Devang Sharma, 2026, 14:6
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

CarbonWise: A Blockchain-Based Carbon Credit Marketplace for Small and Medium Cement Manufacturers

Devang Sharma, Priyansh Sharma
Department of Computer Science and Engineering
Poornima Institute of Engineering and Technology Jaipur, Rajasthan, India

Abstract- CarbonWise is a blockchain-oriented carbon credit marketplace concept designed to help small and medium cement manufacturers document, represent, and exchange carbon-reduction claims through a transparent digital workflow. The proposed platform combines Ethereum-compatible smart contracts, an ECO token for representing approved credit units, and the InterPlanetary File System (IPFS) for storing supporting evidence off-chain while anchoring content identifiers on-chain. A web application provides the user-facing marketplace, while wallet-based authorization supports ownership and transaction signing. The project also explores gasless token transfers through EIP-712 typed-data signatures and a relayer, reducing the need for users to hold native blockchain currency for every transfer. The architecture is intended to improve traceability, auditability, and access to digital carbon-credit workflows. However, blockchain records cannot independently establish that an emissions reduction is real: credible monitoring, reporting, verification (MRV), authorized issuance, and controls against double counting remain essential. This paper presents the problem context, system architecture, proposed workflow, security considerations, implementation approach, and evaluation plan for CarbonWise. Since independently verified deployment and performance measurements are not available in the project information used for this draft, the paper does not claim quantitative experimental results.

Keywords- Carbon credits, blockchain, cement industry, Ethereum, smart contracts, IPFS, ECO token, EIP-712, carbon accounting, MRV.

I. INTRODUCTION

Industrial decarbonization requires mechanisms that can make emissions data, reduction claims, and the movement of environmental assets easier to audit. Cement manufacturing is a particularly relevant context because the sector faces process-related and energy-related emissions, while smaller manufacturers may have limited resources for participating in sophisticated environmental data and carbon-market systems. Carbon-credit markets can channel finance toward eligible mitigation activities, but their usefulness depends on the credibility of the underlying reductions and the integrity of the records that represent them [1], [2].

CarbonWise is proposed as a blockchain-based marketplace focused on small and medium cement manufacturers. Its goal is to provide a digital workflow through which a manufacturer can submit evidence associated with a reduction claim, have the claim reviewed by an authorized verifier, represent an approved unit using an ECO token, and list or transfer that unit in a marketplace. The platform combines on-chain transaction records with off-chain evidence storage so that the system can maintain a traceable history without placing large documents directly on the blockchain.

The central design principle is to separate evidence submission, independent verification, credit issuance, and trading.

A blockchain can make authorized state changes and transfers tamper-evident, but it cannot determine whether a meter reading, emissions factor, or uploaded report is truthful. CarbonWise therefore treats verification as a governance and operational requirement rather than assuming that tokenization alone creates an authentic carbon credit.

© 2026 Devang Sharma. his is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly credited.

Objectives

- Design a traceable digital workflow for carbon-reduction claims and marketplace transactions.
- Use smart contracts to record approved issuance and token transfers under explicit authorization rules.
- Store supporting documents through IPFS and associate their content identifiers with on-chain records.
- Explore wallet-based access and EIP-712 signed messages with a relayer for gasless transfer interactions.
- Define a practical evaluation plan covering correctness, security, traceability, usability, and transaction performance.

Contributions

This paper contributes a project-level architecture for a cement-focused carbon-credit marketplace, a workflow separating verification from token transfer, an evidence-integrity model using content-addressed storage, and an evaluation framework that can be used to assess a working prototype. The contribution is an implementation-oriented system design, not a claim that CarbonWise is an accredited carbon registry or a substitute for an independent verification program.

II. BACKGROUND AND RELATED WORK

Carbon markets rely on rules for defining eligible activities, quantifying reductions, validating project documentation, verifying outcomes, issuing units, and tracking their retirement or use. The World Bank's work on digital monitoring, reporting, and verification describes how digital systems can support carbon-market data collection and reporting while also emphasizing implementation barriers and the need for appropriate institutional processes [1]. The World Bank's carbon-pricing reports further show that carbon pricing and crediting mechanisms operate within evolving policy and market frameworks [2]. These sources support the need for dependable records, but do not imply that a software marketplace can itself certify environmental outcomes.

Distributed ledgers offer shared transaction histories and programmable rules. In a marketplace, these properties can help make token ownership and transfer events auditable across participants. Smart contracts can enforce rules such as role-based issuance, transfer restrictions, and retirement states. Nevertheless, ledger immutability is not equivalent to data accuracy: incorrect information submitted before a transaction can remain incorrect after it is recorded. CarbonWise therefore needs a defined verification authority, documented methodologies, and a clear process for correcting or revoking invalid claims.

IPFS is a content-addressed system in which content is identified by a content identifier (CID). A CID can be stored in a smart contract or associated database record so that a later user can check whether retrieved content matches the referenced content. IPFS does not by itself guarantee permanent availability, confidentiality, or the truth of a document; persistence requires pinning or another storage arrangement, and sensitive information should not be published in plaintext.

