**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC VÀ KĨ THUẬT THÔNG TIN**



**BÁO CÁO ĐỒ ÁN**

**MÔN CÔNG NGHỆ LẬP TRÌNH ĐA NỀN TẢNG CHO ỨNG DỤNG DI ĐỘNG**

**Đề tài: TÌM HIỂU VỀ THIRDPARTY INTERGRATION &**

**REACT NATIVE FIREBASE**

GVHD: ThS. Võ Ngọc Tân

Nhóm sinh viên thực hiện – Nhóm 2:

1. Nguyễn Xuân Thanh MSSV: 22540020

2. Nguyễn Ngọc Ánh MSSV: 22540001

Tp. Hồ Chí Minh, 10/2024

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*……., ngày……...tháng……năm 2024*

**Người nhận xét**

*(Ký tên và ghi rõ họ tên****)***

# DANH MỤC HÌNH ẢNH

[*Hình 1.1 Quy trình tích hợp API của bên thứ ba. 9*](#_Toc181280855)

[*Hình 2.1 Sự kết hợp giữa Firebase và React trong phát triển ứng dụng web hoặc di động. 10*](#_Toc181280856)

[*Hình 2.2 Các tính năng chính của Firebase. 10*](#_Toc181280857)

[*Hình 2.3 Quá trình xác thực bảo mật bằng Firebase Authentication. 11*](#_Toc181280858)

[*Hình 2.4 Quá trình quản lý thông báo sử dụng Firebase Cloud Messaging. 11*](#_Toc181280859)

[*Hình 2.5 Sự kết hợp giữa AdMob của Google và Firebase 12*](#_Toc181280860)

[*Hình 2.6 Phân tích hành vi người dùng bằng Google Analytics for Firebase. 12*](#_Toc181280861)

[*Hình 2.7 Theo dõi sự cố bằng Firebase Crashlytic. 13*](#_Toc181280862)

[*Hình 3.1 Tạo Android Oauth Client 14*](#_Toc181280863)

[*Hình 3.2 Cấu hình Client ID cho ứng dụng Android trên Google Cloud 15*](#_Toc181280864)

[*Hình 3.3 Hướng dẫn lấy Key Hash cho Facebook Console để tích hợp Facebook Sign-In. 16*](#_Toc181280865)

[*Hình 3.4 Cấu hình Key Hash trên Facebook Developer Console. 16*](#_Toc181280866)

[*Hình 3.5 Tạo dự án trên Firebase 17*](#_Toc181280867)

[*Hình 3.6 Tạo dự án trên Facebook Developer Console 18*](#_Toc181280868)

[*Hình 3.7 Thiết lập Google Console cho API 18*](#_Toc181280869)

[*Hình 3.8 Kiểm tra các quyền truy cập trong ứng dụng 19*](#_Toc181280870)

[*Hình 4.1 Thiết lập Google Sign-In trong ứng dụng React Native -cấu hình trong Android 21*](#_Toc181280871)

[*Hình 4.2 Thiết lập Google Sign-In trong ứng dụng React Native - Cấu hình trên Firebase Console 21*](#_Toc181280872)

[*Hình 5.1 Cấu hình Android với Facebook SDK 26*](#_Toc181280873)

[*Hình 5.2 Thêm các quyền và cấu hình liên quan đến Facebook 27*](#_Toc181280874)

[*Hình 5.3 Thiết lập trên Facebook Developer Console 27*](#_Toc181280875)

[*Hình 6.1 YouTube IFrame Player API 31*](#_Toc181280876)

[*Hình 6.2 YouTube Data API 32*](#_Toc181280877)

[*Hình 7.1 Theo dõi hành vi người dùng 36*](#_Toc181280878)

[*Hình 7.2 Firebase Cloud Firestore: Lưu trữ dữ liệu đám mây và tối ưu hóa truy cập nhanh 37*](#_Toc181280879)

[*Hình 7.3 Firebase Cloud Storage: Quản lý và lưu trữ dữ liệu lớn, như hình ảnh và video 38*](#_Toc181280880)

[*Hình 7.4 Kích hoạt In-App Messaging trên Firebase Console 38*](#_Toc181280881)

[*Hình 7.5 Firebase Crashlytics: Quản lý sự cố và tối ưu hiệu suất 39*](#_Toc181280882)

[*Hình 7.6 Firebase Authentication: Xác thực người dùng bằng email, Google, hoặc Facebook 40*](#_Toc181280883)

[*Hình 7.7 AdMob: Cách tích hợp quảng cáo và tạo doanh thu 41*](#_Toc181280884)

[*Hình 7.8 Firebase Cloud Messaging: Thông báo đẩy và giao tiếp thời gian thực 42*](#_Toc181280885)

[*Hình 8.1 Cấu hình API Key cho Google Maps trên Android và iOS 44*](#_Toc181280886)

[*Hình 8.2 Google Maps Directions API 48*](#_Toc181280887)

**MỤC LỤC**

[DANH MỤC HÌNH ẢNH 4](#_Toc181280672)

[Chương 1: GIỚI THIỆU 7](#_Toc181280673)

[Chương 2: TỔNG QUAN VỀ FIREBASE VÀ REACT 8](#_Toc181280674)

[Chương 3: CÁC BƯỚC CHUẨN BỊ BAN ĐẦU 12](#_Toc181280675)

[**3.1** **Cách lấy SHA1 Fingerprint cho Google Console** 12](#_Toc181280676)

[**3.2** **Những yêu cầu cấu hình khác cần thiết** 15](#_Toc181280677)

[3.2.1 Tạo dự án trên Firebase 15](#_Toc181280678)

[3.2.2 Tạo dự án trên Facebook Developer Console 15](#_Toc181280679)

[3.2.3 Thiết lập Google Console cho API 16](#_Toc181280680)

[3.2.4 Kiểm tra các quyền truy cập trong ứng dụng 17](#_Toc181280681)

[Chương 4: TÍCH HỢP GOOGLE SIGN-IN 18](#_Toc181280682)

[4.1 Thiết lập Google Sign-In trong ứng dụng React Native 18](#_Toc181280683)

[4.1.1 Cài đặt thư viện Google Sign-In 18](#_Toc181280684)

[4.1.2 Cấu hình trong Android 18](#_Toc181280685)

[4.1.3 Cấu hình trên Firebase Console 19](#_Toc181280686)

[4.2 Các bước xác thực người dùng với Google Sign-In 20](#_Toc181280687)

[4.2.1 Khởi tạo Google Sign-In 20](#_Toc181280688)

[4.2.2 Tạo hàm đăng nhập Google Sign-In 20](#_Toc181280689)

[4.2.3 Xử lý kết quả đăng nhập và lưu trữ thông tin 21](#_Toc181280690)

[4.3 Đăng xuất Google Sign-In 21](#_Toc181280691)

[4.4 Kiểm tra trạng thái đăng nhập 22](#_Toc181280692)

[Chương 5: TÍCH HỢP FACEBOOK SIGN-IN 23](#_Toc181280693)

[5.1 Cài đặt và cấu hình Facebook SDK cho React Native 23](#_Toc181280694)

[5.1.1 Cài đặt thư viện Facebook SDK 23](#_Toc181280695)

[5.1.2 Cấu hình Android với Facebook SDK 23](#_Toc181280696)

[5.1.3 Thiết lập trên Facebook Developer Console 25](#_Toc181280697)

[5.2 Thiết lập hàm đăng nhập Facebook trong React Native 26](#_Toc181280698)

[5.2.1 Khởi tạo Facebook SDK 26](#_Toc181280699)

