**Nhóm 4:**

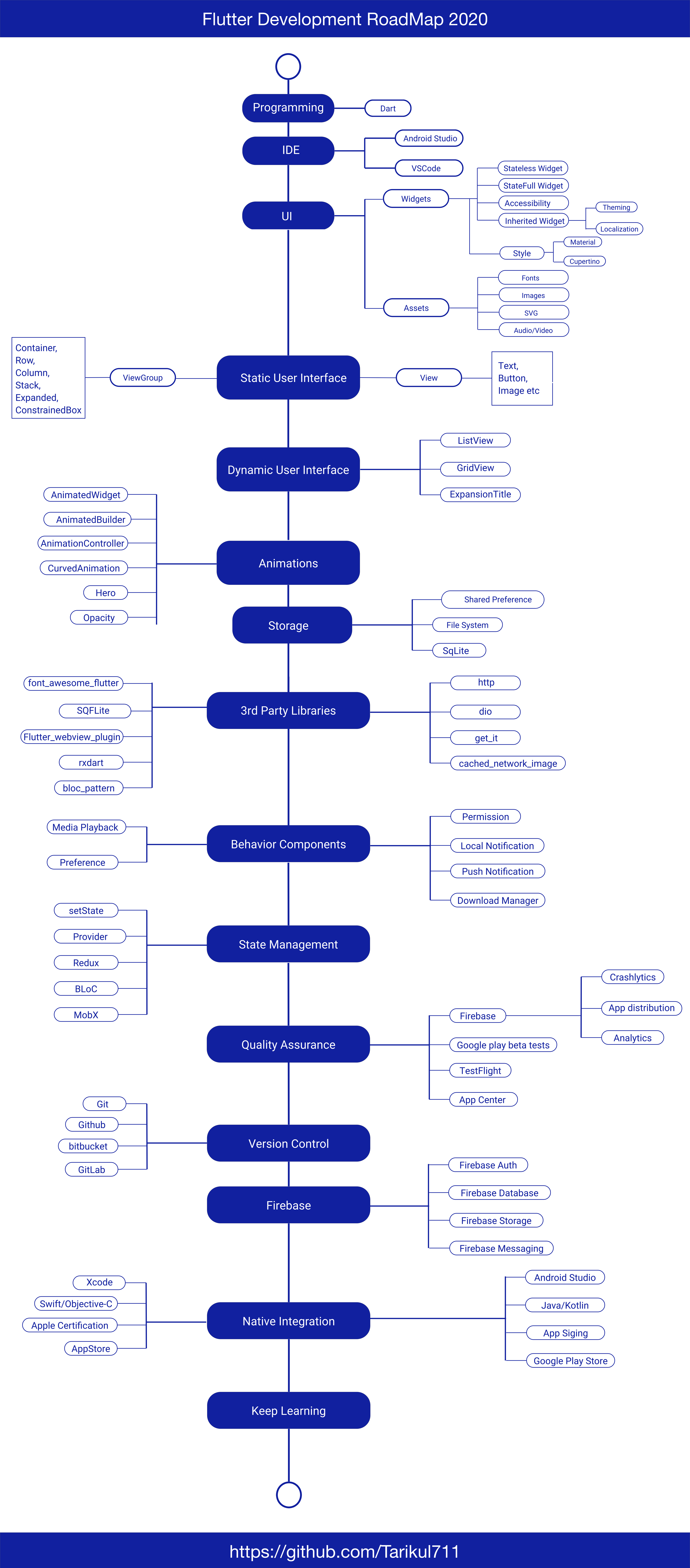
**Họ và Tên:** Đồng Văn Tuấn Đạt

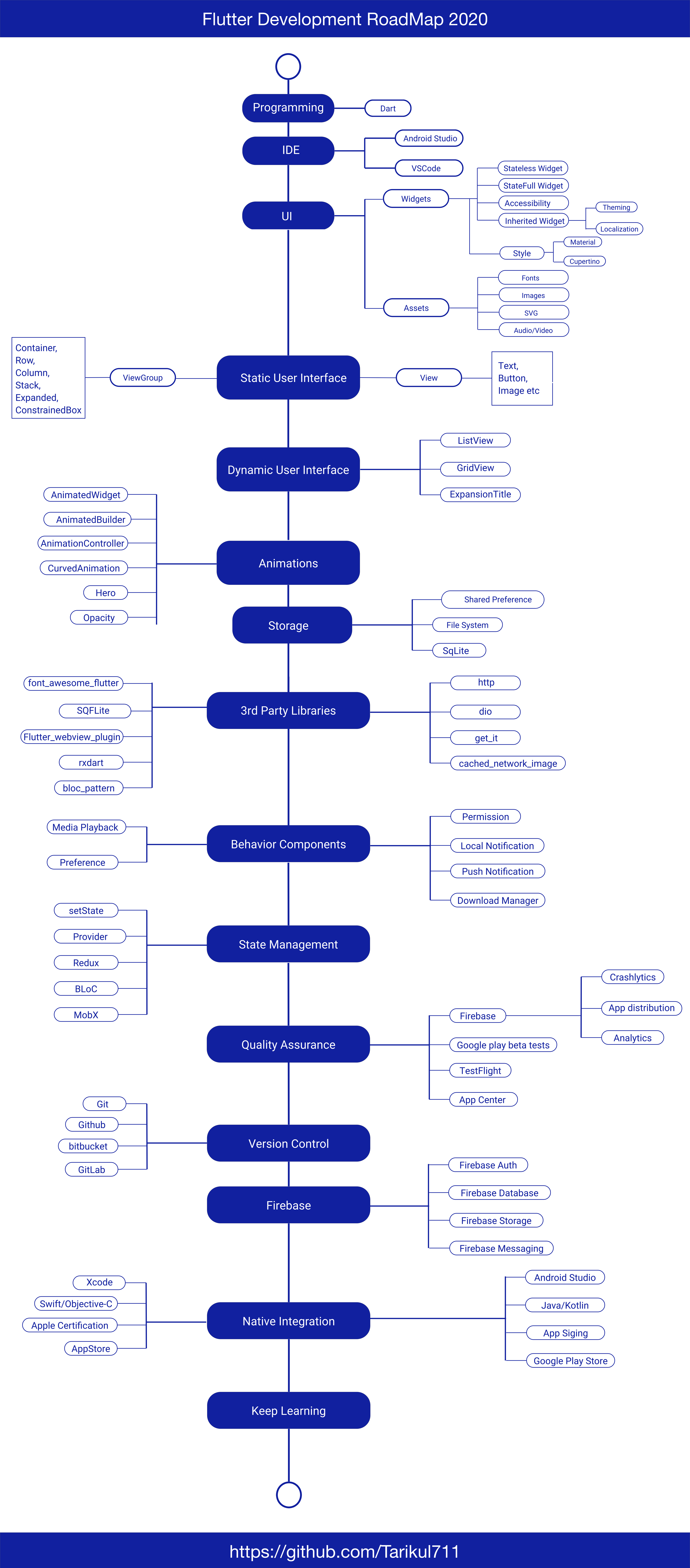
**Lớp**: 08\_DH\_CNPM

**MSSV:** 0850080015

**Đề:** Sinh viên tìm hiểu lí thuyết về

* Storage
* 3rd Party  
  Librabies
* Behavior  
  Components
* Sate  
  Managements
* Quality  
  Assurance
* Version  
  Control
* Firebase
* Native  
  Integration





