A Comparative Analysis of Native and Hybrid Mobile Application Development

Rohan Anand Choudhary[1]
Department of CSE
Amity School of Engineering and Technology
Amity University Uttar Pradesh
Noida, India
anandc.rohan@gmail.com

Tushar Mudi[2]
Department of CSE
Amity School of Engineering and Technology
Amity University Uttar Pradesh
Noida, India
tushar03122@outlook.com

Shilpi Sharma[3]
Department of CSE
Amity School of Engineering and Technology
Amity University Uttar Pradesh
Noida, India
ssharma22@amity.edu

***Abstract*—In the era of 21st century where there is exponential growth in the mobile application development, new technologies are emerging every day has made many companies switch to newer technologies. The two main technologies in the smartphone world currently are Native mobile applications and Hybrid mobile applications. There is numerous mobile operating system (Android, iOS, Windows, BlackBerry OS) in the market and it is difficult for developers to target their applications specifically for each operating system. Both Native and Hybrid have their prevalence and detriment. This paper presents a comparative analysis between Native and Hybrid Application Development. Our analysis primarily focuses on performance, features, user experience native and hybrid mobile application development to provide scientific evidence for researchers and companies to choose vital development mechanism about native and hybrid applications.**

Keywords—Native; Hybrid; Mobile Application; Android; iOS; Windows

1. INTRODUCTION

The prodigious growth and immense popularity of smartphones on the global level continues to expand each and every year. Having exceeded 5 billion mobile device subscribers in 2017 with adding 9 million more subscribers in the next eight years which will get a unique number of 5.9 billion subscribers by the year 2025. With the rising number of users every year there is a need for new technologies to cope up with user expectations. Native mobile applications are built using specific programming languages for specific platforms or devices. The Native Android applications are built on Java or Kotlin specifically and compiled using Android Studio, Native iOS applications are built on Objective C or Swift and compiled using XCode whereas for Native Windows applications are built using C# and compiled using Visual Studio. However, Swift written iOS application would not be compiled in Android Studio similarly Java written Android application would not be compiled on XCode assuming you want to develop your mobile application on both platforms specifically. Unlike web pages or website native mobile applications does not operate on a native browser, for operating of native application they needed to be

downloaded for an individual app store for individual platforms of the specific operating system. Native applications are designed for specific OS typically so it has some added advantages when compared to hybrid applications. Some typical features of native applications are for instance better performance than hybrid applications because of native application interacts directly with native APIs (Application Program Interfaces) without depending on the third party's plugins and web views. The complications with native applications how so ever is that the same mobile applications need to be redesigned and redeveloped by the developer for different operating system separately which leads to the increase in cost, time and resource allocation of the applications with the same features and functionalities which will run in Android or the iOS. Hybrid application development as emerged out as the beginning of the new era in developing mobile application technologies. The name Hybrid associatively suggests that its combination of both web and native solutions [1]. The core hybrid mobile applications are mainly built using web technologies such as CSS, HTML5, and JavaScript, which uses third parties' plugins which makes hybrid applications to have access to the default mobile features such as using geolocation, camera, etc. [2]. The purpose of this paper is to highlight the area of the core native and hybrid mobile applications development cycle and suitable approaches for developing efficient mobile applications on the basis of performance, user experience, resource consumption.

2. CONCEPTUAL COMPARATIVE ANALYSIS BETWEEN NATIVE AND HYBRID APPLICATIONS

A. Native Applications Development

Native mobile applications are developed using native programming languages like Java, Kotlin for platforms like Android, and Swift for iOS [2]. They are built for particular platforms or devices and for the same reason native applications are able to use default features of the operating system as well as some hardware features, which involves the latest technologies such as a camera which are dual, triple, quad lenses these days, magnetic compass which helps for navigation sort of applications, geolocation for applications like google maps and uber, fingerprint sensor for security purposes, contacts, accelerometer etc. [17]. Each native mobile application requires a systematic and specific development approach considering the platform or operating system it is being developed for.

Certain characteristics that native mobile application provides to the mobile user:

- Excellent overall experience. Certain features like 3D touch, crystal clear animation, multi-touch, split-screen etc.
- Frequent updates of new features of the application.
- Different devices require certainly different features for the application like new technologies like the foldable smartphone requires certain different features which are not compatible with the present generation smartphones. So, in addition of different codebase for different platforms now developer also needs to do code different styling for foldable that are not compatible with default touch screen smartphones [7].

