Shape, square

Description automatically generated

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

VIỆT - HÀN

**Khoa Khoa Học Máy Tính**



ĐỒ ÁN CHUYÊN NGÀNH 1

**ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG RẠP CHIẾU PHIM ĐA NỀN TẢNG**

Sinh viên thực hiện: **NGUYỄN ĐĂNG QUÝ**

Lớp: **19IT4**

Giảng viên hướng dẫn: **THS. TRỊNH THỊ NGỌC LINH**

***Đà nẵng, tháng 12 năm 2022***

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

VIỆT - HÀN

**Khoa Khoa Học Máy Tính**



ĐỒ ÁN CHUYÊN NGÀNH 1

**ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG RẠP CHIẾU PHIM ĐA NỀN TẢNG**

Sinh viên thực hiện: **NGUYỄN ĐĂNG QUÝ**

Lớp: **19IT4**

Giảng viên hướng dẫn: **THS. TRỊNH THỊ NGỌC LINH**

**Đà Nẵng, tháng 12 năm 2022**

**LỜI CẢM ƠN**

Đồ án này được được thực hiện bởi nhóm gồm 1 thành viên thuộc sinh viên của trường Đại học Công nghệ thông tin và Truyền thông Việt – Hàn Đà Nẵng. Trước hết, xin gởi lời cảm ơn chân thành đến **Cô giáo Trịnh Thị Ngọc Linh** đã tận tình chỉ dẫn, góp ý để báo cáo của nhóm tránh nhiều sai sót và hoàn thiện hơn. Tiếp đến, xin gởi lời cảm ơn đến các tác giả bài viết, các trang web đã góp phần cung cấp cho chúng tôi các tiện ích và thông tin cần thiết để thực hiện đồ án.

Cuối cùng, mặc dù nhóm đã rất cố gắng nhưng chắc chắn sẽ còn rất nhiều sai sót mà chưa thể khắc phục hết được vì vậy rất mong Cô giáo đóng góp ý kiến, phê bình để nhóm rút thêm kinh nghiệm cho những lần sau.

**Xin chân thành cảm ơn!**

**NHẬN XÉT**

**(Của giảng viên hướng dẫn)**

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

Đà Nẵng, ngày … tháng ... năm 2022

**Giảng viên hướng dẫn**

**ThS. Trịnh Thị Ngọc Linh**

**MỤC LỤC**

Trang

[Chương 1 TỔNG QUAN 1](#_Toc121688015)

[1.1 Giới thiệu 1](#_Toc121688016)

[1.2 Lý do chọn đề tài 2](#_Toc121688017)

[1.3 Giải pháp 2](#_Toc121688018)

[1.4 Mục tiêu 2](#_Toc121688019)

[1.5 Phạm vi 2](#_Toc121688020)

[1.6 Cấu trúc đồ án 2](#_Toc121688021)

[Chương 2 CƠ SỞ LÝ THUYẾT 4](#_Toc121688022)

[2.1 Ngôn ngữ Swift và các thư viện 4](#_Toc121688023)

[2.1.1 Các đặc điểm của ngôn ngữ Swift 4](#_Toc121688024)

[2.1.2 Tìm hiểu về SwiftUI 5](#_Toc121688025)

[2.2 TMDB API: 6](#_Toc121688026)

[2.3 Giới thiệu Firebase 6](#_Toc121688027)

[2.4 Kiến trúc tổng quan 9](#_Toc121688028)

[2.4.1 Giới thiệu mô hình MVVM 9](#_Toc121688029)

[2.4.2 Ưu điểm của MVVM 10](#_Toc121688030)

[2.4.3 Nhược điểm 10](#_Toc121688031)

[Chương 3 PHÂN TÍCH THIẾT KẾ HỆ THỐNG 11](#_Toc121688032)

[3.1 Giới thiệu Xcode 11](#_Toc121688033)

[3.2 Phân tích yêu cầu hệ thống 12](#_Toc121688034)

[3.2.1 Các yêu cầu chức năng 12](#_Toc121688035)

[3.2.2 Các yêu cầu phi chức năng 13](#_Toc121688036)

[3.2.3 Đặc điểm người dùng 13](#_Toc121688037)

[3.2.4 Giả định và phụ thuộc 14](#_Toc121688038)

[3.3 Các sơ đồ Use Case 14](#_Toc121688039)

[3.3.1 Các tác nhân hệ thống 14](#_Toc121688040)

[3.3.2 Mô hình các trường hợp sử dụng của Quản trị viên 15](#_Toc121688041)

[3.3.3 Mô hình các trường hợp sử dụng của Người dùng 16](#_Toc121688042)

[3.3.4 Mô tả chức năng 17](#_Toc121688043)

[3.3.5 Mô tả ca sử dụng 18](#_Toc121688044)

[3.4 Kết quả cài đặt 24](#_Toc121688045)

[3.4.1 Giao diện trên nền tảng điện thoại 24](#_Toc121688046)

[3.4.2 Giao diện trên nền tảng máy tính 25](#_Toc121688047)

**KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN………………………………………….25**

1. KẾT QUẢ ĐẠT ĐƯỢC………………………………………………25
2. HẠN CHẾ…………………………………………………………….26
3. HƯỚNG PHÁT TRIỂN………………………………………………26

**DANH MỤC TÀI LIỆU THAM KHẢO……………………………………………27**

**DANH MỤC CÁC BẢNG**

[Bảng 3.1 - Các tác nhân hệ thống 14](#_Toc121688467)

[Bảng 3.2 - Bảng mô tả chức năng 17](#_Toc121688468)

[Bảng 3.3 - Đặc tả Use Case Đăng ký tài khoản 18](#_Toc121688469)

[Bảng 3.4 - Đặc tả Use Case Đăng nhập tài khoản 19](#_Toc121688470)

[Bảng 3.5 - Đặc tả Use Case Đăng xuất tài khoản 20](#_Toc121688471)

[Bảng 3.6 - Đặc tả Use Case Đặt vé xem phim 21](#_Toc121688472)

[Bảng 3.7 - Đặc tả Use Case Chỉnh sửa thông tin tài khoản 22](#_Toc121688473)

[Bảng 3.8 - Đặc tả Use Case Tìm kiếm 23](#_Toc121688474)

**DANH MỤC CÁC HÌNH**

[Hình 1.1 - Mẫu ứng dụng rạp chiếu phim đa nền tảng hiện nay 1](#_Toc121688523)

[Hình 2.1 - Giao diện website TMDB API 6](#_Toc121688524)

[Hình 2.2 - Logo Firebase 8](#_Toc121688525)

[Hình 2.3 - Các tầng mô hình MVVM 9](#_Toc121688526)

[Hình 3.1 - Logo phần mềm Xcode 11](#_Toc121688527)

[Hình 3.2 - Biểu đồ Use Case của Quản trị viên 15](#_Toc121688528)

[Hình 3.3 - Biểu đồ Use Case của Người dùng 16](#_Toc121688529)

[Hình 3.4 - Giao diện trên nền tảng điện thoại 24](#_Toc121688530)

[Hình 3.5 - Giao diện trên nền tảng máy tính 25](#_Toc121688531)