**Bài làm:**

Trong Roadmap 2020

1. Storage (Lưu trữ):
   * Shared Preferences: Đây là một cách đơn giản để lưu trữ dữ liệu nhỏ dưới dạng key-value pairs. Shared Preferences được sử dụng phổ biến để lưu trữ các cài đặt ứng dụng và dữ liệu người dùng như token xác thực, tùy chọn, đăng nhập/đăng xuất.
   * File Storage: Flutter hỗ trợ lưu trữ và truy cập dữ liệu dưới dạng tệp tin. Điều này cho phép bạn lưu trữ dữ liệu không phải trong bộ nhớ tạm thời của ứng dụng và cho phép bạn lưu trữ dữ liệu lâu dài.
   * SQLite: SQLite là một cơ sở dữ liệu nhúng sẵn trong Flutter, cho phép bạn tạo và quản lý cơ sở dữ liệu cục bộ trong ứng dụng. Nó được sử dụng phổ biến để lưu trữ dữ liệu phức tạp và quản lý cơ sở dữ liệu trong ứng dụng di động.
2. 3rd Party Libraries (Thư viện bên thứ 3):
   * HTTP và Dio: Đều là các thư viện giúp bạn thực hiện các yêu cầu HTTP trong ứng dụng Flutter để giao tiếp với các máy chủ và API bên ngoài. Dio cung cấp nhiều tính năng mạnh mẽ hơn so với HTTP và hỗ trợ quản lý Interceptors, xử lý lỗi, tải xuống tệp, ...
   * get\_it: Thư viện này hỗ trợ quản lý dependency injection trong Flutter, giúp bạn dễ dàng khai báo và sử dụng các dependencies trong ứng dụng.
   * cached\_network\_image: Được sử dụng để tải hình ảnh từ internet và lưu trữ vào bộ nhớ cache để tăng tốc độ hiển thị hình ảnh khi tải lại.
   * Flutter\_webview\_plugin: Cho phép hiển thị trang web bên trong ứng dụng Flutter một cách đơn giản.
   * font\_awesome\_flutter: Thư viện cung cấp hơn 5000 biểu tượng từ thư viện Font Awesome, giúp bạn dễ dàng sử dụng các biểu tượng đẹp trong ứng dụng.
   * SQFLite: Thư viện này là một gói hỗ trợ sử dụng SQLite trong ứng dụng Flutter, giúp bạn dễ dàng thao tác với cơ sở dữ liệu.
   * rxdart: Thư viện này hỗ trợ lập trình reactivie (Rx) trong Dart, giúp bạn quản lý và xử lý luồng dữ liệu dễ dàng hơn.
   * bloc\_pattern: Là một mô hình quản lý trạng thái dựa trên BLoC pattern trong Flutter, giúp bạn tổ chức mã và quản lý trạng thái ứng dụng một cách rõ ràng và dễ dàng hơn.
3. State Management (Quản lý trạng thái):
   * setState: Là phương pháp cơ bản để quản lý trạng thái trong Flutter. Khi trạng thái thay đổi, hàm setState được gọi để cập nhật lại giao diện người dùng.
   * Provider: Đây là một thư viện cung cấp mô hình quản lý trạng thái đơn giản và hiệu quả. Nó sử dụng InheritedWidget để cung cấp dữ liệu và thực hiện giao tiếp giữa các thành phần trong cây widget.
   * Redux: Là một kiến trúc quản lý trạng thái phổ biến trong ứng dụng Flutter, dựa trên nguyên tắc giữ trạng thái ở một nơi duy nhất (Single Source of Truth) và sử dụng hàm reducer để thay đổi trạng thái.
   * BLoC: Là một mô hình quản lý trạng thái và luồng dữ liệu dựa trên BLoC pattern. Nó sử dụng Stream để quản lý và cập nhật trạng thái ứng dụng.
   * MobX: Thư viện này hỗ trợ quản lý trạng thái reactivity dễ dàng hơn. Nó sử dụng các hàm decorator để gắn liền trạng thái với các widget và tự động cập nhật khi trạng thái thay đổi.
4. Behavior Components (Các thành phần hành vi):
   * Permission: Thư viện hỗ trợ xin cấp quyền từ người dùng. Nó cho phép bạn yêu cầu và kiểm tra các quyền truy cập như vị trí, máy ảnh, microphone, ...
   * Local Notification: Để hiển thị thông báo trong ứng dụng Flutter, cho phép bạn lên lịch trình và quản lý thông báo local trong ứng dụng.
   * Push Notification: Cho phép ứng dụng nhận thông báo từ server khi có sự kiện hoặc cập nhật mới.
   * Download Manager: Hỗ trợ quản lý tải xuống các tệp và dữ liệu, cho phép bạn tải xuống dữ liệu và lưu vào bộ nhớ thiết bị.
   * Media Playback: Cho phép phát các phương tiện như âm thanh, video, giúp bạn xây dựng ứng dụng chơi nhạc hoặc xem video.
   * Preference: Thư viện hỗ trợ quản lý các cài đặt ứng dụng, giúp bạn lưu trữ các giá trị cần thiết như token, cài đặt, ...
   * Sharing: Hỗ trợ chia sẻ nội dung từ ứng dụng Flutter sang các ứng dụng khác, giúp bạn dễ dàng chia sẻ liên kết, hình ảnh hoặc văn bản.

