No	Consider	Notes	Remarks React Native	Remarks Flutter	React Native	Flutter
1	Performance		**React Native** For default react native can achieve smooth performance 60fps+ but in some case developers needs to pay attention to rendering method to prevent excessive memory and calculations **(Pure.component / useCallback / useMemo)**	**Flutter** Generally, Flutter applications are performant by default, so you only need to avoid common pitfalls to get excellent performance		
2	Cross Platform		**React Native** Mobile (Android, IOS), Desktop (https://microsoft.github.io/react-native-windows/), Web	**Flutter** Mobile (Android, IOS), Web, Linux Windows, MacOs https://flutter.dev/multi-platform		
3	Language	learning curve ?	**React Native** Use popular javascript language thus it has an extensive base package, if you're react developer you can easily learn to react native, here developers have freedom of choice because it facilitates code reuse and cost-saving	**Flutter** Using Dart. Most of the Dart features are similar to that of static and dynamic languages, and it makes it easy for developers to learn and understand Dart easily and quickly. **even flutter provides documentation for developers of other programming languages** **https://docs.flutter.dev/get-started/flutter-for/react-native-devs**		
4	Customable component		**React Native** Fully Customable	**Flutter** Fully Customable		
5	Material UI	what the theme using Material UI?	**React Native** Has many package that utilities UI based on Material UI Component (https: //www.react-native-material.com/)	**Flutter** We can enable it from pubspec.yaml file if we want to use Material Design in flutter (By default value is true) https://docs.flutter.dev/development/ui/widgets/material		
6	Smooth scrolling & transition		**React Native** Smooth scrolling because rendered component is native and have same mechanism like native android or ios **(Flatlist: based on viewport) (Flashlist: based on recyclerview)**, transition smooth as long developer pay attention to performance best practice	**Flutter** The standard ListView constructor works well for small lists. To work with lists that contain a large number of items, it's best to use the ListView.builder constructor.		
7	Structure Boilerplate (development pattern)		**React Native** Can use many development pattern (Atomic Design, MVC, OOP etc)	**Flutter** We can implement any architectures like (The Clean Architecture, MVVM, MVC, Repository Pattern, Bloc Pattern, etc)		
8	Rich Features	future will go for healthy apps	**React Native** Possible for healty apps, in general healty apps use internal sensor from device and RN has package to access that (Accelerometer, Barometer, Gyroscope, Magnetometer, MagnetometerUncalibrated, Pedometer, GPS) **Hearth BPM Without Integrated Sensors** (Flashlight & Camera) **Oxygen Meter Without Integrated Sensors** (Flashlight & Camera) **Local Notification and Scheduling Notification** (https://docs.expo.dev/versions/latest/sdk/notifications/) (https://github.com/wix/react-native-notifications)	Need to riset about healthy apps		
9	Age of Technology		**React Native** Update SDK Easy With Expo-CLI just one command	**Flutter** Sometimes for major update it will impact to our latest code, but we can use FVM to helps us with the need for consistent app builds by referencing the Flutter SDK version used on a per-project basis **https://fvm.app/**		
10	Rich Library Support	npm should be consider about severity packagase, flutter OK	**React Native** Can use npm package like React Query, Moment.js etc	We can use https://pub.dev/ it is The official package repository for Dart and Flutter apps.		
11	Update packages inside apps		**React Native** Can Update Over The Air With Codepuh or with Expo-Update, or downloading assets remotely with Expo-Assets	Currently futter doesn't have feature for upate in over the air like codepush But we can handle it with playstore in app update for android		
12	Complexity code	extra to enhance performance for react native	**React Native** Same as React, and for performance react native need pay attention to rendering method or how react rendering works	Flutter is powered by Dart, a language optimized for fast apps on any platform		
13	Building Apps	EXPO reactnative, flutter with VM	**React Native** Can build from internal device (Xcode, Android Studio), VM or Cloud (Expo EAS)	There are many options for CI/CD that we can implement to build our flutter apps https://docs.flutter.dev/deployment/cd		
14	Stable Version	Flutter to much update	**React Native** Stable, if there's any update it's major and opt-in-out and not breaks many change	No Comment, I hope next major update in the flutter 4.0 will not much breaks our code Because I have a bad experience when upgrading code to null safety		
15	Client use		**React Native** Facebook, Discord, Microsoft Word, etc (https://reactnative.dev/showcase)	Ebay, Google Pay, Google Ads, Alibaba, etc https://flutter.dev/showcase		
16	Reusable Component with FE		**React Native** for logic we just need to copy paste and done, but for UI there's some extra effort to change the web component to native react native component	Absolutely we cannot use our code running in the our React code		
17	Unit Testing		**React Native** React native can use jest for testing component (https://docs.expo.dev/guides/testing-with-jest/), and use Detox for E2E testing (https://docs.expo.dev/build-reference/e2e-tests/#4-write-e2e-tests)	Flutter has 3 types of test 1. Unit Test (tests a single function, method, or class.) 2. Widget Test (in other UI frameworks referred to as component test) tests a single widget. 3. Integration test tests a complete app or a large part of an app.		
18	File size application		**React Native** React Native With Expo default for android around 15mb and for ios 15mb too in production build, and can be compressed again with some config in native code	If we compare to native's one the flutter app size is much bigger https://medium.com/android-news/comparing-apk-sizes-a0eb37bb36f		
19	Permission Access		**React Native** Easy permission access especially with expo maintained SDK, each feature that need have permissions have easy method to request permission for example ImagePicker (https://docs.expo.dev/versions/latest/sdk/imagepicker/) you just need invoke ImagePicker.requestMediaLibraryPermissionsAsync() method and it will return value granted or not or if you dont want expo there's package that handle it independently (react-native-permissions)	**Flutter** we can use this (https://pub.dev/packages/permission_handler) package for handling permission in flutter		