# Điểm tương đồng giữa Flutter và React Native

Cả React Native và Flutter đều được đánh giá cao và áp dụng rộng rãi các framework phát triển đa nền tảng, cho phép Developer tạo ứng dụng cho cả thiết bị iOS và Android bằng cách sử dụng một codebase duy nhất.

React Native và Flutter còn có cộng đồng người dùng lớn mạnh. Các cộng đồng này cung cấp cho Developer nhiều tài nguyên học tập, thư viện phong phú, các widget đa năng và nhiều plugin. Với sự hỗ trợ cộng đồng mạnh mẽ, các Developer dễ dàng tiếp cận công cụ và kiến ​​thức cần thiết để nổi bật trong hành trình phát triển ứng dụng.

Ngoài việc phát triển ứng dụng đa nền tảng thì một điểm tương đồng khác của 2 framework này là có thể nhúng và hoạt động độc lập khi tích hợp vào các ứng dụng native. Theo trải nghiệm được tiếp xúc và làm việc của cá nhân thì 2 framework này có thể hỗ trợ rất tốt việc phát triển các mini-app tích hợp trên các ứng dụng chủ tạo thành các module con (mô hình đã và đang được ưa chuộng hiện nay).

Bằng cách tận dụng những đặc điểm chung này, React Native và Flutter trao quyền cho các Developer hợp lý hóa quy trình phát triển, nâng cao khả năng tái sử dụng code. Đồng thời cung cấp các ứng dụng giàu tính năng, có tệp người dùng rộng lớn trên nhiều nền tảng.

## Cross-platform app

Cả hai đều được thiết kế để phát triển cross-platform app. React Native và Flutter đều cung cấp hiệu suất ứng dụng vượt trội, nhanh chóng và chất lượng cao trên nhiều platform bao gồm Android, iOS và UWP

## Cộng đồng công nghệ khổng lồ

Cả hai framework được hỗ trợ bởi các cộng đồng công nghệ khổng lồ. React Native được cung cấp bởi Facebook còn Flutter là SDK nguồn mở của Google để phát triển mobile app. Cả hai đều có cộng đồng các developer mạnh mẽ

## Nguồn mở

Nguồn mở, miễn phí và nhanh chóng. React Native và Flutter là các SDK nguồn mở và miễn phí cho phép các developer tạo ra các app tuyệt vời một cách nhanh chóng dựa vào native performance của chúng.

## Chi tiết, luôn được cập nhật

Các cộng đồng của cả hai framework này luôn nỗ lực hết sức để liên tục cập nhập documentation mới nhất với các tài liệu tham khảo API và tài nguyên toàn diện.

## UI và Native Experience

React Native sử dụng các yếu tố xây dựng UI cơ bản trong Android / iOS để mang lại native experience. Flutter cũng sử dụng các widget để cung cấp native experience đáng chú ý trên platform Android và iOS.

## Hot reloading

React Native hỗ trợ “Hot reloading” cho phép đồng thời chạy code mới và giữ trạng thái ứng dụng, thay vì phải recompile. Tính năng này làm cho sự phát triển nhanh hơn, tức thì và hiệu quả.

Flutter cũng không hề kém cạnh. “Stateful Hot Reloading” của nó hỗ trợ phản ánh các thay đổi ngay lập tức mà không cần restart hoặc trong trường hợp mất trạng thái ứng dụng.

Chúng ta đã điểm qua sự giống nhau của React Native và Flutter, bây giờ hãy cùng xem qua những khác biệt của chúng trong process, flow, và feature.