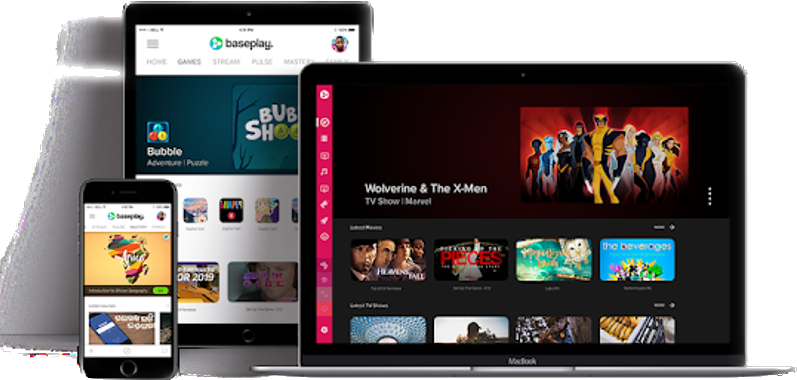
**DANH MỤC CỤM TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| **VIẾT TẮT** | **NỘI DUNG** |
| CNTT | Công nghệ thông tin |
| Obj-C | Objective-C |
| dev | developer |
| MVVM | Model - View - ViewModel |

# TỔNG QUAN

## Giới thiệu

* Trong các rạp chiếu phim hiện nay, việc quản lý và tổ chức chiếu phim, bán vé luôn là vấn đề được quan tâm. Người quản lý luôn gặp khó khăn trong vấn đề quản lý quá trình hoạt động của từng bộ phận như: Phim, lịch chiếu, trang thiết bị, … và đặc biệt là quá trình bán vé. Khách hàng muốn có được suất coi, phải xếp hàng và chờ đợi trong hàng giờ liền để mua được vé làm cho số lượng khách hàng tới rạp thưa thớt dần, khách hàng vì chán ngán cảnh tượng phải chờ đợi, vì thế họ chọn cách xem phim tại nhà (thông qua các trang web chia sẻ phim miễn phí trên mạng) làm cho doanh thu của rạp bị ảnh hưởng. Đứng trước hiện trạng đó, các ứng dụng quản lí rạp chiếu phim là niềm mong muốn và là nhu cầu tất yếu. Ứng dụng quản lí rạp chiếu phim được xây dựng từ các nhu cầu thực tế của khách hàng và nhà quản lí, nhằm giải quyết những khó khăn đang gặp phải, giảm thiểu rủi ro trong quá trình quản lý rạp. Hệ thống hướng tới các đối tượng là khách hàng và những nhân viên quản lý trong rạp. Hệ thống bao gồm các chức năng chính: quản lý khách hàng, quản lý nhân viên, quản lý phim, quản lý lịch chiếu, quản lý phòng chiếu, … Các chức năng này làm cho hoạt động của rạp phim diễn ra một cách ổn định, tối ưu được thời gian đặt mua vé, giúp người quản lý làm việc dễ dàng hơn.



Hình 1.1 - Mẫu ứng dụng rạp chiếu phim đa nền tảng hiện nay

## Lý do chọn đề tài

Người dùng thường cố gắng tìm các ứng dụng miễn phí có chức năng tương tự để hỗ trợ trong việc kinh doanh. Với sự phát triển của hệ sinh thái Apple việc phát triển một ứng dụng quản lý rạp chiếu phim là một nhu cầu tất yếu. Ngoài ra, nhà phát triển ứng dụng cần phải tối ưu chạy trên các nền tảng khác nhau như: macOS, iPadOS.

## Giải pháp

Các rạp chiếu phim được xây dựng nằm ở vị trí địa lý cách xa nhau nên để giải quyết các vấn đề quản lý đồng bộ giữa các rạp phim, nhóm đưa ra giải pháp là xây dựng ứng dụng đa nền tảng để khách hàng có thể tham khảo thông tin phim, lịch chiếu của tất cả các rạp trong hệ thống và thuận tiện hơn trong việc đặt vé và tra cứu các thông tin của từng rạp chiếu phim.

## Mục tiêu

* Hỗ trợ quản trị viên quản lý hệ thống và khách hàng sử dụng hệ thống.
* Hệ thống quản lý rạp chiếu phim có thể: quản lý khách hàng, nhân viên, phim, … một cách tối ưu, thuận tiện và nhanh chóng hơn.
* Giúp người dùng có thể an tâm đặt vé xem phim, lựa chọn các phim đang chiếu rạp thuộc sở thích của mình trong khung giờ phù hợp tại bất cứ nơi nào.
* Tự động hoá việc đặt vé.

## Phạm vi

Do thời gian thực hiện đồ án có hạn, việc thực hiện “Xây dựng ứng dụng rạp chiếu phim đa nền tảng” sẽ tập trung vào các chức năng chính bao gồm: Đăng ký, đăng nhập, chỉnh sửa thông tin tài khoản, đặt vé xem phim, tìm kiếm, quản lý phòng chiếu, quản lý vé, quản lý phim, quản lý lịch chiếu

## Cấu trúc đồ án

* Dưới đây báo cáo được trình bày trong 4 chương, cụ thể như sau:

Chương 1: Tổng quan. Trong chương này, báo cáo trình bày các mục tiêu, lý do chọn đề tài, mục đích nghiên cứu, ý nghĩa khoa học

Chương 2: Cơ sở lý thuyết. Trong chương này, báo cáo trình bày các khái niệm, phương pháp kết quả

Chương 3: Phân tích và xây dựng hệ thống. Bao gồm các tài liệu liên quan đến quá trình phân tích hệ thống như: phân tích các yêu cầu chức năng và phi chức năng

Chương 4: Triển khai thực hiện. Kiểm thử ứng dụng trên thực tế

Chương 5: Kết luận và hướng phát triển. Đưa ra kết luận và đề xuất các hướng mở rộng, phát triển trong tương lai

Cuối cùng là Danh mục tài liệu tham khảo liên quan đến đề tài.

# CƠ SỞ LÝ THUYẾT

## Ngôn ngữ Swift và các thư viện

Ra đời cách đây 8 năm (từ năm 2014), Swift vẫn là một ngôn ngữ khá trẻ và mới mẻ. Rất nhiều lập trình viên vẫn còn thắc mắc ngôn ngữ lập trình Swift là gì? Tuy nhiên, sự thắc mắc đó dần dần được thay thế theo thời gian, số lượng lập trình viên chọn lựa Swift ngày càng tăng lên, cộng đồng phát triển và hỗ trợ cho ngôn ngữ này ngày càng lớn mạnh.

