TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**PHẠM ĐỨC MINH HIẾU – 52100796**

**PHAN HOÀNG PHÚ – 52100086**

**PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG DẠY VÀ HỌC TRỰC TUYẾN ĐA NỀN TẢNG VỚI FLUTTER**

**DỰ ÁN CÔNG NGHỆ THÔNG TIN**

**KỸ THUẬT PHẦN MỀM**

**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**



**PHẠM ĐỨC MINH HIẾU – 52100796**

**PHAN HOÀNG PHÚ – 52100086**

**PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG DẠY VÀ HỌC TRỰC TUYẾN ĐA NỀN TẢNG VỚI FLUTTER**

**DỰ ÁN CÔNG NGHỆ THÔNG TIN**

**KỸ THUẬT PHẦN MỀM**

Người hướng dẫn

**ThS. Mai Văn Mạnh**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

**LỜI CẢM ƠN**

Lời đầu tiên chúng em xin được phép cảm ơn quý thầy cô khoa Công nghệ Thông Tin trường Đại học Tôn Đức Thắng đã tạo mọi điều kiện cho chúng em tạo mọi điều kiện tốt nhất để được thực hiện môn học Dự án công nghệ thông tin.

Chúng em cũng xin được gửi lời cảm ơn đến ThS Mai Văn Mạnh, thầy đã nhiệt tình giảng dạy, trang bị đầy đủ kiến thức để em có thể hoàn thành môn dự án công nghệ thông tin này.

Cuối cùng, do hạn chế về mặt kiến thức, kính mong thầy cô có thể bỏ qua những sai sót nhỏ và chỉ ra được những lỗi sai của chúng em trong bài báo cáo này để những bài báo cáo sau của chúng em được hoàn thiện hơn.

Một lần nữa chúng em xin chân thành cảm ơn thầy và toàn thể quý thầy cô khoa Công Nghệ Thông Tin trường Đại học Tôn Đức Thắng.

*TP. Hồ Chí Minh, ngày 26 tháng 07 năm 2024*

*Tác giả*

*Phạm Đức Minh Hiếu*

*Phan Hoàng Phú*

**PHIẾU ĐÁNH GIÁ CỦA GIẢNG VIÊN HƯỚNG DẪN**

Tên giảng viên hướng dẫn:

Ý kiến nhận xét:

Điểm tổng theo phiếu đánh giá rubrik:

*TP. Hồ Chí Minh, ngày tháng năm 20*

*Giảng viên hướng dẫn*

*(Ký tên và ghi rõ họ tên)*

**CÔNG TRÌNH ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Chúng em xin cam đoan đây là công trình nghiên cứu của riêng chúng em và được sự hướng dẫn khoa học của ThS.Mai Văn Mạnh. 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 Dự 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 chúng em xin hoàn toàn chịu trách nhiệm về nội dung Dự án của mình**. Trường Đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do chúng em gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 26 tháng 07 năm 2024*

*Tác giả*

*Phạm Đức Minh Hiếu*

*Phan Hoàng Phú*

**PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG DẠY VÀ HỌC TRỰC TUYẾN ĐA NỀN TẢNG VỚI FLUTTER**

**TÓM TẮT**

Bài báo cáo này trình bày về quá trình tìm hiểu về việc xây dựng một ứng dụng di động đa nền tảng với Flutter. Mô tả về các công nghệ sử dụng trong dự án, tìm hiểu, thiết kế và xây dựng một ứng dụng dạy và học trực tuyến. Bài báo cáo bao gồm 6 chương:

Chương 1: Trình bày về lý do lựa chọn đề tài, nội dung nghiên cứu và mục tiêu mong muốn hướng đến.

Chương 2: Trình bày về cơ sở lý thuyết, các công nghệ được sử dụng trong dự án, nói về thiết kế hệ thống xây dựng theo mô hình RESTful API – một mô hình phổ biến hiện nay.

Chương 3: Phân tích, thiết kế hệ thống thông qua mô hình quan hệ của hệ thống và các sơ đồ như sơ đồ lớp, sơ đồ use case và sơ đồ tuần tự.

Chương 4: Thiết kế các giao diện cần thiết cho hệ thống.

Chương 5: Các bước xây dựng hệ thống và triển khai lên thực tế.

Chương 6: Kết luận về dự án, những khó khăn còn gặp phải và hướng đi trong tương lai.

**MỤC LỤC**

[DANH MỤC HÌNH VẼ x](#_Toc172986169)

[DANH MỤC BẢNG BIỂU xii](#_Toc172986170)

[DANH MỤC CÁC CHỮ VIẾT TẮT xiv](#_Toc172986171)

[CHƯƠNG 1. MỞ ĐẦU 1](#_Toc172986172)

[1.1 Lý do chọn đề tài 1](#_Toc172986173)

[1.2 Mục tiêu thực hiện đề tài 2](#_Toc172986174)

[1.3 Đối tượng và phạm vi nghiên cứu 2](#_Toc172986175)

[1.3.1 Đối tượng nghiên cứu 2](#_Toc172986176)

[1.3.2 Phạm vi môi trường 2](#_Toc172986177)

[1.4 Phương pháp nghiên cứu 3](#_Toc172986178)

[1.5 Ý nghĩa thực tiễn của đề tài 4](#_Toc172986179)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 5](#_Toc172986180)

[2.1 Flutter 5](#_Toc172986181)

[2.2 NodeJS và ExpressJS 6](#_Toc172986182)

[2.2.1 NodeJS 6](#_Toc172986183)

[2.2.2 ExpressJS 6](#_Toc172986184)

[2.3 ReactJS 7](#_Toc172986185)

[2.4 Firebase 9](#_Toc172986186)

[2.4.1 Firebase Authentication 10](#_Toc172986187)

[2.4.2 Cloud Firestore 10](#_Toc172986188)

[2.5 Google Cloud Stogre 11](#_Toc172986189)

[2.6 Mô hình RESTful API 13](#_Toc172986190)

[2.7 Stripe 14](#_Toc172986191)

[2.8 JWT 15](#_Toc172986192)

[2.9 Vite 17](#_Toc172986193)

[2.10 Tailwind CSS 18](#_Toc172986194)

[2.11 Render 20](#_Toc172986195)

[CHƯƠNG 3. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 22](#_Toc172986196)

[3.1 Khảo sát 22](#_Toc172986197)

[3.2 Use case Diagram 22](#_Toc172986198)

[3.3 Đặc tả các Use case của hệ thống 23](#_Toc172986199)

[3.3.1 Use case đăng nhập 23](#_Toc172986200)

[3.3.2 Use case đăng ký 24](#_Toc172986201)

[3.3.3 Use case đăng xuất 25](#_Toc172986202)

[3.3.4 Use case quên mật khẩu 26](#_Toc172986203)

[3.3.5 Use case đăng nhập bằng Google 28](#_Toc172986204)

[3.3.6 Use case quản lý thông tin cá nhân 29](#_Toc172986205)

[3.3.7 Use case chọn chủ đề học yêu thích 30](#_Toc172986206)

[3.3.8 Use case lưu khóa học yêu thích 31](#_Toc172986207)

[3.3.9 Use case xem chi tiết thông tin của giảng viên 32](#_Toc172986208)

[3.3.10 Use case ghi danh khóa học 33](#_Toc172986209)

[3.3.11 Use case tìm kiếm khóa học 35](#_Toc172986210)

[3.3.12 Use case xem các bài học trong khóa học 36](#_Toc172986211)

[3.3.13 Use case ghi chú trong khóa học 37](#_Toc172986212)

[3.3.14 Use case nộp dự án khóa học 40](#_Toc172986213)

[3.3.15 Use case làm các bài kiểm tra 42](#_Toc172986214)

[3.3.16 Use case thảo luận trên diễn đàn 43](#_Toc172986215)

[3.3.17 Use case thêm khóa học 45](#_Toc172986216)

[3.3.18 Use case thêm khóa học 47](#_Toc172986217)

[3.3.19 Use case lịch sử xem 48](#_Toc172986218)

[3.3.20 Use case quản lý thư mục 49](#_Toc172986219)

[3.3.21 Use case gửi yêu cầu trở thành giảng viên 51](#_Toc172986220)

[3.3.22 Use case xem các yêu cầu trở thành giảng viên 52](#_Toc172986221)

[3.3.23 Use case phê duyệt yêu cầu trở thành giảng viên 53](#_Toc172986222)

[3.3.24 Use case từ chối yêu cầu trở thành giảng viên 54](#_Toc172986223)

[3.3.25 Use case thống kê 55](#_Toc172986224)

[3.3.26 Use case quản lý khóa học 57](#_Toc172986225)

[3.3.27 Use case phê duyệt khóa học 58](#_Toc172986226)

[3.3.28 Use case từ chối yêu cầu duyệt khóa học 58](#_Toc172986227)

[3.3.29 Use case quản lý người dùng 59](#_Toc172986228)

[3.3.30 Use case đăng nhập admin 60](#_Toc172986229)

[3.3.31 Use case đăng xuất admin 62](#_Toc172986230)

[3.4 Mô hình quan hệ 63](#_Toc172986231)

[3.5 Sơ đồ lớp 64](#_Toc172986232)

[3.6 Sơ đồ tuần tự 65](#_Toc172986233)

[3.6.1 Đăng nhập 65](#_Toc172986234)

[3.6.2 Đăng nhập với Google 66](#_Toc172986235)

[3.6.3 Đặt lại mật khẩu 67](#_Toc172986236)

[3.6.4 Tìm kiếm khóa học 67](#_Toc172986237)

[3.6.5 Quản lý thư mục 68](#_Toc172986238)

[3.6.6 Đăng ký 70](#_Toc172986239)

[3.6.7 Lịch sử xem video 70](#_Toc172986240)

[3.6.8 Lưu khóa học yêu thích 71](#_Toc172986241)

[3.6.9 Quản lý thông tin cá nhân 72](#_Toc172986242)

[3.6.10 Đăng ký làm giảng viên 72](#_Toc172986243)

[3.6.11 Theo dõi các chủ đề yêu thích 73](#_Toc172986244)

[3.6.12 Xem các bài học trong khóa học 73](#_Toc172986245)

[3.6.13 Theo dõi, xem chi tiết thông tin giảng viên 74](#_Toc172986246)

[3.6.14 Ghi chú bài học 76](#_Toc172986247)

[3.6.15 Thanh toán khóa học 77](#_Toc172986248)

[3.6.16 Bình luận, trả lời bình luận 78](#_Toc172986249)

[3.6.17 Làm các bài kiểm tra 79](#_Toc172986250)

[3.6.18 Đăng nhập admin 80](#_Toc172986251)

[3.6.19 Quản lý các khóa học: 81](#_Toc172986252)

[3.6.20 Quản lý người dùng 82](#_Toc172986253)

[3.6.21 Thống kê 83](#_Toc172986254)

[3.6.22 Thêm khóa học mới 83](#_Toc172986255)

[3.6.23 Nộp dự án khóa học 84](#_Toc172986256)

[3.6.24 Gửi yêu cầu đăng tải khóa học 85](#_Toc172986257)

[CHƯƠNG 4. THIẾT KẾ GIAO DIỆN 88](#_Toc172986258)

[4.1 88](#_Toc172986259)

[CHƯƠNG 5. XÂY DỰNG ỨNG DỤNG 89](#_Toc172986260)

[5.1 Môi trường cài đặt 89](#_Toc172986261)

[5.2 Môi trường triển khai 89](#_Toc172986262)

[5.2.1 Triển khai mã nguồn server 89](#_Toc172986263)

[5.2.2 Thử nghiệm ứng dụng trên Android và iOS 89](#_Toc172986264)

[CHƯƠNG 6. KẾT LUẬN 91](#_Toc172986265)

[6.1 Kết quả thu được 91](#_Toc172986266)

[6.2 Mặt hạn chế 91](#_Toc172986267)

[6.3 Hướng phát triển 92](#_Toc172986268)

[TÀI LIỆU THAM KHẢO 93](#_Toc172986269)

# DANH MỤC HÌNH VẼ

[Hình 2.1 Biểu tượng Flutter 5](#_Toc172986126)

[Hình 2.2 Biểu tượng NodeJS 6](#_Toc172986127)

[Hình 2.3 Biểu tượng ExpressJS 7](#_Toc172986128)

[Hình 2.4 Biểu tượng ReactJS 8](#_Toc172986129)

[Hình 2.5 Firebase 9](#_Toc172986130)

[Hình 2.6 Cách thức hoạt động của Firebase so với các nền tảng khác 10](#_Toc172986131)

[Hình 2.7 Firebase Authentication 10](#_Toc172986132)

[Hình 2.8 Cloud Firestore 11](#_Toc172986133)

[Hình 2.9 Google Cloud Storage 12](#_Toc172986134)

[Hình 2.10 Mô hình RESTful API 14](#_Toc172986135)

[Hình 2.11 Stripe 14](#_Toc172986136)

[Hình 2.12 JSON Web Token 15](#_Toc172986137)

[Hình 2.13 Cấu trúc của jwt 16](#_Toc172986138)

[Hình 2.14 Vite và React 17](#_Toc172986139)

[Hình 2.15 Tailwind CSS 19](#_Toc172986140)

[Hình 2.16 Render 20](#_Toc172986141)

[Hình 3.1 Sơ đồ use case tổng quan của hệ thống 22](#_Toc172986142)

[Hình 3.2 Mô hình quan hệ hệ thống 63](#_Toc172986143)

[Hình 3.3 Sơ đồ lớp của hệ thống 65](#_Toc172986144)

[Hình 3.4 Sơ đồ tuần tự chức năng đăng nhập 66](#_Toc172986145)

[Hình 3.5 Sơ đồ tuần tự chức năng đăng nhập với Google 66](#_Toc172986146)

[Hình 3.6 Sơ đồ tuần tự chức năng đặt lại mật khẩu 67](#_Toc172986147)

[Hình 3.7 Sơ đồ tuần tự chức năng tìm kiếm khóa học 68](#_Toc172986148)

[Hình 3.8 Sơ đồ tuần tự chức năng quản lý thư mục 69](#_Toc172986149)

[Hình 3.9 Sơ đồ tuần tự chức năng đăng ký 70](#_Toc172986150)

[Hình 3.10 Sơ đồ tuần tự chức năng lịch sử xem video 71](#_Toc172986151)

[Hình 3.11 Sơ đồ tuần tự chức năng lưu khóa học yêu thích 71](#_Toc172986152)

[Hình 3.12 Sơ đồ tuần tự chức năng quản lý thông tin cá nhân 72](#_Toc172986153)

[Hình 3.13 Sơ đồ tuần tự chức năng đăng ký làm giảng viên 73](#_Toc172986154)

[Hình 3.14 Sơ đồ tuần tự chức năng theo dõi các chủ đề yêu thích 73](#_Toc172986155)

[Hình 3.15 Sơ đồ tuần tự chức năng xem các bài học trong khóa học 74](#_Toc172986156)