[5.2.2 Tạo hàm đăng nhập với Facebook 26](#_Toc181280700)

[5.3 Xử lý kết quả đăng nhập và lưu trữ thông tin người dùng 27](#_Toc181280701)

[5.4 Đăng xuất khỏi Facebook 27](#_Toc181280702)

[Chương 6: TÍCH HỢP YOUTUBE VIDEO 29](#_Toc181280703)

[6.2 Sử dụng thư viện React Native YouTube 30](#_Toc181280704)

[6.2.1 Cài đặt thư viện YouTube 30](#_Toc181280705)

[6.2.2 Sử dụng YouTube Player trong ứng dụng 30](#_Toc181280706)

[6.3 Điều chỉnh và tùy biến YouTube Player 32](#_Toc181280707)

[6.3.1 Điều chỉnh kích thước và thiết kế 32](#_Toc181280708)

[6.3.2 Sử dụng nhiều sự kiện hơn 32](#_Toc181280709)

[Chương 7: TÍCH HỢP FIREBASE VÀ CÁC DỊCH VỤ LIÊN QUAN 33](#_Toc181280710)

[7.1 Firebase Analytics: Cách thu thập và phân tích hành vi người dùng 33](#_Toc181280711)

[7.1.1 Cài đặt Firebase Analytics 33](#_Toc181280712)

[7.1.2 Theo dõi hành vi người dùng 33](#_Toc181280713)

[7.2 Firebase Cloud Firestore: Lưu trữ dữ liệu đám mây và tối ưu hóa truy cập nhanh 34](#_Toc181280714)

[7.2.1 Cài đặt và khởi tạo Firestore 34](#_Toc181280715)

[7.2.2 Lấy và đồng bộ dữ liệu 35](#_Toc181280716)

[7.3 Firebase Cloud Storage: Quản lý và lưu trữ dữ liệu lớn, như hình ảnh và video 35](#_Toc181280717)

[7.3.1 Cài đặt và cấu hình Cloud Storage 35](#_Toc181280718)

[7.3.2 Tải lên và truy xuất tệp 35](#_Toc181280719)

[7.4 Firebase In-App Messaging: Gửi thông báo trực tiếp trong ứng dụng 36](#_Toc181280720)

[7.4.1 Kích hoạt In-App Messaging trên Firebase Console 36](#_Toc181280721)

[7.5 Firebase Crashlytics: Quản lý sự cố và tối ưu hiệu suất 36](#_Toc181280722)

[7.5.1 Cài đặt Firebase Crashlytics 37](#_Toc181280723)

[7.5.2 Báo cáo lỗi 37](#_Toc181280724)

[7.6 Firebase Authentication: Xác thực người dùng bằng email, Google, hoặc Facebook 38](#_Toc181280725)

[7.6.1 Cài đặt và cấu hình Firebase Auth 38](#_Toc181280726)

[7.6.2 Tích hợp đăng nhập 38](#_Toc181280727)

[7.7 AdMob: Cách tích hợp quảng cáo và tạo doanh thu 39](#_Toc181280728)

[7.7.1 Cài đặt thư viện AdMob 39](#_Toc181280729)

[7.7.2 Hiển thị quảng cáo 39](#_Toc181280730)

[7.8 Firebase Cloud Messaging: Thông báo đẩy và giao tiếp thời gian thực 40](#_Toc181280731)

[7.8.1 Cài đặt Firebase Messaging 40](#_Toc181280732)

[7.8.2 Xử lý thông báo đẩy 40](#_Toc181280733)

[Chương 8: TRIỂN KHAI REACT NATIVE MAPS 41](#_Toc181280734)

[8.1 Hướng dẫn tích hợp bản đồ với Google Maps API 41](#_Toc181280735)

[8.1.1 Cài đặt thư viện React Native Maps 41](#_Toc181280736)

[8.1.2 Cấu hình API Key cho Google Maps trên Android và iOS 41](#_Toc181280737)

[8.2 Sử dụng React Native Maps trong ứng dụng 43](#_Toc181280738)

[8.3 Tạo điểm đánh dấu trên bản đồ 44](#_Toc181280739)

[8.4 Tìm đường và hiển thị lộ trình 45](#_Toc181280740)

[Chương 9: KẾT LUẬN 49](#_Toc181280741)

[9.1 Đánh giá những lợi ích của việc sử dụng các tích hợp bên thứ ba 49](#_Toc181280742)

[9.2 Tầm quan trọng của việc lựa chọn và triển khai hợp lý các dịch vụ cho ứng dụng 49](#_Toc181280743)

[9.3 Đề xuất cho các nhà phát triển khi tích hợp dịch vụ bên thứ ba vào ứng dụng của mình 50](#_Toc181280744)

[TÀI LIỆU THAM KHẢO 51](#_Toc181280745)

# GIỚI THIỆU

Trong phát triển ứng dụng di động, Third-Party Integration (tích hợp dịch vụ bên thứ ba) là một phần quan trọng giúp gia tăng tính năng và nâng cao trải nghiệm người dùng mà không cần phát triển lại từ đầu các chức năng phức tạp. Các dịch vụ bên thứ ba cung cấp nhiều công cụ và tiện ích hữu ích như hệ thống xác thực, phân tích dữ liệu, lưu trữ đám mây, nhắn tin, quảng cáo, và các tích hợp với mạng xã hội. Nhờ đó, các nhà phát triển có thể tập trung vào việc tối ưu giao diện và trải nghiệm người dùng, đồng thời tiết kiệm thời gian và nguồn lực.

React Native, một framework phát triển ứng dụng đa nền tảng, đã mở ra nhiều cơ hội để tích hợp các dịch vụ này vào ứng dụng di động một cách linh hoạt và hiệu quả. Với sự hỗ trợ từ cộng đồng và các thư viện mạnh mẽ, React Native dễ dàng tương thích với nhiều dịch vụ bên thứ ba, từ Firebase của Google cho đến các API mạng xã hội của Facebook và Google, tạo nên nền tảng lý tưởng cho các ứng dụng di động hiện đại.

Mục tiêu của bài báo cáo là giới thiệu các công cụ và dịch vụ tích hợp phổ biến trong React Native, đồng thời làm rõ cách chúng góp phần nâng cao trải nghiệm người dùng. Bằng cách phân tích các dịch vụ như Firebase, React Native Maps, Google Sign-In, Facebook Sign-In, và YouTube Integration, báo cáo sẽ làm rõ cách từng công cụ đóng góp vào việc tạo ra một ứng dụng di động hoàn thiện, đồng thời cung cấp hướng dẫn cơ bản để các nhà phát triển dễ dàng áp dụng chúng vào dự án của mình.

A diagram of a application

Description automatically generated

Hình 1.1 Quy trình tích hợp API của bên thứ ba.

# TỔNG QUAN VỀ FIREBASE VÀ REACT

Firebase là một nền tảng phát triển ứng dụng di động và web được cung cấp bởi Google, mang đến một hệ sinh thái toàn diện với nhiều dịch vụ đa dạng như lưu trữ dữ liệu, xác thực người dùng, phân tích, và quản lý thông báo. Firebase giúp các nhà phát triển triển khai và quản lý ứng dụng một cách hiệu quả mà không cần phải xây dựng hệ thống từ đầu.

A blue and white background with a blue and white symbol

Description automatically generated

Hình 2.1 Sự kết hợp giữa Firebase và React trong phát triển ứng dụng web hoặc di động.

