STEFAN Anca-Mihaela

1240F

**EX.1**

***Short comparison between Android Studio, Ionic and React Native:***

Android studio uses as languages Java or Kotlin while Ionic uses HTML + CSS + JS and React Native uses Js. When talking about Code reuse, Android has totally separate code bases per platform, React has shared business logic with different UI codebases while Ionic has one codebase, UI codebase stays the same. The offline acces works in all three cases.

In what it concerns API Access / Native Features, Android has separate native API & codebases for each app, while Ionic and React Native have abstracted single-codebase native access through plugins.

If Android has a native UI independent to each platform, React Native has a selection of Native UI elements for iOS and Android UI elements are specific to the target platform and not shared, and Ionic has Web UI elements that are shared across any platform, conforming to the native look & feel of wherever they are deployed.

When talking about the investemnt part, Android has the largest investment in staff and time, React Native has a medium investment in staff and time while Ionic has the lowest investment in staff and time

***Ionic:***

* ***PROS***
* Ionic is a free and open-source
* Easy to learn with built-in components
* Clear and updated documentation
* Build apps for iOS, Android, Windows, Desktop, Web, and PWA
* Rich pre-styled components and great community support
* Rapid development testing cycle
* ***CONS***
* It has performance lags
* You need to use Apache Cordova plugins to access device’s hardware functionality
* It uses WebView to deliver a native-like experience

***React Native:***

* ***PROS***
* Maximum code reusability to develop apps for Android, iOS, and Windows platforms
* Stable and maintained platform to develop large projects
* Rapid development with the efficient code structure
* Wide community and platform-independent code
* ***CONS***
* There is a need for native development skills
* Lack of custom modules

***Android Studio:***

* ***PROS***
* Open system. Android developers receive access to more features.
* Developers use extensive Google design guidelines for developing an intuitive user interface.
* On the one hand, fragmentation may be considered as a disadvantage, but you can develop apps for a broader range of devices, including wearables, TVs, in-car systems, and more.
* Concerning the release, Android apps are easier to publish to Google Play. The whole process may take just a few hours.
* ***CONS***
* Fragmentation may also be an Android drawback. Nowadays, Android devices come in different screen sizes, resolutions, etc. The development team might need more time to adjust the app's features for particular devices.
* Testing. Android versions and devices might vary. Thus, it might take more time for QA specialists to test your app.
* Costly. The more time the development and testing stage takes, the higher will be the Android app price. Still, it depends on the app's features and complexity.

**EX.2** For this lab I followed the steps in the tutorial and I succeded to implement the Ionic app. As an editor I used Visual Studio Code. I encountered a small problem when I run the first command in the terminal because I did not have the .NET framework SDK. After I installed a few things it finally worked and I could follow the next steps. I learned about using Ionic and how easy it is to create a Photo Gallery app which lets you take pictures and also displays them. I attached a couple of screenshots with my app and also the source code.