[Hình 3.16 Sơ đồ tuần tự chức năng theo dõi, xem chi tiết thông tin giảng viên 75](#_Toc172986157)

[Hình 3.17 Sơ đồ tuần tự chức năng ghi chú bài học 76](#_Toc172986158)

[Hình 3.18 Sơ đồ tuần tự chức năng thanh toán khóa học 77](#_Toc172986159)

[Hình 3.19 Sơ đồ tuần tự chức năng bình luận, trả lời bình luận 78](#_Toc172986160)

[Hình 3.20 Sơ đồ tuần tự chức năng làm bài kiểm tra 79](#_Toc172986161)

[Hình 3.21 Sơ đồ tuần tự chức năng đăng nhập admin 80](#_Toc172986162)

[Hình 3.22 Sơ đồ tuần tự quản lý khóa học 81](#_Toc172986163)

[Hình 3.23 Sơ đồ tuần tự chức năng quản lý người dùng 82](#_Toc172986164)

[Hình 3.24 Sơ đồ tuần tự chức năng thống kê 83](#_Toc172986165)

[Hình 3.25 Sơ đồ tuần tự chức năng thêm khóa học 84](#_Toc172986166)

[Hình 3.26 Sơ đồ tuần tự chức năng nộp dự án khóa học 85](#_Toc172986167)

[Hình 3.27 Sơ đồ tuần tự chức năng gửi yêu cầu đăng tải khóa học 86](#_Toc172986168)

# DANH MỤC BẢNG BIỂU

[Bảng 3.1 Đặc tả use case đăng nhập 23](#_Toc172986096)

[Bảng 3.2 Đặc tả use case đăng ký 24](#_Toc172986097)

[Bảng 3.3 Đặc tả use case đăng xuất 26](#_Toc172986098)

[Bảng 3.4 Đặc tả use case quên mật khẩu 27](#_Toc172986099)

[Bảng 3.5 Đặc tả use case đăng nhập bằng Google 28](#_Toc172986100)

[Bảng 3.6 Đặc tả use case quản lý thông tin cá nhân 29](#_Toc172986101)

[Bảng 3.7 Đặc tả use case chọn chủ đề yêu thích 30](#_Toc172986102)

[Bảng 3.8 Đặc tả use case lưu khóa học yêu thích 31](#_Toc172986103)

[Bảng 3.9 Đặc tả use case xem chi tiết thông tin giảng viên 32](#_Toc172986104)

[Bảng 3.10 Đặc tả use case ghi danh khóa học 34](#_Toc172986105)

[Bảng 3.11 Đặc tả use case tìm kiếm khóa học 35](#_Toc172986106)

[Bảng 3.12 Đặc tả use case xem các bài học trong khóa học 36](#_Toc172986107)

[Bảng 3.13 Đặc tả use case ghi chú trong khóa học 38](#_Toc172986108)

[Bảng 3.14 Đặc tả use case nộp dự án khóa học 40](#_Toc172986109)

[Bảng 3.15 Đặc tả use case làm các bài kiểm tra 42](#_Toc172986110)

[Bảng 3.16 Đặc tả use case thảo luận trên diễn đàn 43](#_Toc172986111)

[Bảng 3.17 Đặc tả use case thêm khóa học 45](#_Toc172986112)

[Bảng 3.18 Đặc tả use case đăng tải khóa học 47](#_Toc172986113)

[Bảng 3.19 Đặc tả use case lịch sử xem 48](#_Toc172986114)

[Bảng 3.20 Đặc tả use case quản lý thư mục 49](#_Toc172986115)

[Bảng 3.21 Đặc tả use case yêu cầu trở thành giảng viên 51](#_Toc172986116)

[Bảng 3.22 Đặc tả use case xem các request thành giảng viên 52](#_Toc172986117)

[Bảng 3.23 Đặc tả use case phê duyệt yêu cầu trở thành giảng viên 53](#_Toc172986118)

[Bảng 3.24 Đặc tả use case từ chối yêu cầu trở thành giảng viên 54](#_Toc172986119)

[Bảng 3.25 Đặc tả use case thống kê 56](#_Toc172986120)

[Bảng 3.26 Đặc tả use case quản lý khóa học 57](#_Toc172986121)

[Bảng 3.27 Đặc tả use case phê duyệt khóa học 58](#_Toc172986122)

[Bảng 3.28 Đặc tả use case quản lý người dùng 59](#_Toc172986123)

[Bảng 3.29 Đặc tả use case đăng nhập admin 61](#_Toc172986124)

[Bảng 3.30 Đặc tả use case đăng xuất admin 62](#_Toc172986125)

# DANH MỤC CÁC CHỮ VIẾT TẮT

|  |  |
| --- | --- |
| API | Application Programming Interface |
| HTML | Hypertext Markup Language |
| DOM | Document Object Model |
| CSS | Cascading Style Sheets |
| REST | Representational State Transfer |
| UI | User Interface |
| HTTP | Hypertext Transfer Protocol |
| JWT | JSON Web Token |

# MỞ ĐẦU

## Lý do chọn đề tài

Trong bối cảnh cách mạng công nghiệp 4.0, công nghệ số đang phát triển mạnh mẽ và thay đổi nhiều lĩnh vực, trong đó có giáo dục. Học trực tuyến đã trở thành một phương pháp học tập phổ biến và cần thiết, đặc biệt dưới thời kỳ đại dịch COVID-19 khi việc giãn cách xã hội trở thành yêu cầu bắt buộc. Học trực tuyến giúp tiết kiệm thời gian, chi phí và tạo điều kiện học tập linh hoạt cho người học.

Ngày nay, thiết bị di động như điện thoại thông minh và máy tính bảng đã trở thành một phần không thể thiếu trong cuộc sống hàng ngày. Việc phát triển ứng dụng học trực tuyến trên các nền tảng di động sẽ giúp người học tiếp cận kiến thức mọi lúc, mọi nơi, từ đó nâng cao hiệu quả học tập.

Với sự đa dạng về hệ điều hành trên các loại điện thoại khác nhau, Flutter – một framework phát triển ứng dụng di động đa nền tảng do Google phát triển ra đời cho phép tạo ra các ứng dụng chất lượng cao trên cả iOS và Android và cả Web từ một mã nguồn duy nhất. Điều này không chỉ tiết kiệm thời gian và chi phí phát triển mà còn đảm bảo tính nhất quán và trải nghiệm người dùng tốt hơn.

Ứng dụng Mindify sẽ được thiết kế để đáp ứng nhu cầu cá nhân hóa học tập của từng người dùng. Tính năng này giúp người học có thể lựa chọn nội dung học tập phù hợp với nhu cầu và trình độ của mình, từ đó tối ưu hóa quá trình học tập. Ngoài ra, Mindify sẽ tích hợp các tính năng như video bài giảng, bài kiểm tra, và tài liệu học tập, giúp người học có thể học tập theo tiến độ của riêng mình và nhận được sự hỗ trợ kịp thời từ giáo viên thông qua các diễn đàn thảo luận và phản hồi cá nhân. Điều này giúp tạo ra một môi trường học tập linh hoạt và hiệu quả.

Việc phát triển ứng dụng học trực tuyến đa nền tảng với Flutter không chỉ đáp ứng nhu cầu học tập hiện tại mà còn bắt kịp xu hướng phát triển giáo dục hiện đại. Mindify hướng tới việc trở thành một công cụ hữu ích, hỗ trợ học tập cho mọi đối tượng người dùng, từ học sinh, sinh viên đến người đi làm.

Với những lý do trên, việc chọn đề tài “Phát triển ứng dụng di động dạy và học trực tuyến đa nền tảng với Flutter” là hết sức cần thiết và có ý nghĩa thực tiễn cao. Ứng dụng Mindify hứa hẹn sẽ mang lại nhiều lợi ích cho người học và góp phần vào sự phát triển của giáo dục trong thời đại công nghệ số.

## Mục tiêu thực hiện đề tài

Đề tài này nhằm phát triển một ứng dụng di động giúp cải thiện hiệu quả học tập trực tuyến cho người dùng, với các mục tiêu cụ thể như sau:

* **Xây dựng ứng dụng di động Mindify**: Phát triển một ứng dụng di động dạy và học trực tuyến đa nền tảng sử dụng Flutter, cung cấp các khóa học trực tuyến với nội dung phong phú và linh hoạt.
* **Tạo môi trường học tập hiệu quả**: Đảm bảo ứng dụng cung cấp các tính năng giúp người học có thể học tập một cách hiệu quả, như video bài giảng, bài kiểm tra, tài liệu học tập, và diễn đàn thảo luận.
* **Cá nhân hóa trải nghiệm học tập**: Thiết kế các tính năng cá nhân hóa để người học có thể lựa chọn nội dung học tập phù hợp với nhu cầu và trình độ của mình.
* **Tiết kiệm chi phí và thời gian phát triển**: Sử dụng Flutter để phát triển ứng dụng đa nền tảng, tiết kiệm chi phí và thời gian so với việc phát triển riêng biệt cho từng hệ điều hành.

## Đối tượng và phạm vi nghiên cứu

### Đối tượng nghiên cứu

Đối tượng nghiên cứu: Ứng dụng Mindify hướng đến đối tượng người học đa dạng, bao gồm học sinh, sinh viên, và người đi làm có nhu cầu học tập trực tuyến.

### Phạm vi môi trường

Môi trường để phát triển ứng dụng là các hệ điều hành: iOS, Android và Windows.

Phạm vi nghiên cứu của đề tài bao gồm:

* Nghiên cứu về nhu cầu và thói quen học tập trực tuyến của người dùng.
* Phân tích các tính năng cần thiết để xây dựng và quản lý một ứng dụng học trực tuyến hiệu quả.
* Áp dụng Flutter, Firebase, NodeJS và ReactJS để phát triển ứng dụng đa nền tảng, đảm bảo tính khả dụng và hiệu quả.

## Phương pháp nghiên cứu

Phương pháp thu thập dữ liệu:

* **Khảo sát:** Thực hiện khảo sát trên các trang web và ứng dụng học trực tuyến để thu thập thông tin về nhu cầu và thói quen học tập của người dùng.
* **Nghiên cứu:** Phân tích các ứng dụng học trực tuyến hiện có để xác định những tính năng quan trọng và ưu nhược điểm của chúng.

Phương pháp phân tích:

* **Phân tích dữ liệu:** Xử lý và phân tích dữ liệu thu thập được để hiểu rõ nhu cầu và mong muốn của người dùng.
* **Xác định tính năng:** Xác định các tính năng cần thiết và ưu tiên cho việc phát triển ứng dụng Mindify dựa trên kết quả phân tích.

Phương pháp thiết kế và phát triển:

* **Flutte**r: Sử dụng Flutter để thiết kế và phát triển ứng dụng di động đa nền tảng, đảm bảo ứng dụng hoạt động mượt mà trên cả iOS và Android.
* **Firebase**: Sử dụng Firebase làm cơ sở dữ liệu lưu trữ và quản lý người dùng cho ứng dụng, đảm bảo tính bảo mật và khả năng mở rộng.
* **Node.js**: Sử dụng Node.js để xây dựng backend, triển khai các chức năng cần thiết và kết nối với cơ sở dữ liệu.
* **ReactJS**: Sử dụng ReactJS để phát triển trang web quản lý ứng dụng, cho phép quản trị viên quản lý khóa học, người dùng, và các nội dung liên quan.
* **RESTful API**: Áp dụng mô hình RESTful API để kết nối giữa frontend và backend, đảm bảo khả năng mở rộng và dễ dàng bảo trì hệ thống.
* **Thử nghiệm**: Tiến hành thử nghiệm ứng dụng qua nhiều giai đoạn, bao gồm thử nghiệm nội bộ và beta test với người dùng thực tế.
* **Điều chỉnh**: Thu thập phản hồi từ người dùng và điều chỉnh ứng dụng để cải thiện tính năng và trải nghiệm người dùng.

## Ý nghĩa thực tiễn của đề tài

Đề tài phát triển ứng dụng di động dạy và học trực tuyến đa nền tảng với Flutter mang lại nhiều lợi ích thiết thực cho cả người học và ngành giáo dục:

* **Nâng cao hiệu quả học tập**: Ứng dụng Mindify sẽ giúp người học tiếp cận kiến thức một cách linh hoạt và hiệu quả, tối ưu hóa quá trình học tập cá nhân.
* **Tiết kiệm thời gian và chi phí**: Phát triển ứng dụng đa nền tảng với Flutter giúp tiết kiệm thời gian và chi phí so với việc phát triển ứng dụng riêng biệt cho từng hệ điều hành.
* **Góp phần vào sự phát triển giáo dục**: Mindify hướng tới việc trở thành một công cụ hữu ích, hỗ trợ học tập cho mọi đối tượng người dùng, từ học sinh, sinh viên đến người đi làm, góp phần vào sự phát triển của giáo dục trong thời đại công nghệ số.
* **Thúc đẩy ứng dụng công nghệ trong giáo dục**: Đề tài này sẽ khuyến khích việc áp dụng công nghệ hiện đại trong giáo dục, tạo điều kiện cho sự đổi mới và phát triển trong lĩnh vực này.

# CƠ SỞ LÝ THUYẾT

## Flutter

Flutter là một framework do Google phát triển và lần đầu tiên ra mắt vào tháng 5 năm 2017. Flutter cho phép các nhà phát triển xây dựng các ứng dụng native cho cả iOS và Android bằng cách sử dụng một mã nguồn duy nhất. Điều này giúp tiết kiệm thời gian và công sức so với việc phát triển từng ứng dụng riêng lẻ cho từng nền tảng.

A logo with blue and grey colors

Description automatically generated

Hình 2.1 Biểu tượng Flutter

Flutter cung cấp khả năng gọi mã native (Swift, Kotlin, Java) từ Dart và ngược lại thông qua Platform Channels. Điều này cho phép tích hợp các tính năng và thư viện native mà Flutter chưa hỗ trợ trực tiếp.

Một số ưu điểm của Flutter:

* **Hiệu suất cao**: Flutter sử dụng Dart biên dịch trực tiếp thành mã máy, giúp ứng dụng chạy nhanh và mượt mà.
* **Phát triển nhanh chóng**: Với tính năng Hot Reload, các nhà phát triển có thể thử nghiệm và xem kết quả ngay lập tức.
* **Giao diện tùy chỉnh:** Flutter cung cấp khả năng tùy chỉnh cao, cho phép tạo ra các giao diện phức tạp và độc đáo.
* **Hỗ trợ đa nền tảng**: Một mã nguồn duy nhất có thể chạy trên cả iOS, Android, web và desktop.

## NodeJS và ExpressJS

### NodeJS

Node.js là một nền tảng mã nguồn mở và đa nền tảng cho phép chạy môi trường runtime JavaScript trên máy chủ. Được xây dựng trên nền tảng JavaScript V8 của Google Chrome, Node.js đã trở thành một công cụ phổ biến để phát triển các ứng dụng web, API và các dịch vụ máy chủ khác.

