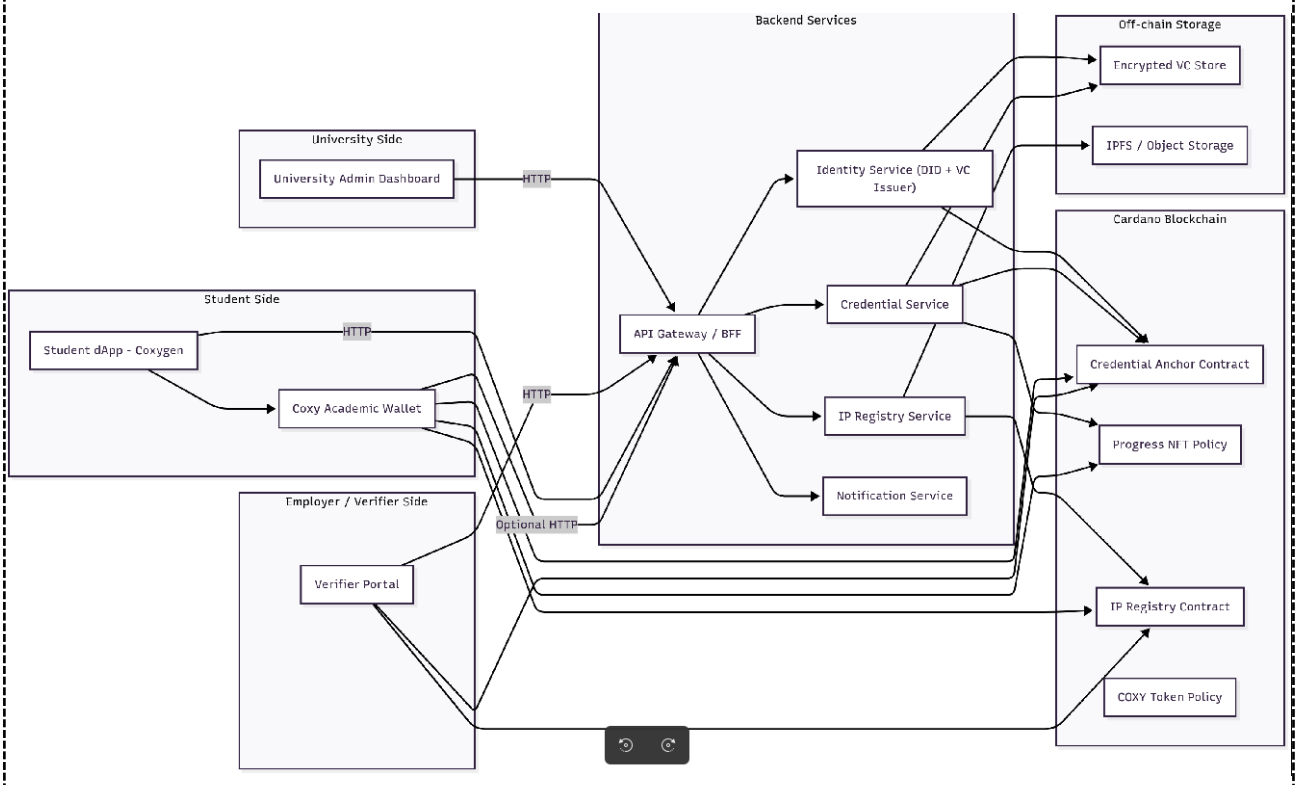
COXY: Tokenizing Tertiary Student Credentials On-Chain