EIP-712 defines a standard for hashing and signing typed structured data [3]. In CarbonWise, typed signatures can describe a transfer request in a structured form, and a relayer can submit the corresponding transaction to a contract that validates the signature and replay-protection fields. A relayer design must carefully validate the chain identifier, verifying contract, nonce, deadline, signer, and transfer parameters. Gasless means that the user does not directly pay the network fee for that interaction; it does not mean that the transaction has no cost.

Research Gap

The project addresses a practical integration problem: connecting evidence records, authorized credit issuance, token ownership, and marketplace transfers in a single user workflow oriented toward smaller cement manufacturers. Its intended value is traceability and reduced friction in digital handling of credit records. Whether it improves access, reduces transaction overhead, or improves trust

must be evaluated empirically; these outcomes are research questions, not assumed results.

III. SYSTEM REQUIREMENTS AND METHODOLOGY

1. System Scope and Actors

The proposed system contains the following actors:

- Manufacturer: registers a facility, submits a reduction claim, uploads supporting documents, and views approved token balances.
- Verifier or administrator: reviews evidence and authorizes issuance only after the applicable verification process is completed.
- Buyer or marketplace participant: views available listings and purchases or receives eligible credit units.
- Smart contracts and relayer: enforce on-chain rules and submit valid signed transfer requests.

The verifier role is deliberately separate from the manufacturer role. In a production deployment, verification should be performed by an independent party under an accepted methodology and applicable program rules. The current project description does not establish that CarbonWise is connected to an accredited registry or an independent verifier.

2. Functional Requirements

- Account and wallet connection: users connect an Ethereum-compatible wallet to identify the address used for authorized actions.
- Claim evidence: a manufacturer submits metadata and supporting files; the application records a CID or equivalent reference and the claim status.
- Review and issuance: only an authorized role can approve a claim and issue the corresponding ECO token units.
- Marketplace: eligible token holders can create listings and execute transfers according to contract rules.
- Traceability: contract events and transaction hashes provide a history of issuance and transfers.
- Optional gasless transfer: the user signs a typed transfer request and a relayer submits it after validation.

3. Non-Functional Requirements

- Integrity: contract permissions and transaction events must support auditable state changes.
- Security: signatures, role permissions, replay protection, and external service credentials must be handled safely.
- Availability: off-chain evidence must be pinned or otherwise retained; IPFS addressing alone does not guarantee persistence.
- Usability: users should understand claim status, token units, transaction outcomes, and any fees.
- Compliance: issuance and marketing claims must align with applicable carbon-credit program rules and must not misrepresent test tokens as certified credits.

4. Proposed Workflow

First, a manufacturer creates a claim and uploads supporting evidence. The application sends the file to IPFS or a compatible pinning service and records the returned CID with claim metadata. Second, a verifier reviews the submitted data using a documented methodology and records an approval or rejection through the authorized application workflow. Third, approval permits the smart contract to issue the corresponding ECO token units, with an event linking issuance to the claim identifier and evidence reference. Fourth, an approved holder can list or transfer eligible units through the marketplace. Finally, transfer and retirement events, if implemented, are retained as an auditable transaction history.

This workflow assumes a clear unit definition. The project must specify whether one ECO token represents one tonne of CO_2-equivalent, a fractional unit, or a project-specific accounting unit. Until that definition and the applicable methodology are finalized, ECO should be described as a project token representing an approved claim within the prototype, not automatically as a universally recognized or registry-issued carbon credit.

IV. PROPOSED ARCHITECTURE

1. Presentation Layer

The web interface is intended to support manufacturer onboarding, evidence submission, claim-status views, token balances, marketplace listings, and transaction history. The remembered project stack includes a React/Next.js-style frontend, wallet connectivity through MetaMask or WalletConnect, and Ethereum interaction through a web3 library such as ethers.js or wagmi. Exact package versions and final deployment configuration should be recorded from the project repository before publication.

2. Smart-Contract Layer

Solidity contracts on an Ethereum-compatible network define token behavior and authorization. The project has used the Sepolia test network during development and explored an ECO token contract and a gasless-transfer component referred to as EcoMetaTransfer. The contract should restrict issuance to authorized roles, emit events for relevant state changes, and prevent unauthorized or repeated execution of signed transfers. Any production contract would require independent review and security testing before real assets or valuable credits were handled.

3. Evidence and Metadata Layer

IPFS is used as the intended evidence-storage layer, with a pinning service such as nft.storage having been explored in the project workflow. The on-chain record should store a CID or a cryptographic digest and a compact claim identifier rather than the full document. Publicly accessible files should be limited to information intended for public disclosure. Personal, commercially sensitive, or confidential facility information should be encrypted or kept in access-controlled storage, with the limitations of key management clearly documented.

4. Gasless Transfer Layer

The planned gasless transfer sequence is: (1) the sender reviews a transfer request in the interface; (2) the wallet signs an EIP-712 typed message; (3) the frontend sends the signature and typed parameters to a relayer endpoint; (4) the relayer validates request format and policy; and (5) the smart contract verifies the signer, nonce, deadline, and domain before executing the transfer. The relayer pays the transaction gas. A direct token transfer from the frontend does not constitute this gasless flow, because it normally requires the sender to authorize and pay for an on-chain transaction.

