# **iOS Wallet app**

# **1 Basics**

A table with an overview of the support status and applicability.

|  |  |
| --- | --- |
| Status: | e.g. **Supported** |
| Architecture(s): | e.g. Web |
| Component(s): | e.g. Angular, Meteor, Ionic |
| Hardware: | *iPhone 6 & Plus - iPhone 8 & Plus, iPhone X* |

# **2 Overview**

The current state of cryptocurrency wallets is still very primitive and not ready for use by masses of people for many reasons. On the usability spectrum, on the far end of good UX design is Apple and on the bad end is every existing cryptocurrency. The Divi Project’s blockchain improvements, coupled with its Smart Wallet will change that.

# **3 User details**

Ease of use

* Send and receive divi instantly with anyone in the world
* Easily request divi from anyone
  + Push notification for request received and funds received

Peace of mind

* You are the only one who has access to your funds
* You stay in control
  + Variety of personalized security features (face id, fingerprint, etc)

High security

* A security center to help protect your funds from unauthorized access
* Advanced Two-Factor Authentication keeps the bad guys out
  + Google authenticator
* Successfully completed security audits by world-class researchers
* PIN Protection

More features

* Hierarchical deterministic address architecture
* Simplified backup and recovery with a 12 word recovery phrase
* 20+ currency conversion rates
  + BTC/DIVI
  + Fiat/DIVI
* 25+ languages
* Dynamic fee display
* Open source
* QR Code Support

# **4 Technical details**

We are considering three frameworks for development [Angular](https://angular.io/), [Meteor](https://www.meteor.com/), and [Ionic](https://ionicframework.com/)

## Angular pros

**Best suited for mobile app development** — The earlier versions of AngularJS, did not pay attention to mobility oriented factors. The latest version focuses more on mobile specific trajectories the developers were longing all along. The latest version is geared towards performance based features such as going easy on the memory, touch based controllers and performance tuning to start with. As the platform is already addressing mobile application development glitches of the earlier versions, its overall efficiency is further boosted for desktop applications making Angular 2 to be a versatile technology for application development.

**Flexibility with web components**— The updated version shows greater flexibility towards all web components. Even components that were not supported earlier are now supported including Shadow DOM, HTML Imports and other custom elements. This certainly extends the horizon and the capability of the platform in developing web friendly applications.

* Developer Friendly — Unlike the earlier version, AngularJS 2.0 is more developer friendly. This was not the case with the earlier version which was essentially designer friendly.
* More user friendly — Those who have used the earlier versions will immediately vouch for the fact that the earlier version of AngularJS was not really user friendly. The learning curve was intense. The recent version has addressed this concern and made the entire platform to be more user friendly from the perspective of the developers.
* Technical glitches addressed effectively — The earlier versions of AngularJS certainly had issues on facets like minification. The latest version tries to overcome these technical downsides.
* Better routing systems — Angular 2.0 is focusing on bringing additional routing features such as nested states and sibling views for simpler and more efficient routing.

**Pros Meteor**

Fast to develop real time, single page, cross-platform application. You can develop a prototype including web app, iOS, Android app during your weekends.

Easy to scale using Galaxy or meteor up. You can also architect your apps into microservices.

**Pros Ionic**

Develop an app once, and deploy across [**iOS**](http://mobile.alliancetek.com/ios-app-development.html), [**Android**](http://mobile.alliancetek.com/android-app-development.html), and Windows devices.

The use of AngularJS creates a powerful SDK, for building rich and robust applications.

Framework’s focus on HTML, CSS, and JS enables quick development, low costs, and minimal maintenance.

# **5 Limitations**

**Cons of Angular**

**Performance.** Dynamic applications don’t always perform that well. Complex SPAs could be laggy and inconvenient to use due to their size.

**Steep learning curve.** As Angular is a versatile instrument, there is always more than one way to complete any task. This has produced some confusion among engineers. However, the abundance of tutorials and issue discussions allowed for resolving most of the problems.

**Cons of Meteor**  
You'll need to keep track of all NPM modules and their versions being used in the application. Sometimes versions conflict or are unsupported when used in conjunction with certain modules and their versions (All of which is contained in package.json).

Although it is easy to learn and develop, you need to deeply understand nodejs in order fully customize your apps. Sometimes easy things can be difficult in Meteor if you do not really understand how it works.

It works well with MongoDB only.

**Cons Ionic**  
In-app performance may not be as swift as if the application were developed natively for each device.  
  
The use of AngularJS demands that developers have a specific skillset for building complex apps.

Building in-app navigation can be notably complex, as the ui-router is tricky to maneuver at present.

# **6 Testing**

We will be farming out testing.

# **7 Areas for improvement**

List of enhancements which could be undertaken, e.g. to improve the feature itself, or improve interaction with other features.

# **8 Known issues**

List of known issues or bugs. For tech preview or experimental features, this section must contain the list of items needing fixing for its status to be upgraded.

# **9 References**

<https://angular.io/>

<https://ionicframework.com/>

<https://www.meteor.com/>