ATTESTIFY

A Blockchain-Based Credential Verification System

Developed By

M. Saad Feroz	**UOC-CS-F2022-034**
Zain Ul Abidin	**UOC-CS-F2022-036**

Supervised By

Mr. Faizan Saleem

Department of Computer Science and Information Technology
Faculty of Computing and Information Technology

The University of Chakwal

Bachelor of Science in Computer Science (2022-2026)

Table of Contents

ATTESTIFY

A Blockchain-Based Credential Verification System

CHAPTER 1

INTRODUCTION

1.1 Introduction

In the digital age, the authenticity of academic and professional credentials has become increasingly difficult to maintain. **Attestify** is a cutting-edge, decentralized application (DApp) designed to revolutionize the way these credentials are issued, stored, and verified. By leveraging the power of Ethereum blockchain technology, Attestify removes the need for centralized intermediaries and manual background checks.

The platform ensures that every credential issued is tamper-proof, immutable, and globally verifiable. Institutions can issue digital certificates that are cryptographically anchored to the blockchain, while third parties (such as potential employers or other academic bodies) can verify their authenticity instantly without needing to contact the original issuer. This creates a "trustless" environment where the validity of a document is guaranteed by mathematics and code rather than institutional reputation alone.

1.2 Scope

The scope of this project involves the design and development of a full-stack, Web3-integrated application. The system caters to three primary stakeholders:

1. **Admins (Issuers):** University registrars or HR departments who have the authority to generate and sign digital certificates.
2. **Students (Holders):** Individuals who receive, store, and present their credentials to third parties.
3. **Verifiers:** External organizations (employers, licensing boards) that need to validate a presented credential.

The lifecycle covered includes:

- **Cryptographic Hashing:** Transforming document data into a unique SHA-256 fingerprint.
- **Decentralized Storage:** Using IPFS (InterPlanetary File System) to store the actual certificate files, ensuring they are not dependent on a single server. [1]
- **On-Chain Recording:** Storing the hash and IPFS CID on the Ethereum Sepolia testnet. [2]
- **Public Verification Portal:** A low-barrier entry point for anyone to verify a file via drag-and-drop.

1.3 Purpose

The fundamental purpose of Attestify is to eliminate **credential fraud**, a multi-billion dollar global issue. Traditional verification is plagued by "degree mills," sophisticated PDF forgeries, and slow, manual administrative processes that can take weeks. Attestify aims to provide an infrastructure where a document's integrity is verified in seconds, reducing administrative overhead and providing graduates with lifelong, self-sovereign control over their achievements.

1.4 Objective

The key objectives of the project are:

- To eliminate counterfeit academic degrees and professional certificates.
- To provide an instant, real-time verification mechanism available 24/7.
- To reduce the administrative burden and costs associated with manual verification for institutions.
- To give individuals ownership and control over their digital credentials.
- To demonstrate the practical application of blockchain technology in solving real-world data integrity problems.

1.5 Tools and Technologies

Frontend:

- **React + Vite:** For building a fast, interactive user interface.
- **TailwindCSS:** For modern, responsive styling.
- **Ethers.js:** For interacting with the Ethereum blockchain and smart contracts. [3]
- **React Router & Axios:** For navigation and API communication.

Backend:

- **Node.js & Express.js:** Server runtime and web framework.
- **MongoDB & Mongoose:** NoSQL database for managing user profiles and off-chain metadata. [4]
- **Nodemailer & Multer:** For email notifications and file upload handling.

Blockchain & Smart Contracts:

- **Solidity:** Language for writing smart contracts.
- **Hardhat:** Environment for compiling, deploying, and testing. [5]
- **Ethereum (Sepolia Testnet):** The underlying blockchain network. [6]
- **OpenZeppelin:** Standard libraries for secure contract development. [7]

CHAPTER 2

LITERATURE REVIEW & EXISTING SYSTEMS

2.1 Literature Review

Research into decentralized identity and credentialing highlights a shift from "Siloed Identity" (centralized) to "Self-Sovereign Identity" (SSI).

- **Blockcerts (MIT Media Lab):** The first major open standard. While it proved the concept, it faced hurdles with Bitcoin's high transaction fees and a complex user onboarding process. [8] [9]
- **OpenCerts (Singapore):** A highly successful national implementation. However, it is largely centralized around the Singaporean government's specific infrastructure, making it less accessible for private global institutions. [10]
- **Credly/Badgr:** These represent the current "Gold Standard" in Web2. While user-friendly, they are centralized. If the company closes, the "trust" and the links to certificates disappear. [11]

2.2 Problem Statement

Current verification methods are:

- **Vulnerable:** Paper and digital PDFs are easily forged using standard design tools.
- **Slow:** Background checks often require manual emails and phone calls.
- **Fragmented:** There is no universal, "trustless" way to verify degrees from different countries or institutions.

