ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

CHUYÊN NGÀNH: HỆ THỐNG THÔNG TIN

ĐỀ TÀI:

XÂY DỰNG ỨNG DỤNG DI ĐỘNG "PERSONAL STYLIST" ĐỀ XUẤT TRANG PHỤC CÁ NHÂN

Người hướng dẫn: **TS. VÕ ĐỨC HOÀNG**

Sinh viên thực hiện: **LÊ THỊ LÂM NHƯ**

Số thẻ sinh viên: **102210223**

Lớp: **21TCLC_DT3**

Đà Nẵng, 06/2025

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

CHUYÊN NGÀNH: HỆ THỐNG THÔNG TIN

ĐỀ TÀI:

XÂY DỰNG ỨNG DỤNG DI ĐỘNG "PERSONAL STYLIST" ĐỀ XUẤT TRANG PHỤC CÁ NHÂN

Người hướng dẫn: **TS. VÕ ĐỨC HOÀNG**
Sinh viên thực hiện: **LÊ THỊ LÂM NHƯ**
Số thẻ sinh viên: **102210223**
Lớp**: 21TCLC_DT3**

Đà Nẵng, 06/2025

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

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

NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP

1. Thông tin chung:

1. Họ, tên sinh viên: Lê Thị Lâm Như Lớp: 21TCLC_DT3 Số thẻ SV: 102210223
2. Tên đề tài: Xây dựng ứng dụng di động Personal Stylist đề xuất trang phục cá nhân.
3. Người hướng dẫn: TS. Võ Đức Hoàng Học hàm/ học vị: Tiến Sĩ

II. Nhận xét, đánh giá đồ án tốt nghiệp:

1. Về tính cấp thiết, tính mới, khả năng ứng dụng của đề tài: (điểm tối đa là 2đ)

   ……..

   ……..

2. Về kết quả giải quyết các nội dung nhiệm vụ yêu cầu của đồ án: (điểm tối đa là 4đ)

   ……..

   ……..

3. Về hình thức, cấu trúc, bố cục của đồ án tốt nghiệp: (điểm tối đa là 2đ)

   ……..

   ……..

4. Đề tài có giá trị khoa học/ có bài báo/ giải quyết vấn đề đặt ra của doanh nghiệp hoặc nhà trường: (điểm tối đa là 1đ)

   ……..

   ……..

5. Các tồn tại, thiếu sót cần bổ sung, chỉnh sửa:

   ……..

   ……..

III. Tinh thần, thái độ làm việc của sinh viên: (điểm tối đa 1đ)

……..

IV. Đánh giá:

1. Điểm đánh giá: ……../10 (lấy đến 1 số lẻ thập phân)
2. Đề nghị: ☐ Được bảo vệ đồ án ☐ Bổ sung để bảo vệ ☐ Không được bảo vệ

Đà Nẵng, ngày tháng năm 2025

Trưởng Bộ môn …….. **Người hướng dẫn**

TÓM TẮT

Tên đề tài: Xây dựng ứng dụng di động "Personal Stylist" đề xuất trang phục cá nhân.

Sinh viên thực hiện:	Số thẻ SV:	Lớp:
Lê Thị Lâm Như	102210223	21TCLC_DT3
Trần Văn Sơn	102210229	21TCLC_DT3

Xây dựng ứng dụng di động Personal Stylist là một đề tài mang đến khả năng hỗ trợ người dùng trong việc quản lý tủ đồ và lựa chọn trang phục một cách thông minh, cá nhân hóa theo nhu cầu. Ứng dụng Personal Stylist giúp người dùng dễ dàng lưu trữ, phân loại quần áo, đồng thời đưa ra các gợi ý phối đồ phù hợp với thời tiết, sự kiện và phong cách cá nhân.

Đề tài áp dụng các kỹ thuật trí tuệ nhân tạo trong nhận diện ảnh trang phục, phân tích ngữ cảnh và thiết kế chatbot đề xuất trang phục, kết hợp cùng cơ sở dữ liệu thời gian thực để đảm bảo trải nghiệm mượt mà, tiện lợi và phù hợp với từng người dùng. Hệ thống hướng đến sự cá nhân hóa tối đa, đồng thời giữ tính đơn giản và dễ sử dụng.

Đề xuất kỹ thuật được áp dụng cho dự án bao gồm:

- Ngôn ngữ lập trình: Kotlin (Android), SwiftUI (iOS)
- Không gian/Công cụ phát triển chính: Android Studio, Xcode, Cursor IDE
- Công cụ: Firebase, NodeJS, MongoDB
- AI: Nhận diện ảnh trang phục, xử lý ngôn ngữ tự nhiên cho chatbot
- Framework/Thư viện: Android UI, SwiftUI, ExpressJS, Firebase Storage

Với mục tiêu đặt ra là thiết kế và triển khai một ứng dụng di động hỗ trợ quản lý tủ đồ và tư vấn trang phục thông minh, đề tài tập trung giải quyết bài toán nhận diện trang phục từ ảnh và đề xuất phối đồ theo ngữ cảnh sử dụng.

Tổng kết lại, ứng dụng Personal Stylist mang lại nhiều lợi ích thiết thực như tiết kiệm thời gian, nâng cao trải nghiệm cá nhân hóa trong thời trang, hỗ trợ định hình phong cách, và mở ra tiềm năng phát triển trong lĩnh vực ứng dụng AI vào đời sống hằng ngày.

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

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

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Họ tên sinh viên: Lê Thị Lâm Như Lớp: 21TCLC_DT3 Mã số sinh viên: 102210223

Khoa: Công nghệ thông tin Ngành: Hệ thống thông tin

1. *Tên đề tài đồ án:*

   Xây dựng ứng dụng di động "Personal Stylist" đề xuất trang phục cá nhân

2. *Đề tài thuộc diện:* ☐ *Có ký kết thỏa thuận sở hữu trí tuệ đối với kết quả thực hiện*
3. *Các số liệu và dữ liệu ban đầu:* Không có
4. *Nội dung các phần thuyết minh và tính toán:*
   - Mở đầu: Giới thiệu về nhu cầu thực tế và lý do thực hiện đề tài, đồng thời giới thiệu mục đích, mục tiêu hướng đến, tính năng và đối tượng.
   - Chương 1: Cơ sở lý thuyết: Giới thiệu tổng quan về các công nghệ và mô hình được sử dụng trong dự án.
   - Chương 2: Phân tích và thiết kế hệ thống: Trình bày thiết hệ thống, hướng tiếp cận vấn đề.
   - Chương 3: Triển khai và đánh giá: Trình bày cách cài đặt, vận hành. Trình bày những kết quả và đánh giá thu được
   - Kết luận và hướng phát triển: Trình bày các kết quả đạt được, chỉ ra những hạn chế còn tồn tại và đề xuất hướng phát triển.
5. *Các bản vẽ, đồ thị:* Không có
6. *Họ tên người hướng dẫn:* TS. Võ Đức Hoàng
7. *Ngày giao nhiệm vụ đồ án: ……/……/2025*
8. *Ngày hoàn thành đồ án: ……/……/2025*

Đà Nẵng, ngày tháng năm 2025

Trưởng Bộ môn …….. **Người hướng dẫn**

LỜI NÓI ĐẦU

Trong lời đầu tiên của báo cáo đồ án tốt nghiệp "Xây dựng ứng dụng di động Personal Stylist đề xuất trang phục cá nhân", em xin được bày tỏ lòng biết ơn chân thành đến tất cả những người đã luôn đồng hành, hỗ trợ và tạo điều kiện thuận lợi cho em trong suốt quá trình học tập và thực hiện đồ án.

Trước hết, em xin gửi lời cảm ơn sâu sắc đến thầy **TS. Võ Đức Hoàng**, giảng viên đã tận tâm hướng dẫn em trong suốt quá trình thực hiện đề tài. Nhờ sự hướng dẫn, hỗ trợ và những góp ý quý báu, chi tiết từ Thầy trong từng buổi báo cáo tiến độ, em đã có thể hoàn thiện đề tài đồ án này. Trong suốt quá trình thực hiện đề tài, em đã có cơ hội tiếp cận thực tế với quy trình phát triển phần mềm, rèn luyện kỹ năng tư duy hệ thống, tổ chức dữ liệu và thiết kế giao diện người dùng, đồng thời nâng cao khả năng vận dụng kiến thức về trí tuệ nhân tạo trong lĩnh vực ứng dụng di động.

Em cũng xin chân thành cảm ơn **Ban Giám hiệu nhà trường**, cùng toàn thể quý thầy cô trong **Khoa Công nghệ Thông tin – Trường Đại học Bách khoa – Đại học Đà Nẵng,** những người đã truyền đạt kiến thức và kinh nghiệm quý báu trong suốt quá trình học tập ngành công nghệ thông tin. Chính nhờ những nền tảng kiến thức vững chắc đó, em đã có đủ hành trang để nghiên cứu, học tập và thực hiện đề tài một cách hiệu quả.

Bên cạnh đó, em muốn gửi lời cảm ơn đến **gia đình, bạn bè và người thân**, những người đã luôn ủng hộ, động viên và tiếp thêm tinh thần cho em trong những thời điểm khó khăn.

Do thời gian thực hiện có hạn và bản thân còn nhiều thiếu sót về kinh nghiệm thực tiễn, đồ án chắc chắn không thể tránh khỏi những hạn chế nhất định. Em rất mong nhận được phản hồi tích cực và góp ý chân thành từ phía thầy cô cũng như hội đồng chấm thi.

Em xin chân thành cảm ơn!

CAM ĐOAN

Tôi xin cam đoan:

Báo cáo đồ án tốt nghiệp với tên đề tài “Xây dựng ứng dụng di động "Personal Stylist" đề xuất trang phục cá nhân” là công trình nghiên cứu của chính cá nhân tôi và bạn Trần Văn Sơn dưới sự hướng dẫn trực tiếp của giảng viên TS. Võ Đức Hoàng

1. Tôi đã tự đọc nghiên cứu, dịch tài liệu và tổng hợp các kiến thức đã làm nên báo cáo này và đảm bảo không sao chép ở bất cứ đâu.
2. Những lý thuyết trong luận văn đều được sử dụng tài liệu như tôi đã tham khảo ở phần tài liệu tham khảo đã có trong báo cáo.

Nếu có vi phạm, tôi xin chịu hoàn toàn trách nhiệm.

Sinh viên thực hiện

Lê Thị Lâm Như

MỤC LỤC

DANH SÁCH HÌNH VẼ

DANH SÁCH BẢNG

DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT

Chữ viết tắt	Tên đầy đủ
CNTT	Công Nghệ Thông Tin
API	Application Programming Interface
XML	eXtensible Markup Language
UI	User Interface
IDE	Integrated Development Environment
NLP	Natural Language Processing
JSON	JavaScript Object Notation
RESTful	Representational State Transfer
NoSQL	Not Only SQL
ERD	Entity Relationship Diagram
OOTD	Outfit of the Day
PNG	Portable Network Graphics
JPEG	Joint Photographic Experts Group
HEIC	High Efficiency Image Coding
RGBA	Red Green Blue Alpha
ViT	Vision Transformer
mAP	mean Average Precision
SDK	Software Development Kit

MỞ ĐẦU

1. Giới thiệu đề tài

Xuất phát từ nhu cầu thực tế hiện nay, nhiều người gặp khó khăn trong việc lựa chọn trang phục phù hợp mỗi ngày do quỹ thời gian hạn chế hoặc không biết cách phối đồ hiệu quả. Việc quản lý tủ đồ và định hình phong cách cá nhân là một thách thức phổ biến, đặc biệt trong bối cảnh nhịp sống hiện đại luôn bận rộn. Bên cạnh đó, thời tiết và hoàn cảnh sử dụng trang phục cũng đòi hỏi người dùng phải thay đổi lựa chọn trang phục linh hoạt, hợp lý.

Trong khi đó, sự phát triển mạnh mẽ của công nghệ thông tin, đặc biệt là trí tuệ nhân tạo (AI), nhận diện hình ảnh và xử lý ngôn ngữ tự nhiên, đã mở ra cơ hội lớn để cá nhân hóa trải nghiệm thời trang. Đề tài "Xây dựng ứng dụng di động Personal Stylist đề xuất trang phục cá nhân" được thực hiện nhằm xây dựng một hệ thống thông minh, hỗ trợ người dùng quản lý tủ đồ và nhận gợi ý phối đồ phù hợp theo thời tiết, sự kiện hoặc sở thích. Ứng dụng còn tích hợp chatbot tư vấn giúp người dùng tương tác nhanh chóng, linh hoạt và mang tính cá nhân cao.

Với mong muốn nâng cao trải nghiệm thời trang cá nhân, tiết kiệm thời gian và hỗ trợ người dùng khẳng định phong cách riêng, đề tài không chỉ đáp ứng nhu cầu thiết thực mà còn có tiềm năng phát triển mở rộng trong tương lai.

2. Mục đích thực hiện đề tài

- Thiết kế và xây dựng một ứng dụng di động hỗ trợ người dùng quản lý tủ đồ cá nhân, đồng thời gợi ý phối đồ thông minh dựa trên thời tiết, sở thích và hoàn cảnh sử dụng.
- Ứng dụng các kiến thức đã học về trí tuệ nhân tạo, lập trình di động và thiết kế hệ thống để giải quyết một bài toán thực tế gần gũi với người dùng trẻ hiện nay.

3. Mục tiêu hướng đến

- Phát triển một ứng dụng có khả năng nhận diện loại trang phục từ hình ảnh.
- Tích hợp chatbot tư vấn và đề xuất trang phục sử dụng xử lý ngôn ngữ tự nhiên.
- Gợi ý trang phục cá nhân hóa theo thời tiết, lịch trình hoặc sự kiện.
- Tạo trải nghiệm thời trang đơn giản, tiện lợi và cá nhân hóa cho người dùng.

4. Đối tượng khách hàng

- Người yêu thích thời trang, muốn tối ưu hóa việc chọn đồ mỗi ngày.
- Người bận rộn, có ít thời gian nhưng vẫn quan tâm đến việc mặc đẹp.
- Người yêu thích ứng dụng công nghệ trong đời sống.

5. Phạm vi và đối tượng

- Phạm vi: ứng dụng được triển khai thử nghiệm trên nền tảng Android và iOS, với chức năng cơ bản như quản lý tủ đồ, gợi ý phối đồ, và tư vấn bằng chatbot.
- Đối tượng sử dụng: người dùng cá nhân sử dụng điện thoại thông minh, có nhu cầu sử dụng trợ lý phối đồ hằng ngày.

6. Công nghệ phát triển

- Frontend Android: Kotlin
- Frontend iOS: SwiftUI
- Backend: NodeJS (ExpressJS)
- Cơ sở dữ liệu: MongoDB, Firebase Cloud Storage
- AI: Gemini AI, OpenAI API
- Quản lý và lưu trữ dự án: Github

7. Cấu trúc đồ án tốt nghiệp

- Mở đầu
- Chương 1: Cơ sở lý thuyết
- Chương 2: Phân tích và thiết kế hệ thống
- Chương 3: Triển khai và đánh giá
- Kết luận và hướng phát triển

8. Phân chia nội dung công việc

Bảng 1 Bảng phân chia công việc

Công việc	**Người thực hiện**
Viết đề cương, mô tả mục tiêu và tính cấp thiết của đề tài	Lê Thị Lâm Như
Thiết kế database	Cả nhóm
Thiết kế giao diện trên figma	Lê Thị Lâm Như
Phân tích nghiệp vụ, vẽ sơ đồ Use Case, sơ đồ tuần tự	Cả nhóm
Triển khai giao diện iOS	Trần Văn Sơn
Triển khai giao diện Android	Lê Thị Lâm Như

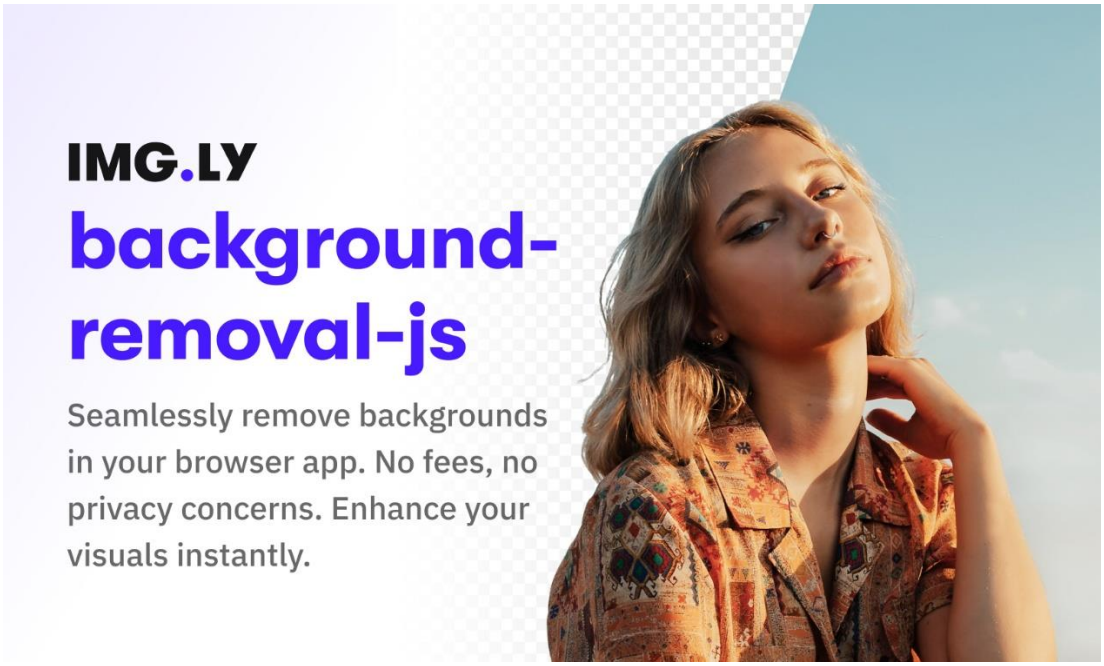
Viết API cho toàn bộ hệ thống	Cả nhóm
Tích hợp Firebase Storage, xử lý upload ảnh và lưu trữ	Trần Văn Sơn
Tích hợp OpenAI Chatbot và mô hình gợi ý phối đồ	Cả nhóm
Xây dựng chức năng xử lý ảnh bằng Gemini AI và tách nền bằng IMG.LY	Cả nhóm
Thiết kế và triển khai hệ thống backend với NodeJS + MongoDB	Trần Văn Sơn
Kiểm thử chức năng ứng dụng trên cả Android và iOS	Cả nhóm
Viết tài liệu báo cáo và chuẩn bị thuyết trình	Lê Thị Lâm Như

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1. Tổng quan về trí tuệ nhân tạo trong nhận diện ảnh

1.1.1. Tách nền

- Một trong những kỹ thuật quan trọng trong quá trình nhận diện trang phục từ hình ảnh người dùng là tách nền (background removal) – thao tác loại bỏ phần nền phía sau để tập trung xử lý đối tượng chính là người và quần áo. Việc tách nền giúp cải thiện đáng kể độ chính xác trong việc phân loại trang phục, trích xuất đặc trưng hình ảnh và hỗ trợ quá trình phối đồ cá nhân hóa.
- Trong đồ án Personal Stylist, nhóm đã tích hợp thư viện mã nguồn mở @imgly/background-removal– một giải pháp WebAssembly chạy hoàn toàn cục bộ (local), không cần gửi ảnh lên máy chủ. Thư viện này là wrapper của mô hình MediaPipe Selfie Segmentation V2 kết hợp kỹ thuật Image Matting, triển khai dưới định dạng ONNX, tương thích với cả môi trường trình duyệt (browser) và NodeJS.



Hình 1 Background Removal IMG.LY

- Quy trình xử lý ảnh: toàn bộ pipeline xử lý ảnh nền được tối ưu hóa như sau:
  - Nhập ảnh: hỗ trợ các định dạng JPEG/HEIC và tự động chuyển đổi về không gian màu sRGB.

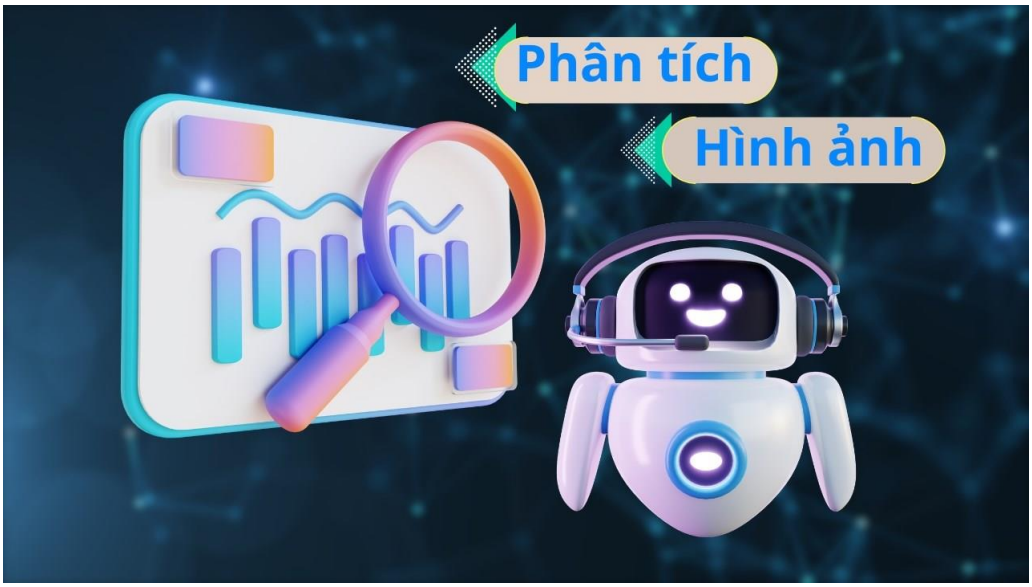
- Phân đoạn sơ bộ: sử dụng hàm removeBackgroundFromImageFile() để tạo mask phân biệt foreground (người/trang phục) và background.
- Làm mịn và lọc nhiễu: áp dụng kỹ thuật morphology (opening, closing) để làm sạch nhiễu và mượt biên.
- Kết xuất kết quả: kết hợp mask đã xử lý với foreground để tạo ảnh đầu ra RGBA (trong suốt) → xuất ảnh định dạng PNG 1:1.
- Tối ưu hóa: nén ảnh WebP (chất lượng 85%) và tạo ảnh thumbnail 300×300 px dùng để hiển thị lưới tủ đồ trong ứng dụng.