5. External Data and Oracles

Chainlink was considered as a potential integration for external data or oracle services. Such an integration should only be described as implemented if the repository and deployed contracts confirm it. An oracle can transport or attest to data under a specified design, but it cannot replace an appropriate emissions methodology or independent verification process. This draft therefore treats Chainlink-based verification as a possible extension rather than a confirmed operational feature.

V. IMPLEMENTATION APPROACH

The remembered development environment includes Solidity and an Ethereum-compatible Sepolia test network, IPFS-based evidence storage, a web frontend, wallet connectivity, and an intended relayer for EIP-712-based transfers. The project has also explored nft.storage for content persistence. Because a complete source repository, deployment logs, contract test output, and end-to-end test records were not provided with the current request, this paper does not assert that every component is currently operational.

1. Smart-Contract State Model

A minimal implementation can represent each claim with a unique identifier, submitting manufacturer, evidence CID, status, verifier address, and issuance amount. Valid states may include Submitted, UnderReview, Approved, Rejected, Issued, and Retired, with transitions controlled by explicit roles. The contract should reject invalid transitions, prevent duplicate issuance for the same approved claim, and emit events that include the claim identifier. If claim documents or verification decisions are updated, the system should preserve a traceable history rather than silently replacing prior records.

2. Application and API Integration

The frontend presents the workflow and obtains wallet signatures. A backend or relayer service can submit gasless requests and maintain operational logs, but it must never hold users' private keys. Relayer credentials should be stored as server-side secrets, protected against abuse, and rate-limited. API requests should validate user input and use transaction receipts—not optimistic UI state alone—to confirm whether an on-chain action succeeded.

3. Development Network

Sepolia is suitable for test-network experimentation, but testnet tokens have no inherent real-world carbon value. The paper should record the chain ID, contract addresses, deployment transaction hashes, compiler version, and source-code commit used for any reported experiment. The remembered project history includes a deployed contract address for an earlier iteration, but that address is intentionally not treated as proof of the current deployment state; it should be re-checked against the latest repository and explorer record.

VI. EVALUATION PLAN AND DISCUSSION

A credible evaluation should separate functional correctness from environmental validity. Blockchain tests can establish whether contract rules behave as intended, while carbon-accounting validity requires an independent verification methodology and supporting evidence. The following plan is proposed for a reproducible assessment; it is not a report of completed experiments.

1. Functional Tests

- Role control: verify that unauthorized addresses cannot approve claims, mint ECO tokens, or alter privileged configuration.
- Issuance integrity: verify that rejected or pending claims cannot mint tokens and that an approved claim cannot be issued twice.
- Transfer correctness: verify balances, events, marketplace settlement, and rejection of insufficient-balance transfers.
- Signature validation: test invalid signatures, wrong chain IDs, wrong contract domains, expired requests, modified parameters, and reused nonces.
- Evidence integrity: retrieve an uploaded file and verify that the content matches its recorded CID; test behavior when the content is unavailable.
- Failure handling: test reverted transactions, unavailable IPFS gateways, relayer downtime, and interrupted frontend requests.

2. Metrics

The evaluation should report successful and failed test counts, gas consumed by key contract operations, end-to-end confirmation time, relayer service response time, evidence retrieval success rate, and the number of duplicate or unauthorized actions correctly rejected. For marketplace usability, a small structured user study could measure task completion time, error rate, and user understanding of claim status and transaction costs. Measurements should include test conditions, number of trials, network congestion, and software versions.

3. Results Status

No verified numeric performance results are available in the project details used to prepare this draft. Accordingly, no gas savings, transaction latency, verification accuracy, carbon reduction, adoption rate, or cost improvement is claimed here. Once test logs and reproducible measurements are available, a Results subsection should present the measured values, baselines, sample sizes, and limitations. Simulated or testnet results must be labeled as such and must not be presented as evidence of real-world emissions reductions.

4. Discussion

CarbonWise's potential strengths are a unified record of claim metadata and token activity, a user-facing marketplace, and a transfer workflow that may reduce direct wallet friction through a relayer. Its principal risks are oracle and evidence trust, inaccurate or unverifiable emissions claims, double counting, loss of off-chain evidence, smart-contract vulnerabilities, relayer abuse, regulatory uncertainty, and confusion between prototype tokens and certified credits. A robust deployment would require independent verification, transparent governance, clear unit definitions, contract audits, incident

response, privacy controls, and alignment with applicable standards and registry rules.

Carbon-Credit Integrity and Governance

A marketplace for environmental assets must distinguish the technical representation of a credit from the environmental claim that gives the credit its meaning. CarbonWise can record the status and movement of a token, but the integrity of the underlying claim depends on how the reduction is defined, measured, verified, issued, and retired. This distinction is particularly important for cement-related activities because emissions may arise from several sources, including process emissions associated with clinker production, fuel combustion, electricity consumption, and upstream inputs. A credible implementation must state which emissions sources are covered and which accounting boundary is used.

