GOOD PRACTICE

How we code and design UIs.

DEBIOTECH

At Debiotech, it has been around 2 years now that we have been working with Flutter for multi-platform development. One of the most important aspects of such development is how we code and design UIs, because each native framework has its own way of defining UI elements, such as Views for iOS, Activities for Android, etc... In this article, we try to explain why Flutter has become our go-to choice for creating flexible and responsive UIs.

One of the standout features that sets Flutter apart is its unified approach to UI design.
Rather than relying on native UI components, Flutter repaints the whole screen and allows us to create UIs using its own set of customizable widgets, which makes it much easier for us to create personalized widgets addressing the distinct and specific needs of our clients. It is worth noting that while Flutter includes fundamental native components of each system (buttons, arrows, etc.), it may not fully align with your design goals if you specifically desire a purely native look and feel. In such cases, exploring native development or considering frameworks like React Native (with potential additional costs) might be more suitable.

Another very important highlight of the Flutter framework is its Hot Reload functionality.
With Hot Reload, we can make code changes and instantly see the results on the running emulator or device, without losing the application's state. This allows us to drastically reduce development time as we can design and test our UIs on the fly. Having multiple emulators of various sizes running simultaneously adds to the pleasure of seeing UI updates happening automatically.

Now, let's talk about the core feature of Flutter, which can also be a double-edged sword: the Widget Library.
Flutter proposes and maintains a massive list of widgets that can almost solve any problem. Think about a design you want to make; Flutter has a widget to meet your needs. It can sometimes be hard to understand and even harder to master, but once you begin to understand the underlying logic (for example how to avoid the “unbounded height” exception 😉), you will be able to create flexible, responsive, animated UIs in a remarkably short amount of time.
However, a potential challenge arises when you think that a widget does not precisely align with your requirements, and you wish to make minor tweaks. Personalizing Flutter widgets can be a bit tricky, but rest assured, we plan to talk about it in another article.

👉 **Don’t hesitate to stay tuned for more software development insights!**

DEBIOTECH