# React Native vs Flutter

|  |  |  |
| --- | --- | --- |
| **Tiêu chí** | **React Native** | **Flutter** |
| **Ngôn ngữ và công cụ** | Không cần học ngôn ngữ mới nếu bạn đã biết JavaScript hoặc TypeScript. Tuy nhiên, việc làm quen với các công cụ và thư viện của React Native có thể phức tạp và tốn thời gian. | Dart là ngôn ngữ mới, nhưng rất dễ học và thân thiện với nhà phát triển, đặc biệt nếu bạn đã quen với TypeScript, C#, hoặc Kotlin. Quá trình thiết lập và sử dụng công cụ của Flutter đơn giản hơn. |
| **Tính năng** | React Native yêu cầu sử dụng nhiều thư viện cộng đồng để bổ sung các tính năng như UI, quản lý trạng thái, và localization. Môi trường và công cụ phức tạp hơn, nhưng cộng đồng hỗ trợ lớn. | Flutter cung cấp rất nhiều tính năng từ đầu, bao gồm Material Design, Cupertino components, và các công cụ phát triển tiên tiến. Có thể phát triển ứng dụng mà không cần đến thư viện bên thứ ba. |
| **Giao diện người dùng** | Sử dụng các thành phần UI native của từng nền tảng, điều này có thể dẫn đến sự không nhất quán giữa các nền tảng khác nhau. Giao diện người dùng tùy biến mạnh mẽ nhờ sử dụng CSS. | Sử dụng engine đồ họa riêng để vẽ UI, giúp giảm thiểu sự khác biệt giữa các nền tảng. Styling khác biệt với CSS nhưng cung cấp nhiều tùy chọn tùy chỉnh và dễ sử dụng sau khi làm quen. |
| **Hiệu suất** | Hiệu suất tốt nhưng có thể bị ảnh hưởng khi xử lý các tác vụ phức tạp do phải thông qua JavaScript Bridge. Không thích hợp cho các ứng dụng yêu cầu đồ họa cao. | Hiệu suất vượt trội hơn nhờ vào engine riêng, cung cấp trải nghiệm người dùng mượt mà và hỗ trợ tốt cho các animation phức tạp. Tuy nhiên, tiêu thụ bộ nhớ có thể cao hơn so với ứng dụng native. |
| **Công cụ phát triển và trải nghiệm** | Expo cung cấp nhiều công cụ phát triển mạnh mẽ và tiện lợi như Expo Go để test app trực tiếp mà không cần sideloading. Cộng đồng và hệ sinh thái phong phú với nhiều thư viện hỗ trợ. | Tích hợp chặt chẽ với VS Code và có công cụ hot reload mạnh mẽ. Tuy không có dịch vụ tương đương Expo, Flutter vẫn cung cấp các công cụ phát triển tốt và dễ sử dụng. |
| **Trải nghiệm phát triển** | Có thể gây khó chịu do cần sử dụng nhiều thư viện bên thứ ba và các vấn đề không đồng bộ giữa các nền tảng. Tuy nhiên, cộng đồng lớn giúp giải quyết vấn đề nhanh chóng. | Trải nghiệm phát triển mượt mà, ít gặp phải các vấn đề phát sinh. Các công cụ và tính năng sẵn có giúp giảm thiểu thời gian và công sức phát triển. |
| **Document** | Tài liệu tổng quát, dựa nhiều vào các bộ dev kit bên ngoài. Tài liệu không trực quan bằng Flutter. | Tài liệu dễ đọc, chi tiết, có cấu trúc rõ ràng và trực quan hơn. |
| **Mức độ phổ biến** | Được sử dụng rộng rãi hơn và phổ biến hơn so với Flutter. Nhiều nhà phát triển sử dụng do quen thuộc với JavaScript. | Flutter là một framework mới hơn và ít được sử dụng hơn, nhưng đang dần thu hút sự quan tâm của cộng đồng. |
| **Cộng đồng và hỗ trợ** | Ra mắt dưới dạng mã nguồn mở trên GitHub vào năm 2015, có cộng đồng lớn và nhiều câu hỏi trên Stack Overflow. | Flutter là framework mới, cộng đồng nhỏ hơn nhưng đang phát triển nhanh chóng. |
| **Cơ hội nghề nghiệp** | Sử dụng phổ biến nên cơ hội việc làm nhiều. | Cơ hội việc làm hiện tại ít hơn, nhưng có thể tăng trong tương lai. |

Bảng trên cung cấp cái nhìn tổng quan về sự khác biệt giữa React Native và Flutter, giúp bạn lựa chọn framework phù hợp dựa trên các tiêu chí như ngôn ngữ lập trình, hiệu suất, cộng đồng hỗ trợ, và cơ hội nghề nghiệp.

Flutter phù hợp với các dự án focus về animation, các layout phức tạp, với thế mạnh sử dụng bộ render tự làm, giao tiếp trực tiếp với GPU và một SDK để viết anim dễ dàng. Cực kỳ phù hợp với các team native (đang có nhu cầu làm thêm các UX có hiệu năng cao vào app native có sẵn). Flutter Developer hiện tại chưa nhiều, sẽ khó tìm người hơn.