Một số lợi ích nổi bật của Firebase trong phát triển ứng dụng di động bao gồm:

* **Lưu trữ dữ liệu thời gian thực**: Firebase Realtime Database và Cloud Firestore hỗ trợ lưu trữ và đồng bộ dữ liệu liên tục giữa các thiết bị, tạo ra trải nghiệm liền mạch cho người dùng.

A yellow and orange folder with black text

Description automatically generated

Hình 2.2 Các tính năng chính của Firebase.

* **Xác thực bảo mật:** Firebase Authentication giúp đơn giản hóa quy trình xác thực người dùng với các phương thức đăng nhập phổ biến như email, Google, Facebook, và các tài khoản mạng xã hội khác.

A screen shot of a phone

Description automatically generated

Hình 2.3 Quá trình xác thực bảo mật bằng Firebase Authentication.

* **Quản lý thông báo:** Firebase Cloud Messaging cho phép gửi thông báo đến người dùng một cách hiệu quả, duy trì tương tác và gắn kết.

A cloud messaging and a cloud computing service

Description automatically generated with medium confidence

Hình 2.4 Quá trình quản lý thông báo sử dụng Firebase Cloud Messaging.

* **Quảng cáo và kiếm tiền**: Firebase AdMob giúp tích hợp quảng cáo để tạo nguồn thu cho ứng dụng.

Hình 2.5 Sự kết hợp giữa AdMobA screenshot of a phone

Description automatically generated của Google và Firebase

* A cellphone with graphs and charts

  Description automatically generated**Phân tích và theo dõi sự cố**: Firebase Analytics và Firebase Crashlytics hỗ trợ theo dõi hành vi người dùng và quản lý lỗi, giúp nhà phát triển nâng cao trải nghiệm người dùng.

Hình 2.6 Phân tích hành vi người dùng bằng Google Analytics for Firebase.

A phone with a error message

Description automatically generated

Hình 2.7 Theo dõi sự cố bằng Firebase Crashlytic.

Với những dịch vụ mạnh mẽ này, Firebase giúp các nhà phát triển dễ dàng triển khai và vận hành ứng dụng trên quy mô lớn, giảm thiểu các yêu cầu về cơ sở hạ tầng và chi phí phát triển.

React Native là một framework do Facebook phát triển, cho phép tạo ra ứng dụng di động đa nền tảng chỉ với một mã nguồn chung bằng JavaScript. Điều này giúp tiết kiệm thời gian và công sức khi phát triển ứng dụng cho cả iOS và Android. React Native nổi bật với khả năng hiệu suất gần tương đương ứng dụng bản địa (native) và có một cộng đồng lớn hỗ trợ, cung cấp nhiều thư viện mở rộng và tài liệu phong phú.

React Native được đánh giá cao trong việc tích hợp với các dịch vụ bên thứ ba như Firebase nhờ khả năng kết hợp dễ dàng các plugin và SDK của các dịch vụ này vào ứng dụng. Cụ thể, React Native Firebase là một tập hợp các thư viện chính thức giúp kết nối liền mạch với các dịch vụ Firebase, từ xác thực, lưu trữ dữ liệu đến phân tích. Khả năng tích hợp nhanh chóng và mượt mà giúp các nhà phát triển có thể dễ dàng áp dụng các dịch vụ bên thứ ba để mở rộng tính năng mà không gặp nhiều rào cản kỹ thuật.

Nhờ vào sự kết hợp này, React Native và Firebase tạo nên một giải pháp mạnh mẽ cho việc phát triển ứng dụng di động hiện đại, giúp các nhà phát triển vừa nhanh chóng triển khai tính năng vừa đảm bảo được hiệu suất và bảo mật cho người dùng.

# CÁC BƯỚC CHUẨN BỊ BAN ĐẦU

Để tích hợp Google Sign-In và Facebook Sign-In vào ứng dụng di động, chúng ta cần thực hiện một số bước chuẩn bị và cấu hình cơ bản trên Google Console, Facebook Developer Console, và Firebase. Các bước này đảm bảo rằng ứng dụng có thể kết nối an toàn với các dịch vụ bên thứ ba và sử dụng các tính năng xác thực, phân tích, và lưu trữ dữ liệu. Dưới đây là hướng dẫn cụ thể:

* 1. **Cách lấy** **SHA1 Fingerprint cho Google Console**

SHA1 Fingerprint là một mã bảo mật duy nhất đại diện cho ứng dụng, giúp Google xác minh ứng dụng và cho phép sử dụng các API của Google, bao gồm Google Sign-In. Để lấy SHA1 Fingerprint, làm theo các bước sau:

**Bước 1:** Sử dụng Android Studio

* Mở Android Studio và mở dự án React Native.
* Chọn Gradle ở thanh công cụ bên phải, sau đó điều hướng đến Tasks > android > signingReport.
* A screenshot of a computer

  Description automatically generatedChạy signingReport, kết quả sẽ hiển thị SHA1 và SHA-256 fingerprints của ứng dụng trong bảng Console.

Hình 3.1 Tạo Android Oauth Client

**Bước 2:** **Sử dụng Terminal** (nếu đã cài đặt keytool)

* Mở Terminal và nhập lệnh sau:

bash

Copy code

keytool -list -v -keystore ~/.android/debug.keystore -alias androiddebugkey -storepass android -keypass android

* Kết quả sẽ hiển thị SHA1 cùng với các thông tin khác.

**Bước 3:** Sau khi có được SHA1 Fingerprint, truy cập vào Google Cloud Console, chọn dự án ứng dụng, và thêm SHA1 vào mục Credentials trong Android API Key để cho phép ứng dụng sử dụng Google Sign-In.

A screenshot of a computer

Description automatically generated

*Hình 3.2 Cấu hình Client ID cho ứng dụng Android trên Google Cloud*

Hướng dẫn lấy Key Hash cho Facebook Console để tích hợp Facebook Sign-In

Key Hash là một mã duy nhất giúp Facebook xác thực ứng dụng khi người dùng đăng nhập bằng Facebook. Để lấy Key Hash, có thể thực hiện các bước sau:

**Bước 1:** Sử dụng lệnh trong Terminal

* + Mở **Terminal** và nhập lệnh sau (thay <path\_to\_your\_keystore> bằng đường dẫn

đến file keystore):

bash

Copy code

keytool -exportcert -alias androiddebugkey -keystore ~/.android/debug.keystore | openssl sha1 -binary | openssl base64

* + Lệnh trên sẽ hiển thị Key Hash. Sao chép Key Hash này để cấu hình trên Facebook Console.

A screenshot of a computer

Description automatically generated

*Hình 3.3 Hướng dẫn lấy Key Hash cho Facebook Console để tích hợp Facebook Sign-In.*

Bước 2: Cấu hình Key Hash trên Facebook Developer Console

* + Truy cập **Facebook Developer Console** và đăng nhập.
  + Tạo một dự án ứng dụng mới nếu chưa có, sau đó điều hướng đến **Settings > Basic**.
  + Trong phần **Android**, dán Key Hash vào trường **Key Hashes** và lưu lại.

A screenshot of a computer

Description automatically generated

*Hình 3.4 Cấu hình Key Hash trên Facebook Developer Console.*

* 1. **Những yêu cầu cấu hình khác cần thiết**

Để triển khai tích hợp thành công, cần hoàn thành các cấu hình cơ bản cho Google, Facebook và Firebase như sau:

### Tạo dự án trên Firebase

* + Truy cập Firebase Console và đăng nhập.
  + Tạo một dự án mới và thêm ứng dụng Android vào dự án.
  + Cấu hình SHA1 Fingerprint như đã hướng dẫn ở mục 3.1 và tải xuống file google-services.json.
  + Đặt file google-services.json vào thư mục Android của dự án React Native.

