Baking Bad / Azguard wallet

Contact Details:

- Telegram: @igorfriz
- Email: if[at]bakingbad[dot]dev

Summary

This proposal introduces Azguard, a secure, feature-rich and user-friendly wallet for the Aztec network. Carefully crafted UX will enable users to get into the Aztec ecosystem and try all its features without having to be a PhD. Common design patterns and minimalistic interface will make both novice and experienced users quickly get to grips with the wallet's functionality. The wallet is going to be built by the responsible team with rich experience and solid reputation, meaning reliability and long-term support.

Estimated Start and End Date

- Start Date: October 14, 2024
- End Date: January 12, 2025

About us

We are from Baking Bad, a team of developers and just friends, started our blockchain development adventure 8 years ago in Tezos. Now we also work with Starknet, TON, Celestia, Astria and Argus.

We have rich experience in building high-quality production-grade blockchain solutions, serving tens thousands clients. The most notable are:

- 3Route, top-1 DEX aggregator in Tezos (~$100M total trading volume,audit report);
- Token bridge for Etherlink, an EVM rollup in Tezos, with its owngovernance solution;
- TzKT, top-1 Tezos explorer with unique indexer and API;
- Better Call Dev, Tezos dev-centric smart contract explorer;
- DipDup, multichain selective indexer inspired by The Graph;
- Celenium, Celestia explorer;

It's worth mentioning that we also developed an open source non-custodial multicurrency HD wallet -Atomex, with built-in true atomic swaps, enabling safe and anonymous cross-chain exchange. We have successfully passed contracts audit and wallet audit and reached ~$70M total swapped volume (swap contracts with most activity:eth, erc20, xtz), but unfortunately had to remove the atomic swaps functionality due to lack of expertise in the legal part. Therefore, the project is currently frozen, but the experience will be with us forever

.

Details & our vision

Browser extension

The Azguard wallet is going to be implemented as a browser extension, which is a fairly familiar format for users in 2024. Moreover, from our own experience, browser extension wallets give the best UX when interacting with web-based dapps due to their ability to communicate directly, that we think is crucial.

User privacy is a priority

The Azguard wallet will be fully client side. No external services, nor logs or analytics data are enabled by default. Only connection to the Aztec node. We will carefully check that sensitive data never leaves the wallet without the user's consent.

Unleashing the power of account abstraction

The Azguard wallet will be backed by its own set of account contracts, starting from a basic implementation that will fit the average user's needs, like Ledger support or social recovery, and ending with more complex solutions like multi-owner accounts. The wallet will support various implementations to cover most popular use cases. One ring

wallet to rule them all.

A place where people can taste Aztec

The Azguard wallet is going to fully support all main features of the Aztec network (primarily, its private execution and AA, including authwits, paymasters, etc.) and present them in an as much user-friendly way as possible, so people can try and get them without the necessity of having a math degree.

A more complete list of the features and functionality planned at the initial stages can be found in the milestones section below. Also, we attach a few screenshots to demonstrate how the wallet may look conceptually:

[

1280×1048 63.3 KB

](https://europe1.discourse-cdn.com/flex013/uploads/aztec/original/2X/0/01eb23578198dda5b278c9d666896e720128ba75.jpeg)

[

1280×1048 42.6 KB

](https://europe1.discourse-cdn.com/flex013/uploads/aztec/original/2X/b/bfef5991690d05f322ab42d098aeb09b4bf92b05.jpeg)

Grant milestones

We split the wallet development into two stages: MVP (see "Milestones 1-3") and actual development (see "Following development").

During the MVP development we do not focus much on UX things, like passkeys, recovery mechanisms, advanced authentication methods, etc., which are nowadays a "must have" functionality to compete with other wallets. Instead, we focus on DevX, because we believe this is what the ecosystem needs at early stages. I.e. users won't come if there are no dapps, and dapps won't be created if developers are blocked by lack of wallets with wallet connect or so.

Therefore, in the MVP we mainly focus on shipping minimal necessary functionality (authwits, token transfers and wallet interaction mechanisms) to unblock dapp developers at first.

Milestone 1 (beginning of Nov):

- design the wallet architecture, taking into account Aztec specifics and ensuring efficient development and further scalability and upgradeability;
- implement basic accounts management functionality: creation, delayed deployment, switching between accounts. At this stage the default SchnorrAccountContract

implementation is used;

- implement basic contract interaction functionality in the wallet for both private and public flows, including contracts and recipients management;
- implement a scaffold/initial version of our own account contract that will be used by default in the future, constantly evolving.

Milestone 2 (beginning of Dec):

- implement minimal necessary functionality in the account contract: deploying, authentication and execution, including batch execution;
- implement tokens functionality in the wallet: enable public and private transfers, display public and private balances;
- fees are paid in Fee Juice at this stage;
- implement authwits functionality both on the account contract and on the wallet side;
- if PXE is not yet ready for working in browser we will consider hosting a centralized PXE service (for testnet only) to lower entry threshold, so users won't have to install docker and run PXE manually to just try the wallet (of course users

will still be able to use their own local PXE);

- initial integration with wallet connect, enabling basic functionality: message signing and batch execution with authwits;
- publish an early version of the wallet.

From this point we will follow an incremental releases model, so all the features will not necessarily be released by the end of the milestone period, but when they are actually ready.

Milestone 3 (beginning of Jan):

- display account's public and private state (active authwits, notes, etc.);
- store transaction history locally and display it in the wallet;
- implement account export and import functionality. At this stage only master secret and auth data are exported/imported (auth data depends on a particular AA implementation, for instance, it's Grumpkin scalar for Schnorr account, and nothing for single key account). Eventually, additional data from PXE will also be handled;
- implement an early version of the Azguard SDK, allowing web-based dapps (for which low latency is a requirement) to interact with the wallet directly.

Following development

Besides constant UX improvements and participation in the ecosystem's standards development, we are going to constantly enhance our account contracts and wallet app with advanced functionality and state of the art technologies.

The following features are planned at the moment:

- support QR Codes;
- make use of passkeys;
- support multiple account contract implementations to enable more complex use cases;
- support authentication via hardware wallets;
- support authentication via multisig;
- implement social recovery mechanism;
- implement account contract upgrade mechanism;
- implement and maintain FPCs, allowing to pay fees in popular tokens and integrate them into the wallet;
- integrate token bridges;
- integrate dexes/aggregators;
- audit contracts and ensure full tests coverage;
- allow to connect explorers, to show all transactions history, not just what has been saved locally on a particular device. By default this feature is disabled. When enabling, the user is warned about privacy risks;
- allow to connect quote providers, to show the value of the user's assets. By default this feature is disabled. When enabling, the user is warned about privacy risks;
- exploring ways to speed up account synchronization (for instance, if a user exactly knows that he created his account not earlier than at a particular time, the PXE can safely skip trial-decryption of all the blocks produced before that time);
- track overall performance, in particular, whether or not the PXE feels good running in a browser extension (if at some point we realize that it doesn't work well, migration from a browser extension to a desktop app might be considered);
- provide users with comprehensive documentation and video guides.

Grant amount & rationale

Request amount is: $100,000.

Team

FTE

Frontend dev (Core & SDK)	1
Frontend dev (UI)	1
Noir developer	0.7
Designer	0.3*
Project manager	0.5*

- All infrastructure costs, as well as the project management and design costs will be covered by the Baking Bad team.