### Các đặc điểm của ngôn ngữ Swift

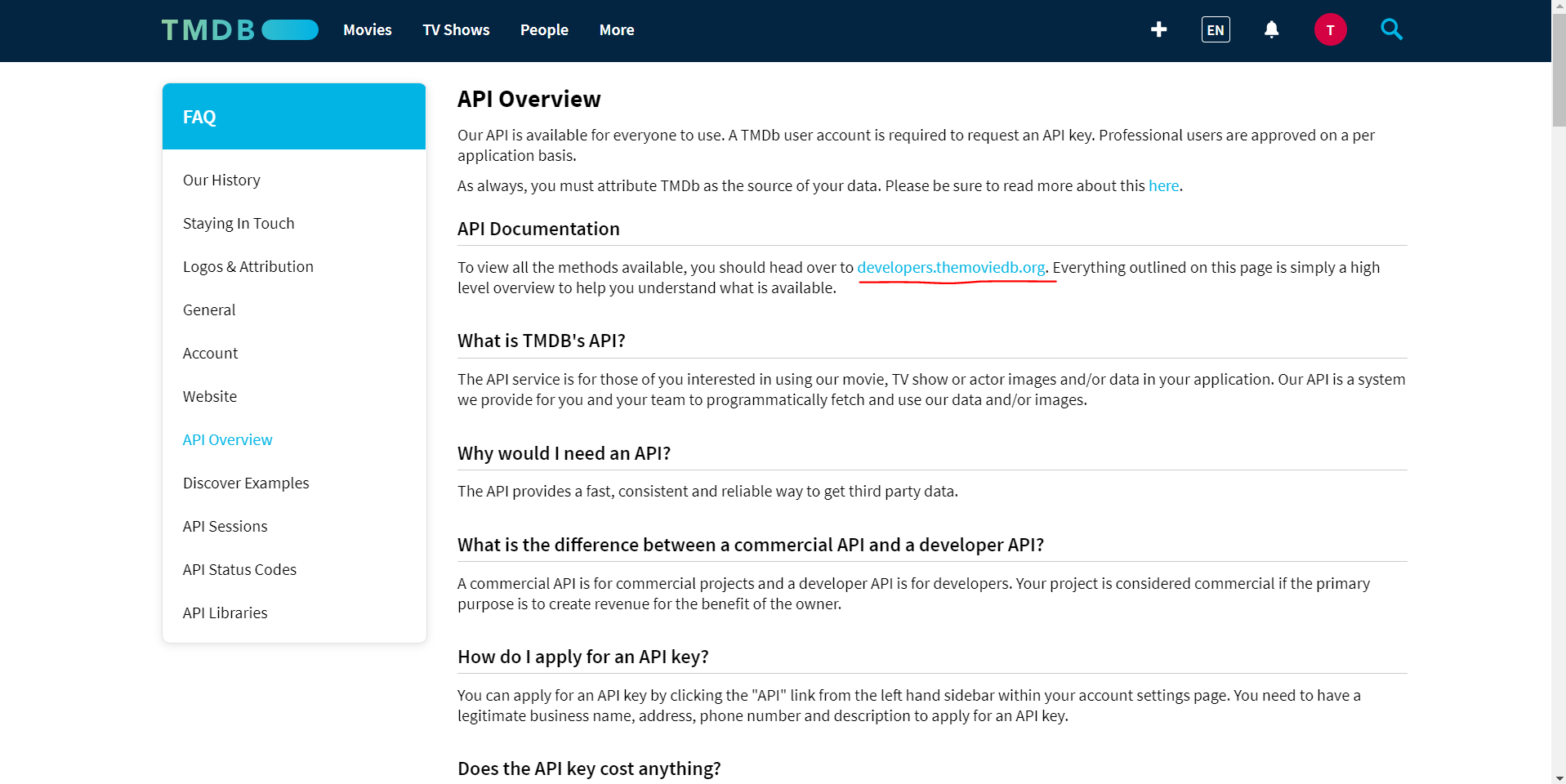
* **Khả năng phát triển ứng dụng di động nhanh và dễ dàng: Các lập trình viên không phải mất công chạy các chương trình compiler và các chương trình test một cách thường xuyên nữa, mà thay vào đó tập trung vào các phần tích hợp với mức độ khó hơn. Điều này cũng giúp tiết kiệm sức mạnh và sự hao mòn của phần cứng cho các nhà phát triển.**
* **Code của ngôn ngữ Swift ngắn, gọn và dễ đọc.** Cụ thể, 3 – 5 dòng code của ngôn ngữ Obj-C có thể được viết chỉ trong 1 dòng code của Swift. Các lớp đối tượng trong Swift được đơn giản hóa, các dòng mã được sắp xếp một cách hợp lý và logic hơn. Điều này giúp cho các lập trình viên tiết kiệm rất nhiều thời gian hoàn thành dự án, cũng như tăng hiệu quá trong quá trình bảo trì và fix các lỗi bug trong tương lai. Swift kế thừa và phát huy tất cả các sức mạnh của Objective-C để lại, nên các nhà lập trình sẽ có một môi trường runtime quen thuộc tuyệt vời, dễ dàng quản lý và phát triển.
* Các ứng dụng được viết ra bởi Swift tương thích và ít gây hao mòn phần cứng hơn Obj-C, điều này giúp người dùng có được những trải nghiệm tuyệt vời hơn khi chơi game trên các thiết bị iOS. Bên cạnh những ưu điểm, thì Swift cũng tồn tại một số điểm yếu như sau:
* Do tuổi đời còn quá trẻ, nên việc bắt đầu học Swift cũng khá khó khăn, các nhà lập trình sẽ phải học lại một ngôn ngữ hoàn toàn mới, thậm chí khó cho những người mới bắt đầu hoặc chưa từng học qua Obj-C với nền tảng căn bản.
* Cộng đồng hỗ trợ và phát triển ứng dụng chưa nhiều: có lẽ Apple sẽ mất một khoản thời gian kha khá để xây dựng một cộng đồng sử dụng Swift hùng hậu.
* Do việc phát triển ứng dụng khá nhanh do được tích hợp nhiều frameworks mạnh mẽ, điều này sẽ dẫn tới nhiều ứng dụng chất lượng kém hoặc rác do những lập trình viên mới vào nghề tạo ra Mặc dù còn tồn tại những mặt khuyết điểm, nhưng với đà phát triển mạnh mẽ của mình, thì Swift vẫn là một ngôn ngữ rất đáng để các bạn đầu tư, học tập. Ở phần tiếp theo của bài viết này, chúng tôi sẽ đề cập sâu hơn những lý do tuyệt vời mà sau khi đọc xong, chắc chắn các bạn sẽ bị hoàn toàn thuyết phục và quyết định ngay việc lựa chọn ngôn ngữ này để phát triển công việc trong tương lai của mình.
* Theo Apple, thì Swift nhanh gần gấp 3 lần Obj-C và nhanh hơn 8 lần so với ngôn ngữ Python. Không những nhanh, mà Swift còn rất mạnh mẽ, giúp cho các nhà lập trình có thể viết được những đoạn mã nâng cao, phức tạp hơn như: generics, closures, tuples, multiple returns, iterators, các mẫu lập trình hàm có sẵn và nhiều hơn thế nữa. Bên cạnh đó, Swift bảo mật và an toàn hơn Objective-C rất nhiều
* **Chúng được sử dụng miễn phí và là mã nguồn mở.** Điều này sẽ giúp cho cộng đồng Swift ngày càng phát triển hơn, người dùng có thể tìm kiếm các lỗ hổng bảo mật, cập nhật các bản vá lỗi, các tính năng, các plugin cải tiến của ngôn ngữ cho cộng đồng.