A screenshot of a computer

Description automatically generated

Hình 3.5 Tạo dự án trên Firebase

### Tạo dự án trên Facebook Developer Console

* + Truy cập [Facebook Developer Console](https://developers.facebook.com/" \t "_new) và đăng nhập.
  + Tạo một ứng dụng mới, sau đó vào phần **Products** và chọn **Facebook Login** để kích hoạt Facebook Sign-In.
  + Cấu hình Key Hash và thêm các quyền cần thiết cho ứng dụng, như email và public\_profile.

A screenshot of a facebook account

Description automatically generated

Hình 3.6 Tạo dự án trên Facebook Developer Console

### Thiết lập Google Console cho API

* + Trong **Google Cloud Console**, vào mục **API & Services > Credentials** và tạo API Key cho ứng dụng Android.
  + Bật các API cần thiết như Google Sign-In API và Maps SDK nếu cần sử dụng bản đồ.

A screenshot of a library

Description automatically generated

*Hình 3.7* *Thiết lập Google Console cho API*

### Kiểm tra các quyền truy cập trong ứng dụng

* + Đảm bảo ứng dụng đã yêu cầu các quyền truy cập cần thiết trong AndroidManifest.xml, bao gồm quyền truy cập Internet và quyền xác thực tài khoản.

A screen shot of a computer screen

Description automatically generated

*Hình 3.8 Kiểm tra các quyền truy cập trong ứng dụng*

Với các bước chuẩn bị ban đầu này, chúng ta đã sẵn sàng cấu hình ứng dụng để tích hợp với các dịch vụ bên thứ ba như Google và Facebook, giúp tối ưu trải nghiệm người dùng và tăng cường tính năng xác thực cho ứng dụng di động của mình.

# TÍCH HỢP GOOGLE SIGN-IN

Để tích hợp **Google Sign-In** vào ứng dụng React Native, chúng ta cần thực hiện một số bước cài đặt và cấu hình cần thiết để ứng dụng có thể tương tác với tài khoản Google của người dùng. Dưới đây là các bước chi tiết để thiết lập và xử lý xác thực người dùng với Google.

## Thiết lập Google Sign-In trong ứng dụng React Native

Để bắt đầu, cần cài đặt thư viện @react-native-google-signin/google-signin, một gói hỗ trợ Google Sign-In trong ứng dụng React Native.

### Cài đặt thư viện Google Sign-In

* + Mở **Terminal** và chạy lệnh sau để cài đặt thư viện:

bash

Copy code

npm install @react-native-google-signin/google-signin

* + Sau khi cài đặt, chạy lệnh để liên kết thư viện với dự án React Native:

bash

Copy code

npx pod-install

### Cấu hình trong Android

* + Mở file android/app/build.gradle, thêm apply plugin: 'com.google.gms.google-services' vào cuối file.

A screenshot of a computer program

Description automatically generated

* + Mở file android/build.gradle và đảm bảo dòng classpath 'com.google.gms:google-services:4.3.10' (hoặc phiên bản mới nhất) đã được thêm vào phần dependencies.

A screenshot of a computer

Description automatically generated

Hình 4.1 Thiết lập Google Sign-In trong ứng dụng React Native -cấu hình trong Android

### Cấu hình trên Firebase Console

* + Truy cập **Firebase Console**, chọn dự án sau đó vào **Authentication > Sign-in Method** và bật **Google Sign-In**.
  + Đảm bảo đã tải file google-services.json về và đặt vào thư mục android/app của dự án.

A screenshot of a computer

Description automatically generated

*Hình 4.2 Thiết lập Google Sign-In trong ứng dụng React Native - Cấu hình trên Firebase Console*

## Các bước xác thực người dùng với Google Sign-In

Sau khi cài đặt và cấu hình, tiếp theo là phần mã hóa xử lý xác thực người dùng trong ứng dụng React Native.

### Khởi tạo Google Sign-In

* + Trong file React Native (ví dụ: App.js), nhập GoogleSignin và khởi tạo nó bằng Web Client ID đã cấu hình trên Firebase.

javascript

Copy code

import { GoogleSignin } from '@react-native-google-signin/google-signin';

GoogleSignin.configure({

webClientId: 'YOUR\_WEB\_CLIENT\_ID', // Thay YOUR\_WEB\_CLIENT\_ID bằng Web Client ID

});

### Tạo hàm đăng nhập Google Sign-In

* + Tạo một hàm để xử lý quá trình đăng nhập bằng Google.

javascript

Copy code

const signInWithGoogle = async () => {

try {

await GoogleSignin.hasPlayServices(); // Kiểm tra dịch vụ Google Play

const userInfo = await GoogleSignin.signIn(); // Bắt đầu quy trình đăng nhập

console.log('Thông tin người dùng:', userInfo);

} catch (error) {

console.error('Lỗi đăng nhập Google:', error);

}

};

* + hasPlayServices() kiểm tra xem thiết bị có hỗ trợ Google Play Services hay không. signIn() sẽ mở giao diện Google Sign-In và trả về thông tin người dùng.

### Xử lý kết quả đăng nhập và lưu trữ thông tin

* + Sau khi người dùng đăng nhập thành công, chúng ta có thể lưu trữ thông tin người dùng (như ID, tên, và email) để sử dụng trong ứng dụng.

javascript

Copy code

const handleLogin = async () => {

const userInfo = await signInWithGoogle();

if (userInfo) {

// Lưu thông tin người dùng vào Redux, Context, hoặc AsyncStorage

}

};

## Đăng xuất Google Sign-In

Để đảm bảo bảo mật và tùy chọn đăng xuất cho người dùng, nên tạo thêm một hàm để xử lý quá trình đăng xuất.

**Tạo hàm đăng xuất:**

javascript

Copy code

const signOutFromGoogle = async () => {

try {

await GoogleSignin.signOut();

console.log('Đã đăng xuất khỏi Google');

} catch (error) {

console.error('Lỗi đăng xuất:', error);

}

};

* + Hàm signOut() sẽ đăng xuất người dùng và xóa thông tin phiên hiện tại.

## Kiểm tra trạng thái đăng nhập

Chúng ta có thể kiểm tra xem người dùng hiện có đang đăng nhập hay không bằng cách sử dụng isSignedIn() từ GoogleSignin.

**Kiểm tra trạng thái đăng nhập**

javascript

Copy code

const checkSignInStatus = async () => {

const isSignedIn = await GoogleSignin.isSignedIn();

if (isSignedIn) {

console.log('Người dùng hiện đang đăng nhập');

} else {

console.log('Người dùng chưa đăng nhập');

}

};

Với các bước trên, chúng ta đã hoàn thành việc tích hợp Google Sign-In vào ứng dụng React Native của mình, cho phép người dùng dễ dàng đăng nhập bằng tài khoản Google. Việc này giúp tăng cường trải nghiệm người dùng và đảm bảo tính bảo mật cho ứng dụng.

# TÍCH HỢP FACEBOOK SIGN-IN

Để tích hợp **Facebook Sign-In** vào ứng dụng React Native, chúng ta cần thiết lập **Facebook App ID** và cấu hình các API liên quan. Dưới đây là các bước chi tiết để thực hiện tích hợp Facebook Sign-In.

## Cài đặt và cấu hình Facebook SDK cho React Native

Trước khi bắt đầu, chúng ta sẽ cài đặt thư viện react-native-fbsdk-next – một gói hỗ trợ Facebook Sign-In cho React Native.