B. Hybrid Application Development

Hybrid Mobile application development is a combination of features of web and native apps. Unlike native applications, Hybrid applications are not developed for a single platform but the single code base can be used to deploy the application on different platforms like Android, iOS, Windows etc respectively [8]. Hybrid apps look similar to native apps but its core functioning and optimization totally vary from native apps. The cost, time and resource consumptions of hybrid application is very low as compared to native application hence a company always consider the hybrid application as a suitable option if they are ready to lower the application's performance, and user experience matters less to them. Hybrid application cannot access the device default features such as camera, contacts, accelerometer, GPS etc directly but it requires the third-party plugs, APIs and frameworks like Xamarin, flutter, React Native, etc. Hybrid Applications are portable, cost- efficient but performance and user experience wise it still lacks behind native application development [9].

C. General Comparison between Native and Hybrid Applications

TABLE I. AN OVERVIEW COMPARISON OF HOW NATIVE MOBILE APPS AND HYBRID MOBILE APPS ARE DIFFERNET ON
BASIS OF SOME CHARACTERISTICS

Attributes	Native Apps	Hybrid Apps
Performance	Maximum efficiency, optimization, and reliable have default access to hardware	Depends on mobile browsers, has a finite access to hardware
User Experience	Smooth experience with native support	Limited integration requires third party frameworks and plugins
Functionality	Default functionality of native operating is there	Only a few functionalities can be used which are very less.
Development Time	Needs additional time for designing application for each operating system platforms separately, different code base separately for each platform	It requires less time as compared to native application development as it can one code base for each operating system.
Cost Consumption	High consumption and expensive to develop native apps as they require additional developers, separate codebase, etc.	Low-cost consumption as it requires a single code base and no additional developers for different platforms.

When a company or organization takes a decision to enter the mobile application development world or thinks of building an application for its existing business or product in the market there are different scenarios and cases they consider initially. Like, which operating system platform should they target first iOS or Android, which development approach should they consider Native or Hybrid Approach, performance, cost, time, resource consumption, etc. [3].

3. COMPARISON BETWEEN NATIVE AND HYBRID APPLICATIONS: LITERATURE SURVEY

A. Performance Analyse of Hybrid and Native Applications

Even the most experienced developers believed that the performance of native mobile applications is higher and more reliable when compared with hybrid mobile applications. It is well defined that the level of abstraction for hybrid application development will be more than Native Applications as in order to be cross-platform the framework and libraries on each platform have to undergo such an abstraction that will allow the hybrid application to be consistent through them. For example, Xamarin uses C# and native libraries wrapped in the .Net layer for cross-platform app development [17]. For very basic apps such difference in abstraction has a negligible effect on performance for hybrid apps with Native applications as benchmarks. But as the complexity of applications increases, the performance of hybrid apps is noticeably less than native apps. With performance-centric tasks in focus, Hybrid apps are not suitable because the native device features that the hybrid apps access are dependent on native plugins which are wrapped and then used by such applications [16]. So, for a developer who wants to create a to-do or checklist application for each of the three major platforms would benefit by creating a hybrid app but on the other hand, video-games for smooth gameplay should be created natively [17].

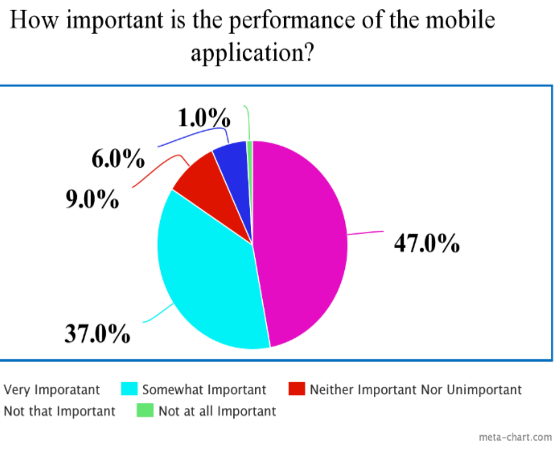


Fig 1. User Performance Reactions for the Application.

The study shows that users consider application performance as one of the most important features hence native mobile application development is always considered first when application needs to have better performance.

B. User Experience Of Hybrid And Native Application

The UI and UX are one of important and deciding factor when it come for an organization or company when they develop an application because no matter how good application performance is if its UI and UX experience is not up to the mark then user certainly decides to uninstall the application from their devices. Today's software and application development are all about User Experience [8]. Native applications have seamless UI and UX experience for the user since native application have direct access to the device's default features such as animations, Multi-touch, 3D touch, etc with makes application smooth whereas hybrid application needs third parties' plugins to do so which are not that up to the mark [14].