A group of logos on a black background

Description automatically generated

Hình 2.2 Biểu tượng NodeJS

Ưu điểm của NodeJS:

* **Hiệu suất cao:** Nhờ vào mô hình I/O không chặn và vòng lặp sự kiện, Node.js có thể xử lý nhiều kết nối đồng thời một cách hiệu quả.
* **Cộng đồng lớn**: Node.js có một cộng đồng lớn và phát triển mạnh mẽ, cung cấp nhiều thư viện và tài nguyên hỗ trợ.
* **JavaScript:** Sử dụng JavaScript cho cả phía máy khách và máy chủ giúp giảm thiểu sự phức tạp và tăng cường tính nhất quán.

### ExpressJS

Express.js là một framework web ứng dụng tối giản nhưng mạnh mẽ cho Node.js, cung cấp một tập hợp các công cụ và tính năng để xây dựng các ứng dụng web và API. Được phát triển và duy trì bởi cộng đồng, Express.js là một trong những framework web phổ biến nhất cho Node.js.

A logo for a company

Description automatically generated

Hình 2.3 Biểu tượng ExpressJS

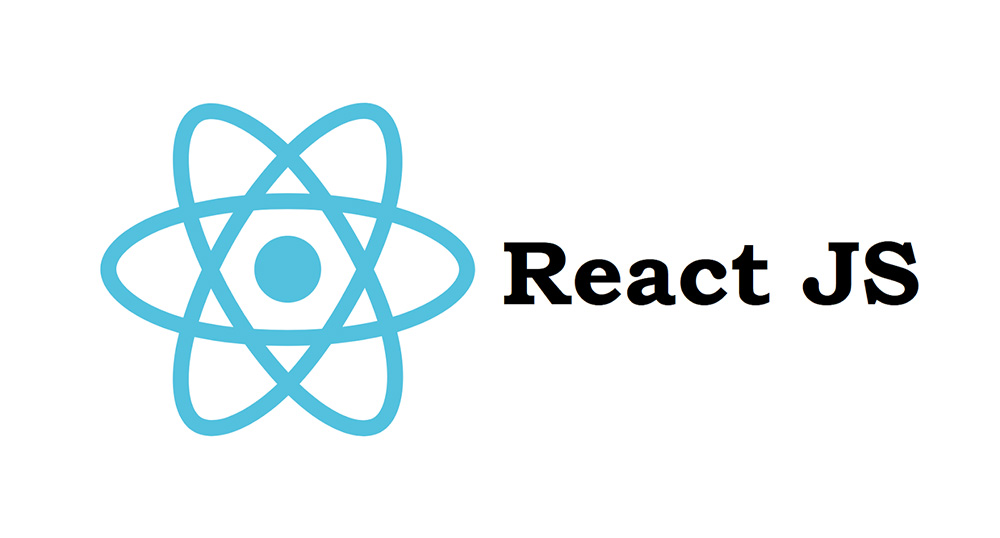
Express.js được ra mắt lần đầu vào tháng 11 năm 2010 bởi TJ Holowaychuk và đã trở thành một công cụ không thể thiếu cho các nhà phát triển Node.js. Express.js giúp đơn giản hóa quá trình xây dựng các ứng dụng web và API bằng cách cung cấp một bộ công cụ mạnh mẽ và linh hoạt.

Ưu điểm của ExpressJS:

* Đơn giản và linh hoạt: Express.js cung cấp một API đơn giản nhưng mạnh mẽ, cho phép các nhà phát triển xây dựng các ứng dụng web và API một cách nhanh chóng và dễ dàng.
* Cộng đồng lớn: Với một cộng đồng lớn và nhiều tài nguyên, các nhà phát triển có thể dễ dàng tìm kiếm sự hỗ trợ và các plugin mở rộng.
* Hiệu suất cao: Express.js được tối ưu hóa cho hiệu suất cao, giúp xử lý hàng nghìn yêu cầu đồng thời một cách hiệu quả.

## ReactJS

React là một thư viện JavaScript mã nguồn mở do Facebook phát triển, dùng để tạo giao diện người dùng (UI) cho các ứng dụng web. Ra mắt lần đầu vào năm 2013, React đã nhanh chóng trở thành một trong những thư viện UI phổ biến nhất trên toàn cầu.



Hình 2.4 Biểu tượng ReactJS

Các đặc điểm chính của ReactJS:

* Virtual DOM (DOM Ảo): là một trong những cơ chế quan trọng nhất của React. Thay vì thao tác trực tiếp trên DOM thực, React sử dụng một bản sao ảo của DOM để theo dõi các thay đổi. Khi trạng thái của một component thay đổi, React sẽ cập nhật Virtual DOM trước. Sau đó, React so sánh Virtual DOM mới với phiên bản cũ để xác định những thay đổi thực sự và cập nhật DOM thực một cách tối ưu nhất, giảm thiểu số lần thao tác DOM và cải thiện hiệu suất
* Components (Thành phần): React xây dựng giao diện người dùng dưới dạng các component. Mỗi component là một khối xây dựng độc lập và tái sử dụng được, có thể chứa logic riêng, trạng thái riêng và có khả năng nhận dữ liệu thông qua props
* Hooks: là một tính năng mới được giới thiệu trong React 16.8, cho phép sử dụng state và các tính năng khác của React trong function components. Các hook phổ biến bao gồm:
* useState(): Quản lý state trong function component.
* useEffect(): Quản lý các side effect như gọi API, đăng ký sự kiện, và dọn dẹp.
* useContext(): Sử dụng context để quản lý dữ liệu toàn cục.

## Firebase

Firebase là một nền tảng phát triển ứng dụng di động và web của Google, cung cấp một loạt các dịch vụ mạnh mẽ giúp các nhà phát triển xây dựng, cải thiện và phát triển ứng dụng của họ một cách nhanh chóng và dễ dàng

A logo with a red and yellow flame

Description automatically generated

Hình 2.5 Firebase

Firebase cung cấp một loạt các dịch vụ như cơ sở dữ liệu thời gian thực, xác thực người dùng, lưu trữ, thông báo đẩy, và nhiều công cụ khác giúp tối ưu hóa quá trình phát triển ứng dụng. Firebase được thiết kế để hoạt động mượt mà với các nền tảng di động và web, giúp các nhà phát triển tập trung vào việc tạo ra trải nghiệm người dùng tuyệt vời mà không phải lo lắng về cơ sở hạ tầng.

A diagram of a server

Description automatically generated

Hình 2.6 Cách thức hoạt động của Firebase so với các nền tảng khác

### Firebase Authentication

Firebase Authentication cung cấp các phương thức dễ sử dụng để xác thực người dùng trong ứng dụng. Nó hỗ trợ nhiều phương thức xác thực như email/mật khẩu, số điện thoại, và đăng nhập qua các tài khoản mạng xã hội như Google, Facebook, Twitter.

A group of phones connected to a group of silver objects

Description automatically generated

Hình 2.7 Firebase Authentication

### Cloud Firestore

Cloud Firestore là một cơ sở dữ liệu NoSQL được phát triển bởi Google, là một phần của Firebase. Dưới đây là một số điểm chính về Cloud Firestore:

* **Realtime Database**: Cloud Firestore cung cấp khả năng cập nhật thời gian thực, cho phép dữ liệu được đồng bộ hóa ngay lập tức trên tất cả các thiết bị kết nối.
* **Scalability:** Được thiết kế để mở rộng dễ dàng, từ ứng dụng nhỏ đến các ứng dụng quy mô lớn với hàng triệu người dùng.
* **Structured Data:** Dữ liệu được lưu trữ dưới dạng bộ sưu tập (collections) và tài liệu (documents), giúp việc tổ chức và truy vấn dữ liệu trở nên dễ dàng hơn.
* **Offline Support**: Cloud Firestore hỗ trợ lưu trữ dữ liệu ngoại tuyến, cho phép ứng dụng tiếp tục hoạt động ngay cả khi không có kết nối mạng.
* **Security:** Tích hợp với Firebase Authentication và cung cấp quy tắc bảo mật mạnh mẽ để kiểm soát quyền truy cập vào dữ liệu.
* **Integration:** Dễ dàng tích hợp với các dịch vụ khác của Firebase và Google Cloud, hỗ trợ cả ứng dụng web và di động.

A computer and mobile devices

Description automatically generated

Hình 2.8 Cloud Firestore

## Google Cloud Storage

Google Storage là một dịch vụ lưu trữ đối tượng được cung cấp bởi Google Cloud Platform.



Hình 2.9 Google Cloud Storage

Dưới đây là một số điểm chính về Google Storage:

* **Object Storage:** Lưu trữ dữ liệu dưới dạng các đối tượng, mỗi đối tượng bao gồm dữ liệu và siêu dữ liệu kèm theo.
* **Buckets:** Đơn vị lưu trữ cơ bản, chứa nhiều đối tượng và được xác định duy nhất bởi tên. Buckets có thể được cấu hình với các thiết lập về kiểm soát truy cập, vị trí địa lý và chính sách vòng đời.
* **Location:** Chọn vị trí địa lý nơi dữ liệu sẽ được lưu trữ, bao gồm multi-region, dual-region và region.
* **Storage Classes**: Các lớp lưu trữ khác nhau phù hợp với nhu cầu truy cập:
* Standard Storage: Cho dữ liệu thường xuyên truy cập.
* Nearline Storage: Cho dữ liệu ít truy cập, khoảng một lần một tháng.
* Coldline Storage: Cho dữ liệu rất ít truy cập, khoảng một lần một năm.
* Archive Storage: Cho dữ liệu gần như không truy cập, nhưng cần lưu trữ lâu dài.
* **Lifecycle Management**: Chính sách vòng đời tự động quản lý các đối tượng dựa trên tiêu chí như tuổi của đối tượng và lớp lưu trữ hiện tại.
* **Security**: Cung cấp nhiều cơ chế kiểm soát truy cập, bao gồm IAM, ACLs và quản lý khóa mã hóa.
* **Integration**: Dễ dàng tích hợp với các dịch vụ khác của Google Cloud, hỗ trợ nhiều loại ứng dụng khác nhau.

## Mô hình RESTful API

RESTful API là một kiểu thiết kế API tuân theo các nguyên tắc của REST. Dưới đây là một số điểm chính về mô hình này:

* Resource-Based: RESTful API dựa trên khái niệm về "resources" (tài nguyên). Mỗi tài nguyên có một URI (Uniform Resource Identifier) duy nhất. Ví dụ: /users, /orders, /products.
* HTTP Methods: RESTful API sử dụng các phương thức HTTP để thực hiện các hành động CRUD (Create, Read, Update, Delete) trên tài nguyên.
* GET: Lấy dữ liệu từ máy chủ.
* POST: Tạo mới một tài nguyên.
* PUT: Cập nhật toàn bộ tài nguyên.
* PATCH: Cập nhật một phần tài nguyên.
* DELETE: Xóa tài nguyên.
* Stateless: Mỗi yêu cầu từ client đến server phải chứa đủ thông tin để server hiểu và xử lý yêu cầu. Server sẽ không lưu trữ trạng thái của client giữa các yêu cầu.
* Representation: Tài nguyên có thể được đại diện ở nhiều định dạng khác nhau, như JSON, XML, HTML. JSON thường được sử dụng phổ biến nhất. Client và Server có thể trao đổi định dạng mong muốn thông qua tiêu đề HTTP Content-Type và Accept.
* Layered System: Kiến trúc của RESTful API có thể được phân thành nhiều lớp, giúp tăng cường bảo mật, cân bằng tải và dễ dàng mở rộng.
* Caching: RESTful API có thể tận dụng cơ chế caching của HTTP để cải thiện hiệu suất. Server có thể chỉ định các header như Cache-Control, ETag để hướng dẫn client về cách lưu trữ tạm thời và làm mới dữ liệu.



Hình 2.10 Mô hình RESTful API

## Stripe

Stripe là một nền tảng thanh toán trực tuyến được phát triển để giúp các doanh nghiệp xử lý giao dịch trực tuyến dễ dàng và hiệu quả.

A blue text on a white background

Description automatically generated

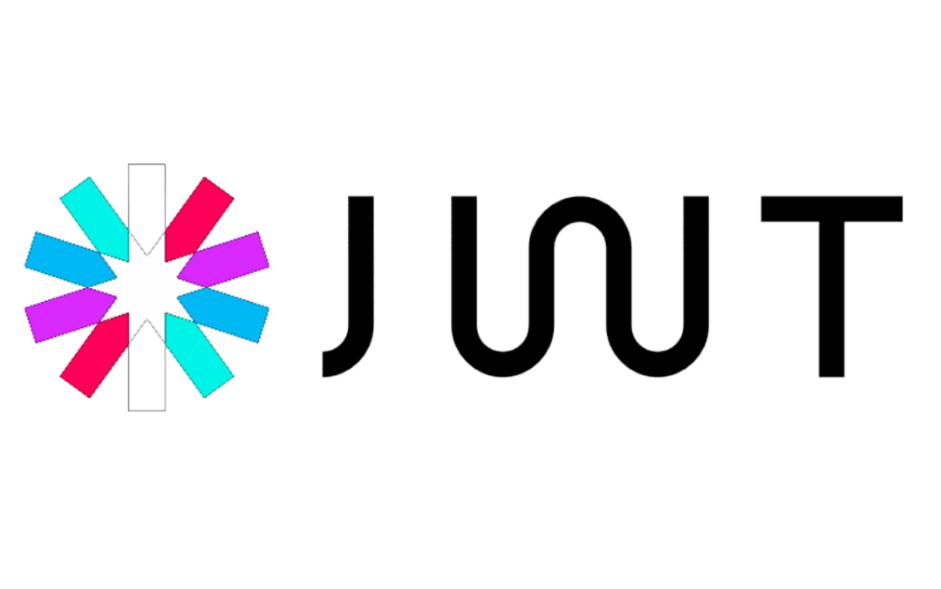
Hình 2.11 Stripe

Dưới đây là một số điểm chính về Stripe:

* **Payment Processing**: Stripe cung cấp khả năng xử lý thanh toán linh hoạt, hỗ trợ nhiều loại thẻ tín dụng và phương thức thanh toán khác nhau trên toàn cầu, bao gồm cả Apple Pay và Google Pay.
* **Scalability**: Được thiết kế để mở rộng dễ dàng, từ các doanh nghiệp nhỏ đến các công ty quy mô lớn với hàng triệu giao dịch hàng ngày.
* **API-Driven**: Stripe cung cấp API mạnh mẽ và linh hoạt, cho phép các nhà phát triển dễ dàng tích hợp và tùy chỉnh giải pháp thanh toán theo nhu cầu của họ.
* **Security**: Stripe tuân thủ các tiêu chuẩn bảo mật PCI DSS và cung cấp các công cụ bảo mật mạnh mẽ như mã hóa dữ liệu và bảo vệ chống gian lận để bảo vệ thông tin thanh toán của khách hàng.
* **Subscription Management**: Stripe hỗ trợ quản lý đăng ký và thanh toán định kỳ, giúp các doanh nghiệp dễ dàng triển khai và quản lý các mô hình kinh doanh dựa trên đăng ký.
* **Integration**: Dễ dàng tích hợp với nhiều nền tảng và dịch vụ khác, bao gồm cả các hệ thống quản lý quan hệ khách hàng (CRM), hệ thống quản lý nội dung (CMS), và các nền tảng thương mại điện tử.
* **Analytics and Reporting**: Stripe cung cấp các công cụ phân tích và báo cáo mạnh mẽ, cho phép các doanh nghiệp theo dõi và quản lý hiệu quả các giao dịch và doanh thu của họ.