React Native phù hợp với các team dùng JS as main language, không có quá nhiều animation phức tạp. RN hiện tại tuyển người khá/rất dễ so với Flutter.

***Điểm khác biệt giữa Flutter vs React Native***

|  |  |  |
| --- | --- | --- |
| **Điểm khác biệt** | **Flutter** | **React Native** |
| **Khái niệm** | Bộ công cụ giao diện người dùng mã nguồn mở để phát triển đa nền tảng từ một codebase duy nhất. | Một framework để phát triển đa nền tảng bằng React. |
| **Thời gian phát hành** | Năm 2018 | Năm 2015 |
| **Được tạo bởi** | Google | Meta |
| **Ngôn ngữ lập trình** | Flutter được viết bằng Dart – ngôn ngữ lập trình được Google giới thiệu vào năm 2011 và hiếm khi được các Developer sử dụng. | React Native hoàn toàn được viết bằng JavaScript. Đây là lợi thế cho React Native vì JavaScript là một trong những ngôn ngữ được sử dụng phổ biến nhất trên thế giới hiện nay. |
| **Nền tảng hoạt động** | * Android * iOS * Windows * macOS * Linux * Web * Embedded System | * Android & iOS * Windows (tất cả các thiết bị hỗ trợ Windows 10 như PC, máy tính bảng Xbox; được Microsoft hỗ trợ) * MacOS (được hỗ trợ bởi Microsoft) * Nền tảng web và nền tảng Android và iOS TV (được hỗ trợ bởi cộng đồng). |
| **Phát triển giao diện người dùng** | Flutter sử dụng các widget có khả năng tùy biến cao, cho phép Developer điều chỉnh thuộc tính và kiểu dáng để đạt được giao diện và hành vi mong muốn trên mọi nền tảng. Flutter cũng đi kèm với một bộ tiện ích có sẵn dựa trên thiết kế Material (Android) và Cupertino (iOS), vì vậy Flutter vẫn giữ được một số thuộc tính của ứng dụng gốc. | React Native sử dụng các tiện ích UI gốc để các Developer có thể tự do giữ nguyên kiểu dáng nền tảng mặc định hoặc điều chỉnh cho phù hợp với giao diện người dùng mong muốn. |
| **Thời gian phát triển** | Flutter vượt trội trong việc tạo ra các ứng dụng có thiết kế đồng nhất trên các nền tảng, tuy nhiên Flutter có thể bị lag nếu thiết kế khác nhau giữa Android và iOS. | React Native có lợi thế hơn do JavaScript cho phép Developer sử dụng bất kỳ IDE hoặc trình soạn thảo văn bản nào. Nếu thiết kế iOS và Android khác nhau, React Native sẽ phát triển nhanh hơn Flutter. |
| **Hiệu suất** | Không có cầu nối trung gian để tương tác với các thành phần gốc của thiết bị. Vì vậy, tương tác với các thành phần gốc nhanh hơn, cải thiện hiệu suất tổng thể của ứng dụng. | React Native sử dụng JavaScript để kết nối với các thành phần gốc thông qua một cầu nối. Do đó, tốc độ phát triển và chạy của ứng dụng chậm hơn Flutter. |
| **Bảo trì** | Dễ dàng cập nhật phiên bản mới nhờ các công cụ tự động tích hợp sẵn. | Việc cập nhật phiên bản có thể khó khăn và cần nhiều thời gian hơn. |
| **Quá trình update version** | * Phát hành bản Beta đầu tiên (2015) * Phiên bản Alpha (5/2017) * Flutter 1.0 – Phiên bản ổn định đầu tiên của framework hỗ trợ iOS và Android (12/2018) * Hỗ trợ web với bản Flutter 1.12 Beta cùng với hỗ trợ alpha macOS (2/2019) * Flutter 2.0 hỗ trợ web và bản preview cho máy tính để bàn (3/2021) * Flutter 2.10 hỗ trợ Windows (2/2022) * Flutter 3.0 hỗ trợ macOS và Linux (5/2022) * Flutter 3.7 cập nhật material 3 cho các Developer (2/2023) | * React Native bắt đầu như một dự án nội bộ của Facebook (2013) * React Native 1 bản preview tại (1/2015) * Hỗ trợ nền tảng Android (9/2015) * Ra mắt chính thức React Native (3/2018) * Giới thiệu Hermes – công cụ JavaScript được tối ưu hóa trên thiết bị di động (2019) * Phiên bản hỗ trợ Android 12 và iOS 15 (10/2021) * Ra đời cấu trúc mới (3/2022) * TypeScript mặc định, cập nhật cấu trúc mới (1/2023) * Version 0.74 giới thiệu chế độ Bridgeless Mode mặc định, các tùy chọn kiểu dáng sáng tạo và API được sắp xếp hợp lý (2024) |
| **Xu hướng** | Theo khảo sát [StackOverflow 2023](https://survey.stackoverflow.co/2023/), Flutter được xếp hạng cao hơn React Native với 9.21% Developer chuyên nghiệp chọn đây là framework yêu thích của họ. Flutter tương đối mới và ít được các công ty sử dụng hơn. Điều này dẫn đến ít cơ hội việc làm hơn. Tuy nhiên, điều này có thể thay đổi trong vài năm nữa.  Khả năng tích hợp linh hoạt vào các ứng dụng viết bằng ngôn ngữ native (iOS, Android). | Cũng theo khảo sát StackOverflow 2023, 9,14% Developer yêu thích React Native (quả là “cuộc chiến” sát sao!). Framework này đã xuất hiện được một thời gian, sử dụng React và JavaScript mang lại cơ hội việc làm tốt hơn. |