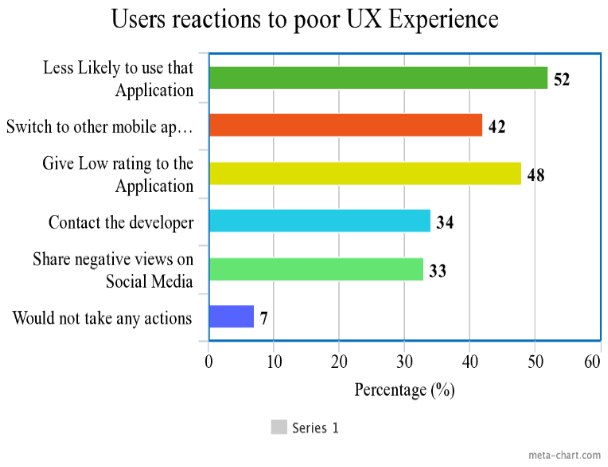


Fig 2. User reactions on UX Experience

While creating native apps, developers have to create platform-specific UIs in their native language while maintaining consistency. Instead, in the case of hybrid apps, they have the flexibility to choose from creating platform-specific UIs in the native or cross-platform language or to create a single cross-platform UI in the cross-platform language. The three major platforms i.e., Android, Windows, and iOS have their own design system. So if, developers decide to create apps natively, such as XAML for UWP apps and XML for Android, the now have to be comfortable with the native UI language. Also due to the difference in the design system, UI elements such as a Toggle button look different on each platform. This is where cross-platform application development for hybrid apps have an edge over Native apps. Frameworks such as Xamarin allows developers to create a shared UI through Xamarin. Forms that can be used for any of the three platforms. Moreover, if a developer just wants to create an Android app using their existing knowledge of XML but suspects it's a future expansion to other platforms then they can use XML for UI and C# for code-behind in a Xamarin. Android application [11]. This simplifies the developer's work and will benefit business-oriented projects as such a platform- specific UI layer makes the app look far more native and generic hybrid apps but adds an abstraction layer which may cause a slight decline in performance as the app UI components are converted into platform-specific interface elements at runtime. This decline can again be overcome by using Xamarin. Android, Xamarin. iOS for custom UI but then again will increase developers' work complexity and development time. However, for rich-UI components used in game-like applications, the developers are better off taking the native approach [13].

4. EXPERIMENTAL FINDINGS AND ANALYSIS

A. Research Analysis of Native and Hybrid Application

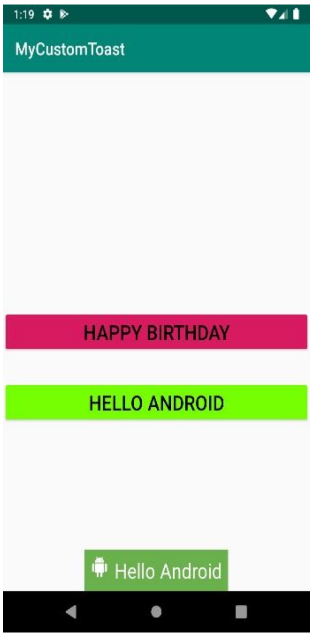


Fig 3. Our Android application shows toast message on clicking the button.

The research methodology used in this study was design science approach [2]. We designed one Native Android Application and one Hybrid Application using Xamarin Framework. First, we performed certain analysis of android application where CPU utilization, main activity performance and memory consumption were taken as consideration.

An example of our native android application is shown above. This native android application as run and installed on native android devices. This application is built on API Level 26 which is Android Nougat OS 7.1. The functionality of this is to display a custom Toast Message “HELLO ANDROID” on clicking the button Hello Android with similar feature in “HAPPY BIRTHDAY” button. This application as two Buttons which both Capable of showing Toast Messages with Vector asset logos which are the default features of this native android application without using and third-party plugins.

