# Helping you build User Centric Apps on Blockchains

uPort is a collection of tools and protocols for building **decentralized user-centric applications**. It is built on **open standards** and open source libraries.

#### As a developer you want Users that are active and come back to your App.

Developing Decentralized Apps is already difficult. Convincing users to try your dApp, create a transaction, and return again and again is even harder.

uPort allows users to take control over their identities and Ethereum transactions. As a developer, you can implement uPort and help users build their own identities. We help you onboard your users in a simple and frictionless manner.

#### Onboarding a user to your dApp

Typically, onboarding new users to dApps is a challenging process. Before they sign up to your app they must first:

* Download a wallet
* Understand what a seed is and back it up
* Buy Ethereum on an exchange
* Understand numerous new concepts
* Now finally remember to come back to your dApp to sign up

#### Users are more than just an Ethereum Address

Most Ethereum development libraries don’t have the concept of a user. Users are just represented as Ethereum addresses.

Behind those addresses are actual people. To be successful you need to build a long-term relationship with them and their data?

#### What is Identity?

A user’s digital Identity is dynamic and constantly changing, just as they are. They interact with many apps, services, and institutions everyday.

A user’s identity is not just based on what others say about them, but also on the things they do. Their interactions with other people, businesses, and blockchains all form part of who they are and how they chose to represent themselves for future interactions.

#### Asking for consent from a user

The core concept of uPort is asking for data about them or for them to do something.

* Asking for their name
* Asking for a verification by a third party of their name
* Asking them to sign a verification on behalf of someone they know
* Asking their Ethereum Account
* Asking the user to sign an Ethereum transaction

This whole process is performed by the private exchange of signed messages. As a developer, you sign and encrypt requests for? a user. They sign and encrypt responses to you.

#### Reasons for asking for Identity Data

For developers and businesses in the traditional world there are three major reasons to ask for identity data:

1. Build a relationship with a user
2. Authenticate actions by a user
3. Risk management

#### The Problem with Centralized Identity Solutions

In a centralized world, different third parties such as social networks and credit agencies record and control this data.

Signing up users in a traditional web or financial app means requesting data from these third parties.

Each third-party effectively runs a silo providing exactly one view of the user’s identity. To get a full picture you would have to request data from each one and figure out how to merge them together.

Obviously, for those of us building decentralized technologies, the whole idea that all of this data is controlled by third parties and not by the end user itself is also extremely problematic.

#### But all I really need is an Ethereum Address

Maybe this is true strictly from a smart contract point of view; however, from your user’s perspective it may not be that simple.

Users want a reason to use your app, not just once but also to come back and feel safe that they are interacting with the same App.

Having a user sign in builds a connection between them and you. Being able to show their previous activity, helps build trust in your App and encourages them to continue interacting with it.

#### Signing Ethereum Transactions

The real value of Ethereum apps, of course, isn’t just in asking them for identity data. You want them to do something. Interact with your smart contracts and/or send funds.

uPort provides a full web3 compatible provider and allows you to call Ethereum smart contract functions from your code, automatically sending the generated function to your user’s uPort mobile app for approval and signing. This without changing any of your existing code, besides the initial setup.

#### Off-chain Data

While uPort works well with on-chain Ethereum transactions, its true strength is being able to exchange private data off the chain.

All of our requests and responses are currently based on industry standard JWTs ([RFC-7519](https://tools.ietf.org/html/rfc7519)), with signatures verified through the proposed [Decentralized Identifiers (DIDs)](https://w3c-ccg.github.io/did-spec/) standard from [W3C](https://www.w3.org/2017/vc/WG/), and the [ERC-1056 Lightweight Ethereum Identity](https://github.com/ethereum/EIPs/issues/1056) standard.

#### Fully Decentralized Apps (dApps)

Decentralized Apps live on a combination of the user’s web browser and decentralized networks such as blockchains and IPFS.

Having no dedicated server component means that logging in a user is not about authenticating access to a server. It is primarily about setting up a good user interface for your users, so they understand their own history with your app and how to create newly signed transactions.

While it is great for developers not to have a server backend to develop, it also presents many challenges.

* Where do we store the user’s state?
* How do we communicate with a user?
* If we have no central user database, how do we connect users together?

Our library – uPort Connect – provides solutions for most of these fairly difficult questions.

#### Decentralized Mobile Applications

Mobile Apps are perfect for interacting with Blockchains. They offer the potential of much stronger security and potentially better UX than web-based dApps.

Our React Native version of uPort Connect brings the full power of uPort Connect to your Mobile Apps.

For native iOS and Android developers, our uPort SDK’s and HDSigner framework can be used to handle key management for you.

#### Hybrid Applications

Sometimes you still need a few centralized components for your App. In particular, if you need to be able to safely sign data from a verified entity, or if you must have the ability to perform backend data analysis.

For these kinds of applications, use uPort Credentials to sign and verify requests from your users. Communication from your server backend is abstracted through the uPort Transports library.

#### Help your user build their Identity

Your user’s identity is built using verified data. Every interaction, request, response, and transaction they perform help improve the value of their identity.

As a developer, you can help your users and build a long-term relationship with them by verifying data about them and their interactions with you.

## uPort Protocols and Libraries

uPort is built using open standards and simple libraries built on them. We use a layered approach and build as much as possible on existing standards and tools.

|  |  |  |  |
| --- | --- | --- | --- |
| Layer | Protocol | Library | Developer Target |
| User Front End |  | Uport-connect, react-native-uport-connect | Front End Devs |
| Transport | [uPort transports](https://developer.uport.me/transports/index) | uport-transports | Backend Devs |
| uPort Signed Messages | [uPort Messages](https://developer.uport.me/messages/index) | uport-credentials | Backend Devs |
| Signed Messages | [JWT RFC 7519](https://tools.ietf.org/html/rfc7519) | did-jwt | Protocol Developers |
| Identity Abstraction | [Decentralized Identifiers (DIDs)](https://w3c-ccg.github.io/did-spec/) | did-resolver | Identity Platform Designers |
| Identity Public Key Lookup | [ERC-1056 Lightweight Ethereum Identity](https://github.com/ethereum/EIPs/issues/1056) | ethr-did-resolver |  |
| Identity Creation/Management | [Ethereum DID Registry Contract](https://github.com/uport-project/ethr-did-registry) | ethr-did | Ethereum Identity Wallet Developers |
| Blockchain API | [Ethereum JSON-RPC](https://github.com/ethereum/wiki/wiki/JSON-RPC) | [Web3](https://github.com/ethereum/web3.js/) or [EthJS](https://github.com/ethjs) |  |
| Blockchain | [Ethereum](https://ethereum.org) |  |  |