

|  |  |
| --- | --- |
| NAME | SABBAHAT AZIZ |
| REG NO | SP20-BSE-034 |
| ASSIGNMNT NO | 01:THEORY |
| SUBMITTED TO | SIR KAMRAN |
| DATE OF SUBMISSION | 2 0CT 2022 |

# A COMPARISON OF CROSS AND NATIVE PLATFORM:

## CROSS PLATFORM:

Cross platform is also known as multiplatform or platform independent.A cross-platform computer product or system is a product or system that can work across multiple types of platforms or operating environments. Different kinds of cross-platform systems include both hardware and software systems, as well as systems that involve separate builds for each platform, as well as other broader systems that are designed to work the same way across multiple platforms.

* The cross-platform or multiplatform software is a type of application / program / software that works on various operating systems or devices, which are often called platforms. A platform means an operating system such as Windows, Mac OS, Android, or iOS.
* developers can use the same code base that connects to native components via the so-called bridges. Cross-platform is achieved by compiling source code for execution on each platform. Each separate compilation will result in a separate executable file.
* Cross-platform apps are less flexible, they are harder to maintain, and they have a lower user experience because this kind of app doesn’t take into account the uniqueness of each platform.

## NATIVE PLATFORM:

A native application is a software program that is developed for use on a particular platform or device. Because a native app is built for use on a particular device and its OS, it has the ability to use device-specific hardware and software.

* Native apps are developed for specific mobile operating systems such as iOS or Android. Platform-specific language is used to build these apps. For example, (native) Android app uses Java, while iOS apps use Objective – C or Swift.
* The term native app development refers to building a mobile app exclusively for a single platform.
* The app is built with programming languages and tools that are specific to a single platform. For example, you can develop a native Android app with Java or Kotlin and choose Swift and Objective-C for iOS apps

# SCENERIOS FOR NATIVE PLATFORM:

* sometimes there’s a need to implement platform-specific features that work in a certain way on one platform and differently on another, like payments or working in background with photos and movies uploads etc. In this case, even if you choose cross-platform development, the team still needs to write separate codes for each platform.
* Creating libraries . if there’s a need to create a new library, native development is the only option.
* Keeping up to date it’s much easier to be up to date with all platform-specific changes in a native app and to maintain it in an effective way.
* Making a high-performance and intuitive app it's good when the user is not surprised by how things look and work. Features introduced in native apps look very similar across the entire system which makes users familiar with them from the moment they’ve downloaded the app, e.g. device’s notification system, buttons placing.

# SCENERIOS FOR CROSS PLATFORM:

* Compared with the obvious benefits offered by a native application, web project encapsulating solutions or cross-platform applications considerably shorten development and deployment times at app stores. There are therefore fewer costs and the risk of opting for projects that will fail is lower.
* Moreover, we should not forget that cross-platform applications not only cut costs, but they also maximize the profit, either through exposure to a larger number of users (the company will have an application in most environments) or on the revenue side.
* It is not necessary to incorporate new profiles to the team other than those that can already be serving a desktop version: having a team knowledgeable in HTML, CSS and JavaScript would be enough.

# LIST OF FRAMEWORKS FOR CROSS PLATFORM:

* Flutter
* Ionic
* React Native
* Xamarin
* NativeScript
* Node.Js
* PhoneGap
* Appcelerator
* Corona SDK
* Sencha Touch

## ICONIC:

* It is an open-source front-end framework meaning it allows alterations in the code structure, suitable to each developer and saves a lot of time.
* Ionic is based on a SAAS UI framework designed specifically for mobile operating systems. It provides numerous UI components for developing robust applications.
* This framework uses Cordova plugins which allow access to devices’ in-built features including Camera, GPS, and Audio Recorder, posing as one of the major benefits of cross-platform development tools.
* The fact that Ionic gives a native-like feel to the apps is what makes it a favorite of developers. It helps develop cross-platform apps and allows them to perform perfectly on various platforms.

## REACT NATIVE

* It is a good thing that React Native is an open-source cross-platform app framework, the same as Ionic. Because of this, it has a large community to support it and improve it by fixing bugs, improvising and introducing features.
* One of the advantages of cross-platform development is that it requires one-time coding for developing apps for platforms as diverse as Android and iOS. This solves one of the biggest challenges of  other frameworks – requiring developers to code separately twice for the same app on different platforms.
* One-time coding instantly reduces the development time of the app along with keeping the [React Native app development cost](https://appinventiv.com/blog/react-native-app-development-cost/) to its lowest.
* Impressively, React Native is highly compatible with third-party plugins, such as Google Maps.
* React Native focuses on UI to a great extent rendering a highly responsive interface. What it means is that the React Native environment eliminates the time taken in loading and delivers a smooth interface to the applications.

