Grant Application Form

Grant Title:

Upgrade CoW Protocol SDK to Ethers.js v6 & Implement Provider-Agnostic Functionality

Author:

- Saldin Ilya (saldin.iliya@gmail.com)

About You:

I am a blockchain developer specializing in Ethereum and EVM-compatible ecosystems. I previously worked at 1inch, where I contributed to various aspects of smart contract integration and blockchain interaction libs. My primary motivation is to make CoW Protocol more accessible by reducing its dependency on a single library (ethers.js v5) and enabling integrators to select their preferred provider. Also I've participiated in developing JS-SDK for dittonetwork.io with provider-agnostic architecture.

Additional Links:

According to the forum's restrictions, I'm not able to add more than two links. After approving post I'll add them via comments.

Grant Category:

Core Infrastructure & Developer Tooling

Grant Description:

This project aims to migrate the CoW Protocol SDK from ethers.js v5 to ethers.js v6, accounting for all breaking changes, while introducing a provider-agnostic structure. Our scope includes:

1. Analyzing the current CoW SDK code to identify all references to ethers.js v5.
2. Refactoring the code and tests to ensure compatibility with ethers.js v6.
3. Implementing an abstraction layer that allows users to switch between different providers (e.g., web3.js, viem) without major code changes.
4. Enhancing test coverage and updating documentation to ensure an easy integration path for the CoW developer community.

Grant Goals and Impact:

- Goals:
- Eliminate technical debt related to the outdated ethers.js v5 dependencies.
- Provide developers with the flexibility to choose their preferred provider.
- Improve overall code quality via expanded test coverage and updated documentation.

- Eliminate technical debt related to the outdated ethers.js v5 dependencies.
- Provide developers with the flexibility to choose their preferred provider.
- Improve overall code quality via expanded test coverage and updated documentation.

- Impact:
- Streamline the process of integrating CoW Protocol into various DeFi applications by removing strict reliance on one provider library.
- Encourage a broader developer base to adopt the CoW SDK.
- Reduce potential bugs tied to incompatible ethers.js versions, thus improving stability and reliability.

- Streamline the process of integrating CoW Protocol into various DeFi applications by removing strict reliance on one provider library.
- Encourage a broader developer base to adopt the CoW SDK.
- Reduce potential bugs tied to incompatible ethers.js versions, thus improving stability and reliability.

Milestones:

Milestone

Due Date

Payment

Milestone 1: Basic Migration and Integration of External Functions

4 weeks from project start

6,000 xDAI

Milestone 2: Library-Agnostic Layer, Extended Testing, and Final Release

5 weeks after Milestone 1

6,000 xDAI

Milestone 1: Basic Migration and Integration of External Functions

Tasks and Deliverables:

:

- Integrate External Functions into the SDK

:

- This includes functions like signOrder or computeOrderUid that may currently reside in external libraries or scripts.
- The goal is to have everything developers need within the SDK, minimizing the need to import separate scripts.

- This includes functions like signOrder or computeOrderUid that may currently reside in external libraries or scripts.
- The goal is to have everything developers need within the SDK, minimizing the need to import separate scripts.

- Migrate to ethers.js v6 (instead of v5)

:

- Replace outdated methods and types (e.g., converting BigNumber to bigint).
- Refactor any portions where function signatures have changed (e.g., Signer in ethers.js).
- Ensure everything compiles and functions as expected at a basic level.

- Replace outdated methods and types (e.g., converting BigNumber to bigint).
- Refactor any portions where function signatures have changed (e.g., Signer in ethers.js).
- Ensure everything compiles and functions as expected at a basic level.

- Basic Testing

:

- After refactoring, verify that primary functions (e.g., order creation, signing, contract interactions) work correctly with the new ethers.js version.
- The main objective is to confirm that migrating to v6 does not break critical functionality.

- After refactoring, verify that primary functions (e.g., order creation, signing, contract interactions) work correctly with the new ethers.js version.
- The main objective is to confirm that migrating to v6 does not break critical functionality.

- (Optional) Preliminary Release of the New Major Version

:

- We can publish, for example, v6.0.0-alpha or v6.0.0-beta for user testing.

- Alternatively, we can hold off on the final release until Milestone 2, keeping just an internal “ready” version after M1.
- We can publish, for example, v6.0.0-alpha or v6.0.0-beta for user testing.
- Alternatively, we can hold off on the final release until Milestone 2, keeping just an internal “ready” version after M1.

Expected Outcome:

- A functioning SDK running on ethers.js v6, with crucial external functions consolidated into its codebase.
- Verified by a basic test suite (not necessarily 100% coverage, but sufficient to rule out major issues).
- Potentially an initial “alpha” or “beta” release that early adopters can begin to test.

Milestone 2: Library-Agnostic Layer, Extended Testing, and Final Release

Tasks and Deliverables:

- Full Implementation of a Provider-Agnostic Approach

: * Develop a blockchain interaction abstraction (e.g., BlockchainProvider) that does not rely on ethers.js.

- Write adapters for ethers.js v6, Viem.
- Separate core logic so that the provider can be swapped at the configuration level without breaking the SDK.
- Develop a blockchain interaction abstraction (e.g., BlockchainProvider) that does not rely on ethers.js.
- Write adapters for ethers.js v6, Viem.
- Separate core logic so that the provider can be swapped at the configuration level without breaking the SDK.
- Extended Test Coverage

: * Add unit and end-to-end tests for different providers (ethers, Viem, etc.).

- Cover all new methods, features, and edge cases.
- If necessary, perform integration testing on a real network (testnet or mainnet fork).
- Add unit and end-to-end tests for different providers (ethers, Viem, etc.).
- Cover all new methods, features, and edge cases.
- If necessary, perform integration testing on a real network (testnet or mainnet fork).
- Documentation and Examples

: * Update the README with guides on how to integrate the library and select a preferred provider.

- Provide a “migration guide” detailing what changes from the previous SDK version (ethers.js v5) to the new setup (v6 + agnostic).
- Update the README with guides on how to integrate the library and select a preferred provider.
- Provide a “migration guide” detailing what changes from the previous SDK version (ethers.js v5) to the new setup (v6 + agnostic).
- Final Release

: * If there was no full production release after Milestone 1, launch the stable major version here (e.g., v2.0.0).

- Publish it on npm and GitHub, complete the review process, and merge all changes into the main branch.
- If there was no full production release after Milestone 1, launch the stable major version here (e.g., v2.0.0).
- Publish it on npm and GitHub, complete the review process, and merge all changes into the main branch.

Expected Outcome:

- A fully updated SDK capable of operating with ethers.js v6, Viem.
- Comprehensive documentation, code examples, and a major release version.

- Extensive tests ensuring stability and reliability.

Funding Request:

I request a total of 12,000 xDAI

for the complete project.

- Milestone 1

: 6,000 xDAI

- Milestone 2

: 6,000 xDAI

This budget reflects the effort required for research, refactoring, extended testing, documentation.

Budget Breakdown:

- Refactoring & Migration (M1: 6,000 xDAI)

: Code audit, transition to ethers.js v6, basic testing.

- Provider-Agnostic Layer, Testing & Documentation (M2: 6,000 xDAI)

: Development of abstraction layer for multiple providers, expanded test coverage, final documentation, and pull request reviews.

Gnosis Chain Address (to receive the grant):

0x399697118F540E47C2b89d6B9715C7C0BFC29cBd

Referral:

I was introduced to the CoW Protocol Grants program by a protocol developer@shoom3301.