### Tìm hiểu về SwiftUI

* SwiftUI là một UI Framework mới được phát triển bởi Apple. Nó cung cấp một API mới cho phép người dùng có thể viết Code và nó còn có thể làm việc được trên tất cả các Apple platforms: **iOS**, **MacOS** và **tvOS**. Framework này đã được đưa ra như là một sự cập nhật và bổ sung đáng được chờ đợi nhất của ngôn ngữ lập trình **Swift** – cung cấp cho các nhà phát triển một giải pháp để có thể mang ứng dụng **iPhone** của họ sang **Mac** và các nền tảng khác của Apple một cách tiết kiệm thời gian và tài nguyên cần thiết.
* Framework này cũng có một tùy chọn để trải nghiệm các tính năng độc đáo như tốc độ liên quan với chuột và bàn phím, các tính năng đặc biệt như Touch Bar và hơn thế nữa – Trong tương lai nó chính là thứ sẽ cạnh tranh trực tiếp với Framework UI Flutter của Google và React của Facebook, là những framework dùng để phát triển ứng dụng đa nền tảng.
* Nếu so sánh SwiftUI với UIKit ở thời điểm hiện tại thì đang cách biệt lớn bởi vì vẫn chưa có nhiều những sự trợ giúp từ các trang nổi tiếng như Stackoverflow, Medium, Hackingswift, ... về những rắc rối hay những vấn đề phức tạp.
* Hiện tại SwiftUI đang được nhiều người lựa chọn sử dụng mặc dù vẫn đang trong quá trình nâng cấp và bổ xung, bởi vì họ tin vào SwiftUI sẽ phát triển mạnh mẽ trong tương lai, và có một điều rằng bạn có thể sử dụng đồng thời cả UIKit và SwiftUI trong một project, vậy nó nói lên rằng nếu bạn đã thông thạo UIKit thì có thêm một sự trợ gúp hoàn hảo khi sử dụng SwiftUI. Có thể nó sẽ là một Framework mạnh mẽ trong tương lai gần, tuy còn nhiều thiếu xót nhưng Apple đã và đang ngày một hoàn thiện cho nó.

## TMDB API:

* The Movie Database (TMDb) là cơ sở dữ liệu cộng tác về phim. Dự án được Travis Bell thành lập vào năm 2008 để thu thập áp phích phim. Cơ sở dữ liệu ban đầu là một khoản đóng góp từ dự án Cơ sở dữ liệu phương tiện mở (omdb) miễn phí.
* TMDb API là một cơ sở dữ liệu trực tuyến chứa thông tin và thống kê về phim, chương trình truyền hình và trò chơi điện tử cũng như các diễn viên, đạo diễn và các chuyên gia trong ngành điện ảnh khác. Thông tin này có thể bao gồm danh sách các diễn viên và đoàn làm phim, ngày phát hành phim và thông tin phòng vé, tóm tắt cốt truyện, đoạn giới thiệu, tiểu sử diễn viên và đạo diễn và các câu đố khác.
* Năm 2010, TMDb được bán cho Fan TV, nhưng trang web vẫn được quản lý bởi người sáng lập. Vào giữa năm 2020, cơ sở dữ liệu chứa 568.729 bộ phim và 1.750.574 người.



Hình 2.1 - Giao diện website TMDB API

## Giới thiệu Firebase

* Firebaselà một nền tảng ứng dụng di động và web với các công cụ và hạ tầng được thiết kế để giúp các lập trình viên xây dựng các ứng dụng chất lượng cao. Nói ngắn gọn, thay vì trực tiếp cung cấp các ứng dụng, họ cung cấp các dịch vụ nền tảng cho các lập trình viên, chính là các bạn, sử dụng để xây dựng ứng dụng cũng như hỗ trợ các bạn tối ưu hóa, tối đa hóa ứng dụng của mình. Với nhiều dịch vụ chất lượng cao đi kèm mức giá phải chăng, Firebase đã và đang, không chỉ là sự lựa chọn hàng đầu cho các lập trình viên đơn thân (single dev) hay các công ty khởi nghiệp (start ups), mà các công ty, tổ chức lớn có tên tuổi cũng sử dụng “Ngọn lửa” để xây dựng các tính năng, các chương trình mới, cũng như chuyển đổi các dịch vụ trước đây sang hệ thống của Firebase. Chẳng hạn như Shazam, Fabulous và cả chính Google nữa, khi nền tảng nhắn tin Allo được xây dựng trên nền tảng Firebase Realtime Database.
* Về mặt lịch sử, Firebase (tiền thân là Evolve) trước đây là một start up được thành lập vào năm 2011 bởi Andrew Lee và James Tamplin. Ban đầu, Evolve chỉ cung cấp cơ sở dữ liệu để các lập trình viên thiết kế các ứng dụng chat (và hiện tại thì để làm quen với realtime db thì bạn cũng làm ứng dụng chat đó thôi). Tuy nhiên, họ nhanh chóng nhận ra tiềm năng sản phẩm của mình khi nhận thấy các khách hàng không sử dụng CSDL để làm ứng dụng chat, mà thay vào đó, để lưu các thông tin như game progress. Bộ đôi Lee và Tamplin quyết định tách mảng realtime ra để thành lập một công ty độc lập – chính là Firebase – vào tháng 4 năm 2012. Sau nhiều lần huy động vốn và gặt hái được những thành công nổi bật, Firebase đã được Google để ý. Vào tháng 10 năm 2014, Firebase gia nhập gia đình Google.
* Cả Google và Firebase đều như hổ mọc thêm cánh. Firebase có điều kiện để phát triển thần tốc, mở rộng số lượng các dịch vụ con, còn Google có được một đội ngũ nhân lực chất lượng cao, năng động, cũng như cơ sở hạ tầng và sự hiệu quả mà các dịch vụ của Firebase mang lại, mà không phải xây dựng lại từ đầu. Hiện tại, Google đã chuyển các dịch vụ nền tảng hỗ trợ các lập trình viên bên ngoài về cho Firebase quản lí, chẳng hạn như Cloud Messaging, AdMob và Analytics.
* Firebase, theo hướng đi của Google, chính thức hỗ trợ Android, iOS và Web. Thực tế, macOS cũng được hỗ trợ vì macOS chia sẻ nhiều dòng code với iOS, song vì Google và Firebase muốn sử dụng web cho ứng dụng desktop thay vì native, nên có khá ít tài liệu chính thức nói về Firebase cho macOS, cũng như các thư viện cho macOS có thể kém chức năng và không ổn định lắm. Còn về Windows, hiện tại tôi chưa thấy họ lên tiếng nào về việc sẽ chính thức phát hành thư viện cho đứa con của Microsoft, nên nếu các bạn muốn làm ứng dụng cho Windows (UWP) thì chỉ nên (và cũng chỉ có mỗi con đường) làm web-based native apps mà thôi.