Before a claim is eligible for tokenization, the platform should require a documented baseline and a defined monitoring period. The baseline describes the counterfactual emissions against which a reduction is assessed, while the monitoring period defines when the reported activity occurred. The methodology should specify the data required, calculation factors, uncertainty treatment, and evidence that must be retained. Where a claim depends on a change in fuel, energy source, production process, or material composition, the platform should record the activity type and relevant assumptions rather than storing only a final number. The platform should also address additionality, permanence where relevant, leakage, and double counting according to the rules of the applicable crediting program. These concepts cannot be resolved by smart-contract code alone. For example, a contract may prevent the same claim identifier from being issued twice within CarbonWise, but it cannot prevent the same reduction from being claimed simultaneously in another registry unless there is a reliable cross-registry process. A production design should therefore identify the authoritative registry or program, define unique identifiers, and make retirement status visible to marketplace participants.

Governance should describe who may submit a claim, who may review it, who may approve issuance, how conflicts of interest are handled, and how errors or suspected fraud are investigated. Administrative powers should be limited and transparent. High-impact actions, such as changing issuance permissions or replacing a verification authority, may require a multi-signature process and a publicly documented reason. The system should preserve a history of decisions so that later users can reconstruct why a claim was accepted, rejected, suspended, or retired.

CarbonWise Data Model and Lifecycle

A clear data model reduces ambiguity between a manufacturer, a facility, a project, a reduction claim, an issued unit, and a marketplace listing. These objects should have separate identifiers. A manufacturer may operate multiple facilities, a facility may submit multiple claims over time, and one approved claim may result in multiple token units. Treating these objects as a single record would make auditing and later correction difficult. The application should therefore link records through stable identifiers while retaining the transaction history associated with each state change.

A claim record can include a claim identifier, manufacturer address, facility identifier, activity category, reporting period, quantity claimed, unit of measurement, evidence CID, submission timestamp, review status, verifier identifier, and final decision reference. The precise fields depend on the chosen methodology and should be finalized before deployment. Only information needed for public traceability should be placed directly on-chain. Detailed operational reports, invoices, sensor exports, and other evidence can remain off-chain, with a CID or cryptographic digest used to detect later content changes.

The lifecycle should be implemented as an explicit state machine. A newly submitted claim may enter a review state; the verifier may approve or reject it; an approved claim may become eligible for issuance; issued units may be listed, transferred, suspended, or retired according to the program rules. Invalid transitions must be rejected. For example, a rejected

claim must not be issued, a retired unit must not be listed as available, and a claim should not be issued a second time merely because a frontend request is repeated after a timeout. These conditions should be enforced in the contract wherever practical and covered by automated tests.

The lifecycle also needs a policy for corrections. If an uploaded report is found to contain an error, deleting the original reference would weaken the audit trail. A better approach is to preserve the original event, record the correction or superseding document, and update the claim status through an authorized process. Where already-issued units are affected, the program must define whether they are frozen, cancelled, replaced, or marked as disputed. The interface should clearly communicate these statuses so that users do not mistake a disputed unit for an unrestricted asset.

Smart-Contract Design and Security Controls

Smart contracts are the enforcement layer for token balances and authorized state transitions. The token contract should use a well-understood token interface where appropriate and should avoid unnecessary custom functionality. Minting authority should be limited to a dedicated role, and role administration should be separated from ordinary marketplace use. If a pause mechanism is included for emergencies, the contract documentation must explain who can invoke it, which operations it affects, and how the system returns to normal operation. Emergency controls improve response options but also introduce trust in the administrators who hold those permissions.

The contract should validate every input that affects balances or authorization. Amounts must be positive where required, recipients must be valid addresses, claim identifiers must be unique, and transfers must respect token balance and status rules. Events should be emitted for claim submission, approval, rejection, issuance, listing, purchase, transfer, and retirement when those functions exist. Events make activity easier to index and audit, but they should not be treated as the sole source of truth if the contract state and event interpretation differ.

Common risks include unauthorized minting, missing access checks, replayed signatures, incorrect nonce handling, re-entrancy in contracts that call external code, integer or unit mistakes, and unsafe administrative changes. A gasless transfer contract adds further requirements because the signer and the transaction submitter are not necessarily the same address. The contract must verify that the typed message is bound to the correct chain and verifying contract, that its nonce has not been used, that the request has not expired, and that the signed parameters match the executed transfer. A signature should never be reusable for a different recipient or amount.

Security testing should combine unit tests, negative tests, static analysis, and independent review. Negative tests are especially important: they intentionally attempt to mint from an unauthorized address, transfer more than the available balance, reuse a nonce, change a signed parameter, submit an expired request, or execute an invalid state transition. The results should record the expected behavior and actual outcome. A testnet deployment is useful for validating integration, but it does not replace a professional audit or prove that a contract is safe for valuable assets.

IPFS Evidence Handling and Privacy

The use of IPFS supports content-addressed evidence references, but evidence handling requires a broader lifecycle policy. Uploading a file and receiving a CID does not guarantee that the file will remain available indefinitely. The operator should define a pinning strategy, backup arrangements, monitoring for retrieval failures, and responsibility for maintaining evidence. The application should also validate file type and size, scan uploads where appropriate, and avoid allowing untrusted content to be rendered unsafely in the browser.