Technical Specification

By Coxygen Global (Pty) Ltd 2025

Table of Contents



1. Document Overview

Project Name

COXY: Tokenizing Tertiary Student Credentials On-Chain

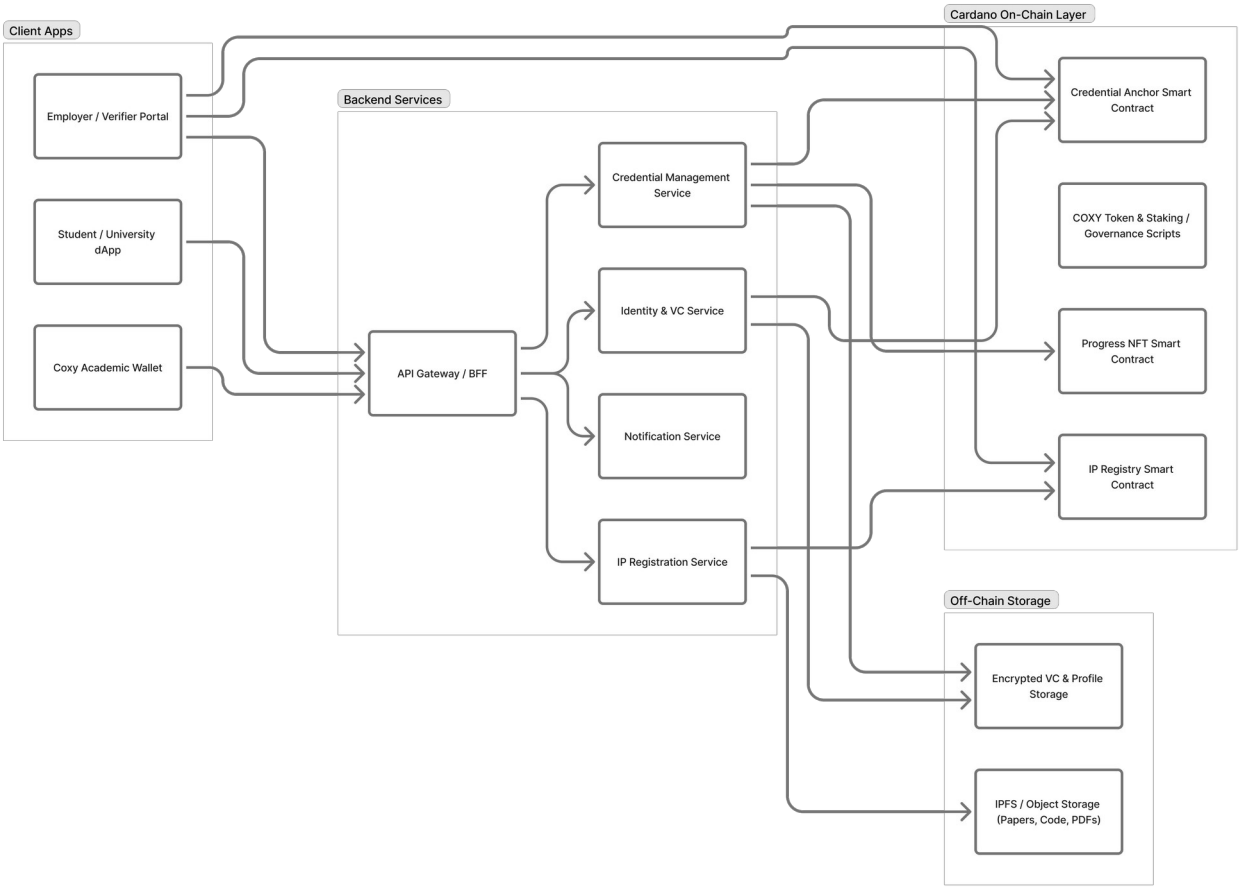
Purpose

This document defines the technical requirements and high-level architecture for a global Cardano-based system that:

- Gives students self-sovereign academic identities (SSI/DIDs).
- Represents academic progress and credentials as verifiable credentials and progress NFTs.
- Registers academic intellectual property (IP) on-chain.
- Uses a tokenized incentive and governance model (the COXY token).

Primary Stakeholders

The main users of this specification are the engineering teams (smart contracts, wallet, backend, and front-end), Coxygen / WIMS-Cardano product and program leads, partner universities and employers, and security, privacy, and compliance reviewers.