**Ưu điểm:**

* Như bạn có thể thấy, Firebase cung cấp rất nhiều dịch vụ hỗ trợ giúp các lập trình viên xây dựng ứng dụng của mình. Việc tích hợp Firebase vào ứng dụng có thể hạ nỗi lo lắng của các dev xuống ở mức coding, đặc biệt là cho các dev viết các ứng dụng có liên quan tới web. Bước đầu tiên, bạn cần có host để chứa cơ sở dữ liệu => đã có Firebase Hosting và Storage. Bạn cần viết code PHP để giao tiếp? Có thể không cần cho lắm, vì Firebase Authentication và Realtime Database đã sẵn sàng cho bạn tích hợp và sử dụng “ngay và luôn”. Bạn muốn gửi thông báo tới các thiết bị? Firebase Cloud Messaging là cái bạn cần. Hơn thế nữa, với Test Lab, Crash Reporting, App Indexing thì Firebase sẽ báo cáo cho bạn các vấn đề có thể xảy ra khi ứng dụng của bạn chạy nhiều trên máy thật.
* Cũng như Remote Config giúp bạn thay đổi các thiết lập trên ứng dụng của người dùng mà không cần thiết phải publish 1 phiên bản mới… Cá nhân tôi, sau khi sử dụng Firebase, mặc dù chỉ ở mức độ “cho biết”, nhưng đã có thể kết luận là sẽ sử dụng Firebase cho các ứng dụng có liên quan tới web sau này, bởi tôi chỉ xếp nó sau việc tự thiết kế toàn bộ hạ tầng từ A-Z, còn nếu đem so với các nhà cung cấp khác thì không ai có thể vượt qua được “Ngọn lửa” cả. Các nhà cung cấp khác có thể có một vài dịch vụ chất lượng hơn, nhưng nhìn chung, họ không cung cấp số lượng dịch vụ nhiều như vậy.

**Nhược điểm:**

* Không có gì là hoàn hảo, và Firebase cũng không là ngoại lệ. Điểm yếu duy nhất của “Ngọn lửa” chính là phần Realtime Database, mà đúng hơn chỉ là phần Database. Cơ sở dữ liệu của họ được tổ chức theo kiểu trees, parent-children, không phải là kiểu Table nên những ai đang quen với SQL có thể sẽ gặp khó khăn từ mức đôi chút tới khá nhiều.

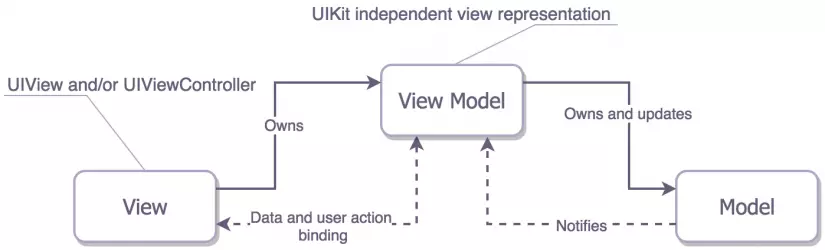


Hình 2.2 - Logo Firebase

## Kiến trúc tổng quan

### Giới thiệu mô hình MVVM

* MVVM là viết tắt của Model - View - ViewModel, đây là mô hình hỗ trợ two-way data binding giữa View và View Model. Cụ thể mô hình MVVM được trình bày như sau:
* View: Tương tự như trong mô hình MVC, View là phần giao diện của ứng dụng để hiển thị dữ liệu và nhận tương tác của người dùng. Một điểm khác biệt so với các ứng dụng truyền thống là View trong mô hình này tích cực hơn. Nó có khả năng thực hiện các hành vi và phản hồi lại người dùng thông qua tính năng binding, command.
* Model: Cũng tương tự như trong mô hình MVC. Model là các đối tượng giúp truy xuất và thao tác trên dữ liệu thực sự.
* ViewModel: Lớp trung gian giữa View và Model. ViewModel có thể được xem là thành phần thay thế cho Controller trong mô hình MVC. Nó chứa các mã lệnh cần thiết để thực hiện data binding, command.
* Trong các mô hình truyền thống, chúng ta thường xử lý sự kiện Click và viết mã thực thi trực tiếp ở trên một Button nhưng với mô hình MVVM không cho phép làm điều này.
* Trong mô hình MVVM, các điều khiển (control) như Button, ListView, SearchBar, ... không thể kết buộc trực tiếp đến dữ liệu mà phải thông qua thuộc tính Command – là một thuộc tính kiểu ICommand.
* Một điểm cần lưu ý là trong mô hình MVVM, các tầng bên dưới sẽ không biết được các thông tin gì về tầng bên trên nó. Như hình minh họa dưới đây:



Hình 2.3 - Các tầng mô hình MVVM

### Ưu điểm của MVVM

* Thực hiện Unit testing bây giờ sẽ rất dễ dàng, vì bạn thực sự không phụ thuộc vào view.
* MVVM sẽ tạo sự tương tác hiệu quả giữa designer và developer.
* Tăng khả năng sử dụng lại các thành phần hay việc thay đổi giao diện chương trình mà không cần phải viết lại code quá nhiều.
* Phát triển ứng dụng nhanh, đơn giản, dễ nâng cấp, bảo trì…

### Nhược điểm

* Khả năng duy trì khi view có thể gán cả biến và biểu thức, các logic không liên quan sẽ tăng dần theo thời gian, ảnh hưởng đến việc thêm code vào XML.
* Đối với dự án nhỏ việc áp dụng mô hình MVVM gây cồng kềnh, tốn thời gian trong quá trình phát triển. Tốn thời gian trung chuyển dữ liệu của các thành phần.
* Đối với dự án lớn hơn, nó gây khó khăn và mất thời gian để thiết kế các ViewModel.
* Việc liên kết dữ liệu cho tất cả các thành phần gây khó khăn trong việc debug khi cơ sở dữ liệu phức tạp.

# PHÂN TÍCH THIẾT KẾ HỆ THỐNG

## Giới thiệu Xcode

* Xcode là một ứng dụng được Apple xây dựng để hỗ trợ cho các lập trình viên phát triển các ứng dụng chạy trên các nền tảng của hãng (iOS, macOS…). Xcode là một công cụ cực kỳ hữu dụng, rất tuyệt vời… và dân lập trình hay ưu ái gọi rằng nó rất mạnh.
* Thực sự vậy, Xcode là công cụ vô cùng hữu ích để xây dựng các ứng dụng trên nền tảng của Apple. Với các lập trình viên ứng dụng nếu đã sử dụng Xcode thì sẽ thấy được sự hỗ trợ rất mạnh mẽ của nó, từ việc viết code đến debug chức năng, tìm lỗi. Tất cả được hỗ trợ vô cùng tuyệt vời mà ít công cụ nào làm được. Tính đến thời điểm hiện tại phiên bản Xcode là 14.1
* Giao diện làm việc của phần mềm Xcode gồm có 5 part chính: **Toolbar**, **Editor area**, **Navigator area**, **Debug area**, **Utility area**.
* Debug area: là vùng hỗ trợ lập trình viên trong quá trình debug lỗi của phần mềm.
* Toolbar area: đây là vùng chứa các công cụ tiện ích giúp lập trình viên có thể đơn giản hơn trong việc chạy, debug ứng dụng, lựa chọn iOS Simulator hay đóng mở các vùng khác, …
* Editor area: vùng để bạn có thể thiết kế giao diện, viết và sửa code của chương trình.
* Utility area: vùng này cho phép lập trình viên tùy chỉnh các tham số, giá trị của các đối tượng trong giao diện, cũng như cho phép người sử dụng kéo thả và sử dụng các đối tượng có sẵn của Xcode như Button, Label, Slider, … hay các đoạn code mẫu (If, Switch, …).
* Navigator area: cung cấp cho lập trình viên một cái nhìn trực quan, tiện lợi trong việc quản lý ứng dụng, xem thông báo lỗi, tìm kiếm một đoạn code trong chương trình hay ra soát mức độ hoạt động của RAM, CPU khi chạy ứng dụng, …