A CID helps detect whether retrieved content matches the referenced content, but it does not prove who created the file or whether its contents are accurate. The evidence record should therefore include contextual metadata, a submission timestamp, the submitting account, and a verification decision. If a verifier signs a decision, the

system can preserve the signature and associated record to strengthen accountability. This still depends on protecting verifier credentials and defining a trustworthy verification process.

Privacy is an important consideration because facility reports may include commercially sensitive production information, employee information, supplier details, or operational data. Public blockchains and public IPFS gateways are not appropriate places for confidential information in plaintext. CarbonWise should minimize personal data, publish only the information required for public audit, and store sensitive material in encrypted or access-controlled systems. Encryption introduces key-management responsibilities: losing a key may make evidence unrecoverable, while exposing a key may compromise confidentiality.

The platform should define retention and deletion policies before collecting evidence. An immutable reference may remain visible even if an off-chain file is later removed, so users should understand that removing a file from one storage service does not necessarily erase all copies or metadata. A privacy review should consider data minimization, lawful basis where applicable, access control, incident response, and the rights of people whose information may appear in uploaded documents.

Gasless Transfers: Design Rationale and Threat Model

Requiring every marketplace participant to acquire native blockchain currency can make a prototype difficult to use, particularly for first-time users. A gasless transfer design allows the user to sign a structured request while a relayer submits the transaction and pays the network fee. The relayer may be funded by the application operator, a marketplace fee, or another disclosed mechanism. The economic model should be transparent because relaying shifts the cost rather than eliminating it.

The EIP-712 domain should include the relevant chain identifier and verifying contract. The signed message should include the sender, recipient, amount, nonce, and deadline, along with any other parameter that affects execution. The contract should consume the nonce only when the transaction succeeds according to the intended semantics. The relayer should enforce rate limits, reject malformed requests, avoid exposing signing credentials, and monitor for abuse. A relayer can refuse to submit a valid request, so the user experience should explain whether a direct-transfer fallback is available.

A gasless design must also guard against misleading signing prompts. The wallet should show the token, recipient, and amount in a human-readable form, and the frontend should ensure that the displayed details correspond exactly to the signed payload. Domain separation reduces the risk that a signature created for one contract or chain is accepted by another. Expiration limits the time during which a signed request can be submitted, while nonces prevent repeated execution. These protections should be tested using both valid and deliberately invalid signatures.

Threat Model and Risk Mitigation

The main assets to protect are token balances, claim status, verifier authority, evidence integrity, relayer credentials, and user privacy. Potential adversaries include a malicious participant attempting to mint or transfer units without authorization, a compromised administrator, an attacker submitting altered evidence, a relayer abuser, or an external service that becomes unavailable. The system should document which threats are addressed by smart contracts and which require organizational controls.

On-chain controls can enforce permissions, valid transitions, and replay protection. Off-chain controls are needed for identity checks, evidence review, file scanning, relayer monitoring, and backup management. Governance controls should address administrator changes, incident response, and conflicts of interest. No single mechanism solves every risk: for example, blockchain immutability can preserve an audit trail but can also preserve an erroneous claim, and IPFS content addressing can reveal a changed file without establishing the truth of the original file.

A risk register should rank each risk by likelihood and impact, identify a responsible owner, and record the mitigation and test evidence. High-priority risks include fraudulent or duplicate claims, unauthorized issuance, signature replay, loss of evidence availability, compromised privileged credentials, and confusion between test tokens and certified credits. Residual risks should be disclosed rather than described as eliminated. The register should be revisited after each major change to contracts, APIs, or marketplace rules.

Deployment, Monitoring, and Operational Plan

A responsible deployment plan should progress through local development, automated tests, testnet integration, controlled demonstration, and only then any production assessment. Each stage should have explicit acceptance criteria. Before a testnet release, the team should record compiler settings, contract addresses, chain identifiers, deployment transactions, frontend build identifiers, and API configuration. Reproducible records allow reviewers to connect reported behavior to the exact software version that produced it.

Operational monitoring should track transaction submission failures, contract reverts, relayer queue depth, evidence retrieval failures, and unusual administrative actions. Alerts should be actionable and should not expose private keys or sensitive uploaded content. The application should distinguish between a request being prepared, submitted, pending, confirmed, or reverted. This is particularly important in blockchain interfaces because a wallet approval or successful API response does not necessarily mean that the on-chain transaction has completed.

Backups should cover application configuration, non-secret operational records, and evidence storage where permitted. Secrets should be held in a suitable secret-management system and rotated when compromise is suspected. Incident procedures should identify who can pause relevant operations, communicate a problem, preserve forensic evidence, and restore service. Recovery procedures should be tested rather than assumed to work. If an administrative pause is available in the contract, its limitations and governance should be documented to avoid creating a single unaccountable point of control.

Evaluation Protocol and Reporting Template

To make the evaluation reproducible, each experiment should state its objective, environment, preconditions, test inputs, number of repetitions, expected outcome, observed outcome, and pass/fail criterion. Functional tests should be run against a known contract deployment and identified source revision. Performance measurements should specify whether they refer to local execution, a test network, or a production chain, because these environments have different latency and fee characteristics.