2. System Goals & Scope

2.1 Goals

The system aims to give each student a self-sovereign academic identity managed in the Coxy Academic Wallet. Academic progress and credentials should be represented as privacy-preserving, verifiable records anchored on Cardano. Universities must be able to issue credentials easily, and employers must be able to verify them online with minimal integration. The system should also allow registration and timestamping of academic IP (such as research, code, and projects) and implement a tokenized protocol (COXY) for fees, staking, and governance.

2.2 In Scope

Phase 1 includes upgrading Coxy Wallet into an “Academic Wallet” with DID and key management, verifiable credential (VC) storage and presentation, and an academic profile view. It also covers the development of on-chain smart contracts for credential anchors, progress NFTs, IP registration, and a basic COXY token policy with staking/governance logic.

On the backend, we will build services for issuing and revoking VCs, verification APIs and portals, and email notifications for transaction hashes and verification links. The system will integrate with the existing Coxygen universities dApp, providing a student dashboard and a university admin view.

2.3 Out of Scope (Phase 1)

Phase 1 will not include full Atala PRISM integration, cross-chain interoperability, or advanced DAO tooling. Governance will be kept lightweight, using simple mechanisms such as snapshot-style voting.

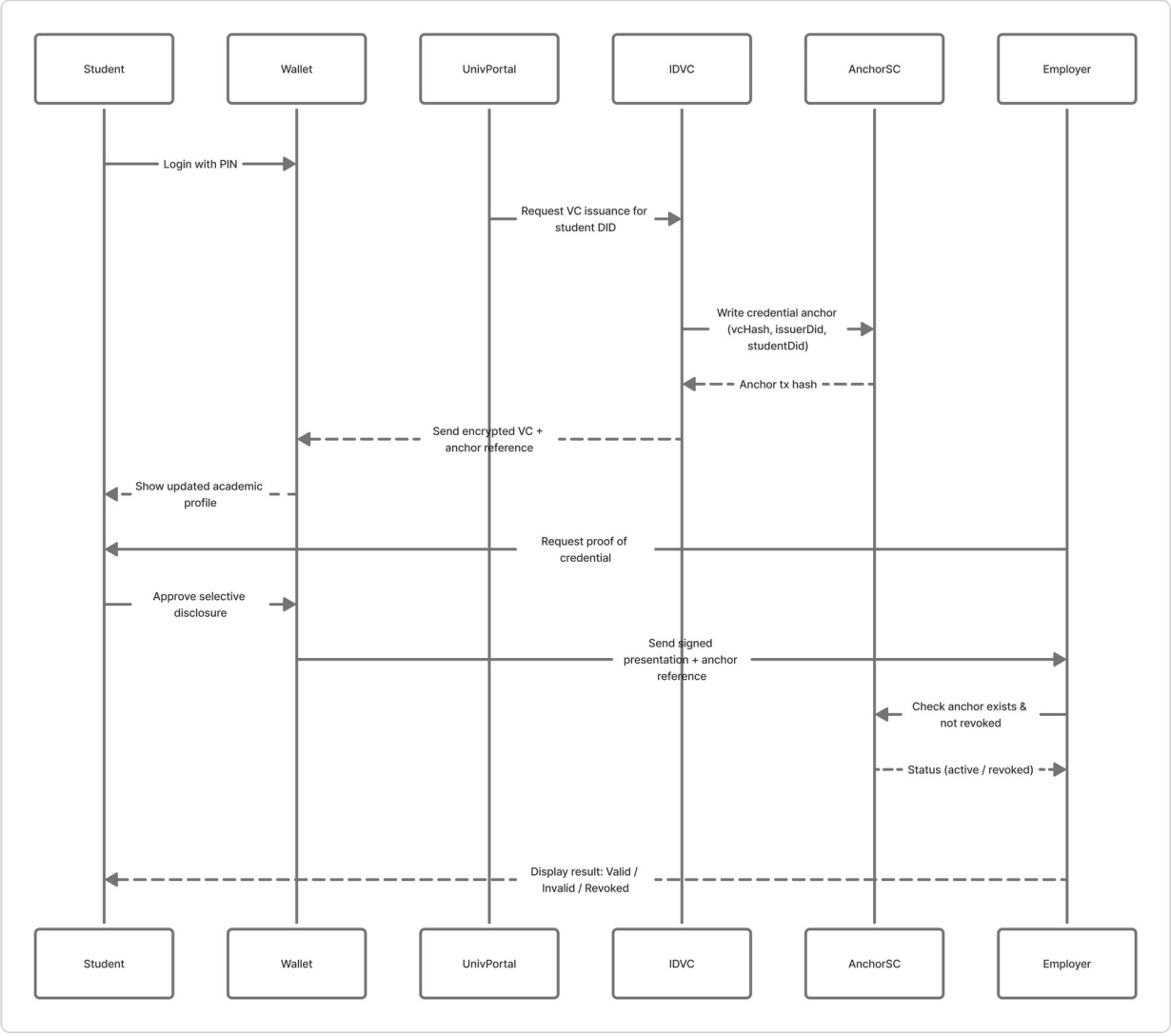
3. High-Level Architecture

3.1 Logical Components

