Most technology users nowadays expect a lot from the software or apps they use. Ease of navigation, quick response, and functional layout are only a few of the features the end users look for. However, if there was one main functionality one could expect from an app or a system, it would be for it to work. Something as simple may however turn out a bit more complicated (or expensive) in practice. How does it happen that an app is created (or rather: written)? There are a few approaches to this topic, and each of them has specific pros, and cons but also specific uses. Let us go through different types of applications based on the methods of their development.

Native apps

Native applications are written and designed to work in one specific system. It could be either Google's Android, Apple iOS, or Windows Phone. Because all parameters are specifically adjusted to work in one given system, native apps work fast, and intuitively. They are fully compatible with the given system. The added value of the native app approach is the ability to use the phone's functions without creating dedicated plug-ins. This includes not only the phone's sensors, such as the camera, microphone, GPS, and fingerprint scanner but also the system functions (calendar, contact list, etc.). The use of these features requires granting some permissions to the app but programming such a module is fast and easy in the case of native apps. Seems like an ideal app! What is the catch? Firstly, native apps need to use API to communicate with other software components, such as the website. API provides mechanisms that enable this communication with the use of a set of definitions and protocols. These 'talks' require a specific API to be built especially for the native app's use. From a practical point of view, the native approach is time-consuming but also quite expensive. Creating native versions of apps for every system from scratch means multiplying the expenses. Additionally, if any changes need to be done to the app, every version will have to be updated separately (unless data can be sent via API). That causes even more time and resources to be devoted to native app development.

Hybrid apps

The code in which hybrid apps are written allows them to be compiled for any system. They simply translate the existing code into a language that is native to a given operating system. Hybrid applications will however require plugins to be installed in order to properly use the system elements. Some elements will have to

be programmed, so they can work as if they belonged to a native app. Unavoidably, hybrid apps will not be as fast or resource-friendly as native apps. However, they will still provide good comfort of use. When it comes to API, similarly to the case of native apps, it must be specially written to allow hybrid apps to communicate effectively with the website. One code also means that any changes we want to introduce do not require building separate apps (providing data can be sent via API). Therefore, another advantage of this method is its cost-effectiveness. If your app will be relatively simple, with contents of a database nature and no use of the phone's core functions, hybrid apps are perfect for the job. The ease of coding and updating makes hybrid apps a very popular choice.

WebView apps

WebView applications are written in many popular programming languages. They simply show a mobile version of the website they were created for. Do simple methods work the best? Not entirely. The functionality of some WebView apps can be very pleasing but others can just be uncomfortable to use. The key here is the website. If the mobile version of the website looks and works well, so will the WebView app. WebView apps are the most beneficial when it comes to the expenses and resources used. Therefore, they are mostly chosen when the budget or the scale of the project is low. Most changes to the app will be done through the website it mirrors, and API is not needed either. Unfortunately, some functions of WebView apps will not work. Furthermore, WebView apps exchange significantly more information than hybrid or native apps. That is why the speed of their responsiveness substantially depends on the speed of the Internet connection.

Which type of mobile app development should you choose?

There is no straightforward answer to this question. It depends on your individual circumstances. If you value high performance no matter the expenses, choose the native approach. It also contributes to the greater speed of the app. Moreover its stable and allows for the use of the phone's functions (such as a microphone, camera, and location). At the early stages of development, many businesses decide to test their apps by releasing them in a native version for only one system. It serves not only the purpose of testing the app but also helps to see the reception. It decreases the possible costs of a failure. If the app is not successful, there will only

be one version that was purchased and created. Are you happy with good performance and want to spend less on the project? A hybrid app will be your best choice. You will also save time and get fast working and easy-to-use app. With little money and time but a very good responsive website, you can get a perfectly working Web view application. It does not require access to phone devices or updates. Most changes will happen within the website code. A WebView approach is also a good option if you do not wish to invest in the further development of the app. At Sailing Byte, we like to keep things as easy as possible. Creating an app might be easier than you think with our Web-to-App converter. If your website is simple and non-responsive, this might be exactly what you need. Are you still unsure which category your future app falls into? You might not need an app at all. Contact us through the form to get a piece of advice on which approach suits best your needs.