Hình 3.1 - Logo phần mềm Xcode

## Phân tích yêu cầu hệ thống

### Các yêu cầu chức năng

* Đăng ký: Cho phép khách hàng tạo tài khoản (user name và password) để đăng nhập vào hệ thống đặt vé online. Bao gồm các thông tin cá nhân như: họ tên, địa chỉ, email, số điện thoại, liên kết ngân hàng, ví điện tử, ...
* Đăng nhập: Cho phép khách hàng sử dụng tài khoản (user name và password) đã đăng ký để đăng nhập vào hệ thống và sử dụng các chức năng khác.
* Xem thông tin phim: Cho phép khách hàng có thể xem được các thông tin của các bộ phim mà rạp đã chiếu đang chiếu và sắp chiếu trong thời gian tới. Các thông tin gồm: tên phim, thể loại, quốc gia sản xuất, thời lượng của bộ phim, tên đạo diễn, diễn viên đóng trong phim, nội dung của bộ phim, trailer, thời gian khởi chiếu...
* Xem lịch chiếu phim: Cho phép khách hàng xem lịch chiếu phim của rạp trong tuần này hay trong 1 hay 2 tuần tới tùy theo sự lựa chọn của khách hàng. Lịch chiếu phim bao gồm thông tin của các buổi chiếu. Mỗi buổi chiếu bao gồm các thông tin: tên phim được chiếu, phòng chiếu, thời gian chiếu.
* Xem thông tin rạp chiếu/phòng chiếu: Giúp khách hàng xem được hình ảnh, địa chỉ các rạp chiếu, khu vực, ...
* Đặt vé: Giúp khách hàng đặt vé xem phim mà không cần phải xếp hàng tại quầy
* Quản lý vé: giúp nhân viên quản lý số lượng vé bán ra của từng loại vé trong mỗi buổi chiếu.
* Quản lý phòng chiếu: Giúp quản lý tất cả các phòng chiếu trong từng rạp. Gồm thời gian, thời lượng, số ghế, thiết bị, ...
* Quản lý phim: Giúp quản lý những phim đang chiếu, sắp chiếu, nội dung, thể loại, điều kiện độ tuổi xem phim, ...
* Quản lý lịch chiếu: Giúp nhân viên lên lịch, sửa, xóa, cập nhật lịch chiếu phim.
* Thanh toán: Hỗ trợ cho khách hàng mua vé online thanh toán chi phí đặt vé. Hệ thống hỗ trợ khách hàng có thể thanh toán trực tiếp hay thông qua chuyển khoản với một số ngân hàng mà hệ thống hỗ trợ và các ví điện tử có liên kết.
* Quản lý thông tin khách hàng: Chức năng này sẽ giúp cho ban quản lý rạp biết được các thông tin cũng như số lượng khách hàng của rạp để từ đó họ có thể phục vụ khách hàng một cách tốt hơn để nâng cao doanh thu của rạp.
* Thống kê báo cáo: Ban quản lý rạp sẽ được hỗ trợ để thống kê các báo cáo: về vé, phim, cơ sở vật chất, các dữ liệu của rạp, các hoạt động bên trong rạp, ...

### Các yêu cầu phi chức năng

* Hiệu suất:
* Thời gian truy vấn của hệ thống là tìm kiếm dưới 5 giây.
* Hệ thống phục vụ được 100 người sử dụng cùng lúc.
* Hệ thống xử lý 100 lượt giao dịch thanh toán trong 1 giờ.
* Tính tiện dụng:
* Giao diện thân thiện, dễ sử dụng.
* Người dùng đăng nhập tài khoản để đặt vé xem phim bằng cách chọn tên phim, ngày, giờ và số lượng vé cần mua.
* Độ tin cậy:
* Ứng dụng này có độ tin cậy cao và nó tạo ra tất cả các thông tin được cập nhật theo đúng thứ tự.
* Khả năng thực thi:
* 90% người dùng có thể sử dụng hệ thống sau 1 giờ trải nghiệm hệ thống.
* Bảo mật:
* Chỉ người dùng xác thực mới có thể truy cập hoàn toàn hệ thống. Nó yêu cầu tài khoản để trở thành thành viên của ứng dụng.
* Cung cấp một bộ quy tắc bảo mật (security rules), kết hợp với Firebase Authentication, giúp giới hạn quyền truy cập của người dùng và phân quyền.
* Khả năng tương thích:
* Đa nền tảng trên các thiết bị sử dụng hệ điều hành iOS, macOS, iPadOS.
* Khả năng chuyển đổi:
* Hệ thống có khả năng chuyển đổi ngôn ngữ

### Đặc điểm người dùng

* Người dùng sử dụng hệ thống để lấy thông tin về các bộ phim có sẵn, giá, ngày và thời gian chủ yếu là để đặt vé.
* Admin quản lý hệ thống và luôn cập nhật. Quản trị viên cũng xem xét các yêu cầu của người dùng.
* Cơ sở dữ liệu của hệ thống phải có khả năng xử lý đồng thời tất cả các yêu cầu gửi đến và giải quyết các lỗi xung đột các chức năng từ thao tác người đặt vé phim (nếu có).

### Giả định và phụ thuộc

* Vai trò tác nhân và nhiệm vụ được xác định trước.
* Người dùng phải tuân theo các tiêu chuẩn quốc tế để đăng ký tài khoản người dùng và phải điền thông tin liên quan ở định dạng phù hợp.
* Mật khẩu phải chứa ít nhất 10 ký tự theo quy tắc.
* Cần có kết nối Internet để sử dụng hệ thống

## Các sơ đồ Use Case

### Các tác nhân hệ thống

Bảng 3.1 - Các tác nhân hệ thống

|  |  |  |
| --- | --- | --- |
| STT | Tên Actor | Giải thích |
| 1 | Người dùng (User) | * Đăng nhập hệ thống: Mỗi một khách hàng đã đăng ký tài khoản đều có quyền đăng nhập vào tài khoản của mình và tài khoản sẽ được quản lý bởi Adminstrator * Đăng ký tài khoản thành viên khi khách hàng muốn đặt vé xem phim * Sửa thông tin tài khoản: khi cần cập nhật, sửa thông tin (họ và tên, địa chỉ, mật khẩu, email) * Đặt vé: Khi người dùng đã chọn các phim cần đặt vé phim thì sẽ được chuyển đến trang thanh toán để điền các thông tin cần thiết (chọn địa chỉ, phương thức thanh toán) và sau đó xác nhận đơn hàng * Xem vé phim: Khi đã đặt được vé phim thành công thì người dùng có thể kiểm tra lại vé phim * Thay đổi mật khẩu: khi muốn thay đổi mật khẩu người dùng chỉ cần nhập đúng mật khẩu cũ và mật khẩu mới 2 lần |
| 2 | Quản trị viên (Admin) | * Admin là tác nhân giữ vai trò chính của hệ thống: Quyền quản lý là quyền cao nhất của hệ thống. * Tác nhân Admin có thể thực hiện được tất cả các chức năng như: Tạo, quản lý tài khoản thành viên tức là Admin có quyền sửa hay xóa tài khoản của các thành viên. Ngoài ra Admin còn tùy chỉnh thông tin phim (thêm, sửa, xóa), lịch chiếu rạp, phòng chiếu phim. |

