



*Hình 2: Actor người dùng*

## Đặc tả use-case:

### 2.5.1 Use-case 1:

|  |  |  |
| --- | --- | --- |
| **Mã use case** | UC1 | |
| **Use Case** | Get Started | |
| **Ngữ cảnh** | Khi vào trang đầu tiên của hệ thống botChat | |
| **Mô tả** | Người dùng vào trang chọn botChat | |
| **Tác nhân** | Người dùng | |
| **Sự kiện kích hoạt** | Khi người dùng ấn vào button “Get Started” | |
| **Điều kiện tiên quyết** | Người dùng phải vào được trang đầu tiên của hệ thống | |
| **Kết quả** | Vào trang chọn botChat | |
| **Luồng sự kiện** | **Actor** | **System** |
| Người dùng ấn button “Get Started” | Hệ thống chuyển hướng đến trang chọn botChat |
| **Ngoại lệ** | Người dùng không khởi chạy được hệ thống. | |

*Bảng 5: Use-case Get started*

### 2.5.2 Use-case 2:

|  |  |  |
| --- | --- | --- |
| **Mã use case** | UC2 | |
| **Use Case** | Get Started | |
| **Ngữ cảnh** | Khi màn hình chọn botChat hiện ra | |
| **Mô tả** | Người dùng chọn botChat để sử dụng | |
| **Tác nhân** | Người dùng | |
| **Sự kiện kích hoạt** | Khi người dùng ấn vào ô chọn botChat | |
| **Điều kiện tiên quyết** | Người dùng phải ấn button “Get Started” | |
| **Kết quả** | Vào trang chat với botChat mà người dùng đã chọn | |
| **Luồng sự kiện** | **Actor** | **System** |
| Người dùng ấn vào ô chọn botChat mà mình muốn sử dụng. | Hệ thống chuyển hướng đến trang trò chuyện với botChat mà người dùng đã chọn |
| **Ngoại lệ** | Người dùng không ấn nút “Get Started”  Người dùng không khởi chạy được hệ thống. | |

*Bảng 6: Use-case chọn botChat*

### 2.5.3 Use-case 3:

|  |  |  |
| --- | --- | --- |
| **Mã use case** | UC3 | |
| **Use Case** | Nhập văn bản | |
| **Ngữ cảnh** | Khi màn hình trờ chuyện với botChat hiện ra | |
| **Mô tả** | Người dùng nhập văn bản | |
| **Tác nhân** | Người dùng | |
| **Sự kiện kích hoạt** | Khi người dùng ấn khung chat để nhập văn bản | |
| **Điều kiện tiên quyết** | Người dùng phải chọn xong botChat | |
| **Kết quả** | Nhập được văn abrn | |
| **Luồng sự kiện** | **Actor** | **System** |
| Người dùng ấn vào ô nhập text | Hệ thống sẽ hiển thị những gì mà người dùng đã nhập |
| **Ngoại lệ** | Người dùng không chọn botChat.  Người dùng không khởi chạy được hệ thống. | |

*Bảng 7: Use-case nhập văn bản*

### 2.5.4 Use-case 4:

|  |  |  |
| --- | --- | --- |
| **Mã use case** | UC4 | |
| **Use Case** | Gửi văn bản | |
| **Ngữ cảnh** | Khi màn hình trờ chuyện với botChat hiện ra | |
| **Mô tả** | Người dùng gửi văn bản đã nhập cho botChat | |
| **Tác nhân** | Người dùng | |
| **Sự kiện kích hoạt** | Khi người dùng ấn button ”Gửi” | |
| **Điều kiện tiên quyết** | Người dùng phải chọn xong botChat | |
| **Kết quả** | Gửi được yêu cầu cho botChat | |
| **Luồng sự kiện** | **Actor** | **System** |
| Người dùng ấn vào button “Gửi” | Hệ thống sẽ hiển thị những gì mà người dùng đã nhập và xử lý yêu cầu của người dùng |
| **Ngoại lệ** | Người dùng không chọn botChat.  Người dùng không khởi chạy được hệ thống. | |

*Bảng 8: Use-case gửi văn bản*

### 2.5.5 Use-case 5:

|  |  |  |
| --- | --- | --- |
| **Mã use case** | UC5 | |
| **Use Case** | Trả lời câu hỏi | |
| **Ngữ cảnh** | Khi hệ thống đã nhận được yêu cầu của người dùng | |
| **Mô tả** | Hệ thống xử lý yêu cầu người dùng | |
| **Tác nhân** | Hệ thống | |
| **Sự kiện kích hoạt** | Khi người dùng ấn button ”Gửi” | |
| **Điều kiện tiên quyết** | Người dùng phải chọn xong botChat | |
| **Kết quả** | Hệ thống gửi lại câu trả lời câu hỏi cho người dùng | |
| **Luồng sự kiện** | **Actor** | **System** |
| Người dùng ấn vào button “Gửi” | Hệ thống sẽ hiển thị những gì mà người dùng đã nhập, xử lý yêu cầu của người dùng và gửi lại câu trả lời cho người dùng. |
| **Ngoại lệ** | Hệ thống không nhận được yêu cầu người dùng  Người dùng không chọn botChat.  Người dùng không khởi chạy được hệ thống. | |