## JWT

JWT là một tiêu chuẩn mở để truyền tải thông tin một cách an toàn giữa các bên dưới dạng JSON.



Hình 2.12 JSON Web Token

Dưới đây là một số điểm chính về JWT:

* + **Compact**: JWT có kích thước nhỏ gọn, giúp dễ dàng truyền tải qua URL, HTTP header hoặc lưu trữ trong các đối tượng HTML5 Web Storage.
  + **Self-contained**: JWT tự chứa thông tin, nghĩa là token bao gồm tất cả thông tin cần thiết cho việc xác thực và uỷ quyền mà không cần truy cập vào cơ sở dữ liệu.
  + **Secure**: JWT sử dụng chữ ký số để đảm bảo tính toàn vẹn và xác thực của thông tin bên trong token. Các chữ ký này có thể là HMAC hoặc cặp khóa RSA/ECDSA.
  + **Interoperability**: Do dựa trên định dạng JSON, JWT có khả năng tương thích cao và dễ dàng sử dụng với nhiều ngôn ngữ lập trình và nền tảng khác nhau.
  + **Claims**: JWT bao gồm các claims, là các thông tin như người dùng, thời gian hết hạn và các quyền hạn. Các claims này được chia thành ba loại: registered claims, public claims và private claims.
  + **Structure**: JWT bao gồm ba phần chính: Header, Payload và Signature.



Hình 2.13 Cấu trúc của jwt

* + **Header**: Chứa thông tin về thuật toán mã hóa và loại token.
  + **Payload**: Chứa claims, là các thông tin cần truyền tải.
  + **Signature**: Được tạo ra bằng cách kết hợp và mã hóa Header, Payload và một secret hoặc khóa riêng tư.
  + **Ứng dụng**: JWT thường được sử dụng cho xác thực người dùng (authentication), phân quyền (authorization), trao đổi thông tin (information exchange) và bảo vệ các API.

JWT là một công cụ mạnh mẽ và linh hoạt, giúp bảo đảm tính an toàn và hiệu quả trong việc trao đổi thông tin và xác thực người dùng trên các hệ thống phân tán.

## Vite

Vite là một công cụ xây dựng (build tool) hiện đại cho các dự án JavaScript, đặc biệt là các ứng dụng dựa trên Vue.js và React.



Hình 2.14 Vite và React

Dưới đây là một số điểm chính về Vite:

* **Fast Development Server**: Vite cung cấp một máy chủ phát triển cực nhanh, cho phép làm mới trang ngay lập tức và hot module replacement (HMR) nhanh chóng. Điều này giúp tăng tốc độ phát triển và nâng cao trải nghiệm lập trình viên.
* **Optimized Build**: Vite sử dụng Rollup dưới hood để tạo ra các gói sản phẩm tối ưu. Điều này giúp đảm bảo rằng các ứng dụng sản xuất có hiệu suất cao và dung lượng nhỏ.
* **ES Module Support**: Vite tận dụng lợi thế của ES Modules trong trình duyệt để cải thiện tốc độ tải trang và khả năng mô-đun hóa mã nguồn, giúp giảm thiểu thời gian xây dựng ban đầu.
* **Flexible Configuration**: Vite cho phép cấu hình linh hoạt thông qua các tệp cấu hình, giúp dễ dàng tùy chỉnh và mở rộng dự án theo nhu cầu cụ thể của bạn.
* **Plugin Ecosystem**: Vite có một hệ sinh thái plugin mạnh mẽ, cho phép tích hợp dễ dàng với nhiều công cụ và thư viện khác như TypeScript, JSX, CSS pre-processors, và nhiều hơn nữa.
* **Out-of-the-box Support**: Vite hỗ trợ mặc định cho nhiều tính năng quan trọng như xử lý tệp tĩnh, CSS, và nhiều ngôn ngữ lập trình phổ biến, giúp giảm thiểu cấu hình và thời gian thiết lập dự án.
* **Production Ready**: Vite không chỉ tối ưu hóa quá trình phát triển mà còn đảm bảo rằng các ứng dụng sản xuất được xây dựng một cách hiệu quả và tối ưu.

## Tailwind CSS

Tailwind CSS là một framework CSS tiện dụng cho phép chúng ta nhanh chóng xây dựng các giao diện người dùng hiện đại và tiện lợi.

A black text on a white background

Description automatically generated

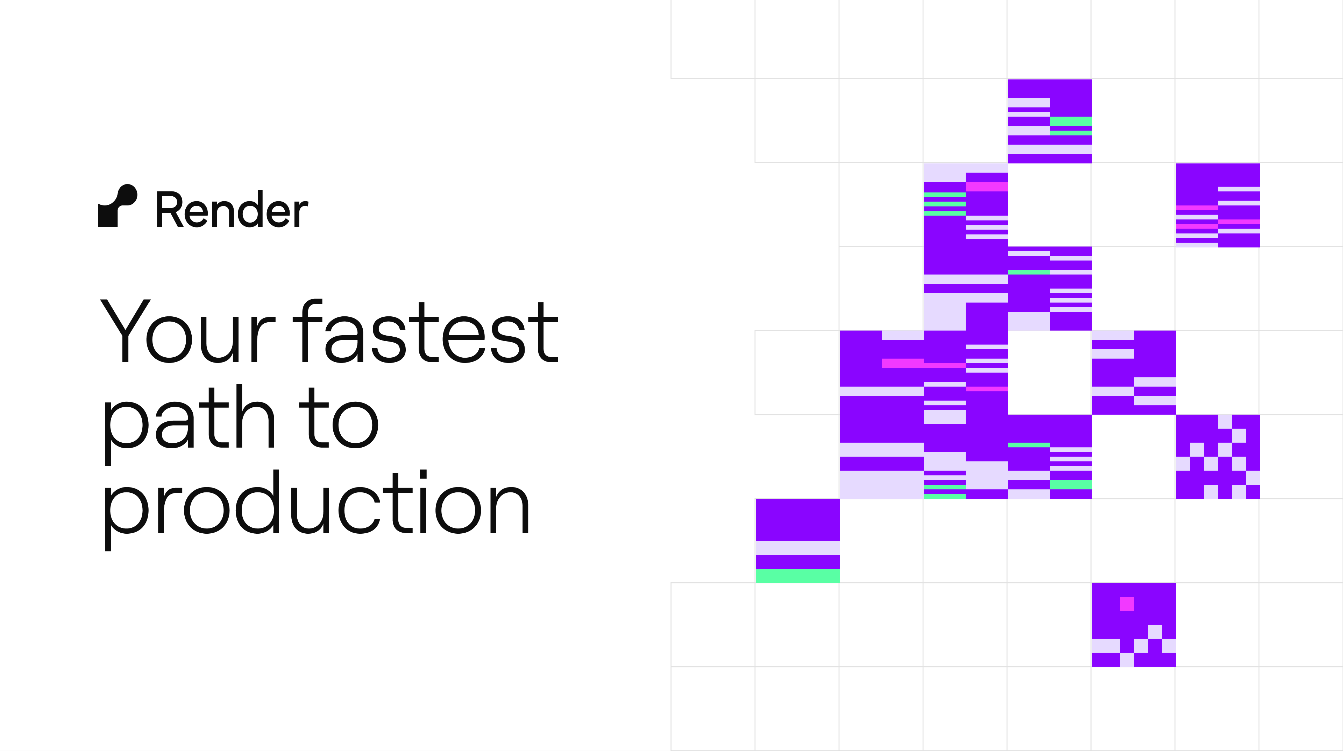
Hình 2.15 Tailwind CSS

Dưới đây là một số điểm chính về Tailwind CSS:

* **Utility-First**: Tailwind CSS sử dụng cách tiếp cận “utility-first”, cung cấp hàng trăm lớp tiện ích nhỏ gọn (utility classes) như *flex, pt-4, text-center, rotate-90,…* giúp bạn xây dựng giao diện một cách nhanh chóng và dễ dàng mà không cần viết CSS tùy chỉnh.
* **Highly Customizable**: Tailwind CSS rất linh hoạt và dễ dàng tùy chỉnh. Bạn có thể cấu hình các giá trị mặc định, tạo các lớp tiện ích tùy chỉnh, và mở rộng framework thông qua tệp cấu hình tailwind.config.js.
* **Responsive Design**: Tailwind CSS hỗ trợ thiết kế đáp ứng (responsive design) một cách hiệu quả. Bạn có thể dễ dàng thêm các lớp tiện ích cho các kích thước màn hình khác nhau bằng cách sử dụng các tiền tố (prefix) như *sm:, md:, lg:, và xl:.*
* **Performance**: Tailwind CSS cung cấp công cụ PurgeCSS để loại bỏ các lớp không sử dụng trong tệp CSS của bạn, giúp giảm kích thước tệp và cải thiện hiệu suất tải trang.
* **Component-Based**: Tailwind CSS hỗ trợ tốt việc xây dựng các component tái sử dụng, giúp mã CSS của bạn trở nên rõ ràng và dễ bảo trì.
* **Design Consistency**: Với Tailwind CSS, bạn có thể dễ dàng duy trì tính nhất quán trong thiết kế của mình bằng cách sử dụng các lớp tiện ích được tiêu chuẩn hóa và cấu hình sẵn.
* **Community and Ecosystem**: Tailwind CSS có một cộng đồng lớn và sôi động, cung cấp nhiều plugin và công cụ hỗ trợ, từ các tiện ích cho việc tạo mẫu nhanh đến các bộ component sẵn dùng như Tailwind UI.
* **Integration**: Tailwind CSS dễ dàng tích hợp với nhiều framework và công cụ khác như React, Vue, Angular, và các hệ thống build như Webpack và Vite.

## Render

Render là một nền tảng đám mây cung cấp dịch vụ lưu trữ và triển khai ứng dụng cho các nhà phát triển. Với Render, chúng ta có thể dễ dàng triển khai các ứng dụng web, dịch vụ API, static sites, cơ sở dữ liệu, và các loại ứng dụng khác mà không cần quản lý cơ sở hạ tầng phức tạp. Render hỗ trợ nhiều ngôn ngữ và framework phổ biến như Node.js, Python, Ruby, Go, và nhiều hơn nữa.



Hình 2.16 Render

Các tính năng chính của Render:

* + Dễ dàng triển khai: Render cung cấp quy trình triển khai tự động từ kho lưu trữ mã nguồn như GitHub, GitLab. Bạn chỉ cần kết nối kho mã nguồn của mình và Render sẽ tự động triển khai ứng dụng mỗi khi có thay đổi trong mã nguồn.
  + Quản lý cơ sở hạ tầng: Render quản lý toàn bộ cơ sở hạ tầng cho bạn, bao gồm các máy chủ, cân bằng tải, và cơ sở dữ liệu, cho phép bạn tập trung vào việc phát triển ứng dụng.
  + Tích hợp CI/CD: Render hỗ trợ tích hợp liên tục và triển khai liên tục (CI/CD), giúp tự động hóa quy trình triển khai và đảm bảo mã nguồn luôn ở trạng thái tốt nhất.
  + Bảo mật: Render cung cấp các tính năng bảo mật như chứng chỉ SSL miễn phí, xác thực hai yếu tố (2FA), và bảo mật dữ liệu.
  + Giá cả hợp lý: Render có các gói giá phù hợp với mọi nhu cầu, từ các dự án cá nhân nhỏ đến các ứng dụng doanh nghiệp lớn.
  + Khả năng mở rộng: Render cho phép mở rộng tài nguyên dễ dàng để đáp ứng nhu cầu lưu lượng và tải của ứng dụng.

# PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## Khảo sát

## Use case Diagram

Dưới đây là use case tổng quan về hệ thống:

A diagram of a company

Description automatically generated

Hình 3.1 Sơ đồ use case tổng quan của hệ thống

Các tác nhân chính của hệ thống bao gồm:

* Người quản lí (Administrator): có nhiệm vụ quản lý về người dùng, khóa học, duyệt các khóa học từ giảng viên và duyệt các yêu cầu làm giảng viên.
* Người dùng (User): người sử dụng ứng dụng di động Mindify, có quyền sử dụng các dịch vụ của ứng dụng.
* Giảng viên (Instructor): cũng là người sử dụng như user nhưng có nhiều quyền hạn hơn như đăng tải một khóa học, tạo các bài kiểm tra, tạo dự án và đánh giá các dự án do phía học viên mình thực hiện. Để trở thành một giảng viên người dùng cần đăng ký trên ứng dụng và chờ để phê duyệt.