- Ưu điểm của giải pháp
  - Bảo mật và hiệu năng cao: không sử dụng API bên ngoài → không phát sinh chi phí vận hành, giảm độ trễ và bảo vệ quyền riêng tư người dùng.
  - Chất lượng tách nền cạnh tranh: độ chính xác tiệm cận các nền tảng thương mại như Remove.bg hoặc ClipDrop, nhưng với độ trễ thấp hơn vì xử lý trực tiếp trên thiết bị người dùng.
  - Dữ liệu đầu vào tối ưu cho nhận diện: ảnh PNG trong suốt là input lý tưởng cho các mô hình phân loại trang phục như DeepLab-v3+ hoặc Vision Transformer (ViT). Thử nghiệm cho thấy việc sử dụng ảnh đã tách nền giúp tăng chỉ số mAP (mean Average Precision).

1.1.2. Phân tích ảnh bằng Gemini AI để sinh metadata cho Item

- Sau bước tách nền, hệ thống Personal Stylist tiếp tục sử dụng trí tuệ nhân tạo để phân tích hình ảnh trang phục và sinh ra các siêu dữ liệu (metadata) cho từng item. Đây là bước quan trọng nhằm chuẩn hóa dữ liệu đầu vào và hỗ trợ hệ thống gợi ý phối đồ chính xác, phù hợp với từng người dùng.
- Quy trình xử lý: Khi người dùng tải ảnh trang phục lên từ ứng dụng di động, hệ thống backend sẽ tiếp nhận ảnh thông qua API POST /item-uploads/process-image. Quá trình xử lý được thực hiện theo các bước sau:
  - Tối ưu hóa và tách nền ảnh: Hình ảnh gốc được xử lý bằng thư viện @imgly/background-removal, loại bỏ phần nền và giữ lại đối tượng chính (trang phục). Điều này giúp cải thiện chất lượng phân tích thị giác và tăng độ chính xác cho các bước sau.

- Lưu ảnh vào Firebase Storage: Ảnh sau khi tách nền sẽ được lưu trữ dưới dạng ảnh PNG trong suốt trên Firebase Storage. Việc này giúp chuẩn hóa đường dẫn ảnh và phục vụ cho các dịch vụ AI sử dụng.
- Gọi dịch vụ phân tích ảnh Gemini: Hệ thống gọi hàm GeminiService.analyzeItemImage() để gửi ảnh đến mô hình Gemini AI – một kiến trúc AI đa mô thức của Google có khả năng hiểu đồng thời hình ảnh và ngữ nghĩa. Gemini tiến hành phân tích nội dung ảnh và trả về kết quả dưới dạng JSON
- Sinh tên và mô tả ngắn cho item: Sau khi nhận metadata, hệ thống tiếp tục gọi GeminiService.generateItemNameAndDescription() để sinh ra tên item cùng mô tả ngắn gọn, phục vụ hiển thị trong tủ đồ và gợi ý phối đồ.
- Trả kết quả về client: Cuối cùng, hệ thống trả về cho ứng dụng một đối tượng JSON



Hình 2 Gemini AI

- Vai trò và lợi ích
  - Tự động hóa nhập liệu: Người dùng không cần thao tác thủ công để khai báo thông tin trang phục.
  - Chuẩn hóa dữ liệu thời trang: Các thuộc tính được sinh ra có định dạng thống nhất, giúp dễ dàng phân loại và tìm kiếm.
  - Nâng cao trải nghiệm người dùng: Metadata được sử dụng để xây dựng các thuật toán đề xuất phối đồ thông minh theo thời tiết, ngữ cảnh, cá tính.

1.2. Tổng quan về chatbot và xử lý ngôn ngữ tự nhiên (NLP)

1.2.1. Khái niệm và lịch sử phát triển của chatbot

- Chatbot là một hệ thống phần mềm có khả năng mô phỏng cuộc hội thoại với con người, thông qua văn bản hoặc giọng nói, nhằm cung cấp thông tin hoặc thực hiện một số tác vụ nhất định. Các chatbot hiện đại thường được xây dựng dựa trên nền tảng trí tuệ nhân tạo (AI) và xử lý ngôn ngữ tự nhiên (Natural Language Processing – NLP) để hiểu, phân tích và phản hồi ngữ nghĩa một cách chính xác và tự nhiên hơn.
- Lịch sử phát triển của chatbot có thể được chia làm nhiều giai đoạn:
    - 1966 – ELIZA: Chatbot đầu tiên do Joseph Weizenbaum phát triển, mô phỏng cuộc trò chuyện đơn giản dựa trên khớp từ khóa.
    - 1995 – ALICE: Sử dụng AIML (Artificial Intelligence Markup Language), tạo nền tảng cho nhiều chatbot AI sau này.
    - 2011 – Siri: Trợ lý ảo đầu tiên tích hợp NLP trong smartphone thương mại (Apple).
    - 2020 – GPT-based Chatbot: Với sự ra đời của GPT-3 và GPT-4, các chatbot có thể xử lý và tạo văn bản một cách linh hoạt, sáng tạo và theo ngữ cảnh cụ thể.

1.2.2. Tích hợp chatbot tư vấn thời trang sử dụng OpenAI API

- Chatbot đóng vai trò là trợ lý thời trang ảo có khả năng tư vấn phối đồ cho người dùng dựa trên thời tiết, sở thích, lịch trình hoặc một item cụ thể trong tủ đồ. Chatbot này được xây dựng trên nền tảng OpenAI API, sử dụng mô hình ngôn ngữ GPT-4 để xử lý ngôn ngữ tự nhiên và sinh câu trả lời ngữ cảnh hóa.

Hình 3 Chatbot tích hợp OpenAI API

- Quy trình hoạt động của chatbot
    - Người dùng đặt câu hỏi: Ví dụ "Hôm nay trời lạnh, nên mặc gì với chiếc áo hoodie?"
    - Gửi câu hỏi lên server: Backend nhận yêu cầu và chuyển tiếp tới ChatbotService.askGpt().
    - Gọi OpenAI GPT-4 API: Câu hỏi được gắn với prompt ngữ cảnh thời trang để định hướng câu trả lời.
    - Xử lý và sinh câu trả lời: GPT phân tích từ khóa, thời tiết, nội dung câu hỏi và sinh phản hồi.
    - Trả về client: Người dùng nhận được câu trả lời theo dạng hội thoại tự nhiên.
- Ưu điểm khi sử dụng OpenAI API
    - Ngữ cảnh hóa tốt: GPT có khả năng nhớ và phân tích đoạn hội thoại trước đó để đưa ra gợi ý hợp lý.
    - Khả năng mở rộng: Có thể tích hợp các tính năng như: phản hồi theo tone giọng, đề xuất theo phong cách cá nhân, hoặc đồng bộ với lịch.
    - Tương tác tự nhiên: Giao diện hội thoại thân thiện, phản hồi mượt mà như một stylist thật sự.
- Việc tích hợp chatbot AI vào hệ thống Personal Stylist giúp cá nhân hóa trải nghiệm thời trang ở mức cao. Thay vì lựa chọn trang phục dựa trên quy tắc cứng, người dùng có thể tương tác linh hoạt với trợ lý ảo như đang trò chuyện với một stylist thật sự. Điều này không chỉ tăng tính tiện lợi mà còn giúp xây dựng một hệ sinh thái thời trang thông minh, phù hợp với xu hướng cá nhân hóa hiện đại.

1.2.3. Chatbot tư vấn thời trang sử dụng NLP

- Xử lý ngôn ngữ tự nhiên (NLP) là kỹ thuật cốt lõi giúp chatbot hiểu được ý định (intent), thực thể (entity) và ngữ cảnh trong câu hỏi của người dùng. Trong hệ thống này, GPT-4 đảm nhận toàn bộ pipeline NLP, từ phân tích cú pháp, ngữ nghĩa cho đến sinh phản hồi.
- Ví dụ phân tích câu hỏi: "Hôm nay trời có mưa, mình nên mặc gì để đi học?"
    - Intent: Tư vấn trang phục
    - Entity: thời tiết = mưa, bối cảnh = đi học
    - Output mong muốn: outfit phù hợp, có thể chống nước, lịch sự, thoải mái

1.3. Tổng quan về ngôn ngữ lập trình Kotlin và SwiftUI

1.3.1. Ngôn ngữ Kotlin trong phát triển Android

- Trong dự án Personal Stylist, nền tảng Android được xây dựng bằng ngôn ngữ lập trình Kotlin kết hợp với XML để tạo giao diện người dùng. Đây là cách tiếp cận truyền thống nhưng hiệu quả, đặc biệt phù hợp khi cần kiểm soát tỉ mỉ bố cục giao diện và tách biệt rõ ràng giữa phần nhìn (UI) và phần xử lý logic (code).
- Kotlin mang đến nhiều lợi ích như cú pháp ngắn gọn, an toàn với null (null safety) và tương thích tốt với Java, cho phép tận dụng hệ sinh thái Android lâu đời. Trong khi đó, việc xây dựng giao diện bằng XML giúp dễ dàng mô phỏng thiết kế từ Figma sang ứng dụng thực tế, hỗ trợ mạnh mẽ trong các màn hình có bố cục phức tạp như phối đồ theo thời tiết, hiển thị tủ đồ cá nhân, hoặc tạo lịch mặc đồ hằng ngày.

Hình 4 Kotlin

- Ưu điểm nổi bật khi dùng Kotlin + XML trong Personal Stylist:
    - Dễ tái sử dụng layout: Các layout như thẻ trang phục, item outfit, khung lịch đều được định nghĩa sẵn bằng XML và tái sử dụng ở nhiều màn hình khác nhau.
    - Dễ kiểm thử và chỉnh sửa theo thiết kế: Nhờ cấu trúc rõ ràng của XML, nhóm phát triển có thể dễ dàng kiểm tra, tinh chỉnh từng thành phần UI cho sát với bản thiết kế Figma.
    - Tách biệt vai trò rõ ràng: Trong dự án, nhóm frontend tập trung xử lý XML và animation, trong khi nhóm backend xử lý dữ liệu và logic bằng Kotlin, đảm bảo quá trình phát triển mạch lạc và hiệu quả.

1.3.2. SwiftUI và giao diện người dùng cho hệ sinh thái iOS

- SwiftUI là framework do Apple phát triển nhằm xây dựng giao diện người dùng một cách hiện đại, trực quan và có tính khai báo (declarative). Trong dự án Personal Stylist, SwiftUI được sử dụng để phát triển phiên bản iOS của ứng dụng, với mục tiêu mang đến trải nghiệm mượt mà, tương tác thời trang cá nhân hóa và đồng bộ với thiết kế đã có trên nền tảng Android.
- Khác với UIKit truyền thống, SwiftUI cho phép mô tả giao diện bằng cú pháp Swift đơn giản, dễ đọc và gắn liền với dữ liệu động thông qua cơ chế @State, @Binding, và ObservableObject. Điều này giúp tăng tốc độ phát triển, dễ bảo trì và thuận tiện cho việc mô phỏng UI từ thiết kế Figma.

Hình 5 SwiftUI

- Ưu điểm khi sử dụng SwiftUI trong dự án:
    - Cập nhật giao diện theo thời gian thực: Mọi thay đổi trong dữ liệu (thêm đồ, chỉnh sửa outfit…) đều được UI phản ánh tức thì thông qua cơ chế @State và ObservableObject.
    - Hỗ trợ preview trực quan: Cho phép xem trước giao diện ngay trong Xcode khi thiết kế từng component.
    - Đa nền tảng: SwiftUI hỗ trợ cả iPhone, iPad và macOS, mở ra khả năng mở rộng ứng dụng sang các thiết bị Apple khác trong tương lai.
    - Tích hợp tốt với AI + ảnh: SwiftUI dễ dàng kết hợp với các module xử lý ảnh (như nhận diện trang phục, tách nền) và tích hợp API thời tiết để gợi ý trang phục theo ngữ cảnh.

1.4. NodeJS và kiến trúc RESTful API

1.4.1. Tổng quan về NodeJS

- Node.js được phát triển từ Javascript vào năm 2009 bởi Ryan Dahl. NodeJS có cách thức hoạt động chủ yếu trên Server sử dụng để xây dựng cho các ứng dụng realtime. NodeJS dùng mô hình I/O lập trình dựa theo sự kiện non-blocking.
- Chính vì vậy, NodeJS tương đối gọn nhẹ, hiệu quả và là một công cụ hoàn hảo dành cho mọi ứng dụng chuyên sâu về dữ liệu dựa theo khoảng thời gian thực khi chạy trên những thiết bị phân tán. NodeJs thường xuyên được dùng cho mục đích xây dựng một số ứng dụng như: Ad Server, Websocket server, Fast File Upload Client, RESTful API, Cloud Services, Any Real-time Data Application
- Ưu điểm khi dùng NodeJS
    - IO hướng sự kiện không đồng bộ, cho phép xử lý nhiều yêu cầu đồng thời.
    - Sử dụng JavaScript – một ngôn ngữ lập trình dễ học.
    - Chia sẻ cùng code ở cả phía client và server. NPM (Node Package Manager) và module Node đang ngày càng phát triển mạnh mẽ.
    - Cộng đồng hỗ trợ tích cực.
    - Cho phép stream các file có kích thước lớn.
- Nhược điểm khi dung NodeJS
    - Không có khả năng mở rộng, vì vậy không thể tận dụng lợi thế mô hình đa lõi trong các phần cứng cấp server hiện nay.
    - Khó thao tác với cơ sử dữ liệu quan hệ.
    - Mỗi callback sẽ đi kèm với rất nhiều callback lồng nhau khác.
    - Cần có kiến thức tốt về JavaScript.
    - Không phù hợp với các tác vụ đòi hỏi nhiều CPU.

1.4.2. ExpressJS là gì

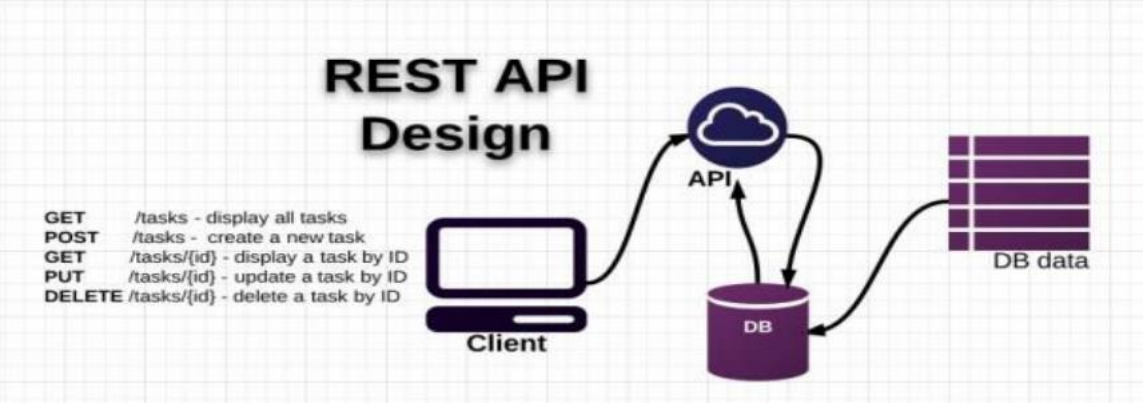
- Express là một web framework phổ biến, được viết bằng JavaScript và chạy trên môi trường Node.js. Mô-đun sẽ giải thích một số lợi ích của framework này, cách để thiết lập môi trường, phát triển cũng như triển khai ứng dụng.

Hình 6 NodeJS & ExpessJS

- Tổng hợp một số chức năng chính của Expressjs như sau:
    - Thiết lập các lớp trung gian để trả về các HTTP request.
    - Define router cho phép sử dụng với các hành động khác nhau dựa trên phương thức HTTP và URL.
    - Cho phép trả về các trang HTML dựa vào các tham số.
- ExpressJS sẽ giúp bạn tổ chức kiến trúc back-end của mình. Các nhà phát triển web thường sử dụng ExpressJS để triển khai kiến trúc MVC, điều này cho phép họ viết một codebase back-end bảo trì tương đối dễ dàng.
- Những lợi ích của ExpressJS
    - Rất dễ học, chỉ cần bạn biết JavaScript, bạn sẽ không cần phải học một ngôn ngữ mới để học ExpressJS
    - Giúp cho việc phát triển back-end dễ dàng hơn nhiều khi sử dụng ExpressJS
    - Mã JavaScript được diễn giải thông qua Google V8 JavaScript Engine của Node.js. Do đó, mã sẽ được thực hiện một cách nhanh chóng và dễ dàng.
    - ExpressJS rất đơn giản để tùy chỉnh và sử dụng theo nhu cầu.
    - Cung cấp một module phần mềm trung gian linh hoạt và rất hữu ích để thực hiện các tác vụ bổ sung theo phản hồi và yêu cầu.

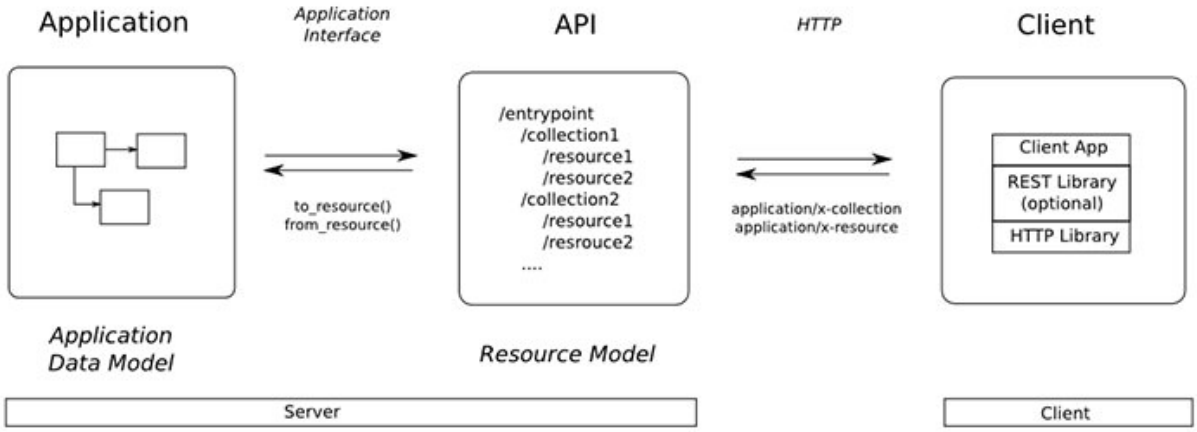
1.4.3. RESTful API

- RESTful API là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. Nó chú trọng vào tài nguyên hệ thống (tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu động…), bao gồm các trạng thái tài nguyên được định dạng và được truyền tải qua HTTP.



Hình 7 RESTful API

- **RESTful hoạt động như thế nào?**



Hình 8 Hoạt động của RESTful API

- REST hoạt động chủ yếu dựa vào giao thức HTTP. Các hoạt động cơ bản nêu trên sẽ sử dụng những phương thức HTTP riêng.
  - GET (SELECT): Trả về một Resource hoặc một danh sách Resource.
  - POST (CREATE): Tạo mới một Resource.
  - PUT (UPDATE): Cập nhật thông tin cho Resource.
  - DELETE (DELETE): Xoá một Resource.
- Những phương thức hay hoạt động này thường được gọi là CRUD tương ứng với Create, Read, Update, Delete – Tạo, Đọc, Sửa, Xóa.

- **Authentication và dữ liệu trả về**

- RESTful API không sử dụng session và cookie, nó sử dụng một access_token với mỗi request. Dữ liệu trả về thường có cấu trúc như sau:
- **Status code:** Khi chúng ta request một API nào đó thường thì sẽ có vài status code để nhận biết sau:

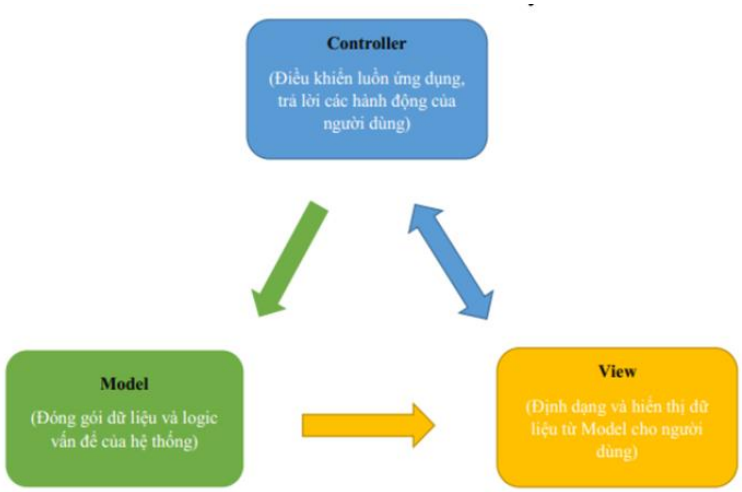
- 200 OK – Trả về thành công cho những phương thức GET, PUT, PATCH hoặc
- DELETE.
- 201 Created – Trả về khi một Resouce vừa được tạo thành công.
- 204 No Content – Trả về khi Resource xoá thành công.
- 304 Not Modified – Client có thể sử dụng dữ liệu cache.
- 400 Bad Request – Request không hợp lệ
- 401 Unauthorized – Request cần có auth.
- 403 Forbidden – bị từ chối không cho phép.
- 404 Not Found – Không tìm thấy resource từ URI
- 405 Method Not Allowed – Phương thức không cho phép với user hiện tại.
- 410 Gone – Resource không còn tồn tại, Version cũ đã không còn hỗ trợ.
- 415 Unsupported Media Type – Không hỗ trợ kiểu Resource này.
- 422 Unprocessable Entity – Dữ liệu không được xác thực
- 429 Too Many Requests – Request bị từ chối do bị giới hạn

- **HTTP Request:** HTTP request có tất cả 9 loại method (2 loại được sử dụng phổ biến nhất là GET và POST):
  - GET: Được sử dụng để lấy thông tin từ server theo URI đã cung cấp.
  - HEAD: Giống với GET nhưng response trả về không có body, chỉ có header.
  - POST: Gửi thông tin tới sever thông qua các biểu mẫu http.
  - PUT: Ghi đè tất cả thông tin của đối tượng với những gì được gửi lên.
  - PATCH: Ghi đè các thông tin được thay đổi của đối tượng.
  - DELETE: Xóa tài nguyên trên server.
  - CONNECT: Thiết lập một kết nối tới server theo URI.
  - OPTIONS: Mô tả các tùy chọn giao tiếp cho resource.
  - TRACE: Thực hiện một bài test loop – back theo đường dẫn đến resource.