Attestify solves this by providing a decentralized registry that is globally accessible and cryptographically secure. For the developer, this project serves as a deep dive into the intersection of traditional backend (Web2) and decentralized blockchain (Web3) architectures. [12]

2.3 Proposed Solution

The solution is a multi-tier DApp that utilizes the **Ethereum Blockchain** as a permanent, public registry.

1. **Hashing Mechanism:** The system does not store personal data on the blockchain. Instead, it stores a **SHA-256 hash**. This preserves privacy while ensuring integrity. [13]

2. **Verification Flow:** A verifier does not need an account. They simply upload the file to the portal; the system calculates the hash locally and checks if that exact fingerprint exists on the blockchain.

2.4 Comparative Analysis

Feature	**Manual/Centralized**	**Proposed Solution (Attestify)**
Data Storage	Centralized Databases	Decentralized Blockchain
Verification Speed	Slow (Days/Weeks)	Instant (Real-time)
Cost	High (Admin fees)	Low (Minimal gas fees)
Security	Low (Forgery risk)	High (Cryptographic proof)
Single Point of Failure	Yes	No

CHAPTER 3

SOFTWARE REQUIREMENTS SPECIFICATIONS

3.1 Functional Requirements

3.1.1 User Management and Authentication

1. **Multi-Role Authentication:** The system shall support distinct login portals for Admins (Issuers) and Students (Holders).
2. **Web3 Wallet Integration:** Admins must be able to connect and authenticate using MetaMask to sign blockchain transactions.
3. **Session Management:** The system shall implement JWT (JSON Web Tokens) for secure off-chain session persistence.

3.1.2 Credential Issuance (Admin Flow)

1. **Digital Certificate Generation:** Admins shall be able to generate PDF certificates by inputting student data (Name, Degree, Graduation Date).
2. **Automated Hashing:** Upon generation, the system shall automatically calculate a SHA-256 hash of the certificate file.
3. **Decentralized Storage (IPFS):** The system shall upload the certificate to IPFS and retrieve a unique Content Identifier (CID).
4. **Blockchain Registration:** The system shall trigger a smart contract transaction to record the student ID, file hash, and IPFS CID on the Ethereum ledger.

3.1.3 Student Features (Holder Flow)

1. **Personal Repository:** Students shall have a dashboard to view all credentials issued to their unique Student ID.
2. **Credential Export:** Students shall be able to download the original PDF certificate and share the IPFS link with third parties.
3. **Verification Status:** Students shall see the real-time status (Valid/Revoked) of their certificates as recorded on the blockchain.

3.1.4 Public Verification (Verifier Flow)

1. **File-Based Verification:** Verifiers shall be able to drag and drop a certificate PDF into a public portal.
2. **Blockchain Querying:** The system shall compute the hash of the uploaded file and check for a match on the blockchain without requiring the verifier to pay gas fees.
3. **Revocation Check:** The system shall explicitly state if a matched hash exists but has been

marked as "Revoked" by the issuer.

3.1.5 Administrative Control

1. **Revocation Logic:** Authorized Admins shall have the functional capability to invalidate a credential on-chain in cases of error or academic fraud.
2. **Audit Trails:** The system shall log all issuance and revocation actions in the off-chain database for institutional tracking.

3.2 Non-Functional Requirements

3.2.1 Security and Integrity

1. **Smart Contract Security:** The code shall be protected against common vulnerabilities such as Reentrancy, Integer Overflow, and Unauthorized Access using OpenZeppelin standards.
2. **Data Privacy:** Personal Identifiable Information (PII) shall never be stored on the public blockchain; only cryptographic hashes shall be recorded.
3. **Immutability:** Once a record is finalized on the blockchain, it shall be impossible for any party (including the developer) to delete the record.

3.2.2 Performance and Scalability

1. **Response Time:** Verification results shall be returned within 3 seconds of file upload under normal network conditions.
2. **Gas Efficiency:** The smart contract shall use optimized data types (e.g., bytes32) to minimize transaction costs for the issuer.
3. **Concurrent Users:** The backend API shall be capable of handling up to 100 concurrent requests during peak graduation periods.