***So sánh ưu điểm của Flutter vs React Native***

|  |  |  |
| --- | --- | --- |
| **Nội dung** | **Flutter** | **React Native** |
| **Hiệu suất** | Flutter được xây dựng bằng C/C++ và Dart gốc, phù hợp để tạo các ứng dụng hiệu suất cao. Flutter cũng hỗ trợ nhiều tính năng tăng tốc phần cứng, giúp cải thiện hiệu suất của ứng dụng. | Các ứng dụng React Native có hiệu suất nguyên bản và phản hồi nhanh. React Native sử dụng các thành phần UI gốc để tạo giao diện người dùng, mang lại tốc độ nhanh hơn. Giao diện người dùng này đảm bảo ứng dụng hoạt động trơn tru và nhanh chóng. |
| **Kích thước mã nguồn** | Mã nguồn của các ứng dụng viết bằng Flutter tương đối lớn so với các framework đa nền tảng khác. Việc sử dụng một codebase duy nhất sẽ tiết kiệm không gian và dễ bảo trì. | Ứng dụng viết bằng React Native có mã nguồn thường nhỏ hơn nhiều so với ứng dụng gốc, do React Native sử dụng JavaScript, nhẹ hơn các ngôn ngữ gốc như Java và Swift. |
| **Phiên bản SDK yêu cầu tối thiểu** | Flutter có phiên bản SDK yêu cầu tối thiểu là 3.19.6 và có thể được biên dịch sang Android 4.1 để tương thích với nhiều thiết bị. | React Native thường yêu cầu phiên bản SDK tối thiểu thấp hơn các ứng dụng khác. Do đó, các ứng dụng React Native thường nhẹ hơn và cần ít code hơn. |
| **Phát triển giao diện người dùng** | Flutter dựa trên react programming, đơn giản và hiệu quả hơn các phương pháp truyền thống. Ngoài ra, Flutter còn cung cấp nhiều thành phần và công cụ UI khác nhau, giúp tạo hiệu ứng động trong giao diện người dùng đẹp mắt, tương tác dễ dàng. | React Native cho phép phát triển giao diện người dùng tốt hơn các loại ứng dụng khác. Do React Native sử dụng phong cách lập trình khai báo, giúp việc tạo giao diện người dùng phản hồi đẹp mắt trên các thiết bị khác nhau. |
| **Giải quyết lỗi** | Flutter hỗ trợ tính năng Hot Reload cho phép các Developer nhanh chóng lặp lại code, từ đó  gỡ lỗi dễ dàng hơn nhiều. Ngoài ra, Flutter còn có môi trường giải quyết lỗi tích hợp, giúp xác định và khắc phục các vấn đề về hiệu suất. | Ứng dụng React Native dễ truy cập hơn ứng dụng gốc truyền thống vì mã hiển thị rõ hơn. |
| **Tái sử dụng code trên các nền tảng di động** | Flutter cho phép các Developer code bằng một ngôn ngữ duy nhất (Dart) và sau đó biên dịch sang Android và iOS. Điều này giúp việc tái sử dụng code giữa hai nền tảng dễ tiếp cận hơn, tiết kiệm thời gian và tài nguyên phát triển. | React Native cho phép tái sử dụng code giữa các nền tảng, tiết kiệm thời gian và công sức trong quá trình phát triển. |