## Đặc tả các Use case của hệ thống

### Use case đăng nhập

Use case đăng nhập biểu hiện cho hành động người dùng đăng nhập vào ứng dụng Mindify. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.1 Đặc tả use case đăng nhập

|  |  |  |
| --- | --- | --- |
| Tên use case: | Sign in | |
| Tác nhân: | User, Instructer | |
| Mô tả: | Người dùng đăng nhập vào ứng dụng Mindify trên thiết bị di động | |
| Điều kiện tiên quyết: | Người dùng đã đăng ký tài khoản và xác nhận email thành công. | |
| Điều kiện sau: | Đăng nhập được vào ứng dụng | |
| Điều kiện thoát: | Người dùng sign out | |
| Luồng hoạt động | Actor | System |
| Nhập email và mật khẩu.  Người dùng nhấn đăng nhập | Hệ thống gửi yêu cầu đăng nhập lên để FirebaseAuth kiểm tra.  FirebaseAuth kiểm tra email và password có khớp hay không.  Nếu tồn tại và khớp thì cấp quyền tương ứng.  Nếu không thì thông báo đăng nhập không thành công.  Ứng dụng trả về kết quả đăng nhập.  Nếu thành công thì chuyển tới trang chủ.  Không thành công hiện thông báo sai email hoặc mật khẩu.  Kết thúc use case |
| Điều kiện ngoại lệ | Không có | |

### Use case đăng ký

Use case đăng ký biểu hiện cho hành động người dùng đăng ký vào ứng dụng Mindify để sử dụng các dịch vụ. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.2 Đặc tả use case đăng ký

|  |  |  |
| --- | --- | --- |
| Tên use case: | Sign up | |
| Tác nhân: | User | |
| Mô tả: | Người dùng đăng ký tài khoản trên ứng dụng Mindify của thiết bị di động. | |
| Điều kiện tiên quyết: | Không có | |
| Điều kiện sau: | Hệ thống xác nhận đăng ký thành công, thêm tài khoản vào hệ thống. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Trên trang đăng nhập, người dùng nhấn vào nút Create one.  Người dùng nhập thông tin vào các trường email, mật khẩu và xác nhận mật khẩu.  Người dùng bấm nút Create.  Người dùng vào ứng dụng Gmail và xác nhận email. | Ứng dụng đưa người dùng tới trang Sign up.  FirebaseAuth kiểm tra email. Sau đó gửi mail xác nhận tài khoản tới email mà người dùng đăng ký.  Nếu người dùng chưa xác nhận email, chuyển người dùng tới trang xác nhận email mỗi khi đăng nhập.  Nếu người dùng đã xác nhận email, lần đăng nhập kế tiếp chuyển người dùng tới trang chủ.  Kết thúc use case |
| Điều kiện ngoại lệ | Không có | |

### Use case đăng xuất

Use case đăng xuất biểu hiện cho hành động người dùng đăng xuất khỏi ứng dụng Mindify. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.3 Đặc tả use case đăng xuất

|  |  |  |
| --- | --- | --- |
| Tên use case: | Sign out | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Khi người dùng muốn đăng xuất khỏi ứng dụng. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập thành công vào ứng dụng | |
| Điều kiện sau: | Quay về trang đăng nhập của ứng dụng | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Người dùng nhấn vào nút đăng xuất ở trang infor | Hệ thống ghi nhận yêu cầu đăng xuất của người dùng và gửi lên FirebaseAuth.  FirebaseAuth xác nhận yêu cầu và thoát phiên đăng nhập.  Ứng dụng chuyển người dùng về trang đăng nhập.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case quên mật khẩu

Use case quên mật khẩu biểu hiện cho hành động người dùng quên mật khẩu và muốn tạo lại mật khẩu mới. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.4 Đặc tả use case quên mật khẩu

|  |  |  |
| --- | --- | --- |
| Tên use case: | Forgot password | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng quên mật khẩu để đăng nhập ứng dụng và muốn tạo lại mật khẩu để đăng nhập lại. | |
| Điều kiện tiên quyết: | Tài khoản người dùng tổn tại trong hệ thống. | |
| Điều kiện sau: | Người dùng tạo lại được mật khẩu mới để đăng nhập vào ứng dụng | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Trên màn hình đăng nhập, người dùng nhấn vào nút Forgot password.  Người dùng nhập email tài khoản cần cấp lại mật khẩu mới.  Người dùng kiểm tra hòm thư của email và bấm vào đường dẫn trong nội dung.  Nhập mật khẩu đặt lại.  Nhấn Submit. | Hệ thống chuyển người dùng đến trang Forgot password.  FirebaseAuth kiểm tra email có tồn tại trong hệ thống không  Nếu tồn tại thì gửi một mail về email để người dùng xác nhận và đặt lại mật khẩu mới.  FirebaseAuth xác nhận và đổi lại mật khẩu của tài khoản ứng với mật khẩu đã đặt lại từ phía người dùng.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có. | |

### Use case đăng nhập bằng Google

Use case đăng nhập bằng Google biểu hiện cho hành động người dùng có thể vào được ứng dụng thông qua tài khoản Google của mình mà không cần phải trải qua bước đăng ký tài khoản. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.5 Đặc tả use case đăng nhập bằng Google

|  |  |  |
| --- | --- | --- |
| Tên use case: | Sign in with Google | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng đăng nhập vào ứng dụng thông qua phương thức đăng nhập bằng Google | |
| Điều kiện tiên quyết: | Tài khoản Google người dùng tồn tại. | |
| Điều kiện sau: | Người dùng đăng nhập vào ứng dụng thành công | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Trên màn hình đăng nhập, người dùng nhấn vào nút Sign in with Google.  Người dùng nhập email và password để đăng nhập vào Google.  Người dùng nhấn Sign in trên browser web. | Hệ thống chuyển người dùng browser web để đăng nhập Google  FirebaseAuth kiểm tra thông tin  Nếu đăng nhập thành công và đăng nhập lần đầu thì thêm thông tin người dùng vào FirebaseAuth và Firebase Firestore, sau đó thì chuyển người dùng tới trang chủ của ứng dụng.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có. | |

### Use case quản lý thông tin cá nhân

Use case quản lý thông tin cá nhân biểu hiện việc người dùng có thể chỉnh sửa các thông tin cá nhân của bản thân mình như tên, mật khẩu hoặc avatar. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.6 Đặc tả use case quản lý thông tin cá nhân

|  |  |  |
| --- | --- | --- |
| Tên use case: | Manage Information | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng chỉnh sửa thông tin cá nhân của bản thân. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào hệ thống | |
| Điều kiện sau: | Thông tin cá nhân của người dùng được cập nhật lại nếu có sự thay đổi. | |
| Điều kiện thoát: | Người dùng nhấn Done nếu có sự thay đổi hoặc thoát ra bình thường nếu không có sự thay đổi nào. | |
| Luồng hoạt động | Actor | System |
| Trên màn hình Profile, người dùng nhấn vào biểu tượng Settings (răng cưa).  Người dùng nhấn vào Edit Profile.  Người dùng chỉnh sửa thông tin cá nhân (tên, avatar, mật khẩu) và nhấn Done.  Người dùng thấy thông tin đã được cập nhật. | Hệ thống chuyển người dùng đến trang Settings.  Ứng dụng chuyển người dùng đến trang Edit Profile.  Ứng dụng xác nhận và gửi yêu cầu cập nhật lên hệ thống.  Nếu mật khẩu nhập lại của người dùng không khớp, thông báo lỗi và yêu cầu người dùng cập nhật lại.  Hệ thống thay đổi thông tin của người dùng, lưu thông tin xuống Firebase.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có. | |

### Use case chọn chủ đề học yêu thích

Use case chọn chủ đề yêu thích giúp cho ứng dụng sẽ đưa ra những khóa học thuộc về những chủ đề mà người dùng yêu thích và hiển thị lên trang chủ. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.7 Đặc tả use case chọn chủ đề yêu thích

|  |  |  |
| --- | --- | --- |
| Tên use case: | Choose favorite categories | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng lựa chọn những chủ đề học mà mình yêu thích | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào hệ thống. | |
| Điều kiện sau: | Hệ thống ghi nhận và hiển thị những khóa học thuộc về chủ đề yêu thích của người dùng lên trang chủ. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Người dùng vào tab Favorite tại trang View Profile.  Người dùng bấm nút add category.  Người dùng chọn các chủ đề yêu thích.  Người dùng quay lại trang chủ sẽ xuất hiện thêm các khóa học dựa trên các chủ đề người dùng đã thêm trước đó. | Hệ thống chuyển người dùng đến trang thêm các category.  Hệ thống ghi nhận các chủ đề đó và ghi lại vào cơ sở dữ liệu.  Từ các chủ đề đó, hệ thống sẽ lấy các khóa học dựa trên chủ đề và cho xuất hiện lên trang chủ.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có. | |

### Use case lưu khóa học yêu thích

Use case lưu khóa học yêu thích nhằm mục đích lưu lại các khóa học mà người dùng yêu thích, để sau này người dùng có thể ghi danh và học khóa học đó. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.8 Đặc tả use case lưu khóa học yêu thích

|  |  |  |
| --- | --- | --- |
| Tên use case: | Save to favorite course | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng thêm các khóa học mình yêu thích vào danh sách lưu. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào hệ thống | |
| Điều kiện sau: | Hệ thống ghi nhận và lưu các khóa học đó vào danh sách đã lưu cho người dùng. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Trên trang chủ, người dùng yêu thích khóa học nào có thể bấm vào biểu tượng bookmark đại diện cho thao tác lưu khóa học vào yêu thích.  Hoặc người dùng có thể bấm vào khóa học mình muốn lưu, ở trong chi tiết khóa học cũng có một biểu tượng bookmark để phục vụ cho thao tác lưu khóa học vào yêu thích cho người dùng.  Để biết khóa học nào mà mình đã lưu, người dùng sẽ vào trang Saved Classes ở Profile tab của ứng dụng. | Hệ thống ghi nhận và lưu id của khóa học đó vào cho người dùng.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có. | |

### Use case xem chi tiết thông tin của giảng viên

Use case xem chi tiết thông tin của giảng viên nhằm mục đích cho người dùng biết được thông tin của giảng viên đó, các khóa học mà giảng viên đã đăng tải, số người dùng đang theo dõi và số người theo dõi của giảng viên đó. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.9 Đặc tả use case xem chi tiết thông tin giảng viên

|  |  |  |
| --- | --- | --- |
| Tên use case: | Instructor detail | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng vào một trang chi tiết khóa học bất kỳ, bấm vào biểu tượng ảnh đại diện hoặc tên của giảng viên, ứng dụng sẽ dẫn người dùng tới trang chi tiết thông tin của giảng viên đó. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào hệ thống và đang ở trang course detail. | |
| Điều kiện sau: | Người dùng xem được thông tin của giảng viên đó, bao gồm các khóa học của giảng viên, số lượng người theo dõi và đang theo dõi của giảng viên đó, đồng thời cũng xem được danh sách theo dõi và đang theo dõi của giảng viên. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Tại trang chi tiết khóa học, người dùng bấm vào hình đại diện hoặc tên giảng viên của khóa học đó.  Tại trang chi tiết giảng viên, người dùng có thể xem các thông tin như các khóa học mà giảng viên đã đăng tải, số lượng người theo dõi và đang theo dõi của giảng viên đó, đồng thời cũng xem được danh sách theo dõi và đang theo dõi của giảng viên. | Ứng dụng chuyển người dùng đến trang chi tiết giảng viên.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có. | |

### Use case ghi danh khóa học

Use case ghi danh khóa học là use case cho người dùng mua khóa học, khi mua khóa học, người dùng có nhiều đặc quyền như xem toàn bộ video của khóa học đó, tải video về máy, tạo ghi chú, được thảo luận trên diễn đàn, được nộp project của khóa học và được nhận phản hồi đánh giá từ giảng viên. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.10 Đặc tả use case ghi danh khóa học

|  |  |  |
| --- | --- | --- |
| Tên use case: | Enroll to course | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng vào một trang chi tiết khóa học bất kỳ, bấm purchase và thực hiện thanh toán để ghi danh vào khóa học | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào hệ thống và đang ở trang course detail, với điều kiện khóa học đó chưa được người dùng mua. | |
| Điều kiện sau: | Người dùng được hưởng các đặc quyền mà chỉ khi ghi danh vào khóa học mới có. | |
| Điều kiện thoát: | Người dùng thanh toán thất bại hoặc hủy thanh toán. | |
| Luồng hoạt động | Actor | System |
| Tại trang chi tiết khóa học, người dùng bấm vào hình nút purchase.  Người dùng bắt đầu mua khóa học, nhập các thông tin cần thiết. Nếu thông tin hợp lệ sẽ cho bấm nút payment của sheet điền thông tin thanh toán.  Tại trang chi tiết khóa học, người dùng được hưởng các đặc quyền mà chỉ khi ghi danh vào khóa học mới có. | Hệ thống sẽ dẫn người dùng tới trang để thanh toán khóa học.  Nếu tiền thanh toán đủ, sẽ cho bấm pay của sheet. Ứng dụng ghi nhận và trả về phía hệ thống, hệ thống sẽ ghi nhận và lưu thông tin thanh toán vào Firebase, đồng thời cũng tạo ra document ghi danh của người dùng với khóa học đó. Sau đó, ứng dụng đưa người dùng về trang chi tiết khóa học.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có. | |

### Use case tìm kiếm khóa học

Use case tìm kiếm khóa học giúp người dùng tìm được các khóa học theo tên mình mong muốn. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.11 Đặc tả use case tìm kiếm khóa học

|  |  |  |
| --- | --- | --- |
| Tên use case: | Search course | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng vào trang tìm kiếm, nhập nội dung tìm kiếm trên thanh tìm kiếm, ứng dụng ghi nhận thông tin và trả về kết quả là các khóa học có nội dung tương tự nội dung tìm kiếm. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào hệ thống và đang ở trang search page. | |
| Điều kiện sau: | Các khóa học hiển thị ra theo nội dung người dùng nhập vào | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Tại trang chi tiết khóa học, người dùng bấm vào hình nút purchase.  Người dùng bắt đầu mua khóa học, nhập các thông tin cần thiết. Nếu thông tin hợp lệ sẽ cho bấm nút payment của sheet điền thông tin thanh toán.  Tại trang chi tiết khóa học, người dùng được hưởng các đặc quyền mà chỉ khi ghi danh vào khóa học mới có. | Hệ thống sẽ dẫn người dùng tới trang để thanh toán khóa học.  Nếu tiền thanh toán đủ, sẽ cho bấm pay của sheet. Ứng dụng ghi nhận và trả về phía hệ thống, hệ thống sẽ ghi nhận và lưu thông tin thanh toán vào Firebase, đồng thời cũng tạo ra document ghi danh của người dùng với khóa học đó. Sau đó, ứng dụng đưa người dùng về trang chi tiết khóa học.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có. | |

### Use case xem các bài học trong khóa học

Use case xem các bài học trong khóa học đã ghi danh mô tả cách người dùng xem các bài học của một khóa học trong ứng dụng học tập. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.12 Đặc tả use case xem các bài học trong khóa học

|  |  |  |
| --- | --- | --- |
| Tên use case: | Watch lessons | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng đã ghi danh khóa học có thể xem danh sách các bài học trong khóa học, xem chi tiết bài học và đánh dấu hoàn thành bài học. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào ứng dụng và đã ghi danh vào khóa học thành công. | |
| Điều kiện sau: | Người dùng xem xong bài học và có thể tiếp tục học các bài học khác hoặc làm các bài kiểm tra liên quan đến khóa học. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Tại trang chi tiết khóa học với điều kiện là người dùng đã ghi danh vào khóa học. Người dùng có thể thấy danh sách các bài học, bao gồm tiêu đề bài học, thời lượng, và trạng thái hoàn thành của từng bài học.  Người dùng chọn một bài học cụ thể để xem chi tiết.  Người dùng có thể xem lại danh sách các bài học và tiếp tục chọn bài học khác để xem. | Hệ thống hiển thị nội dung bài học và cho phép người dùng xem video bài học.  Hệ thống đánh dấu hoàn thành nếu người dùng đã coi xong bài học đó và hiển thị lên hệ thống.  Chuyển đến video bài theo bài học tương ứng mà người dùng chọn.  Kết thúc use case. |
| Điều kiện ngoại lệ | Nếu người dùng chưa ghi danh khóa học, họ chỉ có thể xem trước bài học đầu tiên của khóa học. | |