### Cài đặt thư viện Facebook SDK

* + Mở **Terminal** và chạy lệnh sau để cài đặt:

bash

Copy code

npm install react-native-fbsdk-next

* + Sau đó, chạy lệnh này để liên kết thư viện với dự án:

bash

Copy code

npx pod-install

### Cấu hình Android với Facebook SDK

* + Mở file android/app/src/main/res/values/strings.xml và thêm **Facebook App ID**

xml

Copy code

<string name="facebook\_app\_id">YOUR\_FACEBOOK\_APP\_ID</string>

<string name="fb\_login\_protocol\_scheme">fbYOUR\_FACEBOOK\_APP\_ID</string>

A screenshot of a computer

Description automatically generated

Hình 5.1 Cấu hình Android với Facebook SDK

* + Trong file android/app/AndroidManifest.xml, thêm các quyền và cấu hình liên quan đến Facebook:

xml

Copy code

<application>

...

<meta-data android:name="com.facebook.sdk.ApplicationId" android:value="@string/facebook\_app\_id"/>

<activity android:name="com.facebook.FacebookActivity"

android:configChanges="keyboard|keyboardHidden|screenLayout|screenSize|orientation"

android:label="@string/app\_name" />

<provider android:authorities="com.facebook.app.FacebookContentProviderYOUR\_FACEBOOK\_APP\_ID"

android:name="com.facebook.FacebookContentProvider"

android:exported="true"/>

</application>

A screenshot of a computer

Description automatically generated

Hình 5.2 Thêm các quyền và cấu hình liên quan đến Facebook

### Thiết lập trên Facebook Developer Console

* + Truy cập [Facebook Developer Console](https://developers.facebook.com/), đăng nhập và tạo một ứng dụng mới.
  + Trong **Settings > Basic**, chúng ta sẽ thấy **Facebook App ID** và **App Secret**. Sử dụng **Facebook App ID** để cấu hình ứng dụng.
  + Trong **Products**, chọn **Facebook Login** và bật nó lên, sau đó thiết lập **OAuth Redirect URI** với đường dẫn phù hợp, ví dụ:

perl

Copy code

<https://www.facebook.com/connect/login_success.html>

A screenshot of a login page

Description automatically generated

*Hình 5.3 Thiết lập trên Facebook Developer Console*

## Thiết lập hàm đăng nhập Facebook trong React Native

Sau khi cấu hình Facebook SDK, chúng ta sẽ viết mã để xử lý đăng nhập Facebook trong ứng dụng React Native.

### Khởi tạo Facebook SDK

* + Trong file React Native (ví dụ: App.js), nhập và khởi tạo LoginManager và AccessToken từ react-native-fbsdk-next:

javascript

Copy code

import { LoginManager, AccessToken } from 'react-native-fbsdk-next';

### Tạo hàm đăng nhập với Facebook

* + Tạo một hàm để xử lý quá trình đăng nhập bằng Facebook.

javascript

Copy code

const signInWithFacebook = async () => {

try {

const result = await LoginManager.logInWithPermissions(['public\_profile', 'email']);

if (result.isCancelled) {

console.log('Đăng nhập Facebook bị hủy');

} else {

const data = await AccessToken.getCurrentAccessToken();

if (data) {

console.log('Access Token:', data.accessToken.toString());

}

}

} catch (error) {

console.error('Lỗi đăng nhập Facebook:', error);

}

};

* + logInWithPermissions(['public\_profile', 'email']) mở giao diện Facebook Login và yêu cầu quyền truy cập vào hồ sơ công khai và email của người dùng. getCurrentAccessToken() sẽ trả về token để xác thực người dùng.

## Xử lý kết quả đăng nhập và lưu trữ thông tin người dùng

Sau khi người dùng đăng nhập thành công, chúng ta có thể lưu trữ thông tin người dùng để sử dụng trong ứng dụng.

**Lưu trữ thông tin người dùng:**

javascript

Copy code

const handleFacebookLogin = async () => {

const userInfo = await signInWithFacebook();

if (userInfo) {

// Lưu thông tin người dùng vào Redux, Context, hoặc AsyncStorage

}

};

## Đăng xuất khỏi Facebook

Nên cung cấp một tùy chọn đăng xuất để người dùng có thể đăng xuất khỏi tài khoản Facebook của họ.

**Tạo hàm đăng xuất:**

javascript

Copy code

const signOutFromFacebook = async () => {

try {

await LoginManager.logOut();

console.log('Đã đăng xuất khỏi Facebook');

} catch (error) {

console.error('Lỗi đăng xuất:', error);

}

};

Với các bước trên, chúng ta đã hoàn thành việc tích hợp Facebook Sign-In vào ứng dụng React Native của mình, giúp người dùng dễ dàng đăng nhập bằng tài khoản Facebook. Điều này không chỉ tăng cường trải nghiệm người dùng mà còn giúp thu thập được các thông tin quan trọng từ tài khoản mạng xã hội của họ một cách bảo mật.

# TÍCH HỢP YOUTUBE VIDEO

Tích hợp YouTube Video vào ứng dụng React Native cho phép người dùng xem các nội dung đa phương tiện trực tiếp mà không cần rời khỏi ứng dụng. Điều này nâng cao trải nghiệm người dùng và giúp ứng dụng trở nên đa dạng hơn về mặt nội dung. Để tích hợp YouTube Video, chúng ta có thể sử dụng **YouTube Player API** hoặc thư viện React Native hỗ trợ.

6.1 Các API để tích hợp YouTube video trong ứng dụng

YouTube cung cấp API để nhúng và quản lý các video trong ứng dụng. Dưới đây là một số API chính:

6.1.1 YouTube IFrame Player API

* + API này cho phép nhúng video dưới dạng iframe, hỗ trợ đầy đủ các chức năng như phát, tạm dừng, tua lại, hoặc điều chỉnh chất lượng video.
  + Dễ sử dụng trong ứng dụng web hoặc nếu sử dụng WebView trong React Native.

A screenshot of a browser

Description automatically generated

*Hình 6.1 YouTube IFrame Player API*

6.1.2 YouTube Data API

* + API này cho phép truy xuất thông tin từ YouTube như danh sách phát, thông tin video, và các kênh.
  + Hữu ích nếu muốn tích hợp tìm kiếm video hoặc hiển thị thông tin chi tiết của video.

A screenshot of a browser

Description automatically generated

*Hình 6.2 YouTube Data API*

## Sử dụng thư viện React Native YouTube

Trong React Native, chúng ta có thể sử dụng thư viện react-native-youtube-iframe để nhúng video YouTube. Thư viện này hỗ trợ YouTube IFrame Player API, giúp nhúng video dễ dàng và cung cấp các chức năng điều khiển video trực tiếp.

### Cài đặt thư viện YouTube

* + Mở **Terminal** và chạy lệnh sau để cài đặt:

bash

Copy code

npm install react-native-youtube-iframe

### Sử dụng YouTube Player trong ứng dụng

* + Sau khi cài đặt, có thể sử dụng YoutubePlayer từ thư viện để nhúng video vào trong React Native. Dưới đây là ví dụ cách sử dụng:

javascript

Copy code

import React, { useState, useCallback } from 'react';

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

import YoutubePlayer from 'react-native-youtube-iframe';

const YouTubeVideoScreen = () => {

const [playing, setPlaying] = useState(false);

const onStateChange = useCallback((state) => {

if (state === "ended") {

setPlaying(false);

console.log("Video đã kết thúc!");

}

}, []);

const togglePlaying = useCallback(() => {

setPlaying((prev) => !prev);

}, []);

return (

<View>

<Text>Video YouTube</Text>

<YoutubePlayer

height={200}

play={playing}

videoId={'dQw4w9WgXcQ'} // Thay 'dQw4w9WgXcQ' bằng ID của video muốn nhúng

onChangeState={onStateChange}

/>

<Button title={playing ? "Tạm dừng" : "Phát"} onPress={togglePlaying} />

</View>

);

};

export default YouTubeVideoScreen;

* + **Giải thích**:
    - YoutubePlayer là thành phần chính dùng để phát video, videoId là ID của video trên YouTube.
    - onStateChange giúp theo dõi trạng thái của video (đang phát, đã kết thúc, tạm dừng).
    - togglePlaying cho phép người dùng bắt đầu hoặc tạm dừng video.