The hybrid application was developed using the enterprise edition of Visual Studio 2019 so that exclusive tools such as Xamarin Profiler which is a graphical interface for the Mono log profiler can be used to measure the performance of the application. The application was built using Xamarin. Forms so that both, the UI and code is shared and can be utilized on iOS and Windows as well. Performances of the application were measured on an emulator with the base device as Nexus 5X, processor as x86, ram size as 1024 MB and heap size as 256 MB. For the Xamarin Profiler, all instruments were selected with level of detail of maximum, max frames as maximum 32, sample cycle frequency as maximum, sampling mode as real. Also, track all allocations and automatic snapshots at every one second were enabled. The app was created to provide very basic functionality which was to display to the users the number of times the click me button was clicked. The reason behind such basic functionality was to be able to conclude the performance with respect to complexity of the application because as the complexity of applications increase, a decrease in performance can be expected.

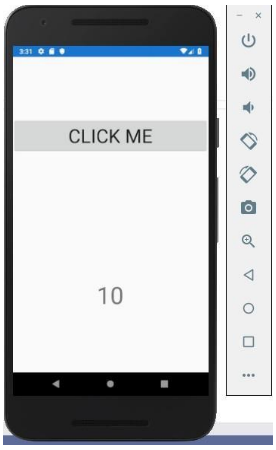
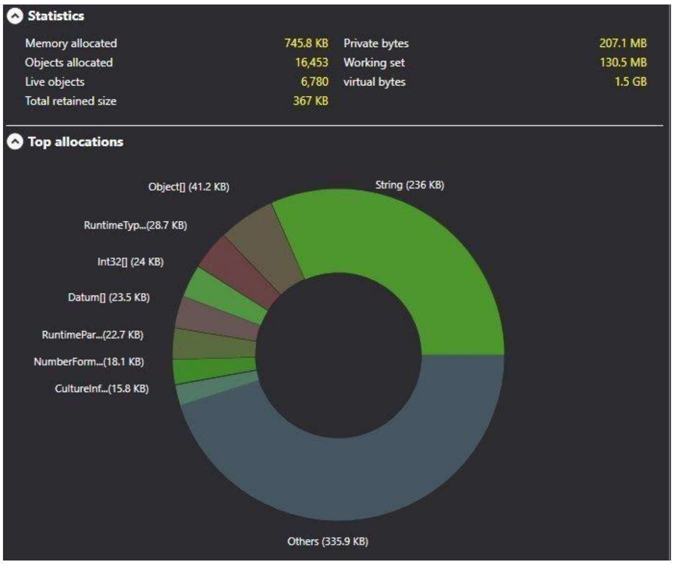


Fig 4. Our Hybrid Application built on Xamarin Framework

B. Result

The It was expected from the hybrid app to lag a bit but the lag in performance was quite noticeable [17]. The allocations graph steeply rises till 2 seconds which is because of the sudden memory consumption when the app starts. From there till 20.674 seconds the memory consumption slowly increases and then becomes constant. It took the app 31.889 seconds to completely load the UI and become functional. The steep increase in memory also matches with the high CPU usage shown in the time profiler till 21.747 seconds.



Statistics

Memory allocated	745.8 KB	Private bytes	207.1 MB
Objects allocated	16,453	Working set	130.5 MB
Live objects	6,780	virtual bytes	1.5 GB
Total retained size	367 KB		

Top allocations

Fig 5. Hybrid Application Performance Statistic

When 33 seconds had passed, the app was constantly interacted with, which very slightly increased the memory usage but significantly increased the CPU usage. During the last 5 seconds, the application hasn't interacted with due to which the CPU usage again went down. When installed on a physic device, the application took 25.47 MB of storage with average memory consumption of 1.1 MB. This is comparable to that of the native application when installed on a physical device. It should also be noted that even though the measurements were taken on an emulator the ration of difference in performance to that of the native application will remain the same nonetheless.