### Use case ghi chú trong khóa học

Use case ghi chú giúp người dùng có thể thêm, chỉnh sửa, xóa và xem các ghi chú liên quan đến các bài học cụ thể trong khóa học. Ghi chú được lưu trữ trong Firestore và người dùng có thể điều hướng đến các mốc thời gian cụ thể trong video dựa trên ghi chú. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.13 Đặc tả use case ghi chú trong khóa học

|  |  |  |
| --- | --- | --- |
| Tên use case: | Note the lessons | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng có thể thêm, chỉnh sửa, xóa và xem các ghi chú liên quan đến các bài học cụ thể trong khóa học. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào ứng dụng và đã ghi danh vào khóa học thành công. | |
| Điều kiện sau: | Hệ thống lưu trữ các ghi chú mới, chỉnh sửa và xóa vào cơ sở dữ liệu.  Người dùng có thể điều hướng đến các mốc thời gian cụ thể trong video dựa trên ghi chú. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Người dùng truy cập vào tab ghi chú của bài học trong khóa học.  Người dùng thêm ghi chú:   * Người dùng nhấn vào nút "Add note". * Người dùng nhập nội dung ghi chú và nhấn lưu.   Người dùng chỉnh sửa ghi chú:   * Người dùng nhấn vào nút chỉnh sửa (biểu tượng cây bút) bên cạnh ghi chú. * Người dùng chỉnh sửa nội dung và nhấn lưu.   Người dùng xóa ghi chú:   * Người dùng nhấn vào nút xóa (biểu tượng thùng rác) bên cạnh ghi chú. * Người dùng xác nhận xóa.   Người dùng xem ghi chú:   * Người dùng có thể xem danh sách các ghi chú trong tab ghi chú. * Ghi chú hiển thị các thông tin: nội dung, tiêu đề bài học, thời gian trong video. * Người dùng nhấn vào ghi chú để điều hướng đến thời điểm tương ứng trong video. | Thêm ghi chú:   * Hệ thống mở một ô nhập liệu để người dùng nhập nội dung ghi chú. * Hệ thống ghi nhận và lưu trữ vào cơ sở dữ liệu. * Hệ thống cập nhật giao diện để hiển thị ghi chú mới.   Chỉnh sửa ghi chú:   * Hệ thống mở ô nhập liệu với nội dung ghi chú hiện tại. * Hệ thống cập nhật nội dung ghi chú trong Firestore. * Hệ thống cập nhật giao diện để hiển thị nội dung mới của ghi chú.   Xóa ghi chú:   * Hệ thống xóa ghi chú khỏi Firestore. * Hệ thống cập nhật giao diện để xóa ghi chú khỏi danh sách. * Hệ thống hiển thị thông báo xác nhận xóa.   Kết thúc use case. |
| Điều kiện ngoại lệ | Nếu người dùng chưa ghi danh khóa học, họ không thể ghi chú vào khóa học. | |

### Use case nộp dự án khóa học

Use case nộp dự án khóa học giúp cho những người đã mua khóa học có thể gửi dự án mà giảng viên đề ra trong khóa học, bao gồm tiêu đề, mô tả, ảnh bìa và các tệp nội dung liên quan như ảnh và tài liệu, đến khóa học đó. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.14 Đặc tả use case nộp dự án khóa học

|  |  |  |
| --- | --- | --- |
| Tên use case: | Submit project | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng đã ghi danh khóa học có thể gửi dự án của họ, bao gồm tiêu đề, mô tả, ảnh bìa và các tệp nội dung liên quan như ảnh và tài liệu, đến khóa học. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào ứng dụng và đã ghi danh vào khóa học đó. | |
| Điều kiện sau: | Dự án được gửi thành công và lưu trữ trong hệ thống. | |
| Ràng buộc | Người dùng phải chọn ít nhất một ảnh bìa và một tệp nội dung để dự án có thể được gửi.  Người dùng cần có kết nối mạng ổn định để tải lên tệp. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| 1. Chọn chức năng "Submit Project" từ giao diện khóa học.  3. Điền vào tiêu đề và mô tả dự án.   * Tiêu đề: Bắt buộc. * Mô tả: Bắt buộc.   4. Chọn và tải lên ảnh bìa dự án.  5. Chọn và tải lên các tệp nội dung (ảnh và tài liệu).  6. Nhấn nút "Submit Project". | 2. Hệ Thống hiển thị form nhập thông tin dự án.  7. Hệ thống ghi nhận và tiến hành thực hiện các bước:   * Kiểm tra tính hợp lệ của thông tin nhập liệu. * Tải lên ảnh bìa đến Firebase Storage. * Tải lên các tệp nội dung đến Firebase Storage. * Lưu trữ metadata của dự án trong Firestore Database.   8. Hệ thống thông báo cho người dùng về kết quả của việc gửi dự án (thành công hoặc thất bại).  Kết thúc use case. |
| Điều kiện ngoại lệ | Nếu người dùng chưa ghi danh khóa học, họ không thể nộp dự án. | |

### Use case làm các bài kiểm tra

Use case làm kiểm tra cho phép người dùng tham gia và làm các bài kiểm tra (quiz) trong khóa học mà họ đã đăng ký. Người dùng có thể chọn một bài kiểm tra cụ thể, xem thông tin chi tiết và bắt đầu làm bài kiểm tra. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.15 Đặc tả use case làm các bài kiểm tra

|  |  |  |
| --- | --- | --- |
| Tên use case: | Do quizzes | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng đã ghi danh khóa học có thể xem danh sách các bài học trong khóa học, xem chi tiết bài học và đánh dấu hoàn thành bài học. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào ứng dụng và đã ghi danh vào khóa học thành công. Khóa học có sẵn các bài kiểm tra. | |
| Điều kiện sau: | Kết quả bài kiểm tra được lưu lại trong hệ thống. Người dùng có thể xem lại kết quả và chi tiết các câu trả lời đúng/sai. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Người dùng đăng nhập vào hệ thống.  Người dùng chọn một khóa học mà họ đã đăng ký.  Người dùng điều hướng đến phần bài kiểm tra của khóa học.  Người dùng chọn một bài kiểm tra cụ thể.  Người dùng bắt đầu làm bài kiểm tra.  Người dùng trả lời từng câu hỏi trong bài kiểm tra.  Người dùng hoàn thành bài kiểm tra và nộp bài. | Hệ thống hiển thị danh sách các bài kiểm tra có sẵn trong khóa học.  Hệ thống hiển thị thông tin chi tiết về bài kiểm tra (tên, số lượng câu hỏi, thời gian làm bài).  Hệ thống ghi nhận các câu trả lời của người dùng.  Hệ thống chấm điểm bài kiểm tra và hiển thị kết quả cho người dùng.  Kết quả bài kiểm tra được lưu lại trong hệ thống.  Kết thúc use case. |
| Điều kiện ngoại lệ | Nếu người dùng chưa ghi danh khóa học, họ không thể làm kiểm tra. | |

### Use case thảo luận trên diễn đàn

Use case thảo luận trên diễn đàn giúp người dùng có thể tham gia vào các cuộc thảo luận trong khóa học, bao gồm đăng bình luận mới và trả lời các bình luận hiện có. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.16 Đặc tả use case thảo luận trên diễn đàn

|  |  |  |
| --- | --- | --- |
| Tên use case: | Discussion forum | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng có thể tham gia vào các cuộc thảo luận trong khóa học, bao gồm đăng bình luận mới và trả lời các bình luận hiện có. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào ứng dụng và đã ghi danh vào khóa học thành công. | |
| Điều kiện sau: | Bình luận hoặc trả lời của người dùng được lưu trữ trong Firestore và hiển thị trong phần thảo luận của khóa học. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Người dùng mở trang thảo luận của một khóa học.  Người dùng nhập nội dung bình luận vào ô nhập liệu.  Người dùng nhấn nút gửi.  Luồng trả lời bình luận:   * Người dùng chọn trả lời một bình luận hiện có. * Ô nhập liệu thay đổi trạng thái thành "Replying...". * Người dùng nhập nội dung trả lời vào ô nhập liệu. * Người dùng nhấn nút gửi. | Hệ thống hiển thị danh sách các bình luận hiện có.  Hệ thống lưu trữ bình luận vào Firestore và cập nhật giao diện để hiển thị bình luận mới.  Luồng trả lời bình luận:   * Hệ thống lưu trữ trả lời vào Firestore dưới bình luận tương ứng và cập nhật giao diện để hiển thị trả lời mới.   Kết thúc use case. |
| Điều kiện ngoại lệ | Nếu người dùng chưa ghi danh khóa học, họ không thể tham gia bình luận. | |

### Use case thêm khóa học

Use case cho phép giảng viên có thể thêm khóa học mới vào hệ thống quản lý khóa học của mình. Khóa học này sẽ bao gồm thông tin chi tiết như tên khóa học, mô tả, bài giảng video, câu đố và danh mục liên quan. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.17 Đặc tả use case thêm khóa học

|  |  |  |
| --- | --- | --- |
| Tên use case: | Add course | |
| Tác nhân: | Instructor | |
| Mô tả: | Giảng viên có thể thêm khóa học mới vào hệ thống quản lý khóa học của mình. Khóa học này sẽ bao gồm thông tin chi tiết như tên khóa học, mô tả, ảnh bìa, dự án, bài giảng video, bài kiểm tra và tài liệu liên quan. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào ứng dụng và có vai trò là giảng viên. | |
| Điều kiện sau: | Khóa học mới đã được thêm vào hệ thống quản lý khóa học.  Thông tin khóa học được lưu trữ trong Firestore.  Giảng viên có thể xem, chỉnh sửa hoặc xuất bản khóa học bất cứ lúc nào. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| * 1. Giảng viên chọn tùy chọn “New course” từ menu quản lý khóa học.   3. Giảng viên nhập thông tin chi tiết khóa học, bao gồm:  Tên khóa học, mô tả, mô tả dự án khóa học, giá tiền, ảnh bìa dự án.  Thêm các bài kiểm tra nếu muốn.  4. Người dùng nhập xong bấm Save. Khi này giảng viên được chuyển tới trang quản lí bài giảng, có thể:  Đăng tải các video bài giảng.  Đăng tải các tài nguyên liên quan đến khóa học và dự án cho khóa học | * 1. Hệ thống hiển thị chi tiết khóa học, bao gồm danh sách các bài giảng video.   5. Hệ thống ghi nhận thông tin và lưu vào FirebaseFirestore.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case thêm khóa học

Use case cho phép giảng viên có thể gửi yêu cầu đăng tải khóa học để hệ thống duyệt và cho xuất hiện lên trên ứng dụng. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.18 Đặc tả use case đăng tải khóa học

|  |  |  |
| --- | --- | --- |
| Tên use case: | Add course | |
| Tác nhân: | Instructor | |
| Mô tả: | Giảng viên có thể gửi yêu cầu đăng tải khóa học để hệ thống duyệt và cho xuất hiện lên trên ứng dụng. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào ứng dụng và có vai trò là giảng viên. | |
| Điều kiện sau: | Khóa học được gửi yêu cầu và ở trạng thái chờ duyệt (pending)  Chi tiết khóa học, bao gồm ảnh bìa, giá và tài nguyên bổ sung, được lưu trong Firestore. Giảng viên có thể xem, chỉnh sửa khóa học bất cứ lúc nào. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Bước 1: Giảng viên chọn khóa học mà họ muốn xuất bản.  Bước 3: Giảng viên kiểm tra lại các thông tin khóa học thật kỹ, sau đó bấm Publish Course. | Bước 2: Hệ thống hiển thị chi tiết khóa học, bao gồm danh sách các bài giảng video.  Bước 4: Hệ thống ghi nhận thông tin và lưu vào FirebaseFirestore. Lúc này một yêu cầu duyệt được tạo ra và sẽ gửi về phía admin để đợi duyệt khóa học.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case lịch sử xem

Use case cho phép người dùng theo dõi tiến trình xem video của mình trong một khóa học. Khi người dùng xem một video, thời gian xem hiện tại sẽ được lưu lại để khi người dùng quay lại, họ có thể tiếp tục xem từ thời điểm đã dừng trước đó. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.19 Đặc tả use case lịch sử xem

|  |  |  |
| --- | --- | --- |
| Tên use case: | Watched history | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng có thể theo dõi tiến trình xem video của mình trong một khóa học. Khi người dùng xem một video, thời gian xem hiện tại sẽ được lưu lại để khi người dùng quay lại, họ có thể tiếp tục xem từ thời điểm đã dừng trước đó. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào ứng dụng. | |
| Điều kiện sau: | Thời gian xem video hiện tại của người dùng được lưu lại thành công.  Khi người dùng quay lại xem video, video sẽ được phát tiếp từ thời điểm đã lưu trước đó. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Người dùng chọn một khóa học từ danh sách khóa học đã đăng ký.  Người dùng chọn một video để xem.  Người dùng xem video.  Người dùng dừng xem video hoặc chuyển sang bài giảng khác. | Hệ thống hiển thị chi tiết khóa học, bao gồm danh sách các bài giảng video.  Hệ thống kiểm tra lịch sử xem của người dùng và lấy thời gian xem gần nhất từ cơ sở dữ liệu.  Hệ thống phát video từ thời điểm đã lưu trước đó.  Hệ thống lưu lại thời gian hiện tại của video vào cơ sở dữ liệu.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case quản lý thư mục

Use case mô tả các bước người dùng thực hiện để quản lý thư mục của mình, bao gồm tạo mới, xem chi tiết, thêm khóa học vào thư mục và xóa thư mục. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.20 Đặc tả use case quản lý thư mục

|  |  |  |
| --- | --- | --- |
| Tên use case: | Folder Management | |
| Tác nhân: | User, Instructor | |
| Mô tả: | Người dùng có thể theo dõi tiến trình xem video của mình trong một khóa học. Khi người dùng xem một video, thời gian xem hiện tại sẽ được lưu lại để khi người dùng quay lại, họ có thể tiếp tục xem từ thời điểm đã dừng trước đó. | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào ứng dụng. | |
| Điều kiện sau: | Thời gian xem video hiện tại của người dùng được lưu lại thành công.  Khi người dùng quay lại xem video, video sẽ được phát tiếp từ thời điểm đã lưu trước đó. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Tạo thư mục:   * Người dùng nhấn vào nút “+” trên giao diện Quản lý thư mục. * Người dùng nhập tên thư mục vào trường tương ứng. * Người dùng nhấn vào nút “Create”.   Xem chi tiết thư mục:   * Người dùng nhấn vào một thư mục cụ thể trong danh sách thư mục.   Thêm khóa học vào thư mục:   * Người dùng nhấn vào biểu tượng thêm khóa học trong giao diện khóa học. * Người dùng chọn một thư mục từ danh sách. | Tạo thư mục:   * Hệ thống hiển thị hộp thoại tạo thư mục mới. * Hệ thống tạo thư mục mới với tên do người dùng nhập và liên kết với người dùng hiện tại. * Hệ thống hiển thị thông báo thành công và cập nhật danh sách thư mục.   Xem chi tiết thư mục:   * Hệ thống chuyển hướng người dùng đến trang chi tiết thư mục. * Hệ thống hiển thị các thông tin chi tiết của thư mục và danh sách các khóa học trong thư mục.   Thêm khóa học vào thư mục:   * Hệ thống hiển thị danh sách các thư mục của người dùng. * Hệ thống thêm khóa học vào thư mục được chọn. * Hệ thống hiển thị thông báo thành công.   Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case gửi yêu cầu trở thành giảng viên