1.4.4. Mô hình MVC

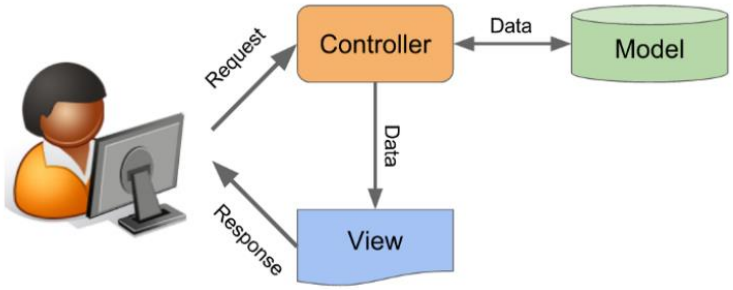
- MVC là viết tắt của cụm từ "Model-View-Controller". Là mô hình thiết kế sử dụng trong kỹ thuật phần mềm. MVC là một mẫu kiến trúc phần mềm để tạo lập giao diện người dùng trên máy tính.

- Mô hình MVC thường được chia làm 3 phần. Mỗi phần đảm bảo một vai trò và nhiệm vụ riêng biệt khác nhau.
- 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.
- View: Đây là phần giao diện dành cho người dùng.MVC là phương tiện hiển thị các đối tượng trong một ứng dụng.
- Controller: Là bộ phận có nhiệm vụ xử lý các yêu cầu người dùng đưa đến thông qua View.



Hình 9 Mô hình MVC

- Luồng xử lý trong MVC
  - Khi một yêu cầu của khách hàng từ máy khách (Client) gửi đến Server. Thì bị Controller trong MVC chặn lại để xem đó là URL request hay sự kiện.
  - Sau đó, Controller xử lý input của user rồi giao tiếp với Model trong MVC.
  - Model chuẩn bị data và gửi lại cho Controller.
  - Cuối cùng, khi xử lý xong yêu cầu thì Controller giữ dữ liệu trở lại View và hiển thị cho người dùng trên trình duyệt.



Hình 10 Luồng xử lý trong MVC

1.5. Tổng quan về cơ sở dữ liệu MongoDB

- NoSQL (Not Only SQL) là một loại cơ sở dữ liệu phi quan hệ, được thiết kế để xử lý khối lượng dữ liệu lớn, có tính chất phi cấu trúc hoặc bán cấu trúc. Không giống như cơ sở dữ liệu quan hệ truyền thống (RDBMS) sử dụng bảng và ràng buộc dữ liệu chặt chẽ, NoSQL linh hoạt hơn, cho phép lưu trữ các dạng dữ liệu đa dạng như JSON, BSON, XML, hoặc key-value. NoSQL được đánh giá là phù hợp với các hệ thống hiện đại cần khả năng mở rộng cao, phản hồi nhanh và đáp ứng tốt với các mô hình dữ liệu linh hoạt.
- MongoDB là một trong những hệ quản trị cơ sở dữ liệu NoSQL phổ biến nhất hiện nay. Dữ liệu trong MongoDB được lưu trữ dưới dạng các tài liệu (document) với cấu trúc BSON – một dạng nhị phân mở rộng từ JSON, cho phép hệ thống dễ dàng thao tác với dữ liệu có cấu trúc không đồng nhất. Mỗi tài liệu là một thực thể độc lập, chứa đầy đủ thông tin và có thể truy xuất nhanh chóng thông qua chỉ mục.

Hình 11 Tổng quan về MongoDB

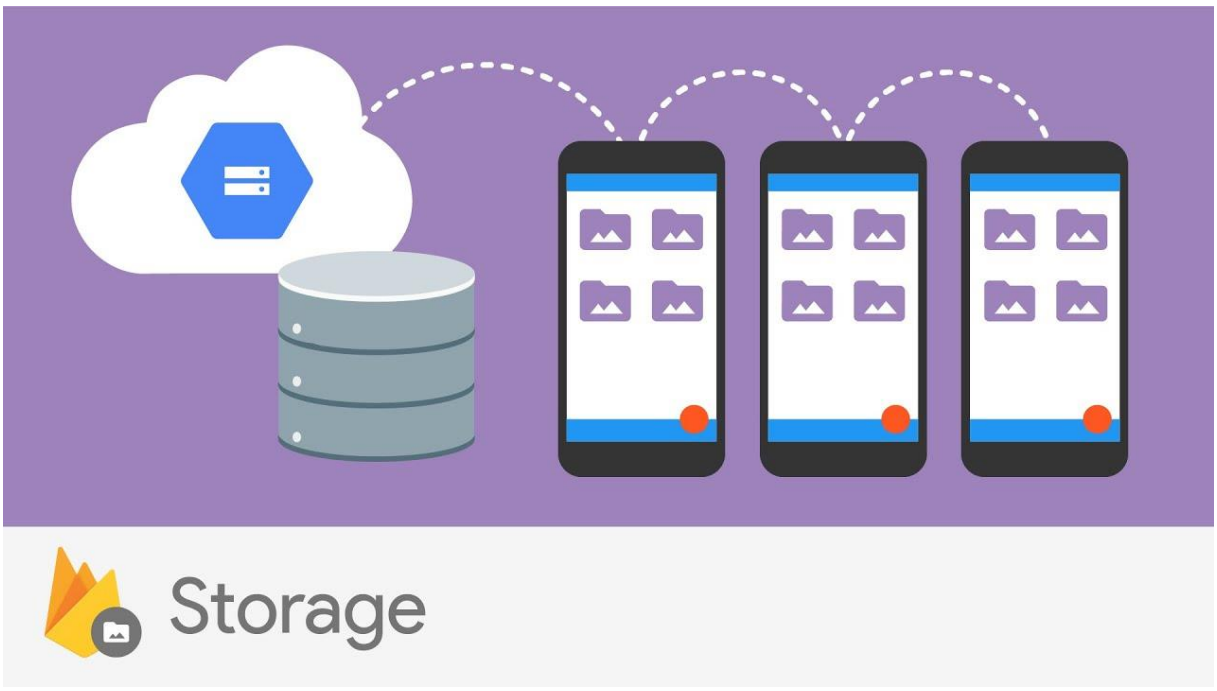
- Ưu điểm của MongoDB trong lưu trữ trang phục và người dùng: trong bối cảnh phát triển ứng dụng Personal Stylist, việc quản lý thông tin trang phục, người dùng, lịch sử phối đồ, ảnh chụp, và thuộc tính thời tiết đòi hỏi một hệ thống cơ sở dữ liệu linh hoạt, mở rộng dễ dàng và có khả năng truy vấn hiệu quả. MongoDB đáp ứng tốt các yêu cầu đó nhờ những ưu điểm nổi bật sau:
    - Lưu trữ dữ liệu không ràng buộc schema: Mỗi người dùng có thể có một tủ đồ với cấu trúc khác nhau, các ảnh trang phục có thể chứa siêu dữ liệu như màu sắc, kiểu dáng, chất liệu,... Dữ liệu không đồng nhất này được MongoDB xử lý mượt mà mà không cần thiết kế cấu trúc bảng cứng nhắc.

  - Truy vấn linh hoạt và hiệu suất cao: Với khả năng lập chỉ mục trên nhiều thuộc tính, hệ thống có thể nhanh chóng truy vấn trang phục phù hợp theo mùa, màu sắc, hoàn cảnh sử dụng hay lịch sử phối đồ.
  - Dễ dàng mở rộng: Khi số lượng người dùng tăng lên, MongoDB hỗ trợ mở rộng ngang thông qua sharding, giúp ứng dụng vẫn giữ được hiệu năng ổn định.
  - Lưu trữ ảnh và file dung lượng lớn: MongoDB có thể kết hợp với GridFS để lưu trữ ảnh trang phục, tạo điều kiện triển khai chức năng nhận diện và hiển thị ảnh từ tủ đồ người dùng.
  - Tích hợp thuận tiện với NodeJS: MongoDB hoạt động tốt cùng với ExpressJS và Mongoose, cho phép backend dễ dàng ánh xạ dữ liệu từ ứng dụng vào cơ sở dữ liệu mà không cần thủ công chuyển đổi định dạng.
- Tuy nhiên, MongoDB cũng tồn tại một số hạn chế như không hỗ trợ trực tiếp các phép Join như cơ sở dữ liệu quan hệ, hoặc có giới hạn kích thước tài liệu (16MB), đòi hỏi thiết kế dữ liệu hợp lý để tránh dư thừa hoặc sai lệch trong lưu trữ. Dù vậy, với tính chất linh hoạt và định hướng document, MongoDB là lựa chọn tối ưu cho ứng dụng Personal Stylist – nơi mà dữ liệu người dùng và trang phục luôn thay đổi và đa dạng.

1.6. Firebase và ứng dụng trong lưu trữ đa phương tiện

1.6.1. Firebase Cloud Storage

- Firebase Cloud Storage là một dịch vụ lưu trữ dữ liệu dạng đối tượng do Google cung cấp, nằm trong hệ sinh thái Firebase – nền tảng phát triển ứng dụng toàn diện hỗ trợ backend, authentication, analytics, và nhiều dịch vụ khác. Cloud Storage được thiết kế để lưu trữ các tập tin dung lượng lớn như hình ảnh, video, âm thanh và tài liệu, với khả năng bảo mật, mở rộng và truy cập tốc độ cao.
- Firebase Cloud Storage cho phép tích hợp trực tiếp với các ứng dụng Android, iOS hoặc Web thông qua SDK. Dữ liệu được lưu trữ dưới dạng "bucket" trên hạ tầng Google Cloud Platform, đảm bảo độ tin cậy cao, đồng thời hỗ trợ phân quyền chi tiết cho từng nhóm người dùng dựa trên Firebase Authentication.



Hình 12 Firebase Cloud Storage

- Một số tính năng chính của Firebase Cloud Storage:
    - Lưu trữ file không giới hạn kích thước (tối đa đến 5TB mỗi tệp).
    - Truy xuất dữ liệu theo URL có kiểm soát truy cập, đảm bảo an toàn.
    - Tự động điều chỉnh hiệu năng truy xuất theo lưu lượng người dùng, đảm bảo khả năng mở rộng.
    - Hỗ trợ SDK đa nền tảng (Web, Android, iOS) với quy trình upload và download dễ tích hợp.

1.6.2. Ứng dụng quản lý ảnh trang phục và đồng bộ đa nền tảng

- Trong ứng dụng Personal Stylist, hình ảnh đóng vai trò then chốt trong việc lưu trữ và hiển thị thông tin trang phục, tủ đồ cá nhân, ảnh gợi ý phối đồ, và ảnh nền phục vụ cho các tác vụ xử lý ảnh (như tách nền, nhận diện kiểu dáng). Do đó, việc chọn giải pháp lưu trữ ảnh hiệu quả là yếu tố thiết yếu để tối ưu trải nghiệm người dùng.
- Firebase Cloud Storage được lựa chọn nhờ những lợi thế sau:
    - Đồng bộ hóa tức thời ảnh giữa các thiết bị: người dùng có thể tải ảnh từ điện thoại Android và xem lại tức thì từ thiết bị iOS hoặc Web.
    - Hỗ trợ tạo URL công khai hoặc có token xác thực, phù hợp với việc hiển thị ảnh trong ứng dụng hoặc xử lý bằng AI phía server.

- Ví dụ, khi người dùng thêm một chiếc áo mới vào tủ đồ, ảnh chụp sẽ được upload lên Firebase Storage. Ảnh này sau đó sẽ được liên kết với metadata trong MongoDB (chứa thông tin mô tả như màu sắc, loại trang phục, mùa sử dụng), và có thể được sử dụng cho các tính năng như:
    - Gợi ý phối đồ từ ảnh
    - Hiển thị ảnh đại diện của từng item
    - Truy vấn tủ đồ theo hình ảnh
- Nhờ Firebase Storage, ứng dụng đạt được độ ổn định cao trong truy xuất ảnh, đồng thời vẫn đảm bảo bảo mật và khả năng mở rộng, đặc biệt hữu ích khi số lượng người dùng hoặc ảnh tăng lên đáng kể.

1.7. Tổng quan về Text Embeddings và ứng dụng trong hệ thống đề xuất trang phục

- **Khái niệm về Text Embeddings:** Text Embedding là một kỹ thuật chuyển đổi văn bản thành các vector số (vector không gian thực) có chiều cố định, sao cho các từ, cụm từ hoặc đoạn văn có ngữ nghĩa tương tự sẽ có vector gần nhau trong không gian này. Đây là nền tảng của hầu hết các ứng dụng xử lý ngôn ngữ tự nhiên hiện đại, đặc biệt là trong việc tìm kiếm, phân loại, gợi ý, và hội thoại.
- **Một số phương pháp embedding nổi bật:**
    - Word2Vec, GloVe: học vector từ theo ngữ cảnh thống kê.
    - BERT, Sentence-BERT (SBERT): embedding theo ngữ cảnh sâu, cho cả câu hoặc đoạn văn.
    - OpenAI Embedding API: cung cấp embedding mạnh mẽ, có thể dùng để so khớp ngữ nghĩa, tìm kiếm tương tự văn bản, v.v.
- Trong ứng dụng Personal Stylist, text embeddings được ứng dụng ở nhiều vị trí, nổi bật nhất là:
    - Gợi ý phối đồ theo ngữ cảnh người dùng nhập vào: Khi người dùng nhập một câu như: "Hôm nay trời lạnh, tôi đi học và muốn mặc gì đơn giản", hệ thống sử dụng OpenAI Embedding API để tạo vector cho câu hỏi. Sau đó, so sánh vector đó với các vector mô tả item hoặc outfit trong tủ đồ (đã embedding trước đó), để tìm ra các trang phục gần nghĩa nhất với yêu cầu.

- Tương tác chatbot ngữ nghĩa hóa: Các câu hỏi và lịch sử hội thoại người dùng được ánh xạ vào vector embedding, giúp chatbot gợi ý chính xác hơn, duy trì ngữ cảnh hội thoại.

1.8. Kết chương 1

- Chương này đã giới thiệu các công nghệ nền tảng được ứng dụng trong dự án: từ AI trong xử lý ảnh và chatbot, đến các công cụ lập trình như Kotlin, SwiftUI, NodeJS, MongoDB và Firebase. Đây là phần trọng yếu giúp hiểu rõ cách hệ thống được xây dựng và triển khai, tạo nền tảng vững chắc để phân tích và thiết kế hệ thống ở chương sau.

CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

2.1. Phân tích nghiệp vụ chính của người dùng

2.1.1. Quản lý tủ đồ

- Thêm, sửa, xóa, phân loại trang phục theo màu sắc, loại, mùa sử dụng,...
- Xem danh sách các item dưới dạng lưới có ảnh đại diện (đã tách nền).
- Lưu trữ và phân loại trang phục theo từng người dùng riêng biệt.

2.1.2. Tạo và lưu Outfit

- Tối ưu hóa việc chọn trang phục hàng ngày.
- Quản lý tủ đồ cá nhân một cách hiệu quả.
- Nhận gợi ý phối đồ theo thời tiết, sự kiện, lịch trình, hoặc phong cách cá nhân.
- Tương tác với chatbot thời trang như một stylist ảo, nhanh chóng và tiện lợi.

2.1.3. Tương tác chatbot

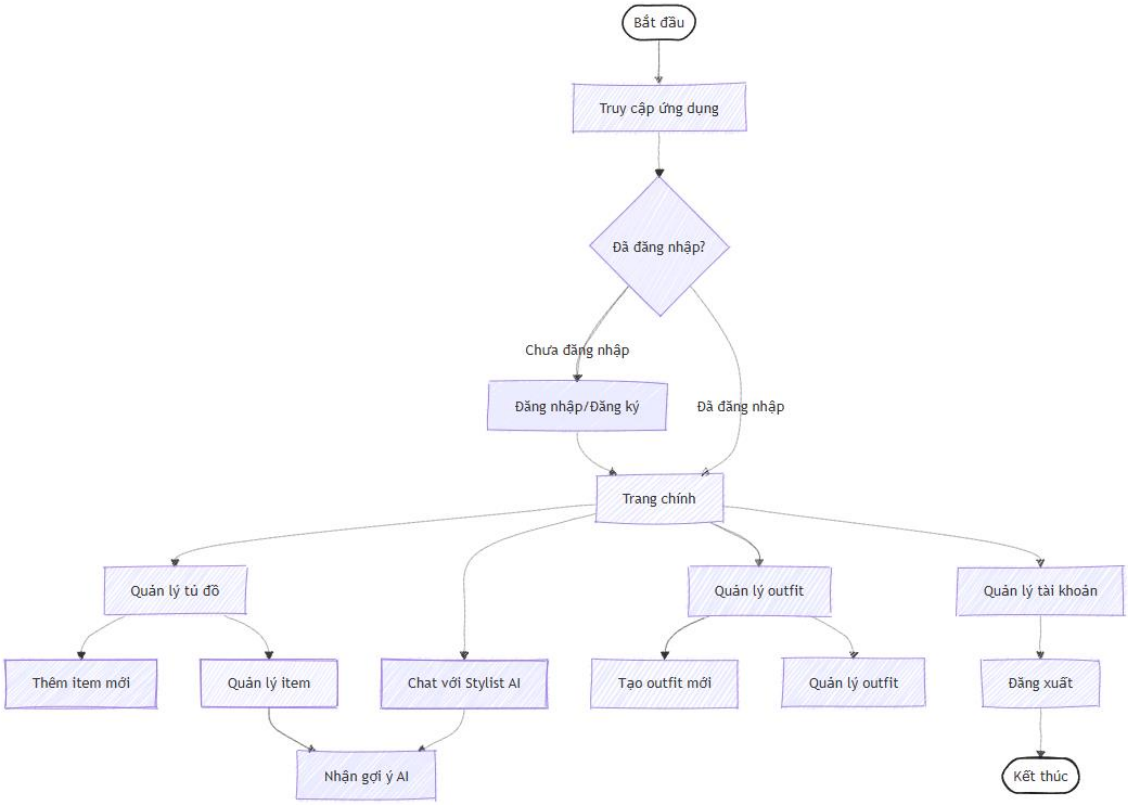
- Gửi câu hỏi tự nhiên đến chatbot (ví dụ: “Nay trời lạnh, mặc gì với váy jeans?”).
- Nhận phản hồi thời trang ngữ cảnh hóa từ GPT-4 thông qua API OpenAI.
- Có thể trò chuyện theo chuỗi hội thoại, lưu ngữ cảnh.

2.1.4. Quản lý tài khoản

- Đăng ký, đăng nhập.
- Quản lý thông tin cá nhân, avatar.
- Đồng bộ dữ liệu cá nhân qua nhiều thiết bị (thông qua Firebase và MongoDB).