***So sánh nhược điểm của Flutter vs React Native***

|  |  |  |
| --- | --- | --- |
| **Nội dung** | **Flutter** | **React Native** |
| **Hiệu suất** | Dù Flutter có lợi thế là code được 2 nền tảng nhưng hiệu suất chắc chắn sẽ không tốt bằng các ứng dụng được viết bằng các ngôn ngữ và tools native do ko được hỗ trợ chuyên sâu và trong phần core của hệ điều hành. | React Native phản hồi nhanh nhưng vẫn không đạt như ứng dụng gốc, do React Native không phải lúc nào cũng sử dụng các thành phần gốc nên không thể sử dụng hết phần cứng của thiết bị. |
| **Kích thước file mã nguồn** | Các ứng dụng Flutter vẫn còn tương đối lớn so với các ứng dụng gốc hoàn toàn. | Mặc dù ứng dụng React Native thường nhỏ hơn nhiều so với ứng dụng gốc nhưng chúng vẫn có thể nổi bật. |
| **Phiên bản SDK yêu cầu tối thiểu** | Phiên bản SDK tối thiểu thấp hơn so với các nền tảng phát triển di động khác, nên ít phù hợp hơn các thiết bị cũ hoặc cấp thấp. | React Native 0.74 có yêu cầu phiên bản SDK Android tối thiểu là 23 (Android 6.0), đồng nghĩa React Native không hoạt động trên các phiên bản Android cũ. |
| **Phát triển giao diện người dùng** | Các thành phần UI của Flutter phải được ghi chép và hỗ trợ hơn các nền tảng phát triển di động khác, khiến việc tạo các thành phần UI này phức tạp hoặc tùy chỉnh trở nên khó khăn hơn. | Việc phát triển giao diện người dùng của React Native phức tạp và tốn thời gian, đòi hỏi kiến ​​thức về React và các thành phần UI gốc. |
| **Giải quyết lỗi** | Môi trường gỡ lỗi của Flutter có thể khó làm quen vì khá khác biệt so với các nền tảng khác. | React Native khó gỡ lỗi hơn các ứng dụng khác. Ngoài ra, React Native có thể chậm hơn ứng dụng gốc và một số tính năng không khả dụng trên tất cả các nền tảng. |
| **Tái sử dụng code trên các nền tảng di động** | Flutter vẫn là công nghệ tương đối mới; ở một số khu vực, việc tái sử dụng code có thể được cải thiện. | Với React Native, các Developer có kinh nghiệm và tài nguyên hỗ trợ đang thiếu khi tái sử dụng code giữa các nền tảng di động. |

Flutter đáp ứng các tiêu chí:

* Ứng dụng không cần bất kỳ chức năng gốc nào.
* Bạn cần tiết kiệm thời gian, ngân sách và nguồn nhân lực.
* Sản phẩm cần sớm được xuất bản và code nhanh chóng.
* Bạn muốn có một codebase duy nhất với sự hỗ trợ đa nền tảng.
* Cần một giao diện người dùng thân thiện và tùy chỉnh với các widget.

React Native đáp ứng các tiêu chí:

* Bạn đang muốn mở rộng quy mô dự án hiện tại bằng các modules đa nền tảng.
* Bạn không bị giới hạn về thời gian và ngân sách.
* Cần phát triển các ứng dụng gốc đơn giản và nhẹ.
* Cần tạo các API được chia sẻ ngay lập tức.
* Muốn tạo một ứng dụng có bản dựng không đồng bộ và giao diện người dùng hấp dẫn.