3.2.3 Availability and Reliability

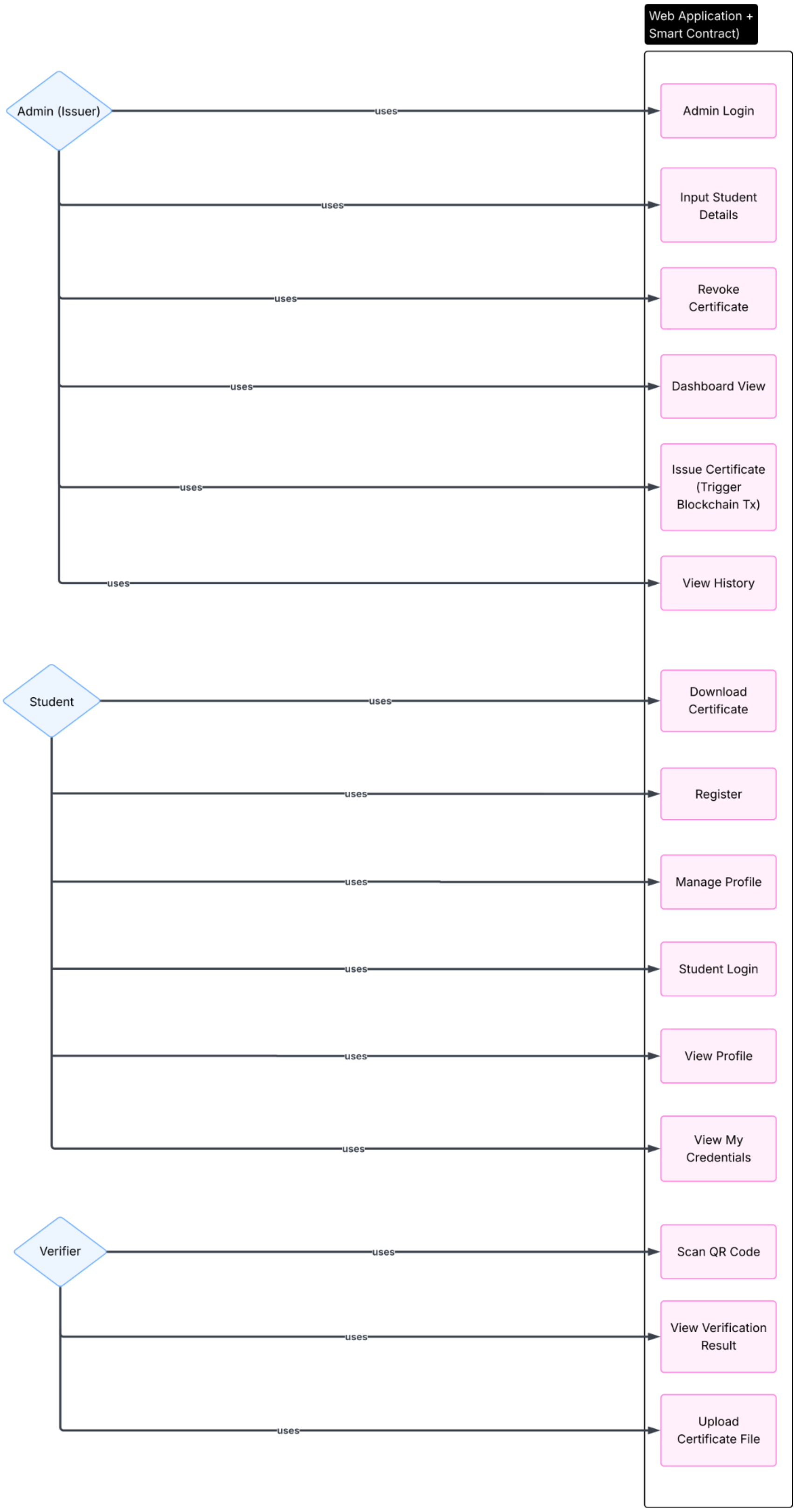
1. **Uptime:** The web interface shall maintain 99.9% availability.
2. **Data Persistence:** Records on the Ethereum blockchain shall be considered 100% available and permanent.
3. **Failover:** The system shall provide clear error messaging if the IPFS gateway or Blockchain RPC provider is temporarily unreachable.

3.2.4 Usability and Accessibility

1. **Responsive Design:** The interface shall be fully functional across desktop, tablet, and mobile browsers using TailwindCSS.

2. **Zero-Gas Verification:** Verification shall remain a "Read-only" operation, ensuring that verifiers do not need a wallet or cryptocurrency to use the system.
3. **User Feedback:** The system shall provide visual indicators (loaders, success/error toasts) for all asynchronous blockchain transactions.