2.2. Thiết kế hệ thống

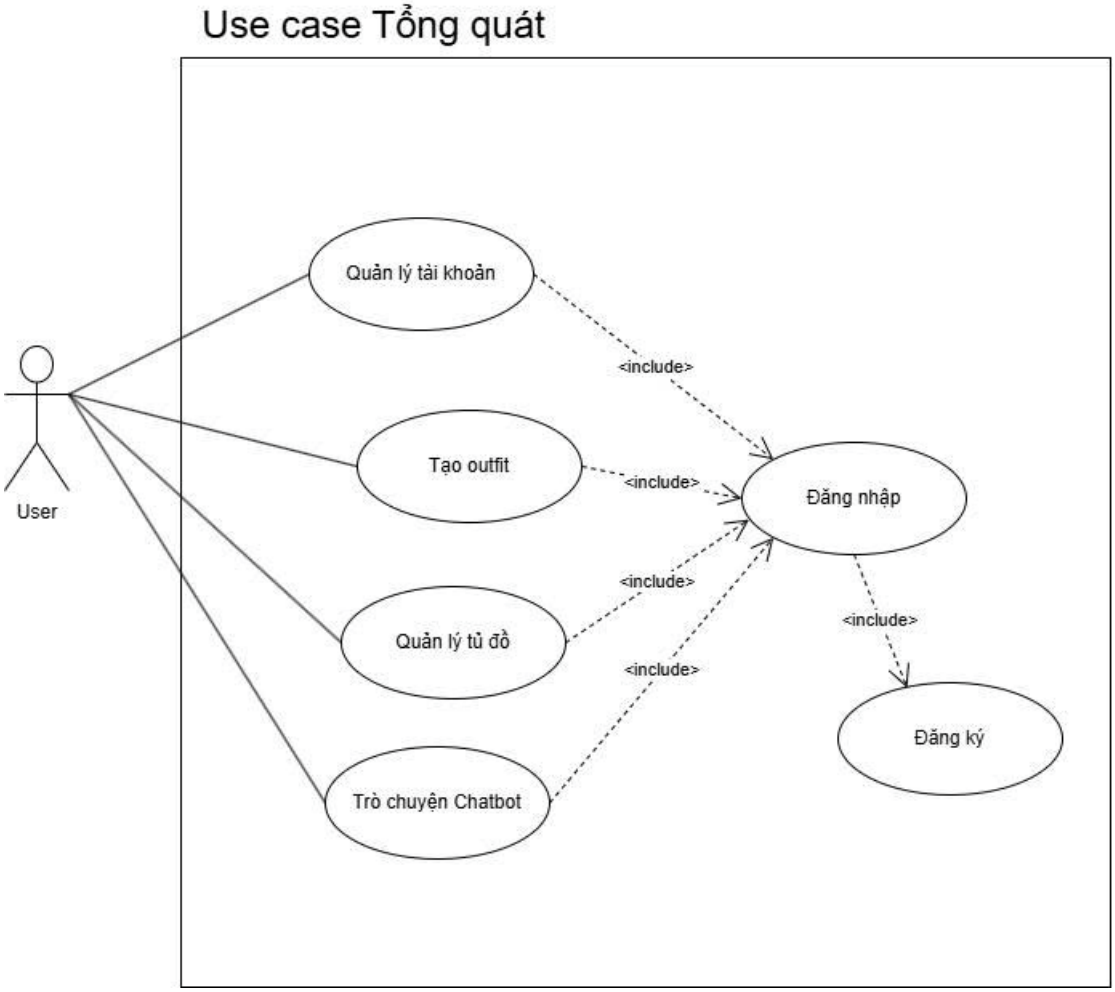
2.2.1. Sơ đồ nguyên lý hoạt động



Hình 13 Sơ đồ nguyên lý hoạt động

2.2.2. Sơ đồ ca sử dụng (Usecase)

- Sơ đồ ca sử dụng là một kỹ thuật được dùng trong kỹ thuật phần mềm và hệ thống để nắm bắt yêu cầu chức năng của hệ thống.
- Usecase tổng quát:



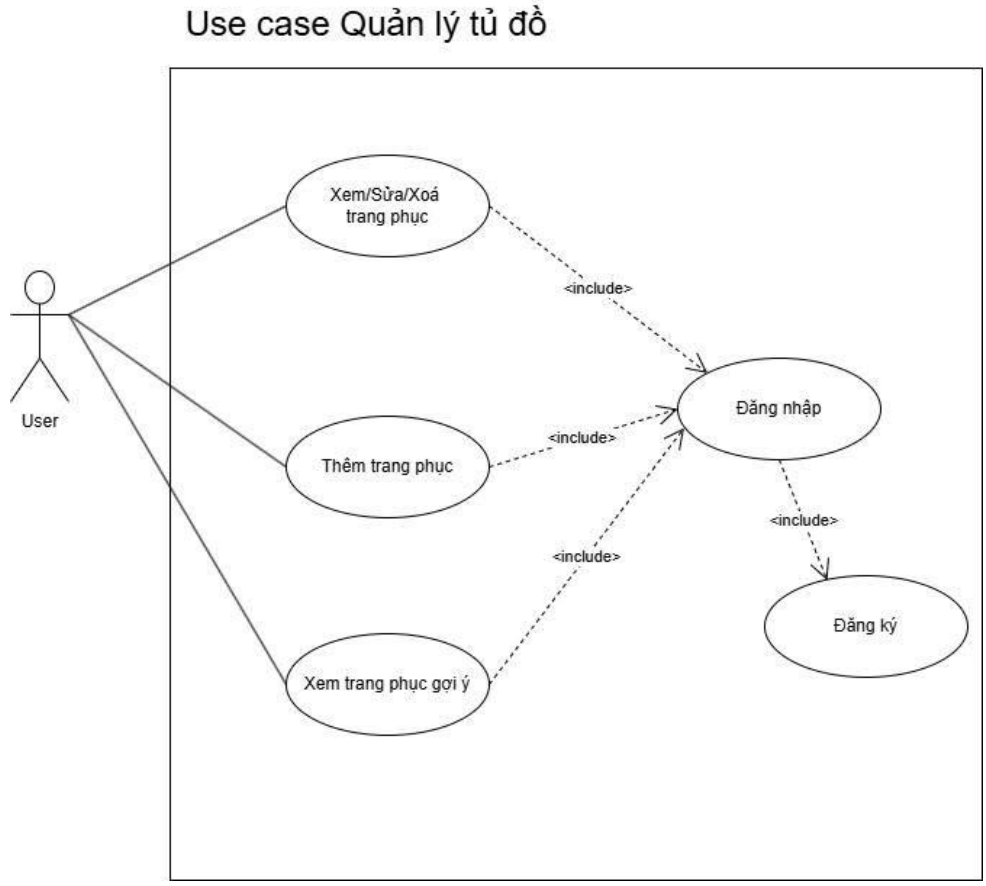
Hình 14 Sơ đồ Usecase tổng quát

Bảng 2 Đặc tả Usecase tổng quát

Mã Use Case	UC-01
Tên Use Case	Tổng quát hệ thống
Tác nhân	Người dùng
Mô tả	Cho phép người dùng trải nghiệm toàn bộ chức năng chính của hệ thống từ quản lý trang phục, phối đồ, tư vấn AI, đến quản lý thông tin cá nhân
Điều kiện tiên quyết	Người dùng đã cài đặt ứng dụng trên thiết bị Android hoặc iOS.

Luồng chính	1. Truy cập ứng dụng 2. Đăng ký/Đăng nhập 3. Sử dụng các chức năng chính (UC02 → UC05)
Kết quả mong muốn	Người dùng quản lý tủ đồ và được tư vấn phối đồ cá nhân hóa hiệu quả.

- Usecase Quản lý tủ đồ:



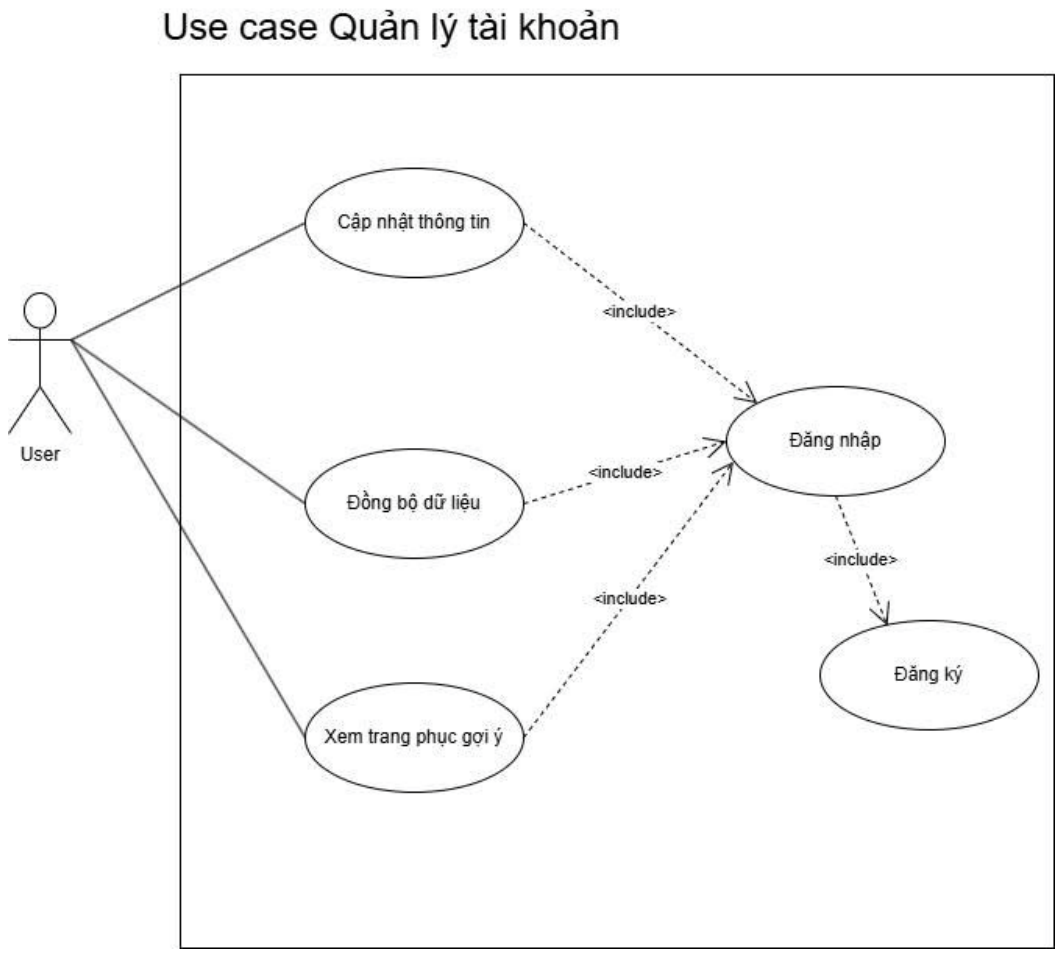
Hình 15 Sơ đồ Usecase Quản lý tủ đồ

Bảng 3 Đặc tả Usecase quản lý tủ đồ

Mã Use Case	UC-02
Tên Use Case	Quản lý tủ đồ
Tác nhân	Người dùng
Mô tả	Người dùng chụp hoặc tải ảnh trang phục > Hệ thống sử dụng công nghệ tách nền và AI (Gemini) để sinh metadata tự động > Sau đó, item được lưu và hiển thị > Người dùng có thể lọc item

Điều kiện tiên quyết	Người dùng đã đăng nhập
Luồng chính	1. Người dùng chọn thêm item 2. Tải ảnh lên 3. Hệ thống tách nền, sinh metadata 4. Lưu item vào MongoDB và Firebase 5. Hiển thị trong tủ đồ
Luồng phụ	Nếu AI không thể sinh metadata → cho phép nhập thủ công
Kết quả mong muốn	Trang phục được lưu trữ, phân loại và hiển thị rõ ràng trong tủ đồ cá nhân.

- Usecase Quản lý tài khoản:

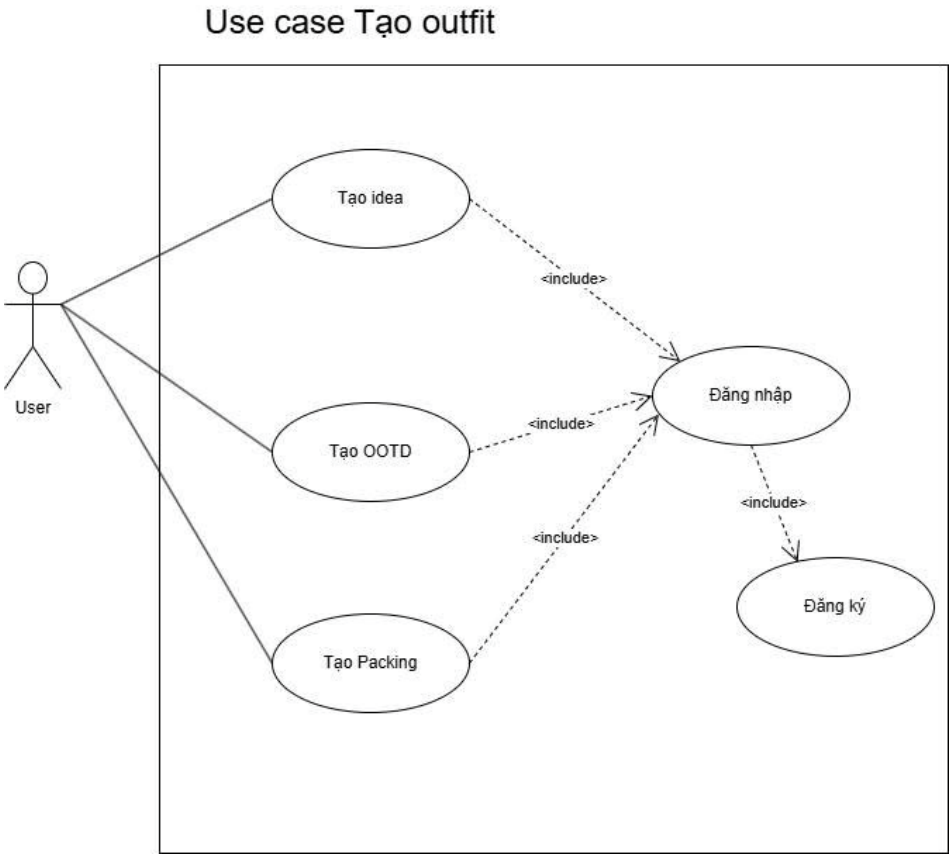


Hình 16 Sơ đồ Usecase Quản lý tài khoản

Bảng 4 Đặc tả Usecase quản lý tài khoản

Mã Use Case	UC-03
Tên Use Case	Quản lý tài khoản
Tác nhân	Người dùng
Mô tả	Người dùng có thể đăng ký bằng email, đăng nhập, chỉnh sửa avatar và thông tin cá nhân. Dữ liệu tài khoản được lưu và đồng bộ giữa các thiết bị nhờ Firebase Authentication và MongoDB.
Điều kiện tiên quyết	Người dùng chưa có tài khoản (đăng ký) hoặc đã có (đăng nhập).
Luồng chính	1. Đăng ký/Đăng nhập 2. Truy cập mục “Tài khoản” 3. Cập nhật thông tin
Kết quả mong muốn	Tài khoản người dùng được thiết lập và cá nhân hóa theo nhu cầu.

- Usecase Tạo Outfit

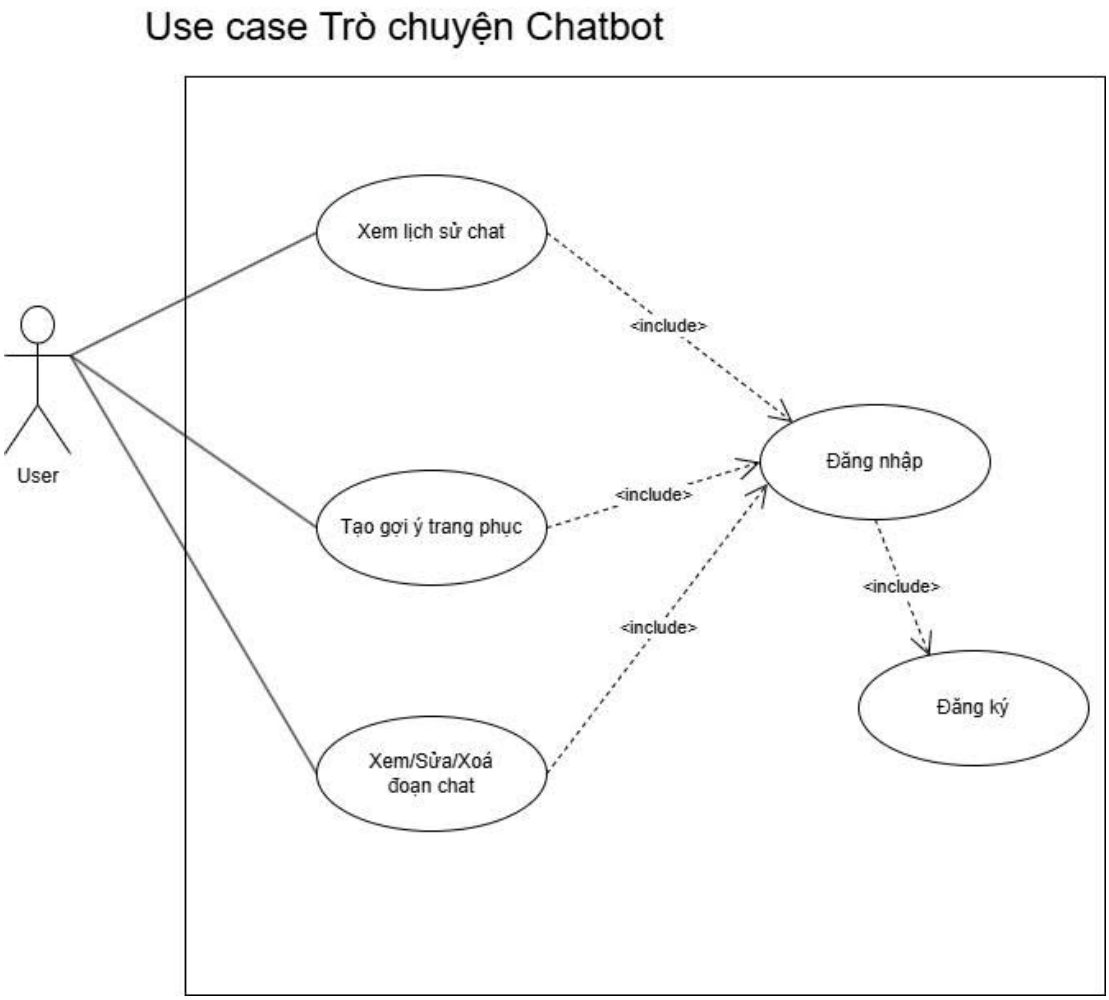


Hình 17 Sơ đồ Usecase Tạo Outfit

Bảng 5 Đặc tả Usecase Tạo và lưu Outfit

Mã Use Case	UC-04
Tên Use Case	Tạo và lưu Outfit
Tác nhân	Người dùng
Mô tả	Người dùng duyệt tủ đồ, chọn các item mong muốn để phối thành một outfit và lưu lại outfit đó. Ngoài ra, hệ thống có thể gợi ý outfit từ AI dựa trên thời tiết
Điều kiện tiên quyết	Người dùng có ít nhất 1 item trong tủ đồ.
Luồng chính	1. Mở trình tạo outfit 2. Chọn item 3. Lưu outfit vào hệ thống
Luồng phụ	Nhận gợi ý outfit từ AI nếu không muốn chọn thủ công
Kết quả mong muốn	Outfit mới được tạo và lưu để sử dụng về sau.

- User Trò chuyện Chatbot:



Hình 18 Sơ đồ Usecase Trò chuyện Chatbot

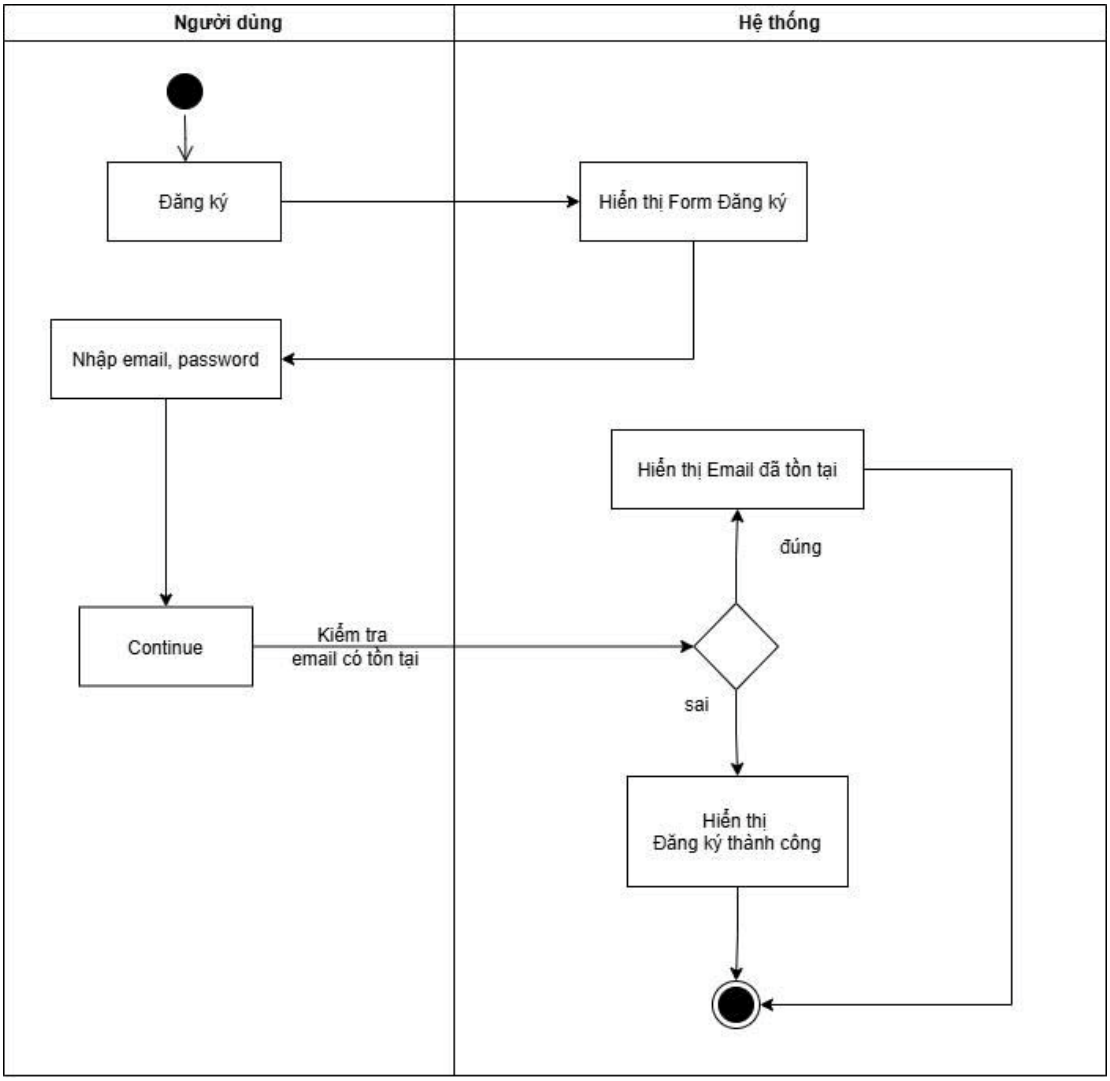
Bảng 6 Đặc tả Usecase Tương tác với Chatbot

Mã Use Case	UC-05
Tên Use Case	Tương tác với Chatbot
Tác nhân	Người dùng
Mô tả	Người dùng đặt câu hỏi tự nhiên về phối đồ. Hệ thống gửi truy vấn đến API GPT-4 với prompt thời trang. Dữ liệu thời tiết và lịch trình được đính kèm để cá nhân hóa phản hồi. Chatbot có thể nhớ hội thoại và phản hồi liên tục nhiều lượt.
Điều kiện tiên quyết	Người dùng đã đăng nhập và có kết nối internet.
Luồng chính	1. Gửi câu hỏi như "Trời mưa mặc gì với váy trắng?"