The system consists of three main application surfaces: the Coxy Academic Wallet (web and/or mobile), the student/university dApp, and an employer/verifier portal.

Behind these, backend services handle identity (DIDs and VC issuing), credential management, IP registration, notifications, and an API gateway or backend-for-frontend layer.

On Cardano, a set of smart contracts manages credential anchors, progress NFTs, IP records, the COXY token policy, and basic staking/governance scripts. Off-chain storage holds encrypted credentials and metadata, with optional IPFS/S3-style storage for larger files like papers or code. Monitoring and observability are provided through logging, metrics, dashboards, and alerts.



4. Technology Choices

The system runs on Cardano (testnet and later mainnet). Smart contracts are written in Helios or Plutus V2, with Helios generating validators. The Coxy Academic Wallet uses WebAssembly and JS/TS and connects via a Cardano light client or a Blockfrost-like API.

Backend services are implemented in TypeScript (Node.js) or Haskell, using frameworks such as Express/Fastify or Yesod/Servant. PostgreSQL stores relational data, and encrypted blob storage holds backed-up VCs if needed. Web dApps and portals are built in React or Next.js.

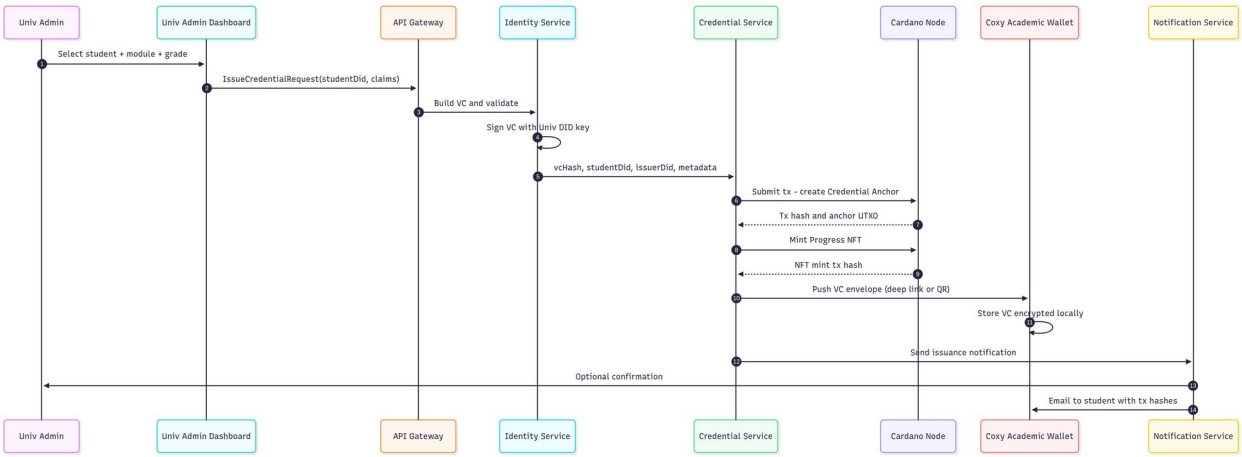
5. Functional Overview

5.1 Coxy Academic Wallet

The wallet generates and manages Cardano payment and stake keys, as well as DID key pairs. Keys are stored encrypted on-device and never sent unencrypted to the server. Users get mnemonic backups and simple PIN-based access.