1. *React Native được Facebook sử dụng để tái cấu trúc công nghệ trên quy mô lớn. Có nhiều cải tiến và tính năng trên nền tảng React Native như:*

* Cung cấp Window dimensions hook;
* Thích hợp cho các frameworks; hỗ trợ CocoaPods;
* Tốc độ refresh nhanh, là sự kết hợp của các tính năng Hot Reload;
* Cung cấp các API ổn định, tạo kho lưu trữ GitHub tốt hơn, tài liệu tốt hơn và các công cụ hỗ trợ trong cộng đồng mã nguồn mở.

1. *Flutter cung cấp nhiều tính năng như:*

* Nâng cao dung lượng bộ nhớ và Máy ảo Dart;
* Tính năng theo dõi được cải thiện;
* Hệ sinh thái tốt hơn;
* Cải thiện lượt xem nền tảng web và hiệu suất nhanh hơn;
* Giảm độ trễ khởi động (50% cho thiết bị cấp thấp và 10% cho thiết bị cao cấp).

# React Native vs Kotlin

Chúng ta hãy cùng xem xét kỹ sự khác biệt chính giữa hai ngôn ngữ này bằng cách xem xét một số khía cạnh quan trọng.

|  | **React Native** | **Kotlin** |
| --- | --- | --- |
| **Platform Support** | React native hỗ trợ phát triển đa nền tảng, cho phép xây dựng ứng dụng Android và iOS với một cơ sở mã duy nhất. | Kotlin được tuyên bố là ngôn ngữ chính thức để phát triển ứng dụng Android. Nó giúp xây dựng các ứng dụng gốc. |
| **Community Strength** | Java Script có số lượng nhà phát triển rất lớn khiến cộng đồng React native trở nên mạnh mẽ. | Kotlin đã được giới thiệu là ngôn ngữ Android chính thức cách đây vài năm nên cộng đồng của nó nhỏ hơn React native nhưng nó đang phát triển rất nhanh do tính phổ biến của nó. |
| **IDE Supports** | Các dự án React native được phát triển bằng nhiều IDE như Atom, Visual Studio Code và các IDE khác. | Android Studio là IDE chính thức để phát triển Android. Nó cung cấp các công cụ, tính năng và Plugin mạnh mẽ giúp các nhà phát triển dễ dàng phát triển. |
| **Performance** | React native cung cấp hiệu suất tốt nhưng vẫn có một số khác biệt về hiệu suất so với native. | Với Kotlin, mã được biên dịch thành mã byte, do đó hiệu suất sẽ tương tự như Java, tốt hơn React Native. |
| **Language** | Phát triển React Native sử dụng Java Script và React. Các nhà phát triển web có kinh nghiệm có thể tận dụng các kỹ năng hiện có của họ nếu họ muốn xây dựng các ứng dụng di động. | Kotlin là ngôn ngữ được gõ tĩnh chạy trên JVM. Kotlin có khả năng tương tác hoàn toàn với Java, do đó các nhà phát triển có thể sử dụng các thư viện Java hiện có. |
| **Learning Curve** | Dễ dàng cho các nhà phát triển có kinh nghiệm phát triển back-end. Ngôn ngữ dễ dàng cho người mới bắt đầu cũng như có cú pháp và thuật ngữ dễ hiểu. | Kotlin đã nhận được phản hồi rất lớn từ các nhà phát triển và trở nên rất phổ biến trong một thời gian rất ngắn. Nó có một cộng đồng nhà phát triển lớn trong một thời gian ngắn như vậy. |

**Yêu cầu hệ thống để phát triển**

**React native**

* Hệ điều hành bạn chọn
* Node.js phải được cài đặt trên hệ thống
* Code Editor như VS Code hoặc Atom
* Android Studio

**Kotlin**

* JDK (Java Development Toolkit) phải được cài đặt trên hệ thống
* IDE như IntelliJ IDEA
* Android Studio