	2. Backend xử lý, gọi GPT-4 API 3. GPT trả lời phối đồ 4. Hiển thị trả lời cho người dùng
Kết quả mong muốn	Người dùng nhận được lời khuyên phối đồ hợp lý theo thời tiết/sự kiện/phong cách.

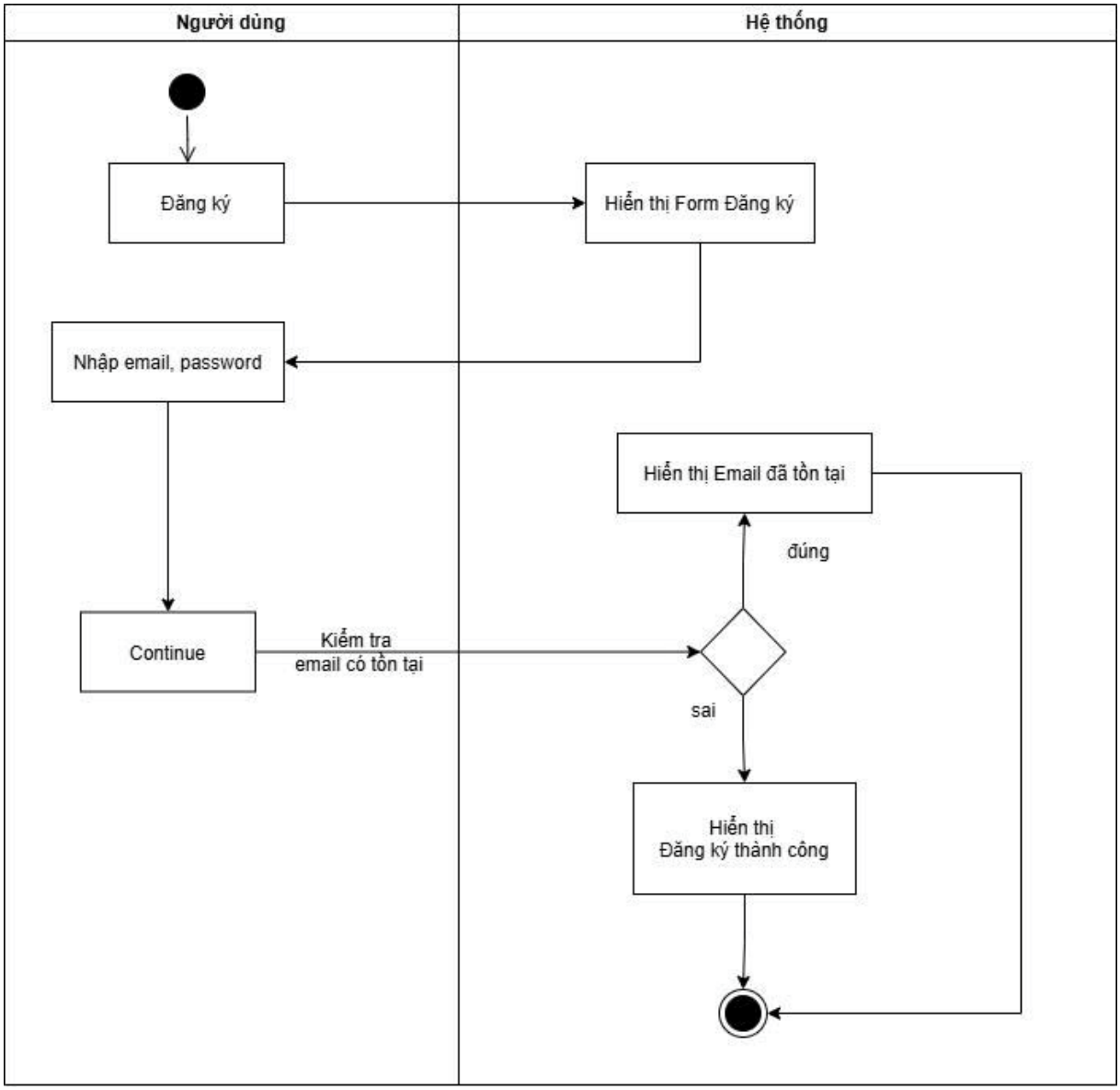
2.2.3. Sơ đồ hoạt động

- Sơ đồ hoạt động chức năng Đăng ký



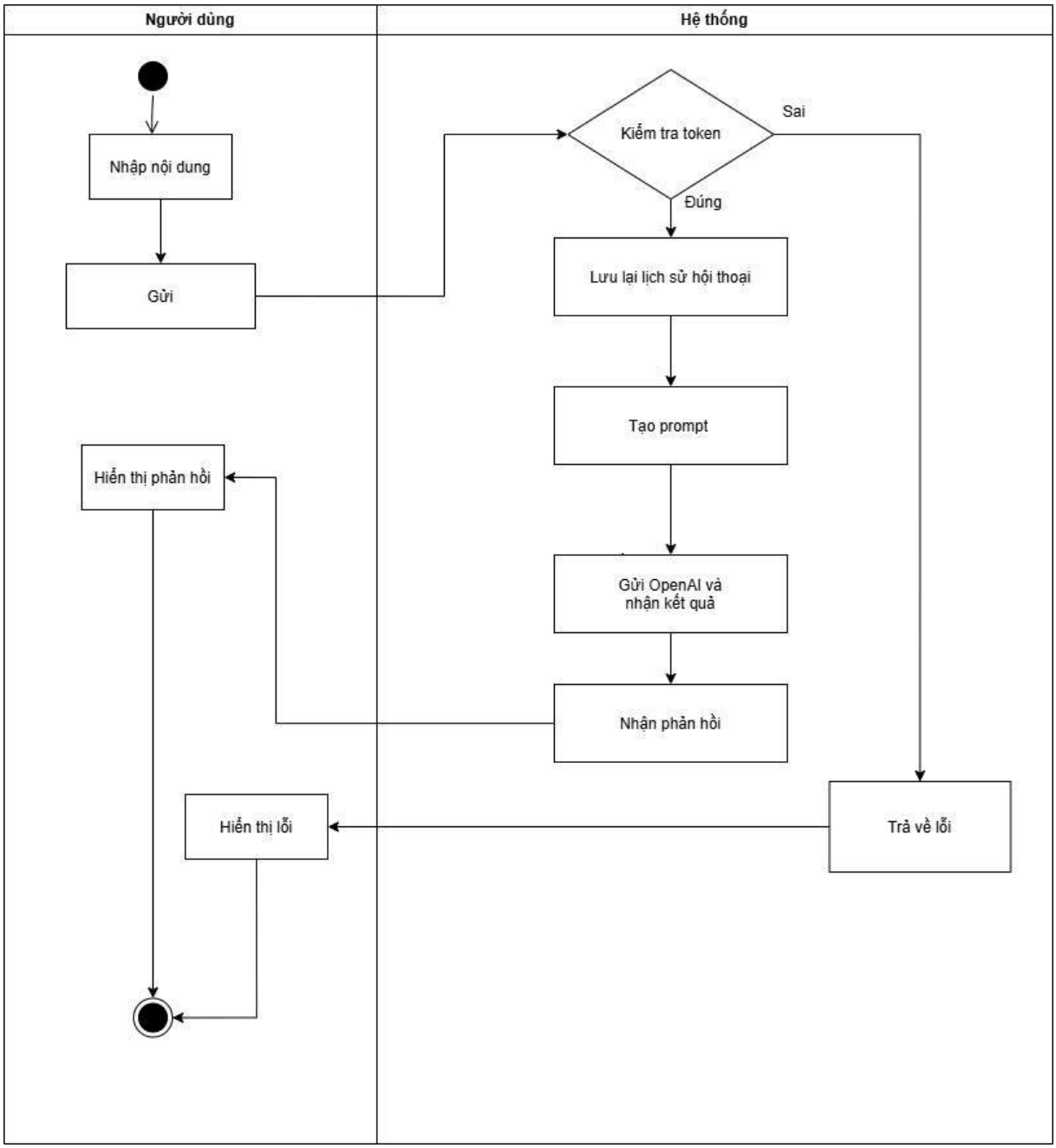
Hình 19 Sơ đồ động chức năng Đăng ký

- Sơ đồ hoạt động chức năng Đăng nhập



Hình 20 Sơ đồ hoạt động chức năng Đăng nhập

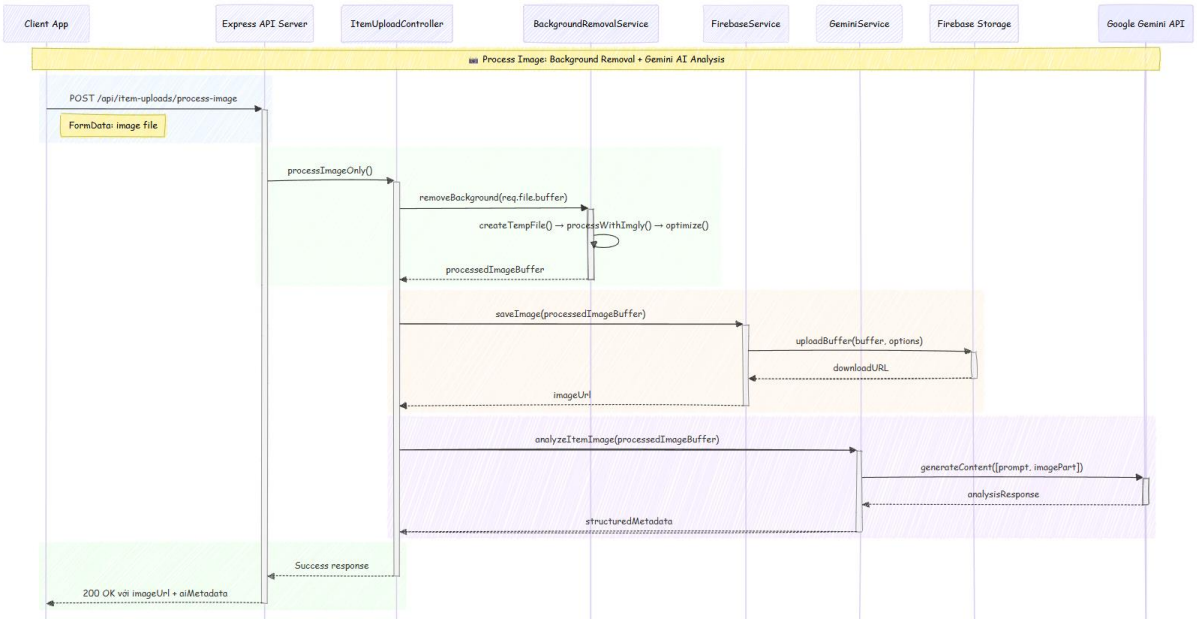
- Sơ đồ hoạt động chức năng trò chuyện với chatbot



Hình 21 Sơ đồ hoạt động chức năng Tương tác với Chatbot

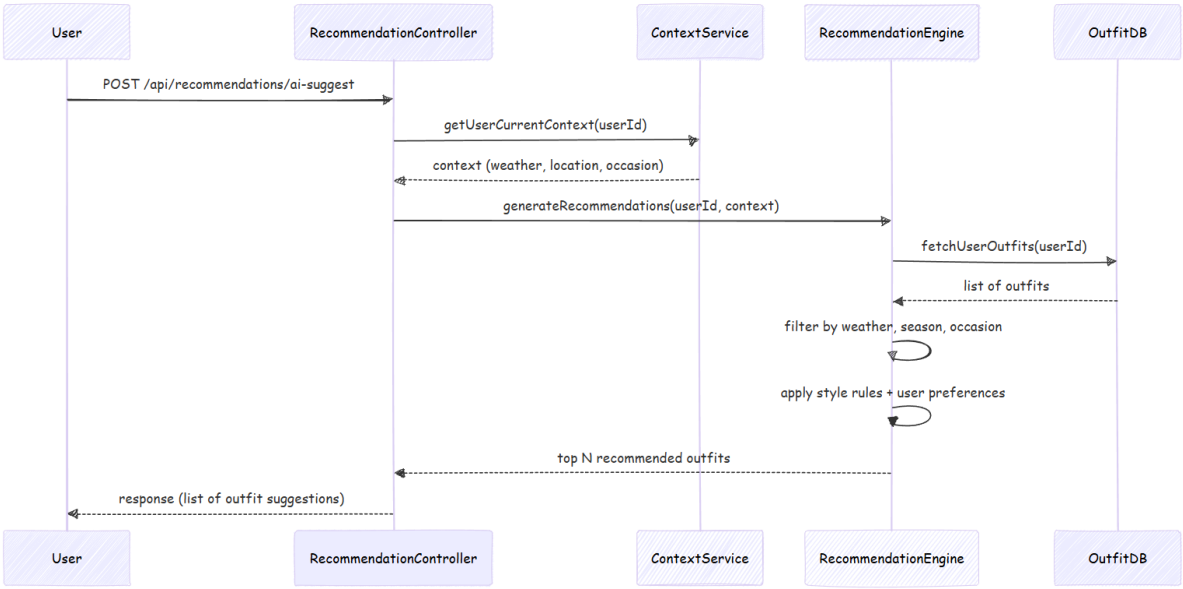
2.2.4. Sơ đồ tuần tự

- Sơ đồ tuần tự chức năng Xử lý ảnh



Hình 22 Sơ đồ tuần tự chức năng Xử lý ảnh

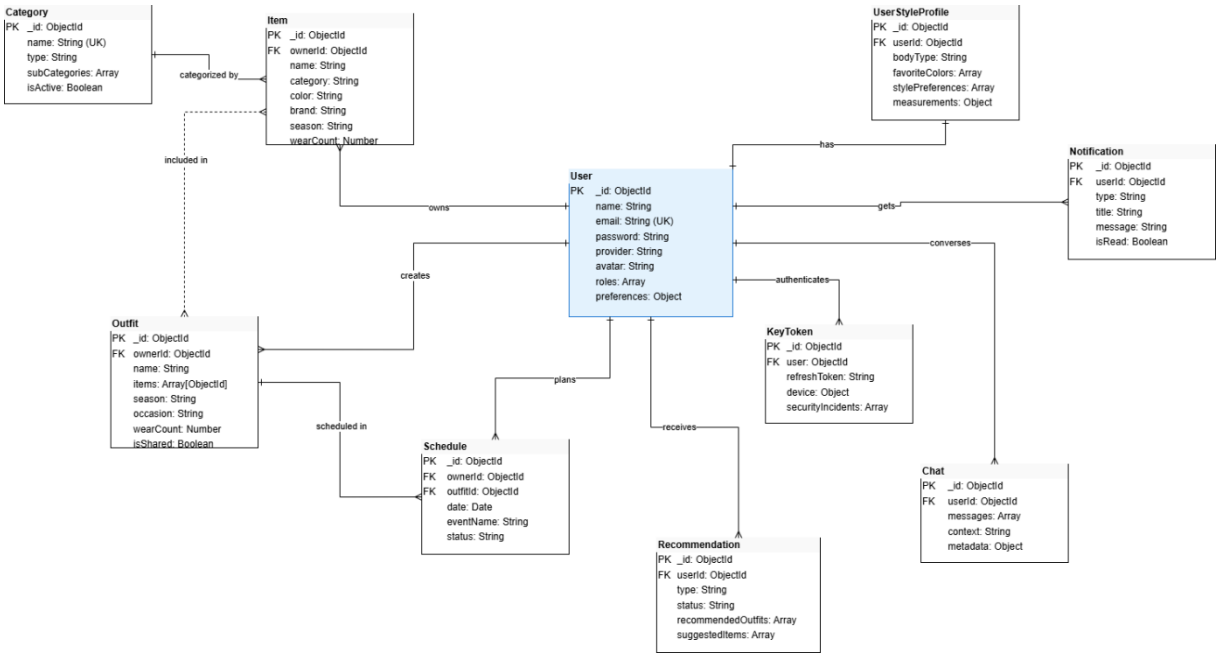
- Sơ đồ tuần tự chức năng Gợi ý trang phục



Hình 23 Sơ đồ tuần tự chức năng Gợi ý trang phục

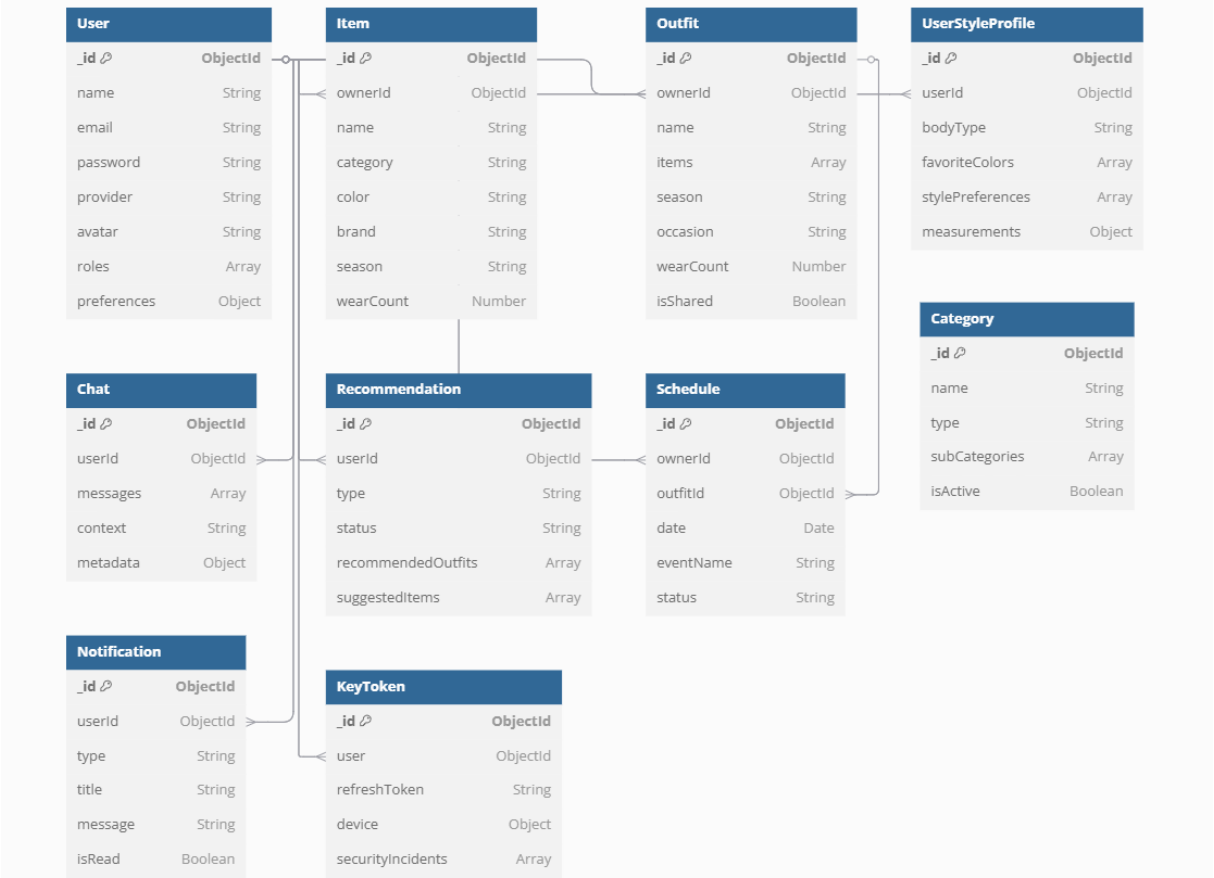
2.3. Thiết kế CSDL

2.3.1. Mô hình thực thể ERD



Hình 24 Mô hình thực thể ERD

2.3.2. Mô hình quan hệ



Hình 25 Mô hình quan hệ

- Để thuận tiện cho quá trình lập trình, các bảng sẽ được chuyển đổi tên và các trường sang tiếng anh.

2.4. Kết chương 2

Chương 2 đã cung cấp một cách tiếp cận toàn diện trong việc phân tích nghiệp vụ và thiết kế hệ thống cho dự án. Các sơ đồ ca sử dụng, và sơ đồ tuần tự không chỉ giúp làm rõ các chức năng của hệ thống mà còn đảm bảo rằng các quy trình nghiệp vụ. Bên cạnh đó, thiết kế cơ sở dữ liệu với các bảng quan hệ và mô hình chuyển đổi từ thực thể đã tạo ra một cấu trúc dữ liệu logic, phù hợp với các yêu cầu lưu trữ và xử lý thông tin. Những nền tảng thiết kế này là tiền đề quan trọng để triển khai hệ thống thực tế, đảm bảo chính xác, ổn định và khả năng mở rộng trong tương lai.