A useful comparison for gasless transfers is between a direct token transfer and a relayed signed transfer. The report should measure the gas consumed by the on-chain execution, total fee paid by the relayer, confirmation time, number of failed requests, and the user steps required. It should not label the gasless path as free; instead, it should describe which party pays and how the application manages that cost. Any claimed reduction in user friction should be supported by a defined usability method.

Evidence integrity can be evaluated by uploading a test file, recording its CID, retrieving it, and checking that the retrieved content matches the expected content. Additional cases should include unavailable gateways, missing pins, altered metadata, and duplicate claim submissions. Contract-level integrity can be evaluated by attempting unauthorized and repeated issuance. These tests can demonstrate technical behavior, but they cannot independently validate the scientific accuracy of a carbon-reduction calculation.

For clarity, a final results table should distinguish measured values from design targets. Each measured value should include the environment and method used to obtain it. Where data are unavailable, the paper should mark the field as 'not measured' rather than inserting an estimate. The current draft intentionally contains no invented numeric performance results; the project team should populate this section only after collecting

reproducible logs and retaining the supporting evidence.

Expected Practical Impact and Adoption Considerations

For small and medium manufacturers, the potential benefit of CarbonWise is not simply the use of a blockchain. The platform would need to reduce the complexity of documenting claims, make review status easier to understand, and provide a transparent path from approved evidence to marketplace activity. If participation requires expensive sensors, complex wallet management, or extensive manual data entry, smaller facilities may still be excluded. A practical pilot should therefore evaluate onboarding effort, evidence preparation cost, connectivity assumptions, language accessibility, and the availability of support.

The marketplace should also avoid suggesting that tokenization automatically creates demand or a reliable market price. Buyers need information about methodology, verification, vintage, project type, status, and applicable restrictions before making a decision. Listing pages should make the source and status of a unit understandable and should distinguish testnet demonstrations from real, recognized credits. Clear disclosures help prevent users from confusing a software token with a legally or environmentally recognized instrument.

Adoption depends on institutional trust as much as technical design. Manufacturers may need confidence that submitted data will not expose sensitive production information, while buyers need confidence that credits are not duplicated or overstated. Independent verification, transparent governance, accessible documentation, and a process for disputes are therefore central parts of the system. These operational arrangements should be designed alongside the contracts and interface, not postponed until after development.

Design Alternatives and Architectural Decisions

On-chain and Off-chain Data Placement

A key design decision is determining which information belongs on the blockchain and which information should remain off-chain. On-chain data is replicated across network participants and can be costly to update, so it is best suited to compact identifiers, authorization state, token balances, and events that need shared verification. Large documents, detailed emissions calculations, facility records, and supporting evidence are more appropriate for controlled off-chain storage. CarbonWise follows this separation by proposing that the chain retain a claim identifier and evidence reference while the document itself is stored through IPFS or a compatible storage workflow.

This separation introduces a dependency: an on-chain record may remain available even if an off-chain document becomes difficult to retrieve. The application should therefore monitor evidence availability and show whether a referenced file can currently be retrieved. A persistence policy should identify who pays for pinning, how redundancy is maintained, and how a file is recovered if a service provider becomes unavailable. The system should not imply that a CID alone is a backup strategy.

Permissioned Roles and Public Verifiability

CarbonWise may use a public blockchain test network for transparent token movement while retaining permissioned controls for claim approval and issuance. Public verifiability does not require every participant to be able to approve claims. Instead, the contract can allow anyone to inspect public state and events while limiting sensitive state changes to designated roles. The trade-off is that users must trust the process by which verifier identities and administrative roles are selected. This trust should be documented in governance rules and supported by reviewable actions.

A more decentralized governance model could be explored in future work, but decentralization is not automatically beneficial for every operation. Dispute resolution, compromised verifier credentials, and erroneous issuance require accountable decision-making. A practical early-stage system may therefore use clearly disclosed administrative roles while adding safeguards such as multi-signature approval, role separation, time delays for sensitive changes, and public event logs. The objective is not to remove

all trust, but to make trust assumptions explicit and auditable.

Marketplace Settlement Options

A marketplace can support direct token transfers, escrow-based purchases, or listings that reference an external payment process. Each model has different risks. Direct transfers are simple but may not guarantee that payment and asset delivery occur together. Escrow can provide atomic settlement within a contract but introduces additional code and requires careful handling of cancellations, fees, and failed transactions. If payment is handled outside the blockchain, the application must clearly distinguish payment confirmation from token transfer confirmation. The appropriate choice depends on the scope of the prototype and the legal and operational rules for the credit units.

Carbon Accounting Workflow for Cement Facilities

The cement sector is a useful context for studying digital carbon accounting because emissions are associated with multiple activities and process stages. A carbon-reduction claim might relate to fuel efficiency, alternative fuels, renewable electricity, process improvements, or changes in material composition. These activities do not all use the same calculation method. CarbonWise should therefore avoid applying one generic formula to every claim and instead associate each claim with a defined methodology and reporting boundary.