The wallet creates and updates DID documents, synchronizes DID anchors with on-chain mechanisms, and receives verifiable credentials from the identity service. VCs are stored encrypted and can be searched by institution, type, or date. When a student shares credentials, the wallet creates a verifiable presentation with selective disclosure and signs it with the DID key.

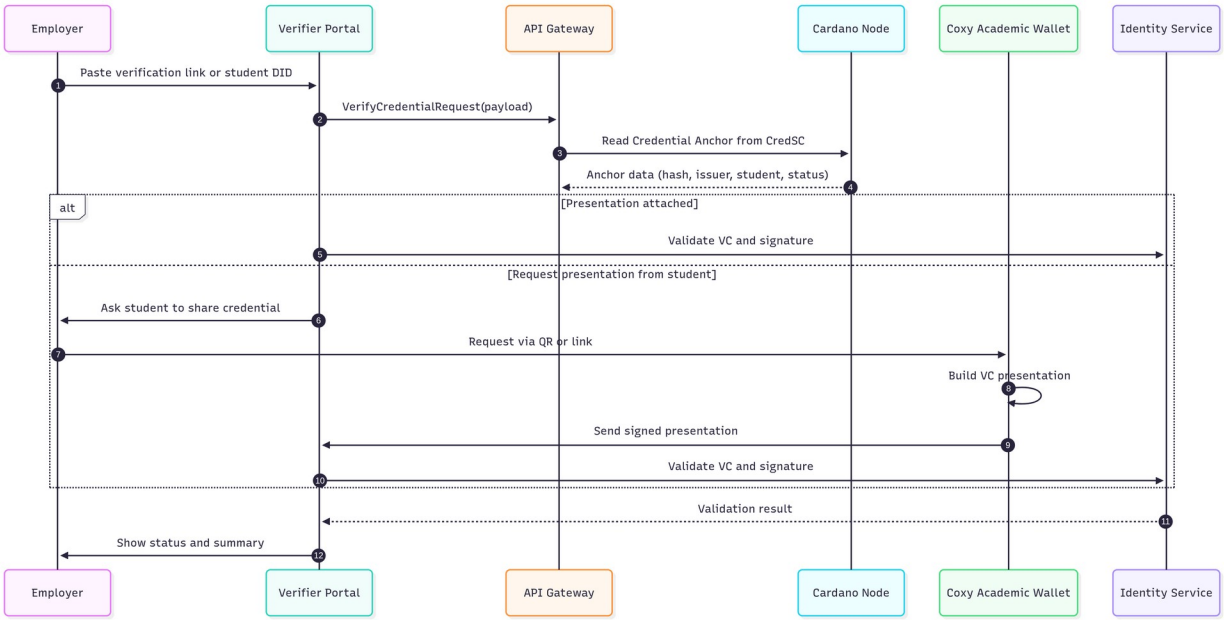
The wallet also shows a unified academic profile, aggregating progress NFTs, credential anchors, and IP registrations, and a regular asset view for ADA, tokens, and academic NFTs. Users can review and approve transactions for credential anchors and IP registration.



5.2 University / Institution System

Institutions use an admin dashboard to manage courses, modules, programs, and student rosters (linked to student DIDs or internal IDs). From this dashboard, staff issue credentials and progress NFTs.

For each credential, the system creates a W3C-compliant VC, signs it with the institution’s DID key, and delivers it securely to the student’s wallet. It then writes a credential anchor to Cardano, storing the hash and minimal metadata. Universities can mint module completion NFTs in bulk or individually, link them to student addresses or DIDs, and perform revocations or updates when needed. Revoked or superseded credentials update both off-chain VCs and their on-chain status, and students are notified.