CHƯƠNG 3: TRIỂN KHAI VÀ ĐÁNH GIÁ

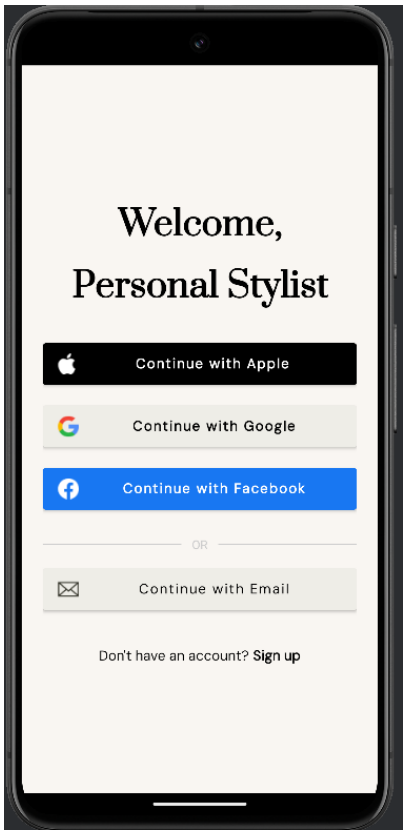
3.1. Môi trường và công cụ lập trình

- Backend: NodeJS sử dụng framework ExpressJS để tạo các RESTFull API.
- Frontend Android: Sử dụng Kotlin kết hợp XML để xây dựng giao diện hiện đại, dễ tùy biến.
- Frontend iOS: Sử dụng SwiftUI giúp khai thác tối đa khả năng native UI trên iOS.
- Cơ sở dữ liệu: MongoDB – cơ sở dữ liệu NoSQL, Firebase phù hợp cho lưu trữ linh hoạt các thông tin như trang phục, hình ảnh, thuộc tính...
- Công cụ hỗ trợ: Android Studio, Xcode, Cursor, Github.
- AI & Chatbot: Mô hình AI xử lý ảnh (tách nền, nhận diện loại trang phục) và chatbot gợi ý phối đồ dựa theo thời tiết, lịch sử, bối cảnh sử dụng.

3.2. Mô tả chức năng kết quả đã đạt được

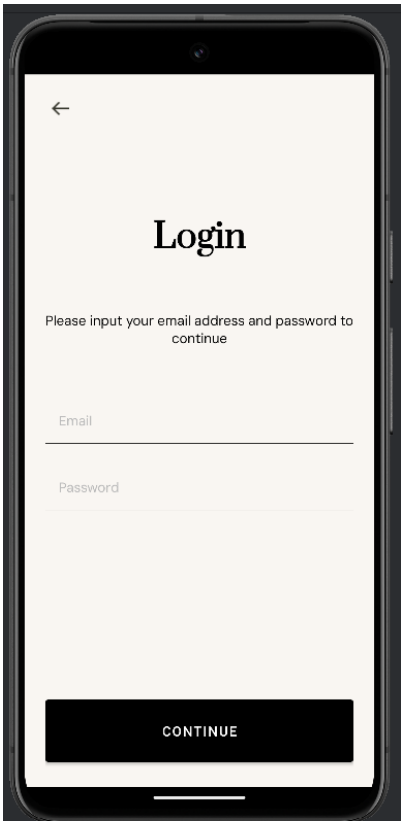
3.2.1. Giao diện chức năng Android

- Màn hình Welcome



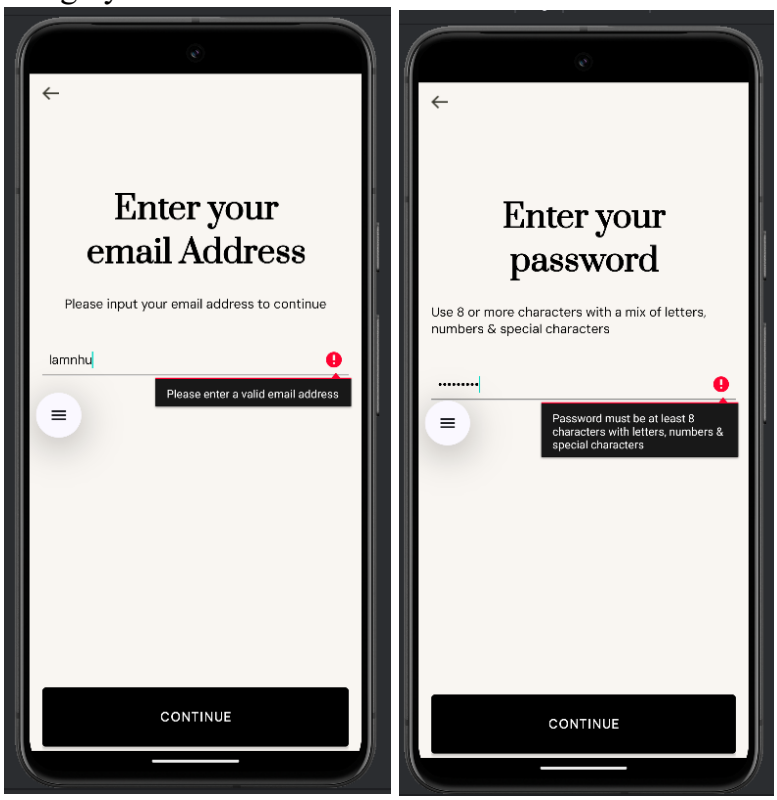
Hình 26 Màn hình Welcome (Android)

- Màn hình Đăng nhập

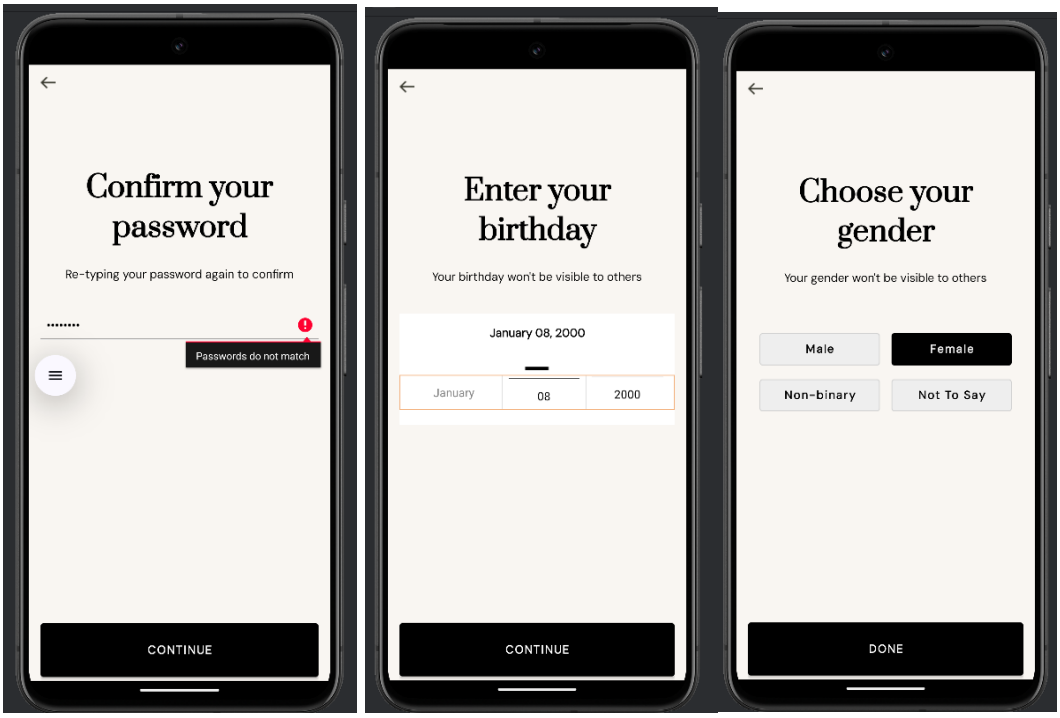


Hình 27 Màn hình Đăng nhập (Android)

- Màn hình Đăng ký

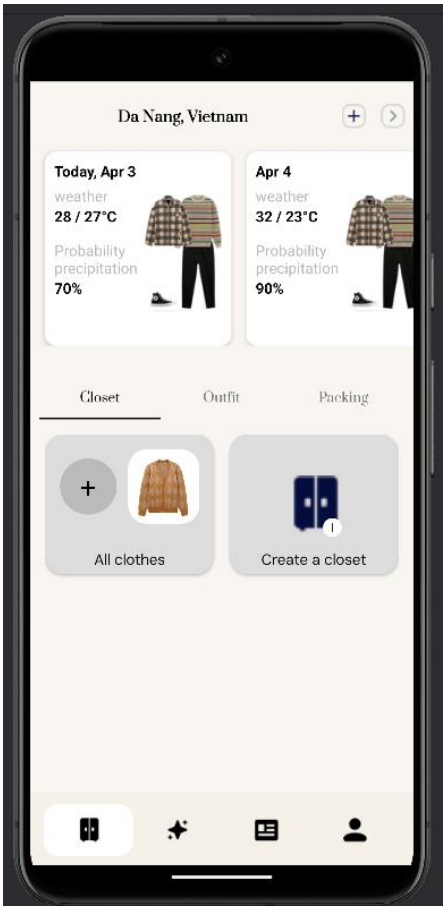


Hình 28 Màn hình Đăng ký – 1 (Android)



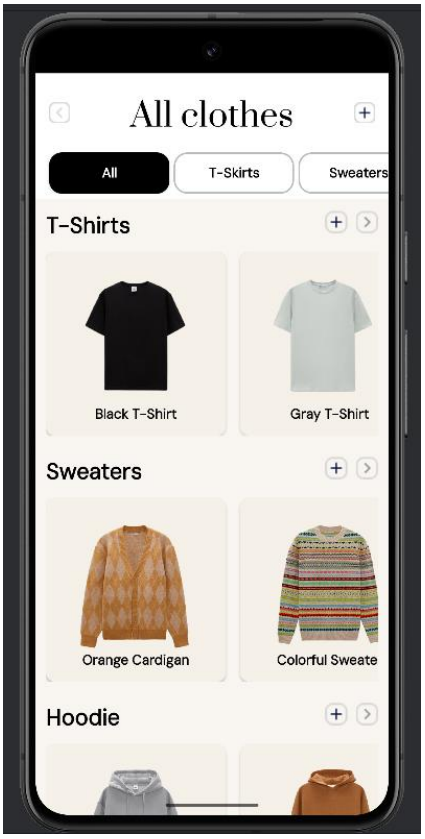
Hình 29 Màn hình Đăng ký – 2 (Android)

- Màn hình chính



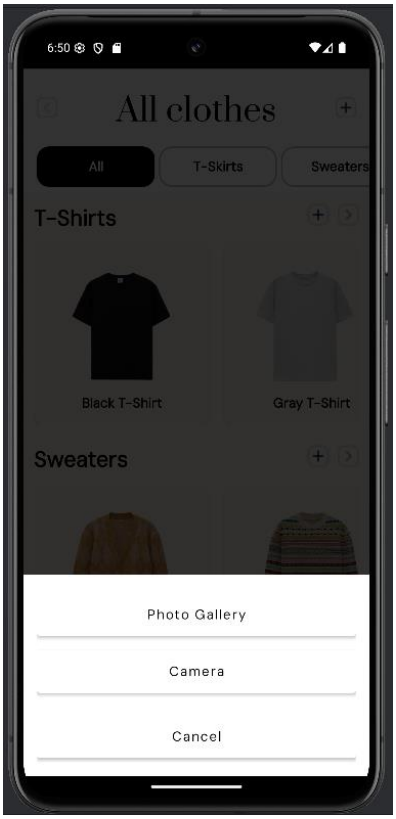
Hình 30 Màn hình chính (Android)

- Màn hình Quản lý tủ đồ

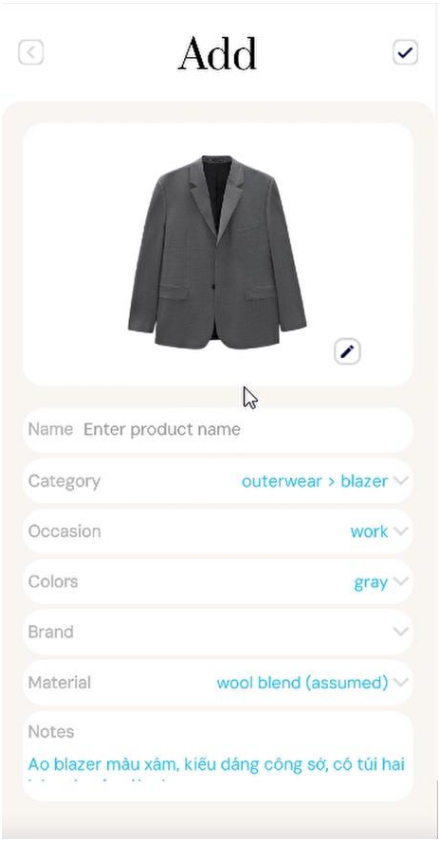


Hình 31 Màn hình Tủ đồ (Android)

- Màn hình Thêm item mới

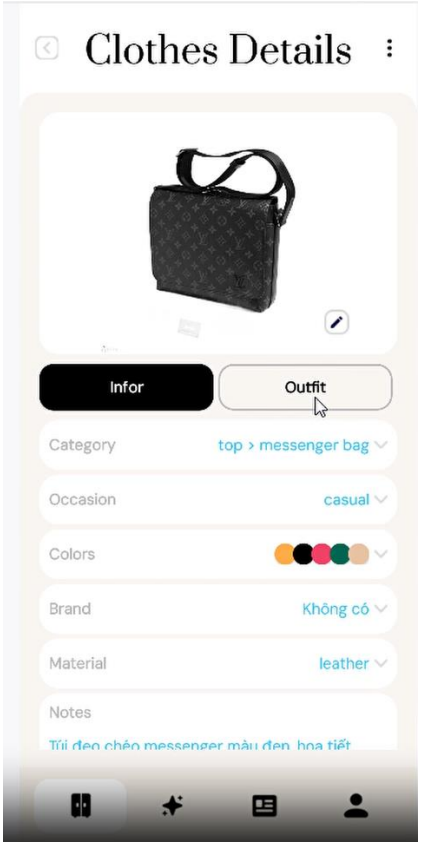


Hình 32 Màn hình Thêm item mới – 1 (Android)



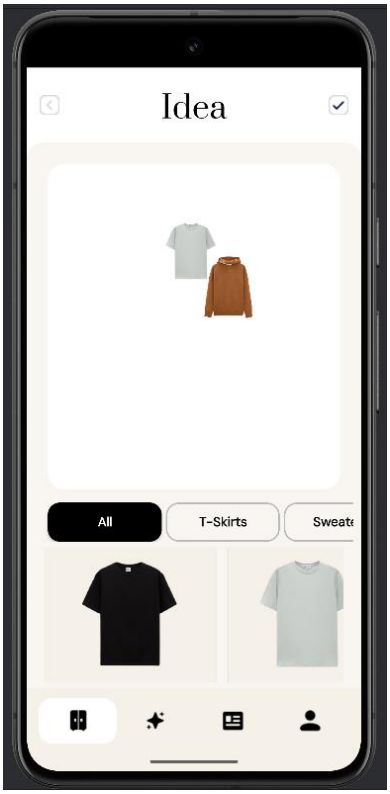
Hình 33 Màn hình them item mới – 2 (Android)

- Màn hình Chi tiết item



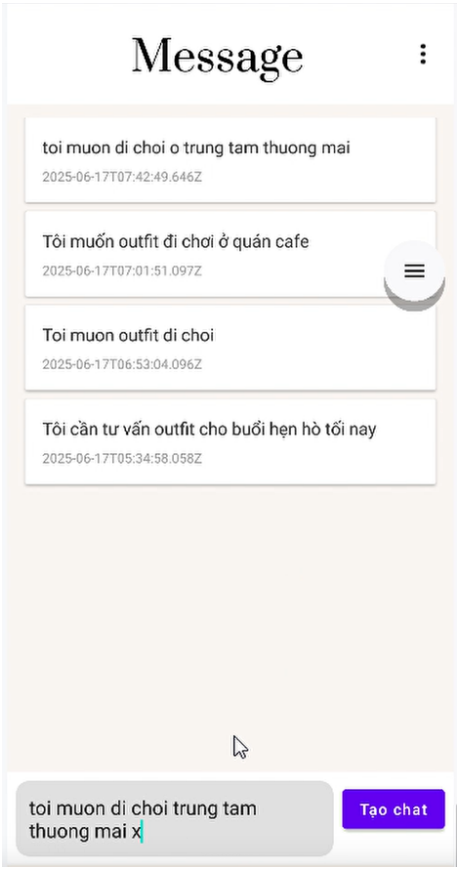
Hình 34 Màn hình Chi tiết item (Android)

- Màn hình Tạo Outfit



Hình 35 Màn hình tạo Outfit (Android)

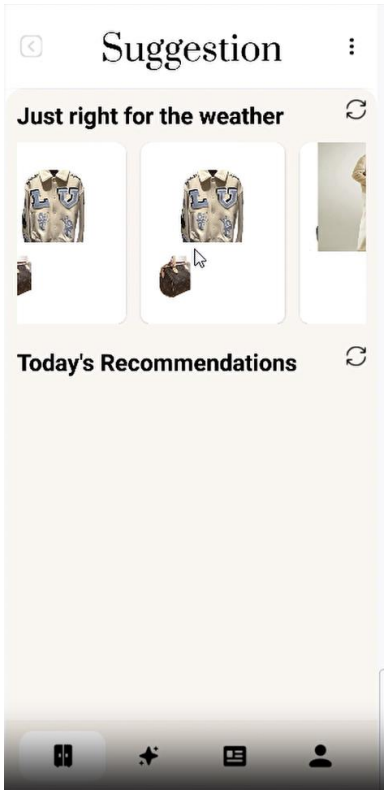
- Màn hình Chatbot



Hình 36 Chatbot - 1(Android)

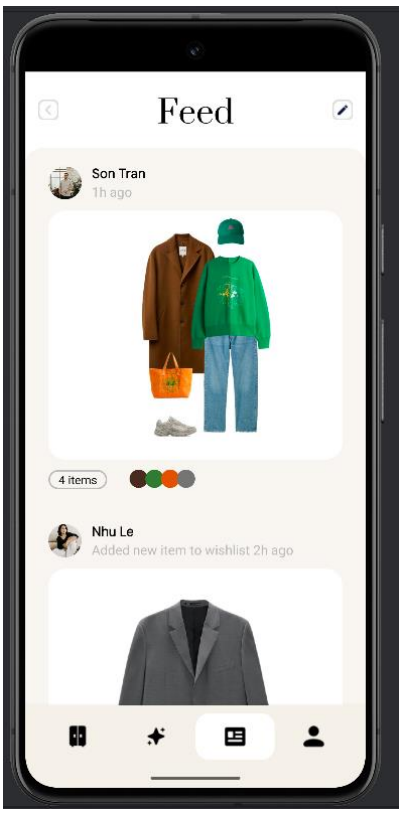
Hình 37 Chatbot - 2 (Android)

- Màn hình Gợi ý trang phục



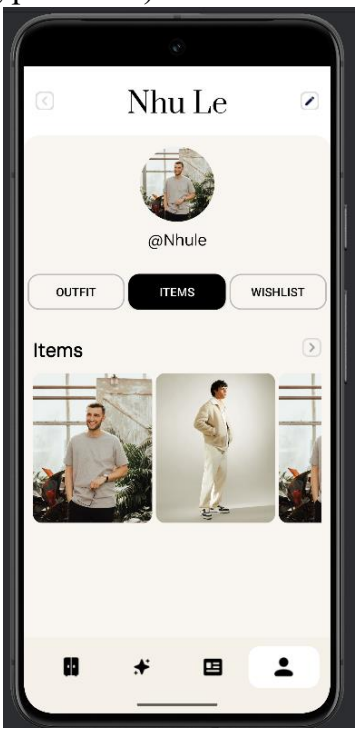
Hình 38 Recommendation (Android)

- Màn hình Feed (Hướng phát triển)



Hình 39 Feed (Android)

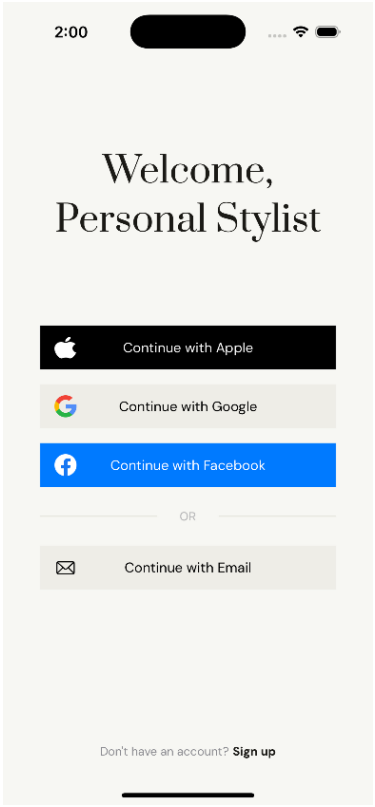
- Màn hình Profile (Hướng phát triển)



Hình 40 Profile (Android)

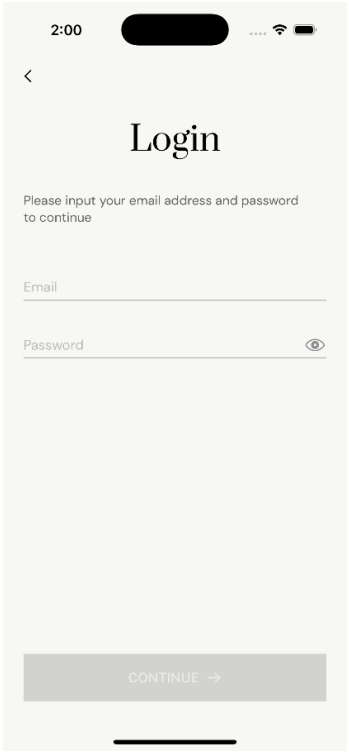
3.2.2. Giao diện chức năng IOS

- Màn hình Welcome



Hình 41 Màn hình Welcome (IOS)