*Bảng 9: Use-case trả lời câu hỏi*

### 2.5.6 Use-case 6:

|  |  |  |
| --- | --- | --- |
| **Mã use case** | UC6 | |
| **Use Case** | Trở về trang chính | |
| **Ngữ cảnh** | Khi hệ thống đã nhận được yêu cầu của người dùng | |
| **Mô tả** | Trở về trang chính lúc đầu | |
| **Tác nhân** | Người dùng, Hệ thống | |
| **Sự kiện kích hoạt** | Khi người dùng ấn button ”Trở về” | |
| **Điều kiện tiên quyết** | Người dùng đang ở trong trang trò chuyện với botChat | |
| **Kết quả** | Trở về trang chính | |
| **Luồng sự kiện** | **Actor** | **System** |
| Người dùng ấn vào button “Trở về” | Hệ thống sẽ chuyển hướng người dùng về tranh chính |
| **Ngoại lệ** | Hệ thống không nhận được yêu cầu người dùng.  Người dùng không chọn botChat.  Người dùng không khởi chạy được hệ thống. | |

*Bảng 10: Use-case Trở về trang chính.*

## Công nghệ React được sử dụng:

- App chatBot được khởi tạo bởi expo và react native.

### 2.6.1 Welcome.js:

import { View, Text, SafeAreaView, Image } from 'react-native'

import React from 'react'

import { TouchableOpacity } from 'react-native-gesture-handler'

import { GestureHandlerRootView } from 'react-native-gesture-handler';

import {widthPercentageToDP as wp, heightPercentageToDP as hp} from 'react-native-responsive-screen';

import { useNavigation } from 'expo-router';

export default function WelcomeScreen() {

    const navigation = useNavigation();

    return (

        <GestureHandlerRootView>

        <SafeAreaView className = "flex-1 flex justify-around bg-white">

            <View className="space-y-2">

                <Text style={{fontSize: wp(10)}} className = "text-center font-blod text-gray-700">

                    CHATGPT CLONE

                </Text>

                <Text style={{fontSize: wp(5)}} className="text-center font-semibold text-gray-800 tracking-wider">

                    Design by Giang,Hậu

                </Text>

            </View>

            <View className = "flex-row justify-center">

                <Image source={require('../../assets/image/icon.png')} style={{width: wp(75), height: wp(75)}}/>

            </View>

            <TouchableOpacity onPress={()=> navigation.navigate('Home')} className="bg-emerald-600 mx-5 p-4 rounded-2xl">

                <Text style={{fontSize: wp(6)}} className = "text-center font-bold text-white text-2xl">Get Started</Text>

            </TouchableOpacity>

        </SafeAreaView>

        </GestureHandlerRootView>

    )

}

### 2.6.2 Features.js:

import { View, Text , Image} from 'react-native'

import React from 'react'

import {widthPercentageToDP as wp, heightPercentageToDP as hp} from 'react-native-responsive-screen';

export default function Features() {

    return (

        <View style={{height: hp(60)}} className = "space-y-4">

            <Text style={{fontSize: wp(6.5)}} className="font-semibold text-gray-700">Features</Text>

            {/\* ChatGPT \*/}

            <View className = "bg-emerald-200 p-4 rounded-xl space-y-1">

                <View>

                    <Image source={require('../../assets/image/chatgpticon.png')}style={{height: hp(4), width: hp(4)}}/>

                    <Text style={{fontSize: wp(4.8)}} className = "font-semibold text-gray-700">Chat GPT</Text>

                </View>

                <Text style={{fontSize: wp(3.8)}} className = "font-medium text-gray-700">

                    ChatGPT là một trợ lý ảo dựa trên trí tuệ nhân tạo của OpenAI, có khả năng hiểu và tạo ra văn bản tự nhiên.