5. Quality Assurance (Đảm bảo chất lượng):

* Quality Assurance (QA): Là quá trình kiểm tra, đảm bảo chất lượng phần mềm, bao gồm kiểm thử, đánh giá và xác minh tính năng, hiệu suất, bảo mật và sự ổn định của ứng dụng.
* Firebase: Firebase là một nền tảng phát triển ứng dụng di động và web của Google, cung cấp nhiều dịch vụ như cơ sở dữ liệu realtime, xác thực, lưu trữ tệp, phân tích, messaging, và nhiều tính năng khác giúp phát triển ứng dụng dễ dàng và hiệu quả.
* Crashlytics: Một dịch vụ của Firebase giúp theo dõi và báo cáo lỗi và sự cố trong ứng dụng của bạn.
* App Distribution: Dịch vụ cho phép bạn phân phối ứng dụng trực tiếp đến người dùng hoặc nhóm người dùng cụ thể để kiểm tra và thu thập phản hồi trước khi phát hành chính thức.
* Analytics: Dịch vụ giúp bạn theo dõi và phân tích các dữ liệu về người dùng và sử dụng ứng dụng để đưa ra quyết định phát triển và cải tiến hiệu suất ứng dụng.

6. Version Control (Quản lý phiên bản):

* Git: Là một hệ thống quản lý phiên bản phân tán (distributed version control system) được sử dụng rộng rãi trong việc quản lý mã nguồn của các dự án phần mềm.
* Github, Bitbucket, Gitlab: Đây là các dịch vụ lưu trữ mã nguồn trực tuyến dựa trên Git, cho phép bạn lưu trữ và quản lý mã nguồn của dự án và làm việc cộng tác với các thành viên khác trong dự án.

7. Native Integration (Tích hợp nền tảng di động):

* Android: Hệ điều hành di động của Google được sử dụng rộng rãi trên các thiết bị di động Android.
  + Android Studio: Là môi trường phát triển tích hợp (IDE) được Google hỗ trợ cho việc phát triển ứng dụng Android.
  + Java và Kotlin: Là các ngôn ngữ lập trình phổ biến được sử dụng để phát triển ứng dụng Android.
  + App Signing: Là quá trình ký và chứng nhận ứng dụng để phát hành lên các cửa hàng ứng dụng (Google Play Store, App Store).
  + Google Play Store: Là cửa hàng ứng dụng dành cho hệ điều hành Android, nơi người dùng có thể tải và cài đặt ứng dụng.
  + In-App Purchase: Cho phép bạn tích hợp việc mua hàng và thanh toán bên trong ứng dụng.
* iOS: Hệ điều hành di động của Apple, chạy trên các thiết bị iPhone và iPad.
* Xcode: Là môi trường phát triển tích hợp (IDE) của Apple dành cho việc phát triển ứng dụng iOS.
* Swift và Objective-C: Là các ngôn ngữ lập trình được sử dụng để phát triển ứng dụng iOS.
* Apple Certification: Quá trình chứng nhận ứng dụng của Apple trước khi đưa lên App Store để đảm bảo ứng dụng tuân thủ các yêu cầu của Apple.
* App Store: Là cửa hàng ứng dụng dành cho hệ điều hành iOS, nơi người dùng có thể tải và cài đặt ứng dụng.