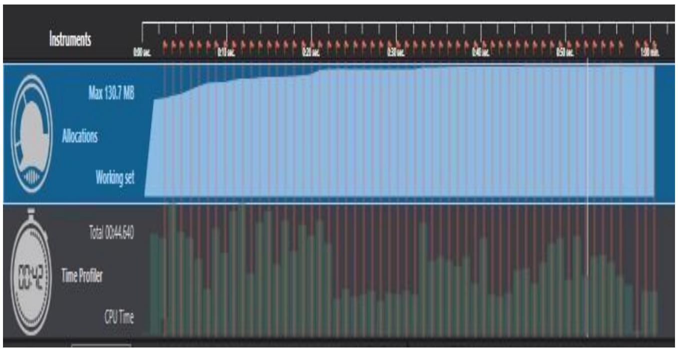


Fig 6. Memory allocation graph for Hybrid Application

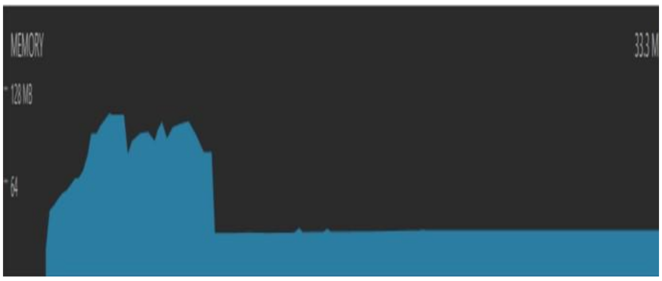


Fig 7. Energy Consumption graph of Native Application

Fig 8. Memory Allocation graph of Native Application

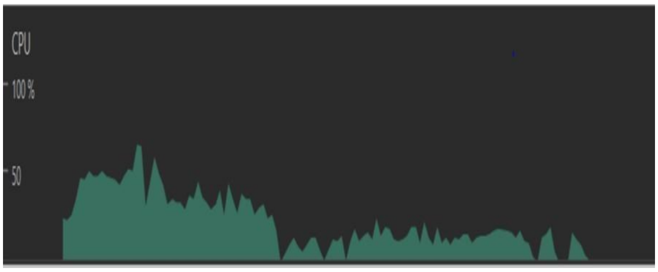
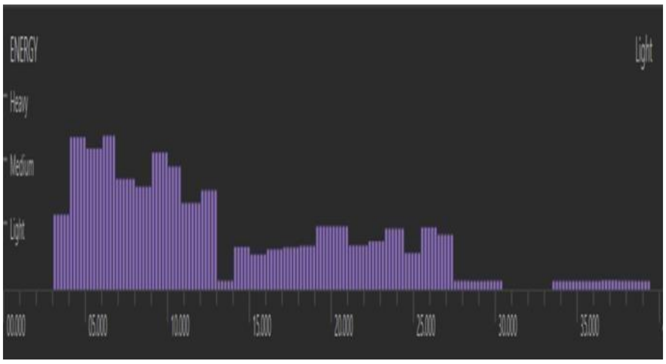


Fig 9. CPU utilization of Native Application

The android application performance was measured using Android Profiler which is available in Android Studio [12]. The result that we got from our Android Application was quite expected as it running smooth and seamless. The android application was made to run till 40 seconds which showed that initially when the buttons of application were clicked the CPU usage, memory consumption and Energy utilization was measured which showed that Memory consumption was initially high but as the time passed around 10 seconds the consumption started to become low. The application memory size is of 8 MB but when it is usage the memory allocation varies between 64 MB to 128 MB and considering CPU usage was initially high but as the interaction with application becomes less the CPU usage became less which proves that native application memory consumption and CPU usage varies significantly when compared with Hybrid Application.

5. CONCLUSION AND FUTURE SCOPE

The conclusion drawn from this research is that native applications are currently the best and suitable for developers, companies and organizations which are switching to mobile application development world because performance, user experience etc. are efficient and best in Native Applications when compared with Hybrid Applications [9] [13]. It is not like Hybrid Applications do not have any advantages over Native Applications. Hybrid Applications have low cost of maintenances and have hassle free integration with single codebase for all platforms. It is certain that both Native and Hybrid Applications have limitations such as performance to development time trade-off, UX across platforms, etc. [10].



However, it is worth noting that most of the limitations such as performance or UI responsiveness and time spent to render it vary across devices' specifications as well as the platform on which the application runs [15].

Also, these days the most common and if not the only way that smartphones and applications work is that the apps have to be installed on the users' devices and then can they run on them [16]. However, cloud gaming services such as Microsoft's cloud and Google's Stadia let users play high-performance demanding video games directly through browsers. Such technologies allow people who do not own medium to high-end devices with dedicated GPUs to play games with such requirements through their browsers. Such technologies can be implemented to fundamentally change the way we use our smartphones and help us overcome limitations such as increased UI rendering time and memory required by hybrid applications and being able to run performance-centric native or hybrid applications with absolutely no difference in performance from the user's perspective [15].