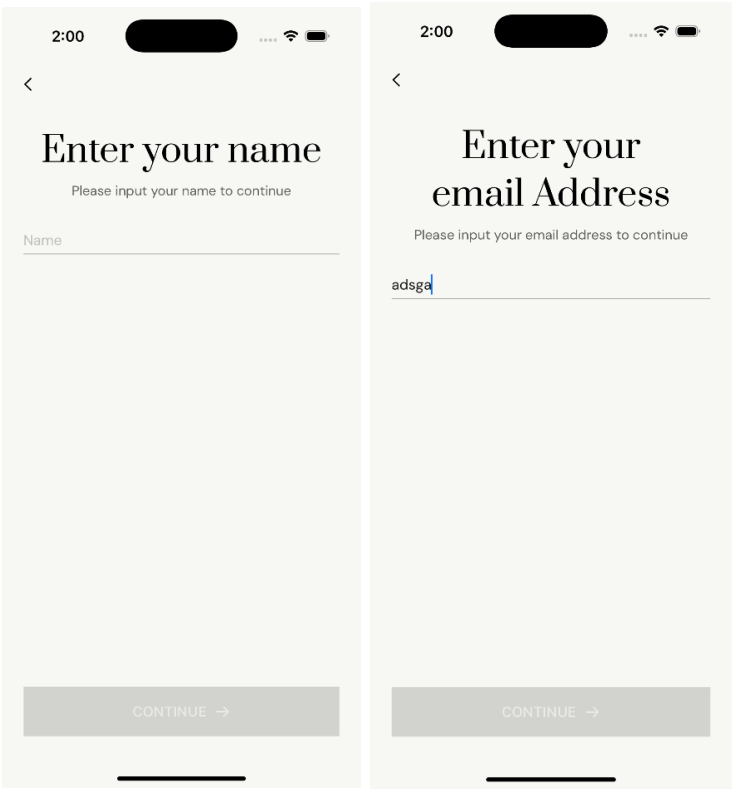
- Màn hình Đăng nhập



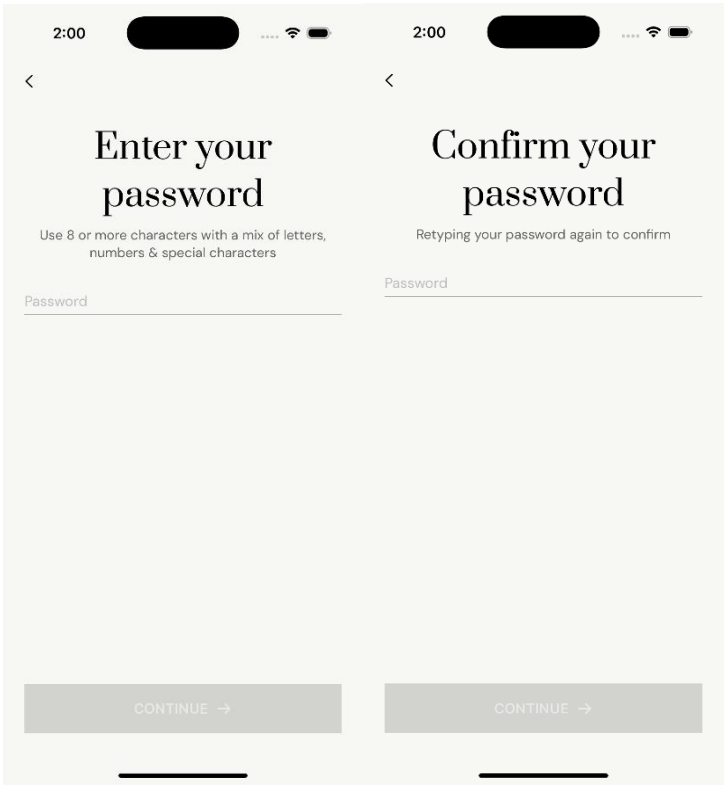
-

Hình 42 Màn hình Đăng nhập (IOS)

- Màn hình Đăng ký

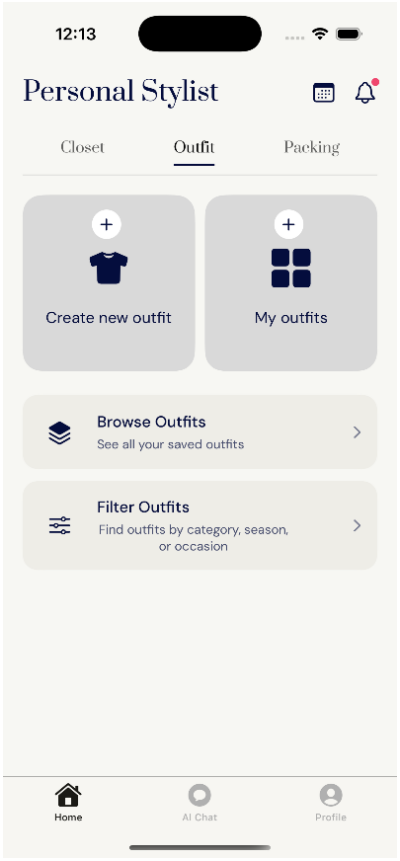


Hình 43 Màn hình Đăng ký – 1 (IOS)



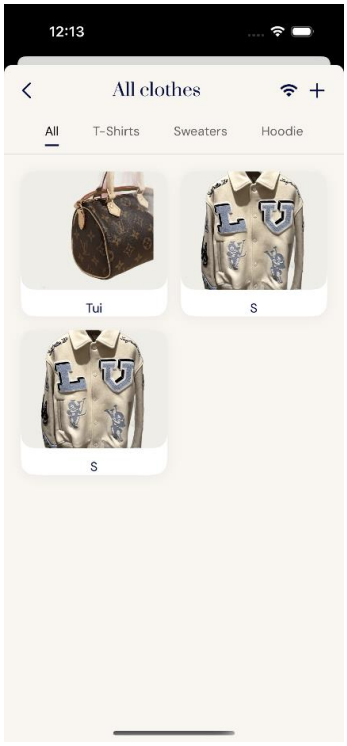
Hình 44 Màn hình Đăng ký – 2 (IOS)

- Màn hình chính



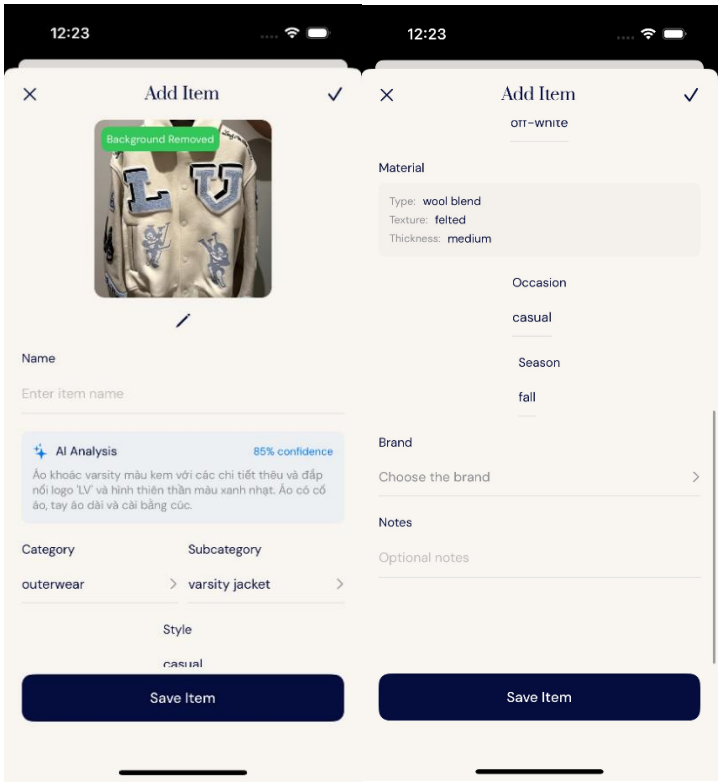
Hình 45 Màn hình chính (IOS)

- Màn hình Quản lý tủ đồ



Hình 46 Màn hình Tủ đồ (IOS)

- Màn hình Thêm item mới

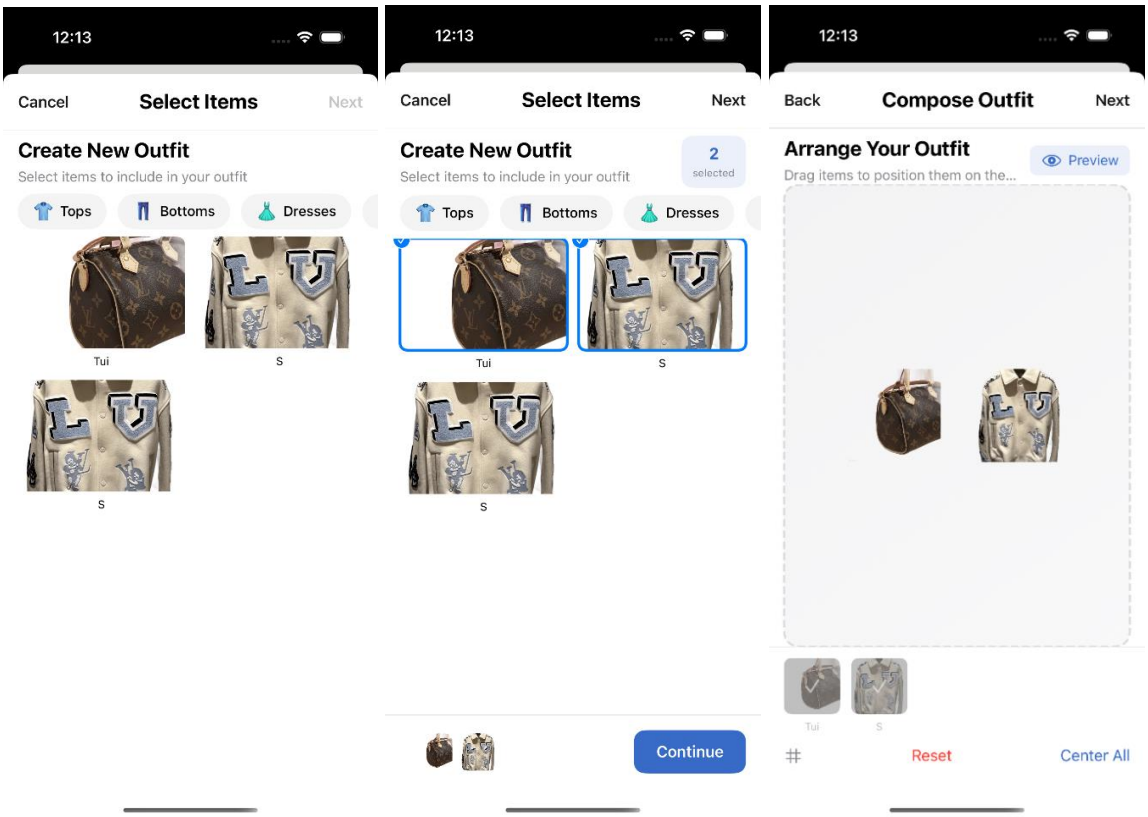


Hình 47 Màn hình Thêm item mới (IOS)

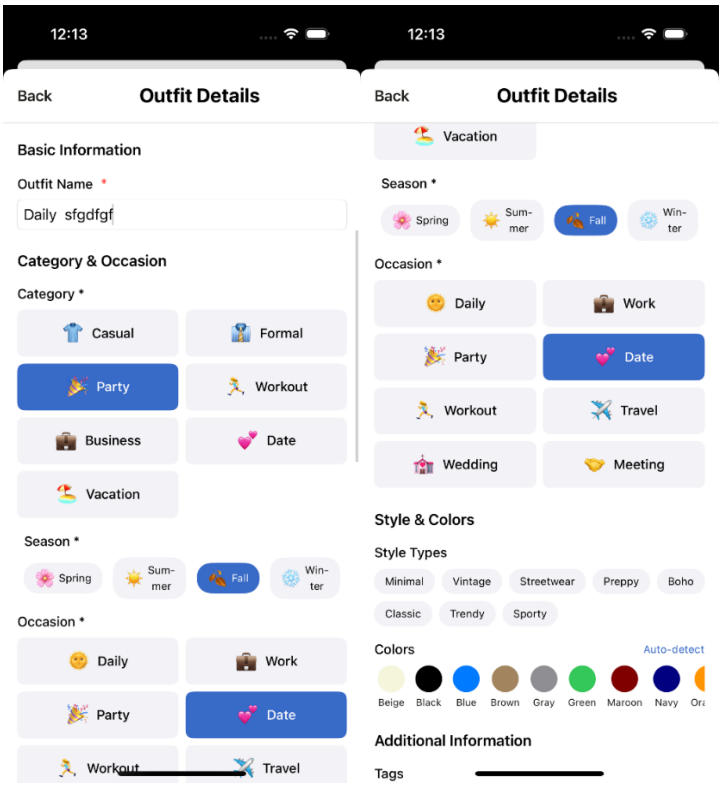
- Màn hình Chi tiết item

Hình 48 Màn hình Chi tiết item (IOS)

- Màn hình Tạo Outfit

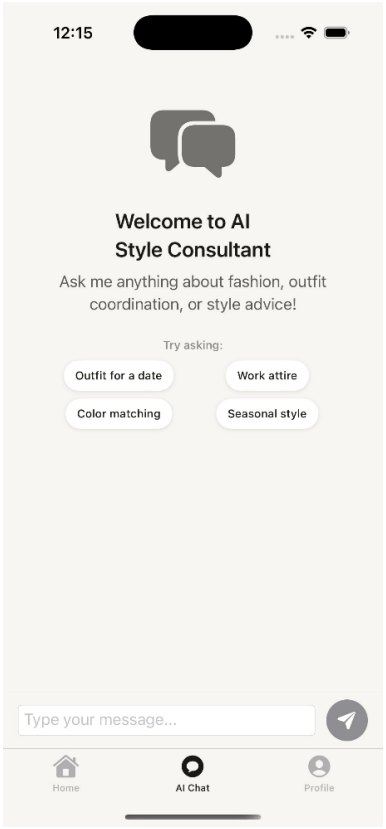


Hình 49 Màn hình tạo Outfit - 1 (IOS)

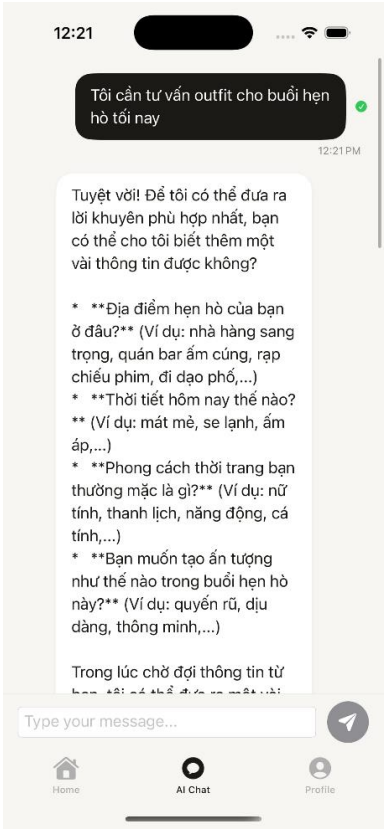


Hình 50 Màn hình tạo Outfit – 2 (IOS)

- Màn hình Chatbot

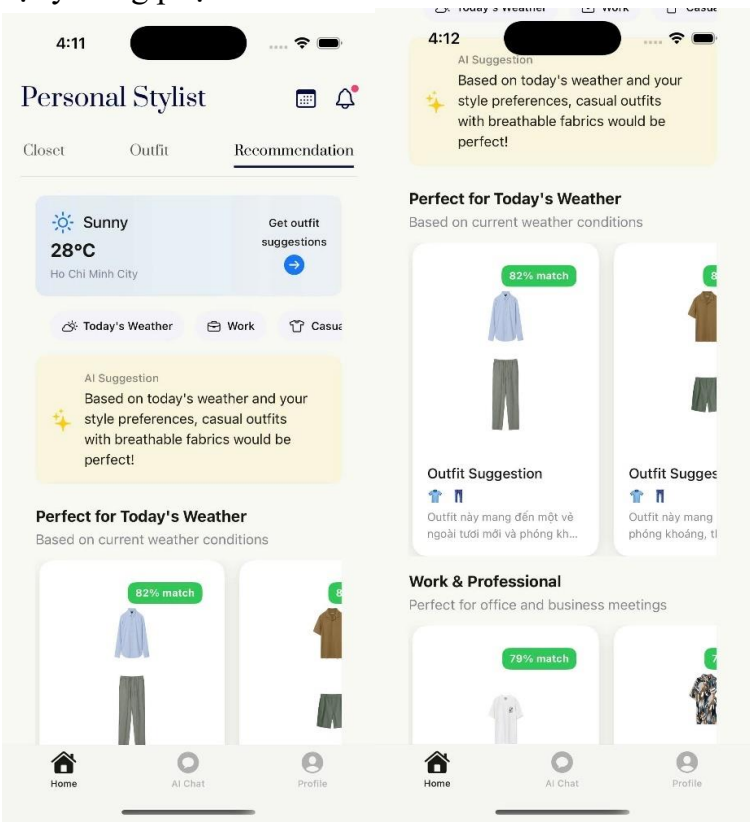


Hình 51 Chatbot - 1 (IOS)



Hình 52 Chatbot - 2 (IOS)

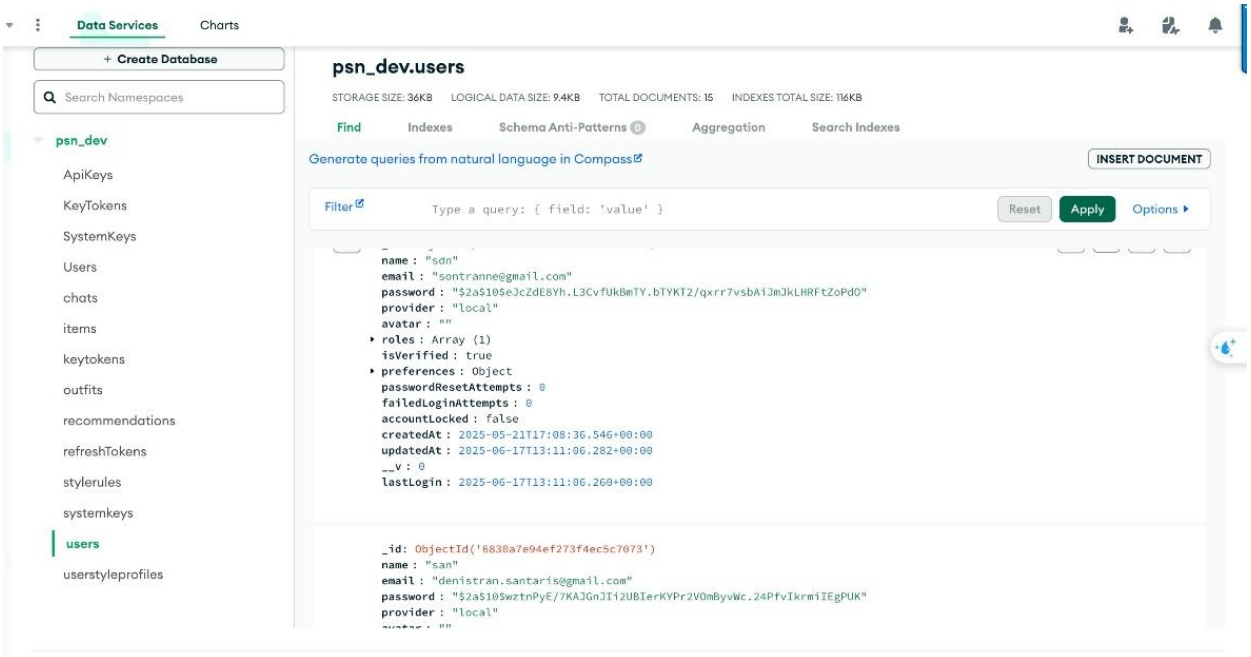
- Màn hình Gợi ý trang phục



Hình 53 Recommendation (IOS)