## Điều chỉnh và tùy biến YouTube Player

Thư viện react-native-youtube-iframe cung cấp các tùy chọn điều chỉnh kích thước và trạng thái của video, giúp dễ dàng tùy biến trải nghiệm xem video trong ứng dụng.

### Điều chỉnh kích thước và thiết kế

* + Chúng ta có thể điều chỉnh height và width của YoutubePlayer để phù hợp với giao diện ứng dụng, đặc biệt khi sử dụng cho màn hình dọc hoặc ngang.

### Sử dụng nhiều sự kiện hơn

* + Chúng ta có thể mở rộng tính năng bằng cách lắng nghe các sự kiện khác như onError, onReady, và onPlaybackRateChange để xử lý các lỗi hoặc cập nhật trạng thái video.

Với các bước trên, chúng ta đã tích hợp thành công YouTube Video vào ứng dụng React Native của mình, giúp người dùng trải nghiệm nội dung phong phú và tương tác cao. Thao tác này giúp tối ưu trải nghiệm người dùng và làm cho ứng dụng trở nên hấp dẫn hơn.

# TÍCH HỢP FIREBASE VÀ CÁC DỊCH VỤ LIÊN QUAN

Firebase cung cấp một loạt dịch vụ giúp nâng cao trải nghiệm người dùng và tối ưu hóa hiệu suất ứng dụng. Dưới đây là hướng dẫn tích hợp các dịch vụ phổ biến của Firebase trong ứng dụng React Native.

## Firebase Analytics: Cách thu thập và phân tích hành vi người dùng

**Firebase Analytics** là công cụ phân tích mạnh mẽ giúp theo dõi và phân tích hành vi người dùng trong ứng dụng, từ đó tối ưu hóa trải nghiệm và hiệu quả kinh doanh.

### Cài đặt Firebase Analytics

* + Mở **Terminal** và cài đặt thư viện:

bash

Copy code

npm install @react-native-firebase/analytics

* + Khởi tạo Analytics trong mã nguồn React Native:

javascript

Copy code

import analytics from '@react-native-firebase/analytics';

// Gửi sự kiện tùy chỉnh

await analytics().logEvent('screen\_view', { screen\_name: 'HomeScreen' });

### Theo dõi hành vi người dùng

* + Chúng ta có thể gửi các sự kiện tùy chỉnh để theo dõi các hoạt động quan trọng trong ứng dụng, như đăng nhập, mua hàng, hoặc xem sản phẩm.

A screenshot of a computer

Description automatically generated

Hình 7.1 Theo dõi hành vi người dùng

## Firebase Cloud Firestore: Lưu trữ dữ liệu đám mây và tối ưu hóa truy cập nhanh

**Firebase Cloud Firestore** cung cấp dịch vụ lưu trữ dữ liệu dạng NoSQL, giúp ứng dụng có thể truy xuất và đồng bộ dữ liệu thời gian thực.

### Cài đặt và khởi tạo Firestore

bash

Copy code

npm install @react-native-firebase/firestore

* + Sau khi cài đặt, khởi tạo Firestore trong mã nguồn:

javascript

Copy code

import firestore from '@react-native-firebase/firestore';

// Thêm dữ liệu vào Firestore

await firestore().collection('users').add({ name: 'Thanh', age: 23 });

### Lấy và đồng bộ dữ liệu

* + Firestore hỗ trợ truy xuất dữ liệu theo thời gian thực, giúp ứng dụng luôn cập nhật thông tin mới nhất cho người dùng.

A screenshot of a computer

Description automatically generated

Hình 7.2 Firebase Cloud Firestore: Lưu trữ dữ liệu đám mây và tối ưu hóa truy cập nhanh

## Firebase Cloud Storage: Quản lý và lưu trữ dữ liệu lớn, như hình ảnh và video

**Firebase Cloud Storage** là giải pháp lưu trữ mạnh mẽ cho các tập tin lớn như ảnh và video.

### Cài đặt và cấu hình Cloud Storage

bash

Copy code

npm install @react-native-firebase/storage

### Tải lên và truy xuất tệp

javascript

Copy code

import storage from '@react-native-firebase/storage';

const reference = storage().ref('images/avatar.jpg');

await reference.putFile('/path/to/image');

A screenshot of a computer

Description automatically generated

*Hình 7.3 Firebase Cloud Storage: Quản lý và lưu trữ dữ liệu lớn, như hình ảnh và video*

## Firebase In-App Messaging: Gửi thông báo trực tiếp trong ứng dụng

**Firebase In-App Messaging** cho phép gửi thông báo khi người dùng đang sử dụng ứng dụng, giúp tăng tương tác.

### Kích hoạt In-App Messaging trên Firebase Console

* + Tạo chiến dịch gửi thông báo từ Firebase Console, cấu hình các điều kiện và thời gian.

A screenshot of a phone

Description automatically generated

*Hình 7.4 Kích hoạt In-App Messaging trên Firebase Console*

## Firebase Crashlytics: Quản lý sự cố và tối ưu hiệu suất

**Firebase Crashlytics** giúp ghi nhận và xử lý các sự cố ứng dụng, tối ưu hóa hiệu suất và bảo trì.

### Cài đặt Firebase Crashlytics

bash

Copy code

npm install @react-native-firebase/crashlytics

### Báo cáo lỗi

javascript

Copy code

import crashlytics from '@react-native-firebase/crashlytics';

crashlytics().log('App crashed due to an unknown error');

A screenshot of a computer

Description automatically generated

*Hình 7.5 Firebase Crashlytics: Quản lý sự cố và tối ưu hiệu suất*

## Firebase Authentication: Xác thực người dùng bằng email, Google, hoặc Facebook

**Firebase Authentication** cung cấp các phương thức xác thực bảo mật, bao gồm đăng nhập qua email, Google và Facebook.

### Cài đặt và cấu hình Firebase Auth

bash

Copy code

npm install @react-native-firebase/auth

### Tích hợp đăng nhập

javascript

Copy code

import auth from '@react-native-firebase/auth';

await auth().signInWithEmailAndPassword('user@example.com', 'password');

A screenshot of a computer

Description automatically generated

*Hình 7.6 Firebase Authentication: Xác thực người dùng bằng email, Google, hoặc Facebook*

## AdMob: Cách tích hợp quảng cáo và tạo doanh thu

**Firebase AdMob** là dịch vụ quảng cáo mạnh mẽ giúp kiếm doanh thu từ ứng dụng.

### Cài đặt thư viện AdMob

bash

Copy code

npm install @react-native-firebase/admob

### Hiển thị quảng cáo

javascript

Copy code

import { BannerAd, BannerAdSize } from '@react-native-firebase/admob';

<BannerAd unitId="ca-app-pub-3940256099942544/6300978111" size={BannerAdSize.FULL\_BANNER} />

A screenshot of a phone

Description automatically generated

*Hình 7.7 AdMob: Cách tích hợp quảng cáo và tạo doanh thu*

## Firebase Cloud Messaging: Thông báo đẩy và giao tiếp thời gian thực

**Firebase Cloud Messaging (FCM)** cho phép gửi thông báo đẩy, giúp giữ liên lạc với người dùng ngay cả khi ứng dụng không hoạt động.