In such an approach, users should be able to interact with their devices the same way as they do now and for this, there should be negligible latency so that results of all the processing done on the powerful servers of service providers to be immediately projected to their devices. In this approach, the users should be provided with the same subscription-based pricing model as the cloud gaming services as they will be at liberty to choose and change the hardware specifications which are simulated on their devices. In such an approach since everything that the user interacts with is a reflection from the Cloud, they would require a constant, high-speed internet connection with almost zero latency but users no longer would have to buy expensive high-end devices due to hardware components they don't directly interact with such as processors. This significantly increases the time for devices to be obsolete and the devices can work with a less complex operating system through which more than one platform i.e., Android, Windows, iOS can be simulated the same way different cloud-gaming services can be accessed from different tabs of the same browser [13]. Moreover, with such a cloud-based approach developer wouldn't have worry much about the memory, performance, UI rendering time or users downloading, installing and updating their apps their hybrid or native apps or targeting a specific platform for more users. This paper presents a comprehensive and systematic analysis of native and hybrid application and shows the contrastive analysis of both respectively. This paper's contrastive analysis can be incorporated by organization and company needs who are coming to mobile application world.

6. REFERENCES

[i] Jobe, W. (2013). Native Apps vs. Mobile Web Apps. *International Journal of Interactive Mobile Technologies*, *7*(4).

[ii] Mohammadi Kho'i, F., & Jahid, J. (2016). Comparing Native and Hybrid Applications with focus on Features.

[iii] Andrade, Paulo Roberto Martins de & Frota, Otavio & Silva, Fátima & Albuquerque, Adriano & Silveira, Robson. (2015). Cross Platform App: A Comparative Study. Journal of Computer Science and Technology. 10.5121/ijcsit.2015.7104.

[iv] Xanthopoulos, Spyridon & Xinogalos, Stelios. (2013). A Comparative Analysis of Cross-platform Development Approaches for Mobile Applications. ACM International Conference Proceeding Series. 10.1145/2490257.2490292.

[v] Huynh, M. Q., Ghimire, P., & Truong, D. (2017). Hybrid app approach: could it mark the end of native app domination? *Issues in Informing Science and Information Technology*, *14*, 049-065.

[vi] Malavolta, I., Ruberto, S., Soru, T., & Terragni, V. (2015, June). End users' perception of hybrid mobile apps in the google play store. *2015 IEEE.*

[vii] anDRe ChaRLanD, B. Y., & Leroux, B. (2011). Mobile application development: web vs. native. *Communications of the ACM*, *54*(5).

[viii] Smutný, P. (2012, May). Mobile development tools and cross-platform solutions. In *Proceedings of the 13th International Carpathian Control Conference (ICCC)* (pp. 653-656). IEEE.

[ix] Khandeparkar, A., Gupta, R., & Sindhya, B. (2015). An introduction to hybrid platform mobile application development. *International Journal of Computer Applications*, *118*(15).

[x] Baldini, I., Castro, P., Cheng, P., Fink, S., Ishakian, V., Mitchell, N., ... & Suter, P. (2016, May). Cloud-native, event-based programming for mobile applications. In *Proceedings of the International Conference on Mobile Software Engineering and Systems* (pp. 287-288). ACM.

[xi] Humayoun, S. R., Ehrhart, S., & Ebert, A. (2013, July). Developing mobile apps using cross-platform frameworks: a case study. In *International Conference on Human-Computer Interaction* (pp. 371-380). Springer, Berlin, Heidelberg.

[xii] NATIVE Vs HYBRID: Things to know before Building your next Mobile Application, "www.fingent.com/blog/native-vs-hybrid-things-to-know-before-building-your-next-your-next-mobile-application".

[xiii] Gokhale, P., & Singh, S. (2014, April). Multi-platform strategies, approaches and challenges for developing mobile applications. In *2014 International Conference on Circuits Systems, Communication and Information Technology Applications (CSCITA) (pp. 289-293). IEEE.*

[xiv] Examining performance differences between Native, Flutter, and React Native mobile development, "https://thoughtbot.com/blog/examining-performance-differences-between-native-flutter-and-react-native-mobile-development*".*

[xv] The Good and the Bad of Xamarin Mobile Development, "www.altexsoft.com/blog/mobile/pros-and-cons-of-xamarin-vs-native/"

[xvi] Performance Comparison: Xamarin. Forms, Xamarin.iOS, Xamarin. Android vs iOS, "www.altexsoft.com/blog/engineering/performance-comparison-xamarin-forms-xamarin-ios-xamarin-android-vs-android-and-ios-native-applications/".

[xvii] What is Hybrid App Development? "www.ionicframework.com/resources/articles/what-is-hybrid-app-development".