5.3 Employer / Verifier Portal

Employers and other verifiers access a web portal or API. They either follow a verification link or enter a student DID or credential ID. The portal fetches the student's presentation or uses the signed payload, validates the VC signature, checks the on-chain anchor, and confirms that the credential is not revoked or expired. It then displays a clear status—valid, invalid, revoked, or expired—along with a human-readable summary.

An API (REST or GraphQL) allows HR systems and applicant tracking systems to integrate verification flows directly. SDKs in JS/TS (and possibly Haskell) simplify this integration.

5.4 IP Registration

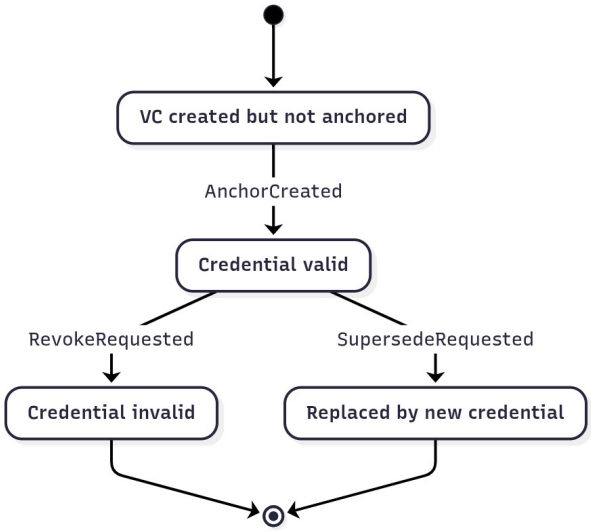
Students or institutions can submit academic IP through the wallet or dApp by providing metadata (title, abstract, type, co-authors) and either content or a precomputed hash. The system computes a content hash (for example, SHA-256), constructs a mint transaction for an IP token, and writes it to the IP registry contract.

The IP token records the hash, owner DIDs, timestamp, and optional reference URLs. Ownership can be a single student or a multi-sig with the institution. A UI and API allow anyone to query IP records by DID, address, or hash and to view the associated metadata and proof.

5.5 Tokenomics & Protocol Logic

The COXY token has a fixed supply of 1,000,000,000 tokens. A multi-sig or DAO-controlled policy mints the full supply once and then locks further minting. Initial allocations are defined in configuration and issued to treasury and distribution addresses.

Every credential anchor, IP registration, or verification call pays a small fee in ADA and optionally COXY. Fees are split between the treasury, the institutions or verifiers involved, and an optional burn or lock component. Institutions and verifiers can also stake COXY, linking their DID or address to a staked amount, which influences their reputation and, in later phases, enables slashing for misbehavior.

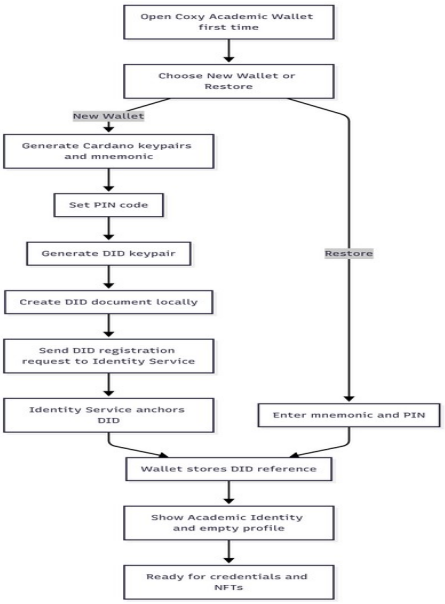


Governance starts with a minimal MVP, using COXY balances and optional reputation tokens (ARP) to signal proposals and record accepted decisions such as updated fee parameters.

6. Smart Contract Layer (Summary)

Smart contracts are written in Helios and compiled to Plutus V2. They use versioned data types for datums and redeemers and are deployed to testnet first, then mainnet after audits.

The **Credential Anchor Contract** stores hashes of VCs and related DIDs, timestamps, and status (active, revoked, or superseded). Only issuers or governance scripts can change status, and no personal data appears on-chain.