## FLUTTER

* Flutter promotes portable GPU, which renders UI power, allowing it to work on the latest interfaces.
* Flutter does not require updating the UI contents manually, as it possesses a reactive framework. [Flutter app Developers](https://appinventiv.com/flutter-app-development/) are only required to update the variables and the UI changes will be visible after that.
* Flutter cross-platform app framework poses as a perfect choice for developing [Minimum Viable Product](https://appinventiv.com/blog/mvp-first-step-towards-ensuring-success-app/) ([MVP](https://appinventiv.com/blog/the-digestible-guide-to-minimum-viable-products-mvp/)) as it initiates an expeditious development process and is also cost-efficient.
* The developers can efficiently remake a widget tree automatically and comprehend the code adjustments.
* Flutter has an inbuilt graphic engine. Along these lines, the developers would not have the headache of making separate interfaces for Android and iOS.

## XAMARIN

* Apps developed on the Xamarin framework are built using C# – a modern cross platform app development language having leverage over Objective-C and Java.
* Xamarin supports a direct inclusion of Objective-C, Java and C++ libraries. This allows developers to reuse many third-party codebases encrypted in Java, Objective-C or C++, making it one of the considerably best cross-platform frameworks.
* Xamarin reduces the time and [cost of mobile app development](https://appinventiv.com/guide/mobile-app-development-cost/) for it supports the WORA and has a humongous collection of class libraries.
* Xamarin offers robust compile-time checking. Because of this facility, developers witness fewer run-time errors and get well-functioning apps.
* Xamarin has an astounding native user interface and controls assisting and allowing developers in designing a native-like app.

## NATIVESCRIPT

* NativeScript renders beautiful, accessible, and platform-native UI, and that too without the WebViews. Developers are only required to define once and let the NativeScript adapt to run everywhere. They can even customize the UI to specific devices and screens.
* As opposed to React Native, NativeScript provides developers with a complete web resource that comes loaded with plugins for all kinds of solutions. This inevitably eliminates the need for third-party solutions.
* NativeScript gives the freedom to easily access native Android and iOS APIs, meaning developers don’t need any additional knowledge of native development languages.
* It uses Angular and TypeScript for programming purposes.
* NativeScript supports segments like Cocoapods and AndroidArsenal and calls local strategies from libraries.

## NODE.JS:

* All of the Node.js APIs are asynchronous, signifying that they are non-blocking in nature, meaning servers based on Node.JS do not essentially wait for data from APIs. It immediately moves on to another API after calling it. A notification mechanism for Node.js allows the server to get a response from the previous API call.
* Node.js library is impressively speedy in its code execution process, for it is built on the Chrome’s V8 engine.
* Node.js cross-platform apps do not buffer, instead, the applications output the data in chunks.
* To deliver smooth and perfectly functioning applications, Node.js uses a single-threaded model with event looping functionality. This event mechanism enables the server to reply in a non-blocking way, making them scalable.
* Node.JS applications lessen reaction time for slow requests and every developer can execute all the information inquiries at the same time.

## APPCELERATOR TITANIUM:

* Appcelerator offers various tools for rapid application development. This indicates that a prototype can be created with much less time and effort to evaluate user interaction with UI.
* It has ArrowDB-  a schema-less data store that allows developers to deploy data models with no additional efforts for setup.
* It enables the integration of existing continuous delivery systems such as SCM solutions and more.
* Appcelerator possesses pre-built connectors available for MS Azure, MS SQL, Salesforce, Box- the list is tediously long.

## PHONEGAP:

* PhoneGap is considered an impeccable cross-platform framework as it enables developers to create cross-platform apps using existing web technologies such as HTML 5, CSS3 and JavaScript.
* Being a cross-platform framework, PhoneGap supports the use of a single code base to create apps for different platforms namely iOS, Android, Windows Phone, BlackBerry, etc.
* It follows an architecture that is plugin-able in nature, meaning it is possible that the access to native device APIs can be extended in a modular way.

## SENCHA TOUCH:

* It is famous for providing built-in native-looking themes for all of the major platforms like Android, iOS, BlackBerry, Windows Phone, etc.
* It comes with an effective agnostic backend data package for working with data sources.
* One of the most celebrated features of Sencha Touch is that it supports Cordova integration for the native API access along with the packaging.
* It offers code compatibility between new and old ones.
* It comes loaded with customizable and 50+ built-in UI widgets. It also has a collection of rich UI like lists, carousels, forms, menus, and toolbars, etc., created specifically for mobile platforms.

## CORONA SDK:

* It has over 1000 APIs that give developers the ability to sprite animations, audio, and music, Box2D physics, object tweening, texture management, native elements, data- the list can go on for pages.
* It responds to the code changes almost instantly while giving a real-time preview of the app’s performance as it would on a real device.
* It supports almost 200 plugins including in-app advertising, analytics, media, and hardware features.
* As mentioned before, Corona SDK depends on Lua programming language which makes the framework quick and powerful.

THANKS……