A practical workflow begins with the facility identifying the activity and the reporting period. The claim record should identify the relevant production quantity, energy or material inputs, applicable emissions factors, and evidence sources required by the chosen methodology. A verifier then checks whether the evidence is complete, whether calculations follow the methodology, and whether the claim is eligible under the relevant program. Only after approval should the application make the claim eligible for issuance. The application can support this process by organizing evidence and recording decisions, but it should not replace technical verification.

Data quality is a major concern. Meter readings may be missing, instruments may be improperly calibrated, and reported production quantities may use inconsistent units. Emissions factors can vary by geography, time period, and source. CarbonWise should record the provenance of the inputs and identify any assumptions used in a calculation. Where uncertainty is material, the methodology should define how it is handled. A token contract can store an approved quantity, but it cannot make an uncertain calculation scientifically reliable.

The project could later explore IoT integrationn for collecting energy, temperature, production, or other relevant operational data. Such integration would require sensor calibration, device identity, secure transmission, timestamp integrity, and controls against tampering. An IoT device can automate data collection, but automated collection does not automatically make a measurement accurate or independently verified. Any AI-based anomaly detection or automated estimation should be treated as decision support and validated against suitable reference data before it is used in a consequential crediting workflow.

Where external datasets or emissions factors are required, an oracle integration may provide a controlled way to deliver selected data to a contract or application. Chainlink was considered in the project context, but this draft does not assume that a live oracle is deployed. The design should specify the source, update frequency, validation rules, failure behavior, and responsible party for any external feed. If a feed is stale or unavailable, the system should flag the condition rather than silently treating old data as current.

User Roles, Interface Design, and Accessibility

The frontend is the primary point of interaction for manufacturers and buyers, so the quality of the interface affects whether the technical system can be used correctly. The manufacturer dashboard should distinguish submitted claims from approved claims and issued units. The marketplace should show unit definitions, claim identifiers, verification status, and any limitations on transfer or retirement. Users should be able to inspect transaction hashes and

evidence references without needing to understand every blockchain implementation detail.

Wallet interaction is a source of possible confusion. A user may connect a wallet without having approved a transaction, may sign a message without submitting an on-chain transaction, or may submit a transaction that later reverts. The interface should explain these states separately. For gasless transfers, the interface should clearly identify the action being signed and indicate whether a relayer has accepted the request. If the relayer fails, the user should receive an actionable explanation rather than a success message based only on a successful API call. Accessibility should be considered from the beginning. The target users may have different levels of technical familiarity, different language preferences, and inconsistent connectivity. Clear labels, plain-language explanations, responsive layouts, and useful error messages can reduce operational mistakes. The platform should avoid making a blockchain wallet the only source of identity or support; users may need onboarding materials and a documented process for recovering access to the application without exposing private keys.

The interface should also make uncertainty visible. A submitted claim is not the same as an approved claim, an approved claim is not necessarily an issued credit, and a testnet token is not a certified environmental asset. Status labels, tooltips, and listing disclosures should reinforce these distinctions. Good interface design is therefore part of risk management, not merely a visual improvement.

Economic and Operational Sustainability

A working marketplace requires a sustainable operating model. Costs may include contract deployment, transaction fees, relayer funding, IPFS pinning, application hosting, monitoring, security reviews, verifier services, and user support. A prototype may subsidize these costs during testing, but a production proposal should state who pays for each service and how continuity is maintained. The gasless-transfer feature shifts transaction costs to the relayer operator and requires safeguards against automated abuse or unexpectedly high demand.

Small manufacturers may be sensitive to the time and cost required to prepare evidence. If the documentation process is more expensive than the potential benefit of participation, the system may not achieve its intended accessibility goal. A pilot should measure onboarding time, evidence preparation effort, verifier turnaround, user support needs, and the proportion of submitted claims that require correction. These measures can help identify whether the workflow is practical for its intended users.

The market value of a credit depends on factors beyond the application, including methodology, verification quality, demand, buyer requirements, and the rules of the relevant program. CarbonWise should not imply that the ECO token has a guaranteed price or that a marketplace listing guarantees a sale. Any fees, redemption rights, transfer restrictions, and unit definitions should be disclosed. If ECO is only a prototype token, the interface and paper should say so clearly.

Operational sustainability also requires a plan for project maintenance. Smart contracts may be difficult or impossible to modify after deployment, while frontend and backend services will evolve. The architecture should document which components are upgradeable, who approves changes, how users are notified, and how older records remain interpretable. Changes to token logic or claim rules should be versioned so that reviewers can distinguish records created under different rules.

Reproducibility and Research Ethics

A research paper describing a software prototype should provide enough detail for another researcher to understand and, where possible, reproduce the evaluation. Useful artifacts include source-code version identifiers, compiler settings, dependency versions, test descriptions, deployment network, contract addresses, transaction hashes, and anonymized test data. Credentials, private keys, confidential documents, and sensitive facility information must never be included in a public

research artifact. The paper should explain which artifacts can be shared and which are restricted.

The research process should avoid overstating the novelty or impact of the project. A prototype that records token transfers can demonstrate a technical workflow, but it cannot by itself demonstrate a reduction in greenhouse-gas emissions. Claims about lower cost, faster verification, improved trust, or increased participation should be supported by a defined baseline and empirical evidence. If only a demonstration has been completed, the results should be described as a proof of concept and its limitations stated explicitly.