### Mô hình các trường hợp sử dụng của Quản trị viên

Diagram

Description automatically generated

Hình 3.2 - Biểu đồ Use Case của Quản trị viên

### Mô hình các trường hợp sử dụng của Người dùng

Diagram

Description automatically generated

Hình 3.3 - Biểu đồ Use Case của Người dùng

### Mô tả chức năng

Bảng 3.2 - Bảng mô tả chức năng

|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả** |
|  | Đăng ký | Cho phép tạo tài khoản để đăng nhập vào hệ thống đặt vé online. Bao gồm các thông tin cá nhân như: họ tên, địa chỉ, email, số điện thoại. |
|  | Đăng nhập | Là chức năng con của Đăng ký, cho phép khách hàng sử dụng tài khoản đăng ký để đăng nhập vào hệ thống và sử dụng các chức năng khác. |
|  | Đặt vé | Cho phép khách hàng đặt vé trực tuyến trên ứng dụng. Khách hàng nhận được vé in hoặc vé điện tử (mã QR) sau khi tiến hành thanh toán. |
|  | Tìm phim | Cho phép khách hàng tìm kiếm phim phù hợp với nhu cầu của mình. Tìm theo suất chiếu, theo phòng chiếu hoặc theo thể loại. Không yêu cầu phải Đăng nhập tài khoản. |
|  | Thanh toán | Là chức năng con của Đặt vé, cho phép khách hàng chọn loại giao dịch để thanh toán: Qua các ngân hàng có hỗ trợ trả tiền online. Nếu không tiến hành thanh toán sau khi đặt vé thì vé sẽ bị huỷ. Khách hàng nhận vé sau khi thanh toán thành công. |
|  | Cập nhật thông tin | Cho phép khách hàng cập nhật thông tin cá nhân. |
|  | Quản lí lịch chiếu phim | Bao gồm các hoạt động nhằm cung cấp thông tin lịch chiếu phim cho khách hàng, hỗ trợ khách hàng tìm kiếm phim. |
|  | Cập nhật thông tin phòng chiếu | Nhân viên cập nhật thông tin phòng chiếu sao cho phù hợp với bối cảnh và quảng cáo. |
|  | Quản lí lịch chiếu phim | Bao gồm các hoạt động nhằm cung cấp thông tin lịch chiếu phim cho khách hàng, hỗ trợ khách hàng tìm kiếm phim và suất chiếu phù hợp. |

### Mô tả ca sử dụng

Bảng 3.3 - Đặc tả Use Case Đăng ký tài khoản

|  |  |
| --- | --- |
| **Use case** | **Nội dung** |
| Tên Use case | Đăng ký |
| Mô tả | Use case cho phép người dùng đăng ký tài khoản để thực hiện những chức năng của mình |
| Actor | Người dùng |
| Điều kiện kích hoạt | Khi người dùng chọn chức năng đăng ký hoặc khi người dùng chọn các chức năng phụ thuộc tài khoản |
| Tiền điều kiện | Người dùng phải có tài khoản email |
| Hậu điều kiện | Người dùng đăng ký thành công |
| Luồng sự kiện chính | Use case này bắt đầu khi người dùng có nhu cầu đăng ký làm thành viên của hệ thống hoặc khi người dùng thanh toán đặt vé xem phim.   1. Hệ thống hiển thị màn hình đăng ký. 2. Người dùng nhập tên, email, số điện thoại và mật khẩu. 3. Hệ thống hiển thị kiểm tra thông tin đăng ký. 4. Nếu thành công hệ thống hiển thị thông báo thành công trên màn hình. 5. Kết thúc Use case. |
| Luồng sự kiện phụ | TH1 - Nếu trong quá trình đăng ký người dùng quyết định hủy bỏ các thông tin đăng ký thì use case sẽ kết thúc và trạng thái của hệ thống không thay đổi.  TH2 - Nếu quá trình kiểm tra thông tin nhập phát hiện không hợp lệ thì hệ thống sẽ trở về trạng thái trước đó (mở trang nhập liệu với các thông tin nhập trước đó vẫn còn để người dùng có thể chỉnh sửa). |

Bảng 3.4 - Đặc tả Use Case Đăng nhập tài khoản

|  |  |
| --- | --- |
| **Use case** | **Nội dung** |
| Tên Use case | Đăng nhập |
| Mô tả | Use case cho phép người dùng đăng nhập vào ứng dụng để thực hiện những chức năng của mình |
| Actor | Người dùng |
| Điều kiện kích hoạt | Khi người dùng chọn chức năng đăng nhập từ trang chủ của website |
| Tiền điều kiện | Người dùng phải có tài khoản trên ứng dụng |
| Hậu điều kiện | Người dùng đăng nhập thành công |
| Luồng sự kiện chính | 1. Hệ thống hiển thị màn hình đăng nhập. 2. Người dùng nhập tên đăng nhập và mật khẩu. 3. Hệ thống hiển thị kiểm tra thông tin đăng nhập. 4. Nếu thành công hệ thống hiển thị ra giao diện của người dùng. 5. Kết thúc Use case. |
| Luồng sự kiện phụ | TH1 – Mật khẩu không hợp lệ: Khi người dùng nhập sai tên đăng nhập và mật khẩu   1. Hệ thống hiển thị lại màn hình đăng nhập để người dùng đăng nhập lại thông tin kèm theo thông báo tên đăng nhập và mật khẩu bị sai. 2. Quay lại bước 2 trong luồng sự kiện chính.   TH2 – Quên mật khẩu: Khi người dùng chọn chức năng quên mật khẩu trên màn hình đăng nhập   1. Hệ thống hiển thị màn hình để người dùng nhập email. 2. Người dùng nhập email và chọn nút chức năng Lấy lại mật khẩu. 3. Hệ thống kiểm tra email hợp lệ và gửi liên kết để reset mật khẩu cho người dùng email. 4. Hệ thống hiển thị màn hình thông báo thành công 5. Use case kết thúc. |