Use case này giúp người dùng có thể gửi yêu cầu trở thành giảng viên và thông tin này sẽ được lưu trữ trong hệ thống để quản trị viên xem xét và phê duyệt. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.21 Đặc tả use case yêu cầu trở thành giảng viên

|  |  |  |
| --- | --- | --- |
| Tên use case: | Send instructor request | |
| Tác nhân: | User | |
| Mô tả: | Người dùng có thể gửi yêu cầu trở thành giảng viên và thông tin này sẽ được lưu trữ trong hệ thống để quản trị viên xem xét và phê duyệt | |
| Điều kiện tiên quyết: | Người dùng đã đăng nhập vào ứng dụng. | |
| Điều kiện sau: | Yêu cầu trở thành giảng viên của người dùng được lưu trữ thành công trong hệ thống Firebase.  Trạng thái yêu cầu của người dùng được cập nhật. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Người dùng mở ứng dụng và điều hướng đến trang đăng ký giảng viên.  Người dùng điền đầy đủ thông tin vào các trường yêu cầu trong biểu mẫu.  Người dùng nhấn nút "Send" để gửi yêu cầu.  Người dùng được chuyển đến trang thông báo gửi yêu cầu thành công | Hệ thống hiển thị biểu mẫu đăng ký giảng viên bao gồm các thông tin cần thiết như danh mục môn học, thông tin cá nhân và mô tả lớp học.  Hệ thống kiểm tra tính hợp lệ của thông tin nhập vào.  Nếu thông tin hợp lệ, hệ thống sẽ:   * Tạo một đối tượng dữ liệu chứa thông tin của yêu cầu. * Gửi yêu cầu này đến dịch vụ Firebase thông qua UserService. * Cập nhật trạng thái yêu cầu của người dùng trong hệ thống.   Hệ thống hiển thị thông báo xác nhận rằng yêu cầu đã được gửi thành công.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case xem các yêu cầu trở thành giảng viên

Use case này giúp quản trị viên có thể xem danh sách các yêu cầu trở thành giảng viên để đánh giá và xử lý các yêu cầu này. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.22 Đặc tả use case xem các request thành giảng viên

|  |  |  |
| --- | --- | --- |
| Tên use case: | Send instructor request | |
| Tác nhân: | Administrator | |
| Mô tả: | Quản trị viên có thể xem danh sách các yêu cầu trở thành giảng viên để đánh giá và xử lý các yêu cầu này. | |
| Điều kiện tiên quyết: | Quản trị viên đã đăng nhập vào hệ thống.  Hệ thống đã có các yêu cầu từ người dùng được lưu trữ. | |
| Điều kiện sau: | Quản trị viên có thể xem danh sách các yêu cầu giảng viên và điều hướng đến trang chi tiết của từng yêu cầu. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Quản trị viên mở web quản trị và điều hướng đến trang danh sách yêu cầu giảng viên.  Quản trị viên có thể nhìn thấy thông tin của các yêu cầu bao gồm họ và tên, email, danh mục môn học, và trạng thái của yêu cầu.  Quản trị viên nhấp vào nút "View Details" để xem chi tiết yêu cầu giảng viên. | Hệ thống gửi yêu cầu để lấy dữ liệu yêu cầu giảng viên.  Hệ thống nhận phản hồi từ API chứa danh sách các yêu cầu giảng viên.  Hệ thống hiển thị danh sách yêu cầu giảng viên trong một bảng.  Hệ thống điều hướng quản trị viên đến trang chi tiết yêu cầu với thông tin chi tiết về yêu cầu giảng viên đó.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case phê duyệt yêu cầu trở thành giảng viên

Use case này giúp quản trị viên có thể phê duyệt yêu cầu trở thành giảng viên của người dùng. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.23 Đặc tả use case phê duyệt yêu cầu trở thành giảng viên

|  |  |  |
| --- | --- | --- |
| Tên use case: | Approve instructor request | |
| Tác nhân: | Administrator | |
| Mô tả: | Tại trang chi tiết yêu cầu, quản trị viên có thể phê duyệt yêu cầu trở thành giảng viên của người dùng. | |
| Điều kiện tiên quyết: | Quản trị viên đã đăng nhập vào hệ thống.  Quản trị viên đang ở trang chi tiết yêu cầu giảng viên. | |
| Điều kiện sau: | Yêu cầu trở thành giảng viên được phê duyệt và trạng thái được cập nhật trên hệ thống.  Người dùng nhận được email thông báo về việc yêu cầu của họ được phê duyệt. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Quản trị viên mở trang chi tiết yêu cầu giảng viên.  Quản trị viên nhấp vào nút "Approve". | Hệ thống hiển thị thông tin chi tiết về yêu cầu.  Hệ thống gửi yêu cầu.  Hệ thống cập nhật trạng thái của yêu cầu giảng viên trên giao diện người dùng.  Hệ thống hiển thị thông báo "Sent email successfully."  Sau 2 giây, hệ thống tự động điều hướng quản trị viên trở lại trang danh sách yêu cầu.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case từ chối yêu cầu trở thành giảng viên

Use case này giúp quản trị viên có thể từ chối yêu cầu trở thành giảng viên của người dùng. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.24 Đặc tả use case từ chối yêu cầu trở thành giảng viên

|  |  |  |
| --- | --- | --- |
| Tên use case: | Reject instructor request | |
| Tác nhân: | Administrator | |
| Mô tả: | Tại trang chi tiết yêu cầu, quản trị viên có thể từ chối yêu cầu trở thành giảng viên của người dùng. | |
| Điều kiện tiên quyết: | Quản trị viên đã đăng nhập vào hệ thống.  Quản trị viên đang ở trang chi tiết yêu cầu giảng viên. | |
| Điều kiện sau: | Yêu cầu trở thành giảng viên bị từ chối và trạng thái được cập nhật trên hệ thống.  Người dùng nhận được email thông báo về việc yêu cầu của họ bị từ chối cùng với lý do từ chối. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Quản trị viên mở trang chi tiết yêu cầu giảng viên.  Quản trị viên nhấp vào nút "Reject".  Quản trị viên nhập lý do từ chối và nhấp vào nút "Submit". | Hệ thống hiển thị thông tin chi tiết về yêu cầu.  Hệ thống hiển thị popup yêu cầu nhập lý do từ chối.  Hệ thống gửi yêu cầu.  Hệ thống cập nhật trạng thái của yêu cầu giảng viên trên giao diện người dùng.  Hệ thống hiển thị thông báo "Sent email successfully."  Sau 2 giây, hệ thống tự động điều hướng quản trị viên trở lại trang danh sách yêu cầu.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case thống kê

Use case này quản trị viên có thể xem thống kê số lượng ghi danh, doanh thu theo tháng trong một năm cụ thể. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.25 Đặc tả use case thống kê

|  |  |  |
| --- | --- | --- |
| Tên use case: | Statistic | |
| Tác nhân: | Administrator | |
| Mô tả: | Quản trị viên có thể xem thống kê số lượng ghi danh theo tháng trong một năm cụ thể. | |
| Điều kiện tiên quyết: | Quản trị viên đã đăng nhập vào hệ thống.  Dữ liệu về số lượng ghi danh theo tháng đã được lưu trữ trong hệ thống. | |
| Điều kiện sau: | Quản trị viên xem được thống kê số lượng ghi danh theo tháng dưới dạng biểu đồ cột. và các thông số tổng hợp khác. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Quản trị viên mở trang Dashboard.  Quản trị viên chọn bộ lọc theo tháng và chọn năm cụ thể từ danh sách. | Hệ thống hiển thị giao diện trang Dashboard với các tùy chọn bộ lọc.  Hệ thống gửi yêu cầu và nhận phản hồi từ API và cập nhật biểu đồ cột với dữ liệu số lượng ghi danh theo tháng.  Hệ thống hiển thị biểu đồ cột cho quản trị viên với các cột tương ứng với các tháng trong năm và số lượng ghi danh tương ứng.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case quản lý khóa học

Use case này cho phép quản trị viên hệ thống quản lý các khóa học bao gồm việc xem danh sách khóa học, chi tiết khóa học, và phân trang các khóa học. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.26 Đặc tả use case quản lý khóa học

|  |  |  |
| --- | --- | --- |
| Tên use case: | Course Management | |
| Tác nhân: | Administrator | |
| Mô tả: | Quản trị viên hệ thống quản lý các khóa học bao gồm việc xem danh sách khóa học, chi tiết khóa học, và phân trang các khóa học. | |
| Điều kiện tiên quyết: | Quản trị viên đã đăng nhập vào hệ thống.  Có sẵn một số khóa học trong hệ thống. | |
| Điều kiện sau: | Quản trị viên có thể xem danh sách và chi tiết các khóa học, cũng như thay đổi số lượng và trang khóa học hiển thị. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Quản trị viên mở giao diện quản lý khóa học.  Quản trị viên thay đổi số lượng khóa học hiển thị.  Quản trị viên bấm nút "Previous" hoặc "Next" để điều hướng qua các trang.  Hệ thống gửi yêu cầu tới API với điểm bắt đầu tương ứng và cập nhật danh sách khóa học.  Quản trị viên xem chi tiết khóa học. | Hệ thống hiển thị danh sách khóa học.  Hệ thống điều hướng quản trị viên tới trang chi tiết khóa học của khóa học đó.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case phê duyệt khóa học

Use case này cho phép quản trị viên phê duyệt yêu cầu tạo khóa học từ người dùng. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.27 Đặc tả use case phê duyệt khóa học

|  |  |  |
| --- | --- | --- |
| Tên use case: | Approve Course Request | |
| Tác nhân: | Administrator | |
| Mô tả: | Quản trị viên phê duyệt yêu cầu tạo khóa học từ người dùng. | |
| Điều kiện tiên quyết: | Quản trị viên đã đăng nhập vào hệ thống.  Có yêu cầu khóa học từ người dùng cần được phê duyệt. | |
| Điều kiện sau: | Quản trị viên có thể xem danh sách và chi tiết các khóa học, cũng như thay đổi số lượng và trang khóa học hiển thị. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Quản trị viên xem danh sách yêu cầu khóa học.  Quản trị viên phê duyệt yêu cầu khóa học. | Hệ thống cập nhật trạng thái yêu cầu.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case từ chối yêu cầu duyệt khóa học

Use case này cho phép quản trị viên từ chối yêu cầu tạo khóa học từ người dùng. Bảng dưới đây biểu thị đặc tả use case:

|  |  |  |
| --- | --- | --- |
| Tên use case: | Reject Course Request | |
| Tác nhân: | Administrator | |
| Mô tả: | Quản trị viên từ chối yêu cầu tạo khóa học từ người dùng. | |
| Điều kiện tiên quyết: | Quản trị viên đã đăng nhập vào hệ thống.  Có yêu cầu khóa học từ người dùng cần được từ chối. | |
| Điều kiện sau: | Yêu cầu khóa học được từ chối thành công và trạng thái được cập nhật trong hệ thống. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Quản trị viên xem danh sách yêu cầu khóa học.  Quản trị viên từ chối yêu cầu khóa học. | Hệ thống cập nhật trạng thái yêu cầu.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case quản lý người dùng

Use case này cho phép quản trị viên quản lý người dùng, bao gồm việc xem danh sách người dùng, khóa/mở khóa tài khoản của người dùng, và phân trang danh sách người dùng. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.28 Đặc tả use case quản lý người dùng

|  |  |  |
| --- | --- | --- |
| Tên use case: | User Management | |
| Tác nhân: | Administrator | |
| Mô tả: | Quản trị viên quản lý người dùng, bao gồm việc xem danh sách người dùng, khóa/mở khóa tài khoản của người dùng, và phân trang danh sách người dùng. | |
| Điều kiện tiên quyết: | Quản trị viên đã đăng nhập vào hệ thống.  Có sẵn một số người dùng trong hệ thống. | |
| Điều kiện sau: | Quản trị viên có thể xem danh sách người dùng, thay đổi số lượng và trang người dùng hiển thị, và khóa/mở khóa tài khoản người dùng. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Quản trị viên mở giao diện quản lý người dùng.  Quản trị viên thay đổi số lượng người dùng hiển thị.  Quản trị viên điều hướng qua các trang người dùng.  Khóa/mở khóa tài khoản người dùng:   * Quản trị viên bấm vào nút “Lock” hoặc “Unlock” bên cạnh tài khoản người dùng cần khóa/mở khóa. * Quản trị viên xác nhận hành động khóa/mở khóa tài khoản người dùng. | Hệ thống hiển thị danh sách người dùng.  Khóa/mở khóa tài khoản người dùng:   * Hệ thống hiển thị hộp thoại xác nhận hành động khóa/mở khóa. * Hệ thống gửi yêu cầu tới API để khóa/mở khóa tài khoản người dùng. * Hệ thống cập nhật trạng thái tài khoản của người dùng thành khóa hoặc mở khóa và thông báo cho quản trị viên về việc thực hiện thành công.   Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case đăng nhập admin

Use case này cho phép quản trị viên đăng nhập vào hệ thống quản lý của họ bằng cách nhập email và mật khẩu. Nếu đăng nhập thành công, quản trị viên sẽ được điều hướng đến trang chính của hệ thống quản lý. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.29 Đặc tả use case đăng nhập admin

|  |  |  |
| --- | --- | --- |
| Tên use case: | Sign in admin | |
| Tác nhân: | Administrator | |
| Mô tả: | Quản trị viên đăng nhập vào hệ thống quản lý của họ bằng cách nhập email và mật khẩu. Nếu đăng nhập thành công, quản trị viên sẽ được điều hướng đến trang chính của hệ thống quản lý | |
| Điều kiện tiên quyết: | Quản trị viên đã có tài khoản với email và mật khẩu hợp lệ. | |
| Điều kiện sau: | Quản trị viên được đăng nhập vào hệ thống quản lý và có thể thực hiện các tác vụ quản lý. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| * Quản trị viên mở giao diện đăng nhập. * Quản trị viên nhập thông tin đăng nhập. * Quản trị viên gửi yêu cầu đăng nhập. | Hệ thống xử lý yêu cầu đăng nhập:   * Hệ thống gửi yêu cầu đăng nhập tới API với thông tin email và mật khẩu. * API xác nhận thông tin đăng nhập và phản hồi với token đăng nhập nếu thông tin hợp lệ. * Hệ thống lưu token đăng nhập vào localStorage nếu đăng nhập thành công. * Nếu thông tin đăng nhập không hợp lệ, hệ thống hiển thị thông báo lỗi “Invalid email or password. Please try again.”   Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