The **Progress NFT Contract** mints tokens representing course or module achievements. Token names encode course and module IDs and a student-specific suffix, while detailed metadata lives off-chain or in standard CIP metadata.

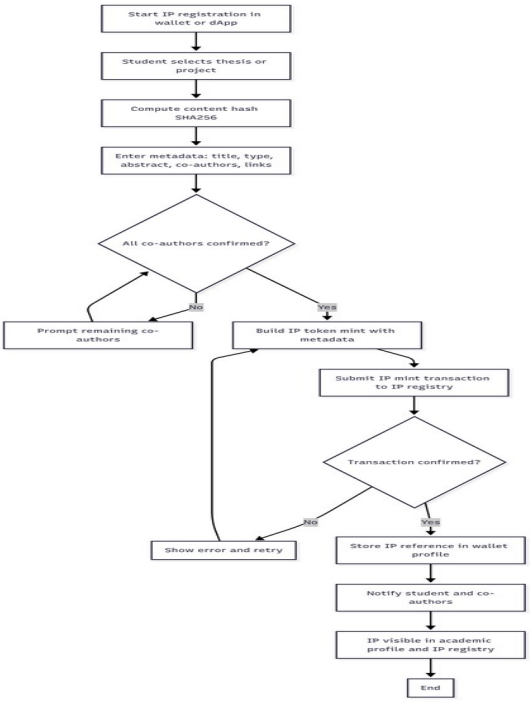
The **IP Registry Contract** creates IP tokens that record a content hash, owner DID hash, timestamp, and lightweight metadata. Revocation is handled by additional state rather than deletion.

The **COXY Token Policy** mints the total supply once, under a multi-sig. After this, the policy is effectively locked. Simple **staking and governance scripts** allow users to lock COXY, record staking data, and store minimal governance results in Phase 1.

7. Data & API Models (Summary)

Off-chain data is primarily JSON. Verifiable Credentials follow W3C standards, with contexts, types, issuer DIDs, issuance dates, credential subjects (program, module, grade, completion date), and cryptographic proofs (e.g., Ed25519 signatures).

APIs handle tasks such as issuing VCs to a student DID using a template and claims, and return both the VC and the transaction hash of the associated credential anchor.



8. Non-Functional Requirements

From a security perspective, all keys are generated client-side and never transmitted unencrypted. Off-chain VCs are encrypted with keys derived from student secrets. All communication uses TLS and strong authentication, and smart contracts are audited before mainnet deployment.

For privacy, no PII is stored on-chain. Off-chain PII is encrypted at rest and in transit and stored only with user consent. The right to be forgotten is supported by deleting off-chain data while leaving on-chain proofs unlinkable without the user’s keys.

The system should scale to at least 10,000 credential anchors per day at launch, with horizontally scalable backend services and cached on-chain reads via an indexer. The verification API targets 99.5% uptime with proper monitoring, alerting, and incident response.

9. DevOps & Deployment

There are separate environments for local development, public testnet, an optional staging environment, and mainnet. CI/CD pipelines run automated tests for smart contracts, backend services, and front-ends. Deployments to dev and testnet are automatic after merges; mainnet deployments require manual approval.

Observability includes centralized logging, metrics (e.g., credential issuance and verification volume, error rates), and on-chain analytics dashboards.

10. Implementation Milestones

The technical implementation is divided into milestones:

- **M1 (Months 1–2):** Finalize data models, basic DID and key management in the wallet, and dev/testnet infrastructure.
- **M2 (Months 3–4):** Implement the credential anchor contract, VC issuing and storage services, and integrate issuance into the university dashboard and wallet.
- **M3 (Months 5–6):** Add progress NFTs, academic profile views in the wallet and dApp, and extended email notifications.
- **M4 (Months 7–8):** Implement the IP registry and verifier portal, delivering end-to-end verification for employers.
- **M5 (Months 9–10):** Implement the COXY token policy, fee routing to the treasury on testnet, and basic staking.
- **M6 (Months 11–12):** Harden security, fix findings, deploy core components to testnet, and run pilots with selected universities and employers.