                </Text>

            </View>

        </View>

    )

}

### 2.6.3 HomeScreen.js:

import React, { useState } from 'react';

import { Image, SafeAreaView, ScrollView, Text, TextInput, TouchableOpacity, View } from 'react-native';

import { heightPercentageToDP as hp, widthPercentageToDP as wp } from 'react-native-responsive-screen';

import { getChatGPTResponse } from '../api'; // Import hàm gọi API ChatGPT

import Features from '../components/Features';

import { dummyMessages } from '../constants/App';

export default function HomeScreen() {

    const [messages, setMessages] = useState(dummyMessages);

    const [result, setResult] = useState('');

    const [loading, setLoading] = useState(false);

    const clear = () => {

        setMessages([]);

    }

    const handleSend = async () => {

        if (result.trim()) {

            const userMessage = {

                role: 'user',

                content: result

            };

            setMessages([...messages, userMessage]);

            setResult('');

            setLoading(true);

            try {

                const response = await getChatGPTResponse([...messages, userMessage]);

                const assistantReply = {

                    role: 'assistant',

                    content: response.choices[0].message.content

                };

                setMessages(prevMessages => [...prevMessages, assistantReply]);

            } catch (error) {

                console.error('Failed to get response from ChatGPT:', error.response ? error.response.data : error.message);

                if (error.response && error.response.data && error.response.data.error) {

                    const errorMessage = error.response.data.error.message;

                    if (error.response.data.error.type === "insufficient\_quota") {

                        alert('You have exceeded your quota. Please check your OpenAI account and billing details.');

                    } else {

                        alert(`Failed to get response from ChatGPT: ${errorMessage}`);

                    }

                } else {

                    alert('Failed to get response from ChatGPT. Please try again later.');

                }

            } finally {

                setLoading(false);

            }

        }

    }

    return (

        <View className="flex-1 bg-white">

            <SafeAreaView className="flex-1 flex mx-5">

                <View className="flex-row justify-center">

                    <Image source={require('../../assets/image/icon.png')} style={{ height: hp(15), width: hp(15) }} />

                </View>

                {

                    messages && messages.length > 0 ? (

                        <View className="space-y-2 flex-1">

                            <Text style={{ fontSize: wp(5) }} className="text-gray-700 font-semibold ml-1">

                                Assistant

                            </Text>

                            <View style={{ height: hp(60) }} className="bg-neutral-200 rounded-3xl p-4">

                                <ScrollView bounces={false} className="space-y-4" showsVerticalScrollIndicator={false}>

                                    {

                                        messages.map((message, index) => {

                                            if (message.role === 'assistant') {

                                                if (message.content.includes('https')) {

                                                    return (

                                                        <View key={index} className="flex-row justify-start">

                                                            <View className="p-2 flex rounded-2xl bg-emerald-200 rounded-tl-none">

                                                                <Image source={{ uri: message.content }} className="rounded-2xl" resizeMode='contain' style={{ height: wp(60), width: wp(60) }} />

                                                            </View>

                                                        </View>

                                                    )

                                                } else {

                                                    return (

                                                        <View key={index} style={{ width: wp(70) }} className="bg-emerald-200 rounded-xl p-2 rounded-tl-none">

                                                            <Text>{message.content}</Text>

                                                        </View>

                                                    )

                                                }

                                            } else {

                                                return (

                                                    <View key={index} className="flex-row justify-end">

                                                        <View style={{ width: wp(70) }} className="bg-white rounded-xl p-2 rounded-tr-none">

                                                            <Text>{message.content}</Text>

                                                        </View>

                                                    </View>

                                                )

                                            }

                                        })

                                    }

                                </ScrollView>

                            </View>

                        </View>

                    ) : (

                        <Features />

                    )

                }

                <View className="flex justify-center items-center my-4 px-4">

                    <View className="w-full flex-row items-center bg-white rounded-full shadow-md p-2">

                        <TextInput

                            style={{

                                height: 40,

                                borderColor: 'transparent',

                                borderWidth: 1,

                                borderRadius: 20,

                                paddingHorizontal: 10,

                                flex: 1

                            }}

                            placeholder="Nhập văn bản"

                            onChangeText={text => setResult(text)}

                            value={result}

                            editable={!loading}

                        />

                        <TouchableOpacity onPress={handleSend} className="bg-blue-500 rounded-full p-2 ml-2" disabled={loading}>

                            <Text className="text-white font-semibold">{loading ? 'Đang gửi...' : 'Gửi'}</Text>

                        </TouchableOpacity>

                    </View>

                    {

                        messages.length > 0 && (

                            <TouchableOpacity onPress={clear} className="bg-neutral-300 rounded-full p-2 mt-2">

                                <Text className="text-black font-semibold">Trở về</Text>

                            </TouchableOpacity>

                        )

                    }

                </View>

            </SafeAreaView>

        </View>

    );

}

### 2.6.4 index.js:

import React, { useEffect } from 'react';

import AppNavigation from './navigation/App';

// import { apiCall } from '../app/api/openAI';

export default function App() {

  return (

    <AppNavigation />

  );