### Use case đăng xuất admin

Use case này cho phép quản trị viên đăng xuất khỏi hệ thống quản lý. Khi quản trị viên bấm nút đăng xuất, hệ thống sẽ xóa token đăng nhập và điều hướng quản trị viên tới trang đăng nhập. Bảng dưới đây biểu thị đặc tả use case:

Bảng 3.30 Đặc tả use case đăng xuất admin

|  |  |  |
| --- | --- | --- |
| Tên use case: | Log out admin | |
| Tác nhân: | Administrator | |
| Mô tả: | Quản trị viên đăng xuất khỏi hệ thống quản lý. Khi quản trị viên bấm nút đăng xuất, hệ thống sẽ xóa token đăng nhập và điều hướng quản trị viên tới trang đăng nhập. | |
| Điều kiện tiên quyết: | Quản trị viên đã đăng nhập vào hệ thống và đang ở trong giao diện quản lý. | |
| Điều kiện sau: | Quản trị viên được đăng xuất khỏi hệ thống và cần đăng nhập lại nếu muốn tiếp tục sử dụng. | |
| Điều kiện thoát: | Không có | |
| Luồng hoạt động | Actor | System |
| Quản trị viên yêu cầu đăng xuất. | Hệ thống xử lý yêu cầu đăng xuất. Xóa token khỏi LocalStorage.  Điều hướng tới trang đăng nhập.  Kết thúc use case. |
| Điều kiện ngoại lệ | Không có | |

## Mô hình quan hệ

Đối với cơ sở dữ liệu NoSQL như Firebase Firestore không có quy tắc thống nhất và linh hoạt hơn rất nhiều so với cơ sở dữ liệu quan hệ (SQL). Tuy nhiên để dễ hình dung được các đối tượng trong hệ thống, dưới đây là mô tả của các bảng và mối quan hệ trong mô hình cơ sở dữ liệu:

A screenshot of a computer screen

Description automatically generated

Hình 3.2 Mô hình quan hệ hệ thống

Với hệ thống trên, chúng ta có thể thấy:

* + **Collection users**: Lưu trữ thông tin người dùng bao gồm username, email, ảnh đại diện, danh sách người theo dõi và các lớp đã lưu. Có tồn tại các subcollections như watchHistories, notifications.
  + **Collection courses**: Lưu trữ thông tin về khóa học như tên khóa học, mô tả, giá,… và các subcollections như lessons, comments (bên trong chứa subcollection replies), resources, projects.
  + **Collection folders**: Lưu trữ các khóa học mà người dùng đã ghi danh và được thêm vào thư mục.
  + **Collection enrollments**: Chứa thông tin về việc ghi danh của người dùng vào các khóa học, chứa mã của giao dịch thanh toán.
  + **Collection transactions**: Lưu trữ thông tin giao dịch thanh toán của người dùng khi mua các khóa học trên ứng dụng.
  + **Collection quizzes**: Lưu trữ các bài kiểm tra do giảng viên tạo ra cho khóa học đó, khóa học có thể có hoặc không có bài kiểm tra. Collection này bên trong chứa subcollection là questions chứa câu hỏi và câu trả lời kèm theo giải thích.
  + **Collection requests**: Lưu trữ những yêu cầu người dùng đăng ký trở thành giảng viên của hệ thống.
  + **Collection course\_requests**: Lưu trữ những khóa học mà giảng viên muốn đăng tải lên ứng dụng để người quản lý phê duyệt.

## Sơ đồ lớp

Để dễ dàng hình dung các đối tượng trong hệ thống, việc sử dụng sơ đồ lớp là vô cùng cần thiết để mang được một sự thống nhất, có quy tắc thiết kế và dễ dàng bảo trì, sửa đổi nếu có sự thay đổi. Dựa trên mô hình quan hệ, chúng ta xây dựng 16 lớp: User, Course, Folder, Enrollment, Notification, Request, Quizz, Lessons, Comment, Reply, Project, Question, WatchedHistory, CourseRequest, Transaction, Resource. Dưới đây là sơ đồ lớp cho hệ thống Mindify:

A diagram of a computer program

Description automatically generated

Hình 3.3 Sơ đồ lớp của hệ thống

## Sơ đồ tuần tự

### Đăng nhập

A screenshot of a computer

Description automatically generated

Hình 3.4 Sơ đồ tuần tự chức năng đăng nhập

### Đăng nhập với Google

A screenshot of a computer

Description automatically generated

Hình 3.5 Sơ đồ tuần tự chức năng đăng nhập với Google

### Đặt lại mật khẩu

A screenshot of a computer screen

Description automatically generated

Hình 3.6 Sơ đồ tuần tự chức năng đặt lại mật khẩu

### Tìm kiếm khóa học

A screenshot of a computer

Description automatically generated

Hình 3.7 Sơ đồ tuần tự chức năng tìm kiếm khóa học

### Quản lý thư mục

A close-up of a document

Description automatically generated

Hình 3.8 Sơ đồ tuần tự chức năng quản lý thư mục

### Đăng ký

A screenshot of a computer

Description automatically generated

Hình 3.9 Sơ đồ tuần tự chức năng đăng ký

### Lịch sử xem video

A diagram of a flowchart

Description automatically generated

Hình 3.10 Sơ đồ tuần tự chức năng lịch sử xem video

### Lưu khóa học yêu thích

A screenshot of a computer

Description automatically generated

Hình 3.11 Sơ đồ tuần tự chức năng lưu khóa học yêu thích

### Quản lý thông tin cá nhân

A white paper with black text

Description automatically generated

Hình 3.12 Sơ đồ tuần tự chức năng quản lý thông tin cá nhân

### Đăng ký làm giảng viên

A screenshot of a computer

Description automatically generated

Hình 3.13 Sơ đồ tuần tự chức năng đăng ký làm giảng viên

### Theo dõi các chủ đề yêu thích

A diagram of a company

Description automatically generated

Hình 3.14 Sơ đồ tuần tự chức năng theo dõi các chủ đề yêu thích

### Xem các bài học trong khóa học

A screenshot of a computer

Description automatically generated

Hình 3.15 Sơ đồ tuần tự chức năng xem các bài học trong khóa học

### Theo dõi, xem chi tiết thông tin giảng viên

A screenshot of a computer screen

Description automatically generated

Hình 3.16 Sơ đồ tuần tự chức năng theo dõi, xem chi tiết thông tin giảng viên

### Ghi chú bài học

A screenshot of a computer

Description automatically generated

Hình 3.17 Sơ đồ tuần tự chức năng ghi chú bài học

### Thanh toán khóa học

A screenshot of a computer screen

Description automatically generated

Hình 3.18 Sơ đồ tuần tự chức năng thanh toán khóa học

### Bình luận, trả lời bình luận

A diagram of a website

Description automatically generated with medium confidence

Hình 3.19 Sơ đồ tuần tự chức năng bình luận, trả lời bình luận

### Làm các bài kiểm tra

A screenshot of a computer

Description automatically generated

Hình 3.20 Sơ đồ tuần tự chức năng làm bài kiểm tra

### Đăng nhập admin

A diagram of a software company

Description automatically generated with medium confidence

Hình 3.21 Sơ đồ tuần tự chức năng đăng nhập admin

### Quản lý các khóa học:

A screenshot of a computer

Description automatically generated

Hình 3.22 Sơ đồ tuần tự quản lý khóa học

### Quản lý người dùng

A screenshot of a computer screen

Description automatically generated

Hình 3.23 Sơ đồ tuần tự chức năng quản lý người dùng

### Thống kê

A screenshot of a computer

Description automatically generated

Hình 3.24 Sơ đồ tuần tự chức năng thống kê

### Thêm khóa học mới

A diagram of a company

Description automatically generated

Hình 3.25 Sơ đồ tuần tự chức năng thêm khóa học

### Nộp dự án khóa học

A screenshot of a computer

Description automatically generated

Hình 3.26 Sơ đồ tuần tự chức năng nộp dự án khóa học

### Gửi yêu cầu đăng tải khóa học

A document with text and images

Description automatically generated with medium confidence

Hình 3.27 Sơ đồ tuần tự chức năng gửi yêu cầu đăng tải khóa học

# THIẾT KẾ GIAO DIỆN

## 

# XÂY DỰNG ỨNG DỤNG

## Môi trường cài đặt

Cài đặt bên phía Server:

* + NodeJS: dùng để xây dựng phần backend của ứng dụng.
  + Firebase: dùng để lưu trữ dữ liệu, thực hiện về các dịch vụ như FirebaseFirestore, FirebaseAuth, FirebaseStorage.

Cài đặt bên phía Client:

* + Admin: Sử dụng nền website được xây dựng bằng ReactJS để tương tác.
  + User: Sử dụng các thiết bị di động có hệ điều hành Android hoặc iOS để sử dụng.

## Môi trường triển khai

### Triển khai mã nguồn server

Ở phần backend, mã nguồn được host lên Render – một nền tảng đám mây cung cấp dịch vụ lưu trữ và triển khai ứng dụng cho các nhà phát triển.

### Thử nghiệm ứng dụng trên Android và iOS

Với việc xây dựng ứng dụng di động dựa trên framework Flutter, để kiểm thử ứng dụng trên thiết bị Android và iOS, chúng ta sẽ thực hiện một số bước chuẩn bị và cấu hình cụ thể. Dưới đây là quy trình triển khai mã nguồn Flutter để kiểm thử trên cả hai nền tảng:

* + Đối với Android:
* Cài đặt môi trường phát triển: Cài đặt Android Studio và các công cụ cần thiết cho phát triển Android, bao gồm Android SDK và các gói liên quan.
* Kết nối thiết bị Android: Kết nối thiết bị Android, đảm bảo chế độ gỡ lỗi USB (USB Debugging) đã được bật trên thiết bị thật. Đối với thiết bị ảo (Android Emulator), chỉ cần tạo ra trong Android Studio và cho kết nối trong Visual Studio Code.
* Chạy ứng dụng trên thiết bị Android: Dùng Visual Studio Code. Sử dụng lệnh flutter run để xây dựng và chạy ứng dụng trực tiếp trên thiết bị Android.
  + Đối với iOS:
* Cài đặt môi trường phát triển: Cài đặt Xcode và các công cụ cần thiết cho phát triển iOS. Xcode chỉ khả dụng trên macOS. Cài đặt thêm máy ảo iOS (gọi là Simulator).
  + - Cấu hình dự án cho iOS: Mở dự án Flutter trong Visual Studio Code.
  + Chạy ứng dụng trên thiết bị iOS: Dùng flutter run để xây dựng và chạy chương trình trên thiết bị.

# KẾT LUẬN

## Kết quả thu được

Đối với dự án này, chúng em đã đạt được những thành tựu quan trọng sau:

* **Xây dựng ứng dụng di động đa nền tảng với Flutter:** Chúng em đã nắm vững và áp dụng thành công Flutter để phát triển một ứng dụng có thể chạy trên cả Android và iOS, giúp tiết kiệm thời gian và công sức so với việc phát triển riêng lẻ cho từng nền tảng.
* **Hiểu sâu hơn về Firebase, NodeJS, và ReactJS:** Dự án đã giúp chúng em củng cố kiến thức về các công nghệ quan trọng này, bao gồm cách thức chúng vận hành và cách tích hợp chúng để xây dựng các tính năng phức tạp trong ứng dụng.
* **Triển khai mã nguồn server thực tế:** Chúng em đã học được cách triển khai mã nguồn phía server lên môi trường thực tế, sử dụng nền tảng Render để lưu trữ và quản lý ứng dụng một cách hiệu quả và chuyên nghiệp.
* **Hoàn thiện ứng dụng và kiểm thử đa nền tảng:** Ứng dụng đã được xây dựng hoàn chỉnh bằng Flutter và đã trải qua quá trình kiểm thử trên cả thiết bị Android và iOS, đảm bảo tính ổn định và hiệu suất tốt trên cả hai hệ điều hành.

Những kết quả này không chỉ giúp chúng em nâng cao kỹ năng kỹ thuật mà còn cung cấp một trải nghiệm thực tế quý báu trong việc phát triển và triển khai ứng dụng di động.

## Mặt hạn chế

Trong quá trình thực hiện dự án, chúng em gặp phải một số vấn đề sau:

* **Số lượng requests thử nghiệm trên Firebase bị hạn chế:** do số lượt request trên FirebaseFirestore chỉ 50 nghìn lượt/ngày, việc gọi quá nhiều request sẽ dẫn tới quá tải và phải trả phí cho các lượt request sau này.
* **Nguồn tài nguyên video không quá lớn:** Do chỉ được hỗ trợ miễn phí 5GB với Firebase Storage (thuộc về Google Cloud Storage), các video bài giảng của chúng em trong ứng dụng không quá nhiều và có thể lặp đi lặp lại.
* **Kinh nghiệm lập trình di động đa nền tảng chưa quá nhiều:** việc còn thiếu kinh nghiệm trong xây dựng ứng dụng với Flutter đôi khi khiến mã nguồn quá dài và không tối ưu.
* **Khó khăn trong việc thiết kế giao diện:** Cần phải thiết kế giao diện sao cho phù hợp và dễ dàng sử dụng, dễ dàng trải nghiệm trên cả hai nền tảng.
* **Thời gian hạn chế**: với chỉ hai tháng xây dựng ứng dụng, có thể chúng em còn thiếu sót và không tối ưu về logic mã nguồn.

## Hướng phát triển

Sau đây sẽ là hướng phát triển của chúng em trong tương lai:

* **Thêm các chức năng mới:** dự kiến trong tương lai ứng dụng có thể thêm được các chức năng liên quan đến AI để theo kịp xu hướng hiện nay.
* **Cải thiện giao diện:** một số giao diện có thể chưa đẹp, chúng em sẽ ghi nhận và sửa đổi để sau này ứng dụng được mượt mà về mặt trải nghiệm và thẩm mĩ về góc nhìn.
* **Triển khai sâu hơn về CI/CD:** Chúng em sẽ tập trung vào việc hoàn thiện quy trình tích hợp liên tục và triển khai liên tục (CI/CD). Điều này sẽ giúp tối ưu hóa quy trình phát triển, giảm thiểu lỗi và đảm bảo rằng mọi bản cập nhật đều được phát hành một cách nhanh chóng và hiệu quả.
* **Nâng cao bảo mật:** Cải thiện và nâng cấp khả năng bảo mật, bảo vệ các API.

# TÀI LIỆU THAM KHẢO

Tiếng Việt

* 1. Hoàng

Tiếng Anh

* 1. Refactoring