If future evaluation involves interviews, usability studies, or operational data from manufacturers, the team should obtain appropriate consent and institutional approval where required. Data collection should be limited to what is necessary, and reporting should avoid exposing commercially sensitive information. Results should include failures and limitations as well as successful cases. This supports a credible evaluation and helps other researchers judge whether the findings are transferable to different facilities or market settings.

Implementation Milestones

A staged roadmap can help the project move from a demonstration to a more thoroughly evaluated prototype. The first milestone is to stabilize the smart-contract interface and document the claim and token lifecycle. The second is to complete automated tests for role permissions, issuance, transfers, and replay protection. The third is to connect the frontend to the deployed test contracts and verify that the displayed state matches confirmed transactions. The fourth is to test evidence upload and retrieval, including unavailable-file cases. The fifth is to validate the gasless transfer path and record the relayer's operational and fee behavior.

A later milestone should focus on governance and environmental methodology rather than adding more blockchain features. The project should define the unit represented by ECO, identify the relevant crediting rules, document verifier responsibilities, and establish a process for duplicate-claim detection and retirement. Only after these foundations are clear should a pilot with representative users be considered. This order reduces the risk of building a technically polished marketplace whose environmental claims remain ambiguous.

Each milestone should have an acceptance checklist and retained evidence. For example, the smart-contract milestone can include test reports and a code review; the frontend milestone can include end-to-end recordings and screenshots; the IPFS milestone can include retrieval checks; and the relayer milestone can include valid and invalid signature tests. Keeping these artifacts makes later paper revisions easier and supports precise claims about what was actually implemented.

Limitations and Future Work

This paper is based on the project description available to the author and does not constitute an independent audit of the source code or deployment. The current information does not establish accreditation, connection to an official carbon registry, completion of all gasless-transfer integrations, independent verification of emissions, or measured performance in a production environment. These limitations should remain explicit until supporting evidence is collected.

- Complete and document end-to-end tests for claim submission, review, issuance, listing, transfer, and retirement.
- Define the ECO unit, methodology for quantifying reductions, additionality requirements where applicable, baseline rules, and double-counting controls.
- Integrate an independent verifier and, if appropriate, an established registry or recognized crediting program.
- Conduct smart-contract security review, access-control testing, signature replay testing, and relayer threat modeling.
- Improve evidence persistence and confidentiality through documented retention, encryption, key management, and access policies.
- Evaluate usability and transaction costs with representative small and medium cement manufacturers.

- Assess applicable Indian carbon-market rules and other relevant jurisdictional requirements before any real-world issuance or sale.

Future work should prioritize environmental integrity and governance before marketplace scale. Technical transparency is valuable only when the underlying credit is credibly quantified, independently verified, uniquely tracked, and governed by rules that prevent duplicate claims.

VII. CONCLUSION

CarbonWise proposes a blockchain-based marketplace workflow for documenting and trading carbon-reduction units associated with small and medium cement manufacturers. Its design combines Solidity smart contracts, an ECO token, IPFS-based evidence references, wallet-connected application workflows, and an explored EIP-712 relayer approach for gasless transfers. The architecture can make token issuance and movement easier to trace, while separating evidence storage from on-chain transaction records. However, blockchain immutability alone cannot prove that an emissions reduction occurred. Credible MRV, independent authorization, careful unit definition, double-counting prevention, security testing, and regulatory alignment are prerequisites for real-world use. The next step is to validate the complete prototype through reproducible tests and report only measured outcomes. This makes CarbonWise a promising implementation project whose claims can be strengthened through transparent evaluation rather than unsupported performance assertions.

Acknowledgment

The author may acknowledge project mentors, teammates, institutional support, and open-source software maintainers here. Names and contributions should be added only with the relevant individuals' consent and after confirming the exact wording.

REFERENCES

1. World Bank, "Digital Monitoring, Reporting, and Verification Systems and Their Application in Future Carbon Markets," Public-Private Partnership Resource Center, Jun. 2022. [Online]. Available: https://ppp.worldbank.org/library/digital-monitoring-reporting-and-verification-systems-and-their-application-future-carbon-markets
2. World Bank, "State and Trends of Carbon Pricing 2024," Washington, DC, 2024. [Online]. Available: https://www.worldbank.org/en/programs/pricing-carbon/reports
3. R. Bloemen, L. Logvinov, and J. Evans, "EIP-712: Typed structured data hashing and signing," Ethereum Improvement Proposals, EIP-712, 2017. [Online]. Available: https://eips.ethereum.org/EIPS/eip-712
4. J. Benet, "IPFS—Content Addressed, Versioned, P2P File System," arXiv:1407.3561, 2014. [Online]. Available: https://arxiv.org/abs/1407.3561
5. Ethereum Foundation, "Sepolia," Ethereum.org Documentation. [Online]. Available: https://ethereum.org/en/developers/docs/networks/
6. World Bank, "State and Trends of Carbon Pricing 2025," Washington, DC, 2025. [Online]. Available: https://www.worldbank.org/en/news/press-release/2025/06/10/global-carbon-pricing-mobilizes-over-100-billion-for-public-budgets