### Cài đặt Firebase Messaging

bash

Copy code

npm install @react-native-firebase/messaging

### Xử lý thông báo đẩy

javascript

Copy code

import messaging from '@react-native-firebase/messaging';

messaging().onMessage(async remoteMessage => {

console.log('Thông báo mới:', remoteMessage);

});

Với các dịch vụ này, Firebase cung cấp một giải pháp toàn diện, giúp ứng dụng trở nên phong phú hơn về tính năng và tối ưu về hiệu suất, từ đó nâng cao trải nghiệm người dùng.

A screenshot of a computer

Description automatically generated

*Hình 7.8 Firebase Cloud Messaging: Thông báo đẩy và giao tiếp thời gian thực*

# TRIỂN KHAI REACT NATIVE MAPS

Việc tích hợp bản đồ vào ứng dụng React Native giúp cải thiện trải nghiệm người dùng, đặc biệt đối với các ứng dụng yêu cầu định vị và chỉ đường. **React Native Maps** là một thư viện phổ biến để tích hợp Google Maps hoặc Apple Maps vào ứng dụng.

## Hướng dẫn tích hợp bản đồ với Google Maps API

Để bắt đầu, cần cài đặt thư viện react-native-maps và cấu hình Google Maps API cho ứng dụng của mình.

### Cài đặt thư viện React Native Maps

* + Mở **Terminal** và cài đặt thư viện

bash

Copy code

npm install react-native-maps

* + Đối với iOS, chạy lệnh để cài đặt Pod

bash

Copy code

npx pod-install

### Cấu hình API Key cho Google Maps trên Android và iOS

* + **Android**:
    - Trong file AndroidManifest.xml, thêm API Key vào phần <application>:

xml

Copy code

<meta-data

android:name="com.google.android.geo.API\_KEY"

android:value="YOUR\_GOOGLE\_MAPS\_API\_KEY"/>

* + **iOS**:
    - Mở file AppDelegate.m và thêm API Key như sau:

objc

Copy code

#import <GoogleMaps/GoogleMaps.h>

- (BOOL)application:(UIApplication \*)application didFinishLaunchingWithOptions:(NSDictionary \*)launchOptions {

[GMSServices provideAPIKey:@"YOUR\_GOOGLE\_MAPS\_API\_KEY"];

return YES;

}

A screenshot of a computer

Description automatically generated

Hình 8.1 Cấu hình API Key cho Google Maps trên Android và iOS

## Sử dụng React Native Maps trong ứng dụng

Sau khi cấu hình xong API Key, có thể sử dụng MapView từ react-native-maps để hiển thị bản đồ và thêm các tính năng tùy chỉnh.

Hiển thị bản đồ và cài đặt cơ bản:

* + Ví dụ cơ bản để hiển thị bản đồ với MapView:

javascript

Copy code

import React from 'react';

import { View, StyleSheet } from 'react-native';

import MapView from 'react-native-maps';

const MapScreen = () => {

return (

<View style={styles.container}>

<MapView

style={styles.map}

initialRegion={{

latitude: 37.78825,

longitude: -122.4324,

latitudeDelta: 0.0922,

longitudeDelta: 0.0421,

}}

/>

</View>

);

};

const styles = StyleSheet.create({

container: { flex: 1 },

map: { flex: 1 }

});

export default MapScreen;

* + **Giải thích**:
    - initialRegion đặt khu vực ban đầu của bản đồ với tọa độ trung tâm, đồng thời latitudeDelta và longitudeDelta xác định phạm vi hiển thị bản đồ.

## Tạo điểm đánh dấu trên bản đồ

Chúng ta có thể thêm điểm đánh dấu để hiển thị các vị trí quan trọng trên bản đồ.

Thêm điểm đánh dấu (Marker):

* + Để thêm điểm đánh dấu, sử dụng Marker từ react-native-maps:

javascript

Copy code

import React from 'react';

import { View, StyleSheet } from 'react-native';

import MapView, { Marker } from 'react-native-maps';

const MapWithMarkers = () => {

return (

<View style={styles.container}>

<MapView

style={styles.map}

initialRegion={{

latitude: 37.78825,

longitude: -122.4324,

latitudeDelta: 0.0922,

longitudeDelta: 0.0421,

}}

>

<Marker

coordinate={{ latitude: 37.78825, longitude: -122.4324 }}

title="Điểm đánh dấu"

description="Mô tả vị trí"

/>

</MapView>

</View>

);

};

export default MapWithMarkers;

* + **Giải thích**:
    - Marker cho phép thêm điểm đánh dấu trên bản đồ tại tọa độ cụ thể. Chúng ta có thể tùy chỉnh title và description để hiển thị thông tin vị trí.

## Tìm đường và hiển thị lộ trình

Để tìm đường từ vị trí hiện tại của người dùng đến một địa điểm khác, có thể sử dụng Google Maps Directions API kết hợp với Polyline để vẽ lộ trình trên bản đồ.

Sử dụng Google Maps Directions API để lấy lộ trình:

* + Gửi yêu cầu đến Google Maps Directions API để lấy dữ liệu lộ trình và sau đó sử dụng Polyline để vẽ lộ trình lên bản đồ.

A screenshot of a application

Description automatically generated

*Hình 8.2 Google Maps Directions API*

* + Ví dụ:

javascript

Copy code

import React, { useState, useEffect } from 'react';

import { View, StyleSheet } from 'react-native';

import MapView, { Marker, Polyline } from 'react-native-maps';

const DirectionsExample = () => {

const [routeCoords, setRouteCoords] = useState([]);

useEffect(() => {

// Gọi Google Maps Directions API và xử lý dữ liệu lộ trình

const fetchDirections = async () => {

try {

const response = await fetch(

`https://maps.googleapis.com/maps/api/directions/json?origin=37.78825,-122.4324&destination=37.7749,-122.4194&key=YOUR\_GOOGLE\_MAPS\_API\_KEY`

);

const data = await response.json();

const points = decode(data.routes[0].overview\_polyline.points);

setRouteCoords(points.map(point => ({ latitude: point[0], longitude: point[1] })));

} catch (error) {

console.error("Error fetching directions:", error);

}

};

fetchDirections();

}, []);

return (

<View style={styles.container}>

<MapView

style={styles.map}

initialRegion={{

latitude: 37.78825,

longitude: -122.4324,

latitudeDelta: 0.0922,

longitudeDelta: 0.0421,

}}

>

<Marker coordinate={{ latitude: 37.78825, longitude: -122.4324 }} title="Điểm xuất phát" />

<Marker coordinate={{ latitude: 37.7749, longitude: -122.4194 }} title="Điểm đến" />

<Polyline coordinates={routeCoords} strokeWidth={5} strokeColor="blue" />

</MapView>

</View>

);

};

export default DirectionsExample;

* + **Giải thích**:
    - fetchDirections() gửi yêu cầu đến Google Maps Directions API và sử dụng phương thức Polyline để vẽ lộ trình giữa điểm xuất phát và điểm đến.
    - decode là hàm giải mã chuỗi polyline từ API để chuyển thành mảng tọa độ.