**Use Cases**

**React Native**

* **Hỗ trợ đa nền tảng cho ứng dụng di động:** React native cho phép phát triển đa nền tảng, nghĩa là với một cơ sở mã duy nhất, các nhà phát triển có thể xây dựng các ứng dụng di động Android và iOS. Vì vậy, đối với các yêu cầu như vậy, React Native là một lựa chọn tốt.
* **Phát triển nhanh:**React Native cung cấp hỗ trợ Hot-Reload giúp các nhà phát triển phát triển và lặp lại nhanh chóng, nó cũng phù hợp với các dự án có thời hạn rất chặt chẽ.
* **Chi phí phát triển tiết kiệm:**Với cơ sở mã duy nhất, React Native cho phép xây dựng ứng dụng Android và iOS, giúp tiết kiệm chi phí hơn. Vì vậy, với chi phí của một nền tảng duy nhất, cả hai ứng dụng nền tảng đều có thể được phát triển.
* **Cộng đồng và thư viện:** React Native có một cộng đồng rất lớn, nơi các nhà phát triển đang tích cực làm việc để hỗ trợ các thư viện hiện có. Hệ sinh thái rộng lớn này có số lượng lớn các thành phần.

**Kotlin**

* **Android(native) App Development:** Kotlin được tuyên bố là ngôn ngữ phát triển Android chính thức, vì vậy Kotlin là lựa chọn tốt nhất nếu ứng dụng Android là yêu cầu duy nhất.
* **Dự án hiện tại với ngôn ngữ Java:**Đối với một số dự án đã được phát triển bằng Java và cần hỗ trợ hoặc di chuyển hoặc hỗ trợ công cụ mới nhất thì Kotlin phù hợp cho những trường hợp như vậy.
* **Ứng dụng doanh nghiệp lớn:** Doanh nghiệp lớn như Ngân hàng, Y tế hoặc Thương mại điện tử, nơi độ chính xác, tính ổn định là ưu tiên hàng đầu, trong những trường hợp đó, Kotlin là lựa chọn tốt hơn.
* **Yêu cầu dành riêng cho nền tảng:** Đối với các dự án dành riêng cho nền tảng như BLE, Bluetooth, Ứng dụng dựa trên cảm biến, nền tảng gốc là lựa chọn tốt nhất để có trải nghiệm tốt nhất.

**Phân tích thị trường - Áp dụng của nhà phát triển**

**React Native**

* **Phổ biến:** Đối với phát triển ứng dụng di động, React Native đã trở nên phổ biến đáng kể, đặc biệt là trên thị trường đa nền tảng. Các nhà phát triển nắm vững Java Script đã áp dụng React Native rộng rãi.
* **Được các công ty lớn áp dụng:** Nhiều gã khổng lồ công nghệ, các công ty khởi nghiệp đã áp dụng React Native do hỗ trợ cơ sở mã đơn cùng với tính ổn định của ứng dụng tốt. Nhiều ứng dụng phổ biến trên thị trường như Facebook, Instagram và AirBnB được phát triển bằng React Native.

**Kotlin**

* **Tăng trưởng trong việc áp dụng:** Kotlin đã nhận được phản hồi đáng kể từ các nhà phát triển. Nhiều dự án mới đang bắt đầu bằng Kotlin như ngôn ngữ chính của nó để phát triển Android. Các nhà phát triển đang đánh giá cao tính an toàn null, tính súc tích và hỗ trợ lambda của nó.
* **Hỗ trợ của Google:** Trong sự trỗi dậy của Kotlin, Google đã đóng một vai trò quan trọng. Kotlin được hỗ trợ đầy đủ trong Android studio và Google quảng bá Kotlin cho phát triển Android. Google cũng tổ chức các phiên KeyNote hàng năm để công bố các tính năng mới giúp các nhà phát triển luôn cập nhật.
* **Tính năng hiện đại:** Kotlin đã giới thiệu các tính năng ngôn ngữ hiện đại như Coroutine giúp lập trình không đồng bộ dễ dàng hơn. Các hàm mở rộng, Android KTX đã nhận được phản hồi đặc biệt từ các nhà phát triển ứng dụng Android.

**Kết luận**

Lựa chọn giữa React native và Kotlin phụ thuộc vào Yêu cầu cụ thể, Nhóm phát triển khả dụng và ngân sách khả dụng cho Dự án. Cả hai nền tảng đều có điểm mạnh và lợi thế riêng cần được cân nhắc khi lựa chọn ngôn ngữ để phát triển ứng dụng di động. Đối với ứng dụng đa nền tảng, React native là lựa chọn tốt nhất, trong khi đối với các tính năng dành riêng cho nền tảng, Kotlin là lựa chọn tốt hơn cho các nhà phát triển.