}

### 2.6.5 api.js:

import axios from 'axios';

const API\_KEY = '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';

const API\_URL = 'https://api.openai.com/v1/chat/completions';

const sleep = (ms) => new Promise(resolve => setTimeout(resolve, ms));

const retryRequest = async (fn, retries = 3, delay = 1000) => {

    for (let i = 0; i < retries; i++) {

        try {

            return await fn();

        } catch (error) {

            if (error.response && error.response.status === 429 && i < retries - 1) {

                console.warn(`Rate limit exceeded. Retrying in ${delay}ms...`);

                await sleep(delay);

                delay \*= 2;

            } else {

                throw error;

            }

        }

    }

};

export const getChatGPTResponse = async (messages) => {

    return retryRequest(async () => {

        const response = await axios.post(

            API\_URL,

            {

                model: 'gpt-3.5-turbo',

                messages: messages,

            },

            {

                headers: {

                    'Authorization': `Bearer ${API\_KEY}`,

                    'Content-Type': 'application/json'

                }

            }

        );

        return response.data;

    });

};

- Các công nghệ react native được sử dụng gồm:

+ **React Native Core Components:** View, Text, Image.

+ **React-native-responsive-screen:** widthPercentageToDP as wp, heightPercentageToDP as hp: Đây là các hàm giúp tính toán kích thước màn hình theo phần trăm, giúp tạo ra giao diện người dùng phản hồi tốt hơn trên nhiều kích thước màn hình khác nhau.

+ **Tailwind CSS Classes:** Các lớp className như space-y-4, font-semibold, text-gray-700, bg-emerald-200, p-4, rounded-xl, space-y-1: Đây là các lớp tiện ích của Tailwind CSS được sử dụng thông qua một thư viện như tailwind-rn hoặc twrnc để áp dụng các kiểu CSS nhanh chóng và tiện lợi trong React Native.

+ **Thư viện third-party:** react-native-responsive-screen: cung cấp các hàm widthPercentageToDP (wp) và heightPercentageToDP (hp).

**+ Styling với Tailwind CSS:** Sử dụng các lớp tiện ích của Tailwind CSS thông qua một thư viện như tailwind-rn hoặc twrnc để áp dụng styling.

+ **expo-router:** useNavigation: Được sử dụng để điều hướng giữa các màn hình trong ứng dụng Expo.

+ **Constants:** dummyMessages

## Hình ảnh demo ứng dụng chatBot:

### 2.7.1 Welcome.js:



*Hình 3: Welcome.js*

### 2.7.2 Features.js:



*Hình 4: Features.js*

- Demo:

+ Bước 1: Chọn botChat(ChatGPT).

+ Bước 2: Nhập văn bản(ví dụ: Chào bạn), sau đó hệ thống sẽ chuyển sang màn hình HomeScreen.js và trò chuyện với botChat

### 2.7.3 HomeScreen.js:



*Hình 5: HomeScreen.js*

## Kết luận và hiệu quả khi sử dụng ứng dụng:

- Giao diện thân thiện.

- Ứng dụng có phần chuyển cảnh mượt.

- botChat có api từ chatGPT có thể trả lời câu hỏi mà chúng ta đặt ra.

- Tương lai sẽ phát triển thêm voice-to-text cho ứng dụng dễ dàng sử dụng hơn

# TÀI LIỆU THAM KHẢO

**Tiếng Việt**

* 1. Lê Đức Sơn (2020), “[React Native là gì? Tất tần tật những điều cần biết (topdev.vn)](https://topdev.vn/blog/react-native-la-gi/?utm_source=google&utm_medium=cpc&utm_campaign=topdev&utm_content=performance&gad_source=1&gclid=CjwKCAjw9cCyBhBzEiwAJTUWNb30gJzKEWAsKoRW0S5E7oqIH3bQD3gPVOU4VWVs8UcXd0JmmnSxPxoCYX8QAvD_BwE)”.
  2. Bùi Quy Hoạt (2023), “[Kivy: Một framework cho Python cho phép bạn phát triển ứng dụng một cách nhanh chóng | Desde Linux](https://blog.desdelinux.net/vi/khung-kivy-cho-python/)”
  3. Trần Trung Sơn (2021), “[React Native hay Android/ iOS Native để lập trình App? (topdev.vn)](https://topdev.vn/blog/react-native/)”

**Tiếng Anh**

* 1. Russell Ballestrini(2023), “[Mobile App Development Frameworks Comparison in 2023 (thecyberiatech.com)](https://www.thecyberiatech.com/blog/mobile-app/mobile-frameworks-comparison/)”
  2. Chat GPT (2024), “compare React native and kivy make app” From open ai
  3. John.HC (2018), “[React Native - Wikipedia](https://en.wikipedia.org/wiki/React_Native#:~:text=React%20Native%20is%20an%20open,along%20with%20native%20platform%20capabilities.)”

# PHỤ LỤC