Với các bước trên, chúng ta đã tích hợp thành công bản đồ và các tính năng tùy chỉnh vào ứng dụng React Native của mình, bao gồm hiển thị vị trí, thêm điểm đánh dấu và tìm đường. Điều này giúp người dùng dễ dàng tương tác và điều hướng, mang lại trải nghiệm sử dụng tốt hơn cho ứng dụng.

# KẾT LUẬN

Các dịch vụ tích hợp bên thứ ba đóng vai trò vô cùng quan trọng trong việc phát triển ứng dụng di động hiện đại. Chúng mang đến cho các nhà phát triển nhiều lợi ích nổi bật, từ việc tiết kiệm thời gian, giảm chi phí phát triển, đến việc nâng cao trải nghiệm người dùng thông qua các tính năng phong phú và chất lượng cao.

## Đánh giá những lợi ích của việc sử dụng các tích hợp bên thứ ba

Việc tích hợp các dịch vụ bên thứ ba như Firebase, Google Sign-In, Facebook Sign-In, YouTube, và React Native Maps mang đến cho ứng dụng nhiều lợi ích thiết thực:

* **Tiết kiệm thời gian phát triển**: Các dịch vụ bên thứ ba cung cấp sẵn các tính năng đã được xây dựng và tối ưu, giúp nhà phát triển không cần phải tái tạo lại từ đầu. Điều này giúp rút ngắn thời gian phát triển ứng dụng đáng kể.
* **Nâng cao trải nghiệm người dùng**: Các dịch vụ này cung cấp nhiều tính năng phong phú và hiện đại, chẳng hạn như xác thực nhanh chóng, bản đồ tương tác, phát video trực tuyến, và thông báo đẩy. Những tính năng này tạo ra trải nghiệm liền mạch, thuận tiện và đáng tin cậy cho người dùng.
* **Dễ dàng mở rộng và bảo trì**: Các dịch vụ bên thứ ba thường xuyên được cập nhật và bảo trì bởi các nhà cung cấp, giúp ứng dụng luôn ở trạng thái tối ưu và dễ dàng mở rộng khi cần.

## Tầm quan trọng của việc lựa chọn và triển khai hợp lý các dịch vụ cho ứng dụng

Mặc dù tích hợp bên thứ ba mang lại nhiều lợi ích, việc lựa chọn và triển khai hợp lý là điều cần thiết để đảm bảo tính ổn định và bảo mật của ứng dụng:

* **Phù hợp với nhu cầu của ứng dụng**: Không phải tất cả các dịch vụ bên thứ ba đều cần thiết cho mọi ứng dụng. Nhà phát triển nên xem xét kỹ lưỡng các yêu cầu của ứng dụng để chọn các dịch vụ phù hợp, tránh việc tích hợp quá nhiều dịch vụ không cần thiết dẫn đến quá tải và phức tạp không cần thiết.
* **Bảo mật và quản lý quyền riêng tư**: Khi tích hợp các dịch vụ bên thứ ba, đặc biệt là các dịch vụ yêu cầu dữ liệu người dùng như xác thực, cần đảm bảo rằng quyền riêng tư của người dùng được bảo vệ theo các tiêu chuẩn bảo mật và tuân thủ các quy định pháp luật về dữ liệu.
* **Khả năng tương thích và hiệu suất**: Các dịch vụ tích hợp cần phải tương thích tốt với nền tảng và framework của ứng dụng (như React Native) để không gây ảnh hưởng đến hiệu suất.

## Đề xuất cho các nhà phát triển khi tích hợp dịch vụ bên thứ ba vào ứng dụng của mình

Khi tích hợp các dịch vụ bên thứ ba vào ứng dụng, các nhà phát triển nên xem xét các đề xuất sau đây để đạt hiệu quả cao nhất:

* **Thử nghiệm kỹ lưỡng trước khi triển khai**: Nên thử nghiệm các tính năng tích hợp trong nhiều tình huống khác nhau để đảm bảo chúng hoạt động mượt mà và đáp ứng đúng yêu cầu.
* **Theo dõi và tối ưu hóa thường xuyên**: Việc tích hợp bên thứ ba không nên là "thiết lập một lần và quên đi". Hãy theo dõi các bản cập nhật từ nhà cung cấp và tối ưu các tính năng để phù hợp với nhu cầu thay đổi của người dùng.
* **Cân nhắc chi phí sử dụng dịch vụ**: Một số dịch vụ bên thứ ba yêu cầu phí sử dụng, đặc biệt khi ứng dụng phát triển với lượng người dùng lớn. Do đó, cần xem xét chi phí và tính toán lợi ích so với ngân sách để đảm bảo hiệu quả kinh tế.
* **Xem xét và cải tiến bảo mật**: Với các dịch vụ yêu cầu quyền truy cập và sử dụng dữ liệu người dùng, nhà phát triển nên áp dụng các tiêu chuẩn bảo mật và mã hóa, đồng thời thường xuyên kiểm tra các vấn đề bảo mật.

Kết luận, việc sử dụng các dịch vụ bên thứ ba trong ứng dụng di động không chỉ giúp phát triển nhanh chóng, mà còn tạo ra trải nghiệm tối ưu cho người dùng. Tuy nhiên, để đảm bảo hiệu quả và an toàn, nhà phát triển cần lựa chọn, triển khai và duy trì hợp lý các dịch vụ này. Điều này sẽ giúp ứng dụng đạt được sự ổn định, bảo mật và dễ dàng mở rộng trong tương lai.

# TÀI LIỆU THAM KHẢO

1. **Firebase và các dịch vụ liên quan**

Firebase Docs: <https://firebase.google.com/docs>

Firebase Authentication: <https://firebase.google.com/docs/auth>

Firebase Cloud Firestore: <https://firebase.google.com/docs/firestore>

Firebase Analytics: <https://firebase.google.com/docs/analytics>

Firebase Cloud Messaging: <https://firebase.google.com/docs/cloud-messaging>

1. **React Native Firebase Integration (Thư viện Firebase chính thức cho React Native)**

React Native Firebase Docs: <https://rnfirebase.io/>

1. **Google Sign-In**

Google Sign-In API: <https://developers.google.com/identity/sign-in>

Tạo OAuth Client ID cho Android: <https://console.cloud.google.com/apis/credentials>

1. **Facebook SDK và Facebook Login**

Facebook for Developers (Facebook Login): <https://developers.facebook.com/docs/facebook-login>

Facebook Developer Console: <https://developers.facebook.com/apps/>

1. **Google Maps API**

Google Maps API Docs: https://developers.google.com/maps/documentation

React Native Maps (thư viện không chính thức): <https://github.com/react-native-maps/react-native-maps>

1. **YouTube API**

YouTube Data API: <https://developers.google.com/youtube/v>3

YouTube IFrame Player API: <https://developers.google.com/youtube/iframe_api_reference>

1. How to Integrate Firebase in React Native Android and iOS App

[How to Integrate Firebase in React Native Android and iOS App](https://aboutreact.com/integrate-firebase-in-android-and-ios-app/)

1. Integration of Google map in React Native using react-native-maps

<https://aboutreact.com/react-native-map-example>

1. Getting SHA1 Fingerprint for Google API Console

<https://aboutreact.com/getting-sha1-fingerprint-for-google-api-console/>

1. Example of Google Sign In in React Native Android and iOS App

<https://aboutreact.com/example-of-google-sign-in-in-react-native/>

1. Getting Key Hash for Facebook Console

<https://aboutreact.com/getting-key-hash-for-facebook-console/>

1. Example of Facebook Sign In integration in React Native

<https://aboutreact.com/react-native-facebook-login/>

1. Example of YouTube Video Integration in React Native

<https://aboutreact.com/youtube-video-integration-in-react-native/>