Bảng 3.5 - Đặc tả Use Case Đăng xuất tài khoản

|  |  |
| --- | --- |
| **Use case** | **Nội dung** |
| Tên Use case | Đăng xuất |
| Mô tả | Use case cho phép người dùng đăng xuất khỏi tài khoản |
| Actor | Khách hàng |
| Điều kiện kích hoạt | Khi người dùng chọn chức năng đăng xuất từ thông tin cá nhân |
| Tiền điều kiện | Người dùng phải phải đăng hhập vào hệ thống |
| Hậu điều kiện | Người dùng đăng xuất thành công |
| Luồng sự kiện chính | 1. Hệ thống hiển thị nút chọn đăng xuất. 2. Hệ thống sẽ kiểm tra tài khoản. 3. Nếu thành công hệ thống hiển thị thông báo thành công trên màn hình. 4. Kết thúc Use case. |
| Luồng sự kiện phụ | TH - Nếu trong quá trình đăng ký người dùng quyết định hủy bỏ các thông tin đăng ký thì use case sẽ kết thúc và trạng thái của hệ thống không thay đổi. |

Bảng 3.6 - Đặc tả Use Case Đặt vé xem phim

|  |  |
| --- | --- |
| **Use case** | **Nội dung** |
| Tên Use case | Đặt vé xem phim |
| Mô tả | Use case cho phép người dùng đặt vé xem phim |
| Actor | Người dùng |
| Điều kiện kích hoạt | Người dùng phải đăng nhập thành công vào hệ thống |
| Tiền điều kiện | Người dùng phải đăng nhập vào hệ thống |
| Hậu điều kiện | Người dùng xem thông tin đặt vé thành công |
| Luồng sự kiện chính | 1. Hệ thống hiển thị thông tin vé xem phim trên màn hình. 2. Người dùng chọn tên phim, ngày giờ, ngôn ngữ, loại ghế và số lượng vé. 3. Hệ thống hiển thị thông tin cụ thể về đơn hàng như: ngày đặt hàng, tên khách hàng, số điện thoại khách hàng.... 4. Kết thúc Use case. |
| Luồng sự kiện phụ | TH: Nếu người dùng chưa chọn ngày hoặc giờ hoặc loại ghế thì thông báo lỗi sẽ hiển thị cho người dùng và sau đó người dùng cần chọn lại |

Bảng 3.7 - Đặc tả Use Case Chỉnh sửa thông tin tài khoản

|  |  |
| --- | --- |
| **Use case** | **Nội dung** |
| Tên Use case | Chỉnh sửa thông tin tài khoản |
| Mô tả | Use case cho phép người dùng chỉnh sửa hồ sơ của mình |
| Actor | Người dùng, Quản trị viên |
| Điều kiện kích hoạt | Khi người dùng chọn chức năng profile trên thanh menu của ứng dụng |
| Tiền điều kiện | Người dùng phải đăng nhập vào hệ thống |
| Hậu điều kiện | Người dùng có thể chỉnh sửa và cập nhật hồ sơ |
| Luồng sự kiện chính | 1. Hệ thống hiển thị hồ sơ. 2. Người dùng nhập tên, họ, email, số điện thoại. Sau đó nhấn nút cập nhật hoặc hủy bỏ 3. Hệ thống hiển thị kiểm tra thông tin cập nhật. 4. Nếu thành công hệ thống hiển thị thông báo thành công trên màn hình. 5. Kết thúc Use case. |
| Luồng sự kiện phụ | TH – Nhập không đúng thông tin yêu cầu: Khi người dùng nhập sai định dạng email hoặc số điện thoại.   1. Hệ thống hiển thị lại màn hình cập nhật hồ sơ để người dùng điền lại thông tin kèm theo thông báo định dạng email, số điện thoại bị sai. 2. Quay lại bước 2 trong luồng sự kiện chính. |

Bảng 3.8 - Đặc tả Use Case Tìm kiếm

|  |  |
| --- | --- |
| **Use case** | **Nội dung** |
| Tên Use case | Tìm kiếm |
| Mô tả | Use case này cho phép người dùng tìm kiếm tất tả các bộ phim. |
| Actor | Người dùng |
| Điều kiện kích hoạt | Khi người dùng chọn chức năng tìm kiếm trên thanh menu ứng dụng |
| Tiền điều kiện | Không có |
| Hậu điều kiện | Người dùng có thể tìm kiếm theo chữ cái đầu tiên |
| Luồng sự kiện chính | 1. Người dùng nhập từ khóa tìm kiếm (Search) trên menu. 2. Hệ thống sử dụng cơ chế tìm kiếm để tìm kiếm trong cơ sở dữ liệu của hệ thống. 3. Hệ thống sẽ lọc ra các từ khóa có liên quan đến từ khóa mà người dùng tìm kiếm. 4. Kết thúc Use case. |
| Luồng sự kiện phụ | TH: Hệ thống không hiển thị thông tin tìm kiếm, người dùng cần nhập lại từ khóa hay hủy bỏ thao tác tìm kiếm. |

## Kết quả cài đặt

### Giao diện trên nền tảng điện thoại

Graphical user interface, application

Description automatically generated

Hình 3.4 - Giao diện trên nền tảng điện thoại

### Giao diện trên nền tảng máy tính

Graphical user interface, website

Description automatically generated

Hình 3.5 - Giao diện trên nền tảng máy tính

**KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

1. **Kết quả đạt được**

Với đồ án lần này, ngoài việc xây dựng một ứng dụng có thể thanh toán đặt vé xem phim, tôi còn được rèn luyện kĩ năng code trên Xcode, biết thêm về cách viết mã đa nền tảng, sử dụng Firebase trong ngôn ngữ lập trình Swift. Nâng cao khả năng lập trình, tự học hỏi nghiên cứu tài liệu, kĩ năng làm báo cáo. Cùng với sự giúp đỡ của ThS. Trịnh Thị Ngọc Linh, nhóm tôi đã có thể hoàn thành tốt Đồ án lần này.

1. **Hạn chế**

Tuy đã đạt được những chức năng đề ra ban đầu nhưng ứng dụng vẫn còn rất nhiều hạn chế:

* Các chức năng chỉnh sửa chưa thực sự hoàn thiện: chạy còn chậm
* Các tính năng chưa tối ưu, giao diện còn đơn giản
* Chưa có tính năng mới tạo nên sự khác biệt với các ứng dụng chỉnh sửa ảnh hiện nay
* Chưa có các chức năng nâng cao

1. **Hướng phát triển**

Ứng dụng vẫn còn thiếu sót khá nhiều về mặt chức năng cũng như khả năng tương tác với người dùng, do đó trong thời gian tới sẽ tiến hành cập nhập bổ sung các tính năng sau:

* Tiến hành nâng cấp giao diện của ứng dụng.
* Chức năng Quản lý nhân viên
* Có giới hạn đặt trước một bộ phim. Nếu hội trường có nhiều người thì người dùng không thể đặt phim vào thời điểm đó.

**DANH MỤC TÀI LIỆU THAM KHẢO**

[1] “<https://www.kodeco.com/22096649-getting-a-redux-vibe-into-swiftui#toc-anchor-011>”

[2] “[https://www.raywenderlich.com/](https://www.raywenderlich.com/30195423-core-image-tutorial-getting-started)”

[3] “<https://kavsoft.dev/>”

[4] “<https://www.hackingwithswift.com/learn>”

[5] “<https://liquidcoder.com/course/movie-booking-app>”

[6] “<https://github.com/alfianlosari/SwiftUITMDbV2>”

[7] “<https://medium.com/firebase-developers/firebase-swiftui-note-taking-app-tutorial-gcp-5a12a1b1154b>”