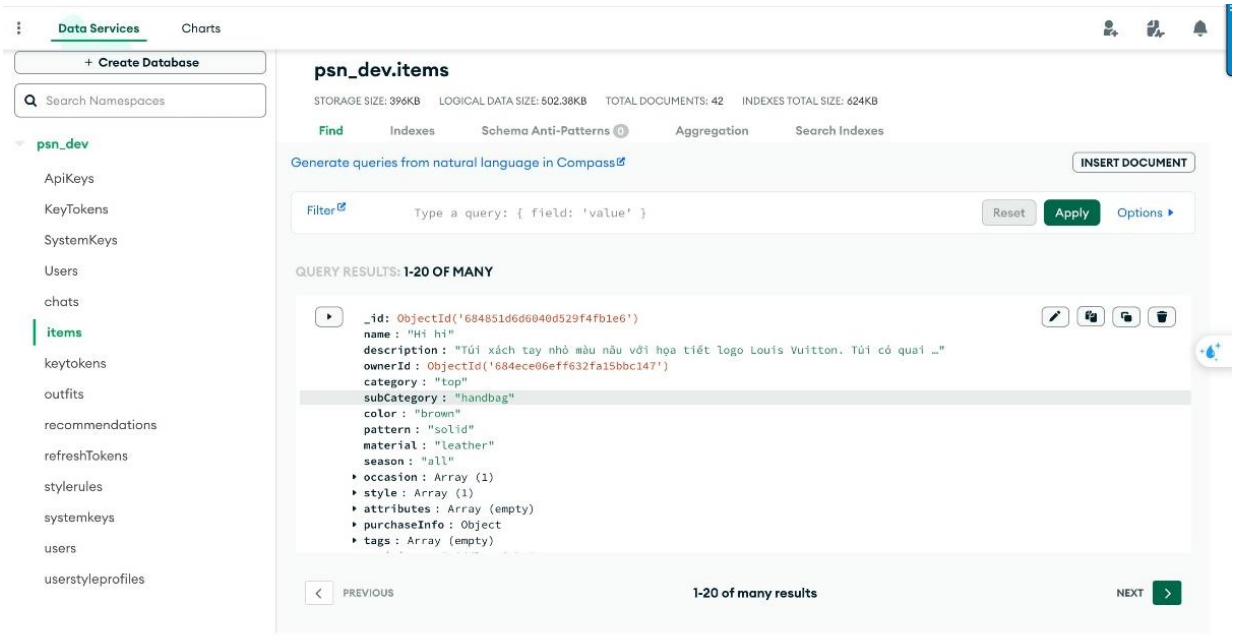
3.2.3. Quản lý dữ liệu trên MongoDB

- Quản lý người dùng



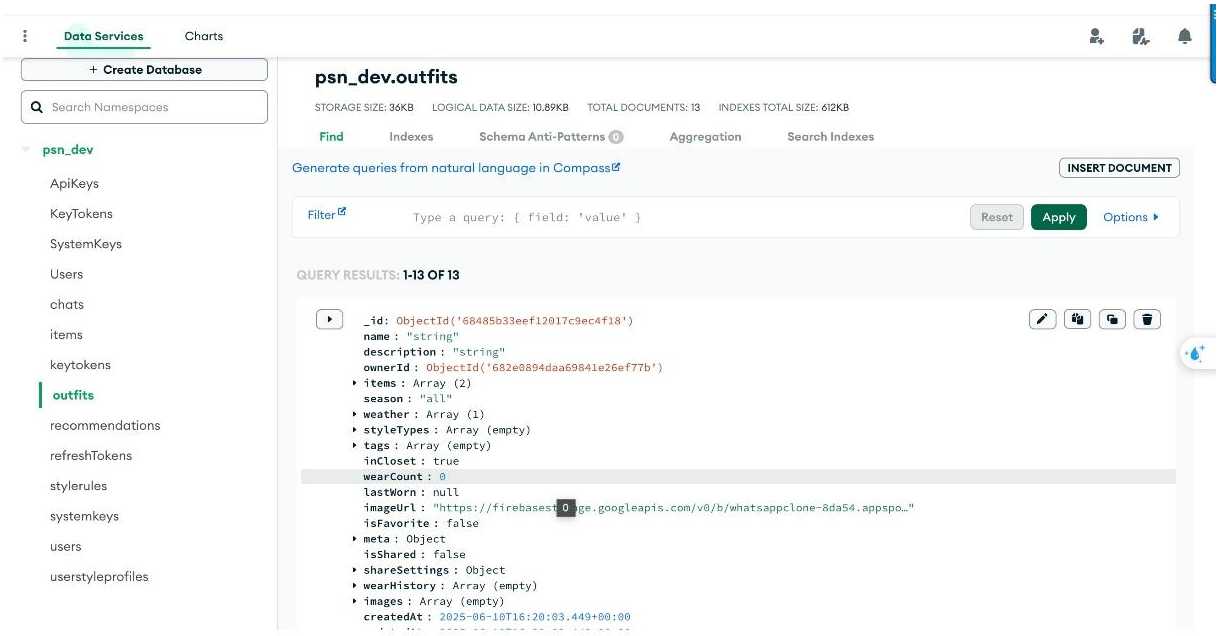
Hình 54 Quản lý người dùng

- Quản lý trang phục



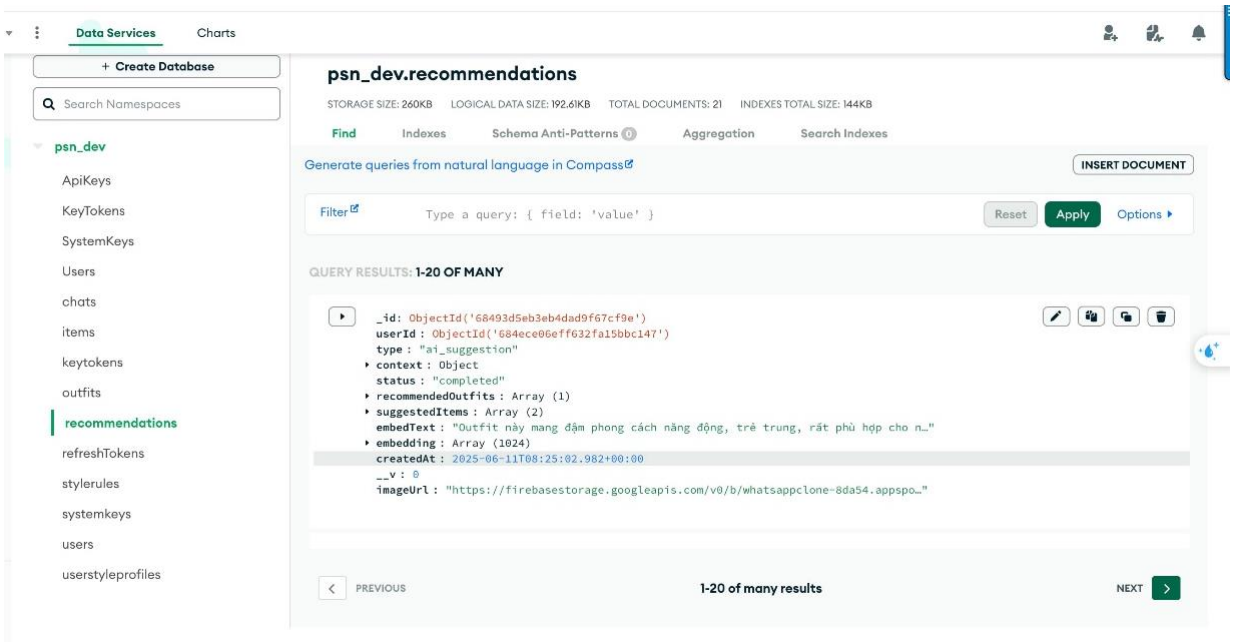
Hình 55 Quản lý trang phục

- Quản lý Outfit



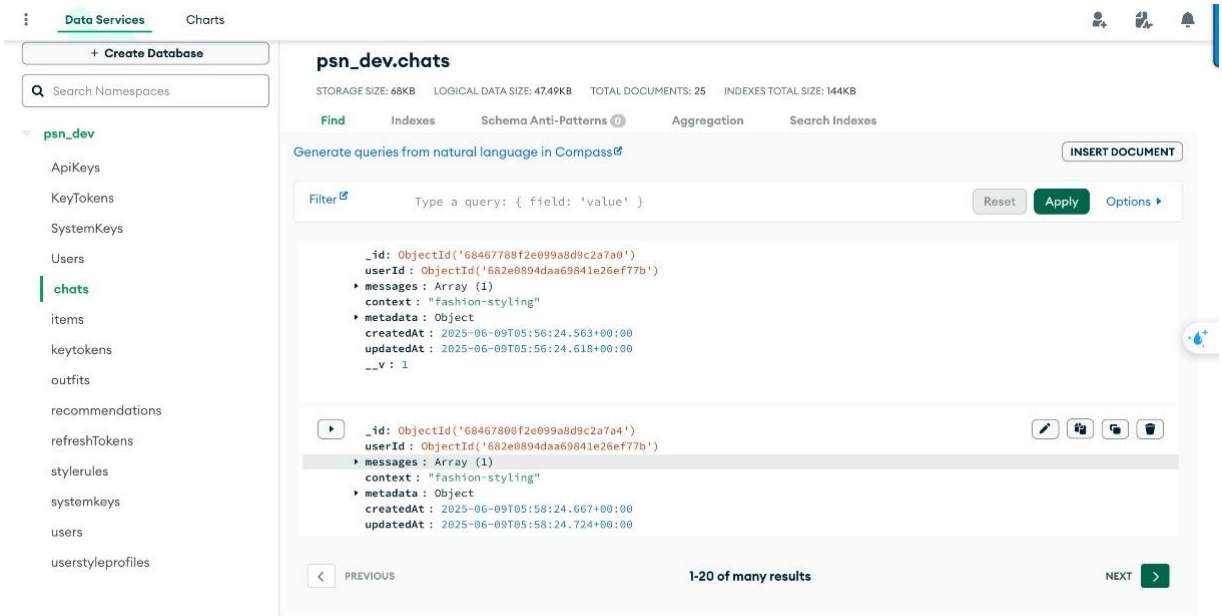
Hình 56 Quản lý outfit

- Quản lý Gợi ý



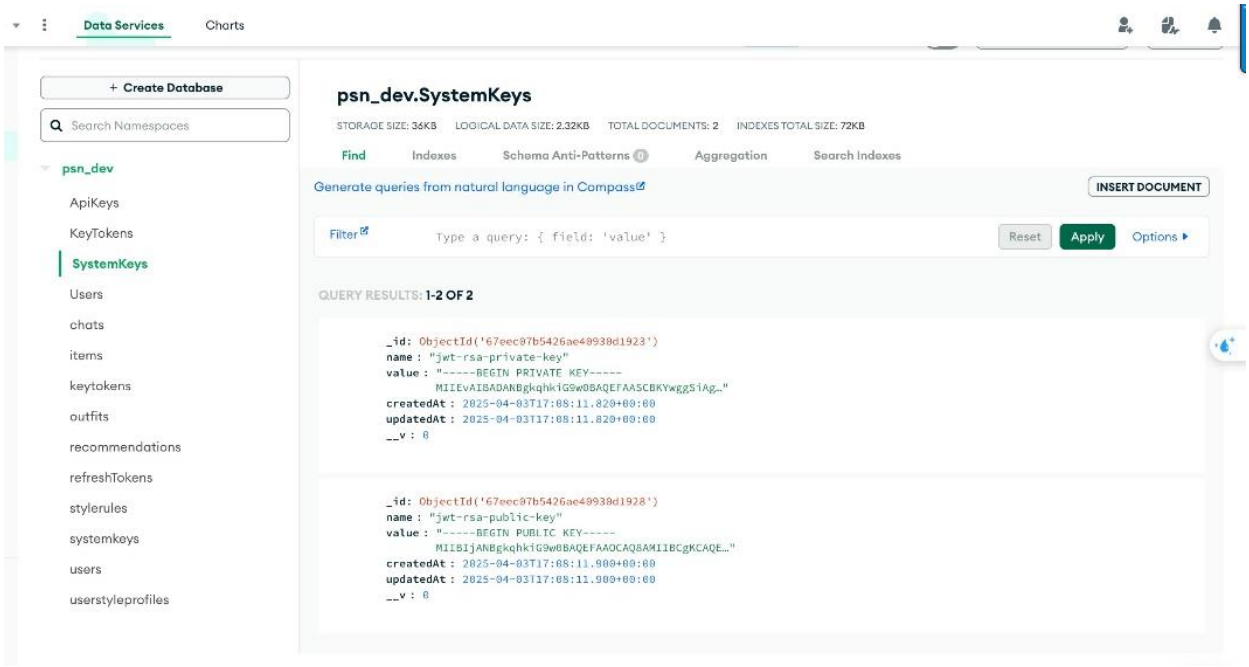
Hình 57 Quản lý gợi ý

- Quản lý chat



Hình 58 Quản lý chat

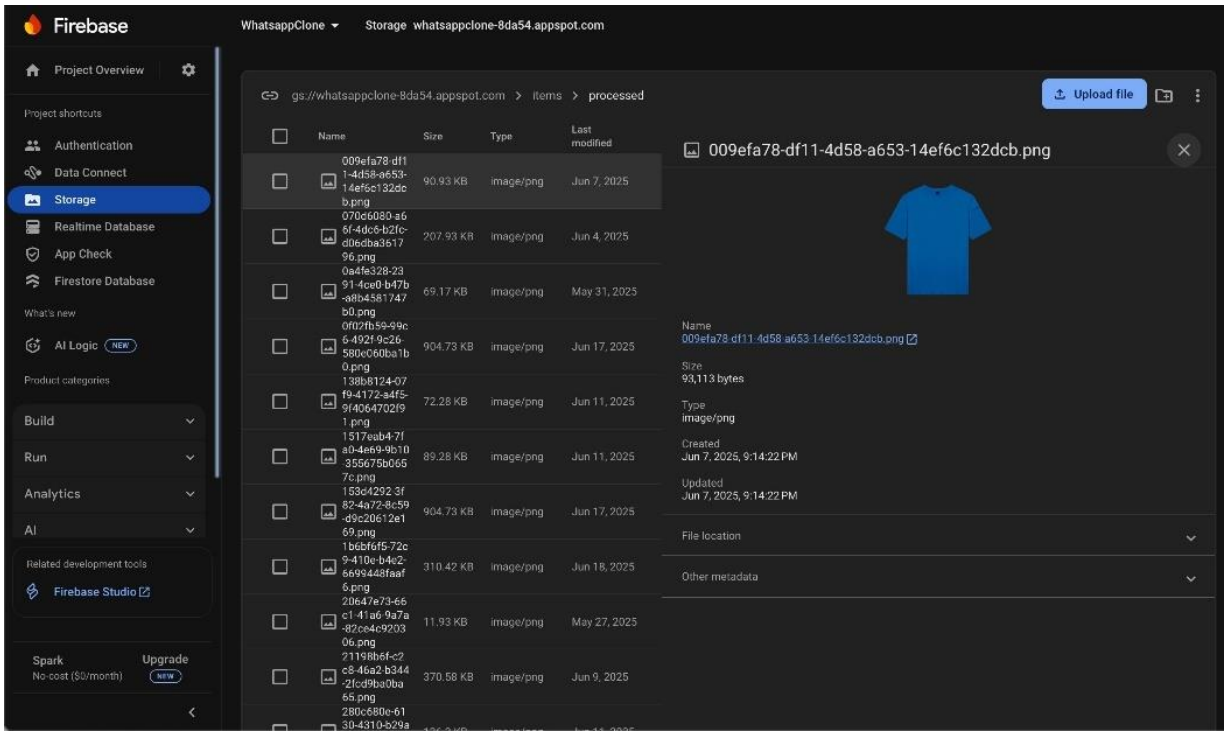
- Quản lý Key của hệ thống



Hình 59 Quản lý Key của hệ thống

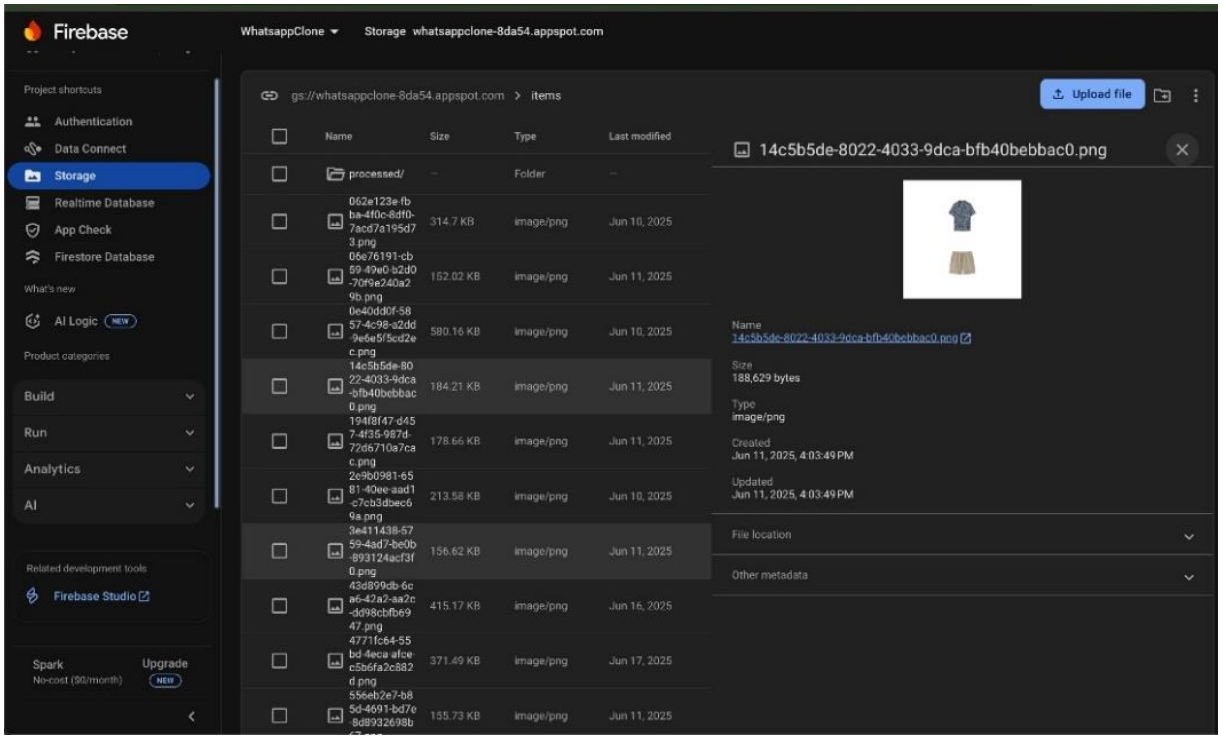
3.2.4. Quản lý hình ảnh trên Firebase Cloud Storage

- Quản lý hình ảnh trang phục sau khi tách nền



Hình 60 Quản lý hình ảnh trang phục

- Quản lý outfit



Hình 61 Quản lý hình ảnh outfit

3.3. Đánh giá kết quả

- Hoàn thiện cơ bản ứng dụng
- Giao diện được thiết kế gần giống với Figma
- Tích hợp thành công AI xử lý ảnh (tách nền), sinh metadata (Gemini) và chatbot thời trang (OpenAI GPT-4).
- Backend vận hành ổn định với kiến trúc RESTful, MongoDB và Firebase.

3.4. Kết chương 3

- Trong chương 3, quá trình xây dựng và hiện thực hóa hệ thống đã được trình bày chi tiết, từ môi trường và công cụ lập trình cho đến các giao diện và chức năng chính.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. Kết quả đạt được

Trong quá trình nghiên cứu tìm hiểu kiến thức, lý thuyết và triển khai ứng dụng đồ án đã đạt được những kết quả sau:

- **Về mặt lý thuyết:**
    - Trong quá trình tìm hiểu và thực thi dự án đã giúp em nắm vững và áp dụng được những kiến thức cơ bản để xây dựng nên một ứng dụng
    - Biết các phân tích nghiệp vụ và hệ thống.
- **Về mặt ứng dụng:**
    - Đã xây dựng và triển khai thành công ứng dụng thời trang. Hoàn thiện hệ thống ứng dụng quản lý tủ đồ có khả năng quản lý tủ đồ và tư vấn outfit.
    - Tích hợp thành công trí tuệ nhân tạo vào ứng dụng

2. Hạn chế

Ngoài những kết quả đã đạt được trong quá trình thực hiện đề tài. Hệ thống còn một số hạn chế như:

- Chưa tích hợp tính năng "ma-nơ-canh ảo" như mô tả trong đề cương.
- Chưa có đánh giá từ người dùng thực tế để kiểm chứng trải nghiệm.
- Giao diện Android và iOS chưa đồng nhất

Vì thời gian có hạn, kinh nghiệm thực tế chưa nhiều nên việc phân tích bài toán về cơ bản đã thực hiện tương đối đầy đủ, tuy nhiên chưa mô tả đầy đủ mọi khía cạnh của vấn đề.

3. Hướng phát triển

- Hoàn thiện tính năng: Calendar (OOTD), Feed, bổ sung các chức năng nâng cao như tạo ma-nơ-canh ảo với thông số cá nhân hóa
- Mở rộng quy mô: Phát hành phiên bản chính thức trên App Store và Google Play
- Cải thiện hiệu quả vận hành: Tối ưu hóa thuật toán xử lý ảnh và phản hồi của chatbot để rút ngắn thời gian xử lý. Nâng cấp hệ thống backend để đáp ứng số lượng lớn người dùng đồng thời.
- Nghiên cứu và phát triển thêm: Ứng dụng mô hình học sâu (deep learning) để cá

nhân hóa trải nghiệm người dùng tốt hơn, như gợi ý theo lịch sử phối đồ, xu hướng thời trang theo mùa hoặc phân tích dữ liệu từ mạng xã hội. Bên cạnh đó, có thể kết hợp AI với công nghệ AR (Augmented Reality) để hỗ trợ người dùng thử đồ ảo một cách trực quan hơn.

TÀI LIỆU THAM KHẢO

[1] G. Sun and J. Zhang, "Clothing style recognition using deep learning for personalized fashion recommendation," IEEE Access, vol. 8, pp. 123456–123467, 2020.

[2] X. Liu, Z. Song, and T. Zhang, "A survey on fashion recommendation: Challenges and techniques," in Proc. IEEE Int. Conf. Big Data, Seattle, WA, USA, Dec. 2018, pp. 2233–2242.

[3] Google. "Jetpack Compose – Modern toolkit for building native UI," Android Developers, 2023. [Online]. Available: https://developer.android.com/jetpack/compose

[4] OpenAI, "GPT-4 Technical Report," arXiv preprint arXiv:2303.08774, 2023. [Online]. Available: https://arxiv.org/abs/2303.08774

[5] Y. Kalantidis, M. Kennedy, and L. Li, "Getting the look: Clothing recognition and segmentation for automatic product tagging," in Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV), 2013, pp. 1–8.

[6] N. Liu, "Designing for Fashion UX: Personalization and User Engagement," Interaction Design Foundation, 2022. [Online]. Available: https://www.interaction-design.org/literature/topics/fashion-ux

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[8] M. Sandler et al., "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Salt Lake City, UT, USA, 2018, pp. 4510–4520.

PHỤ LỤC