3.3 Use Case Diagram



3.4 Methodology: Agile Development

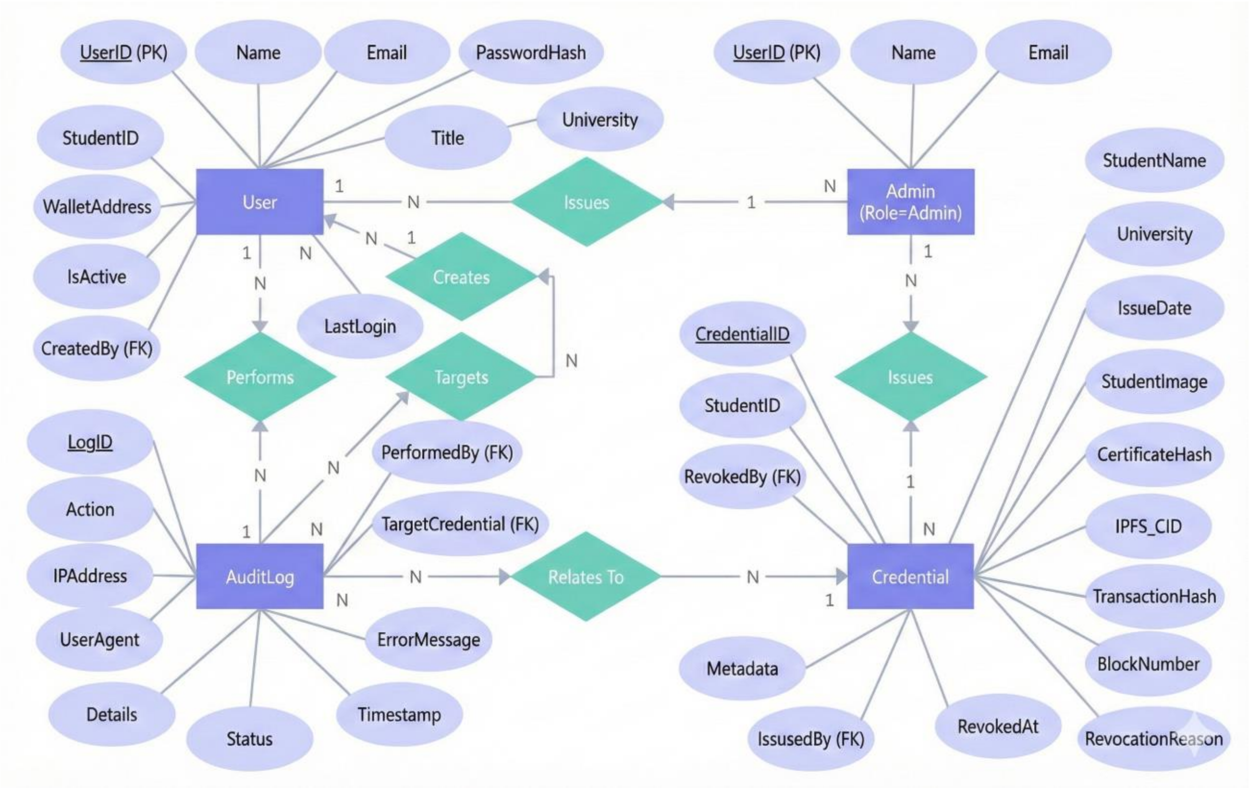
For the Attestify project, the **Agile Methodology** was chosen to manage the inherent complexities of Web3 and DApp development. Unlike traditional waterfall models, Agile facilitated an iterative and flexible workflow that was essential for the following reasons:

1. **Iterative Blockchain Development:** Smart contract logic is immutable once deployed. Agile allowed us to develop the core logic in small increments, performing rigorous testing on local networks (Hardhat) and testnets (Sepolia) before moving to the next feature set.
2. **Rapid Prototyping and Feedback:** The integration between Web2 (Express/Node) and Web3 (Ethers.js/Solidity) required frequent adjustments. Agile Sprints enabled us to build functional prototypes of the verification portal and student dashboard early, allowing for immediate UX/UI refinements.
3. **Risk Mitigation:** Blockchain development is highly susceptible to security vulnerabilities. Continuous integration and testing (CI/CD) within the Agile framework ensured that security patches and gas optimizations were implemented during the development cycle rather than at the very end.
4. **Flexibility to Tech Changes:** Since the DApp landscape evolves rapidly (e.g., changes in RPC providers or gas fee structures), the Agile approach allowed the team to pivot and adapt tools without disrupting the entire project timeline.

CHAPTER 4

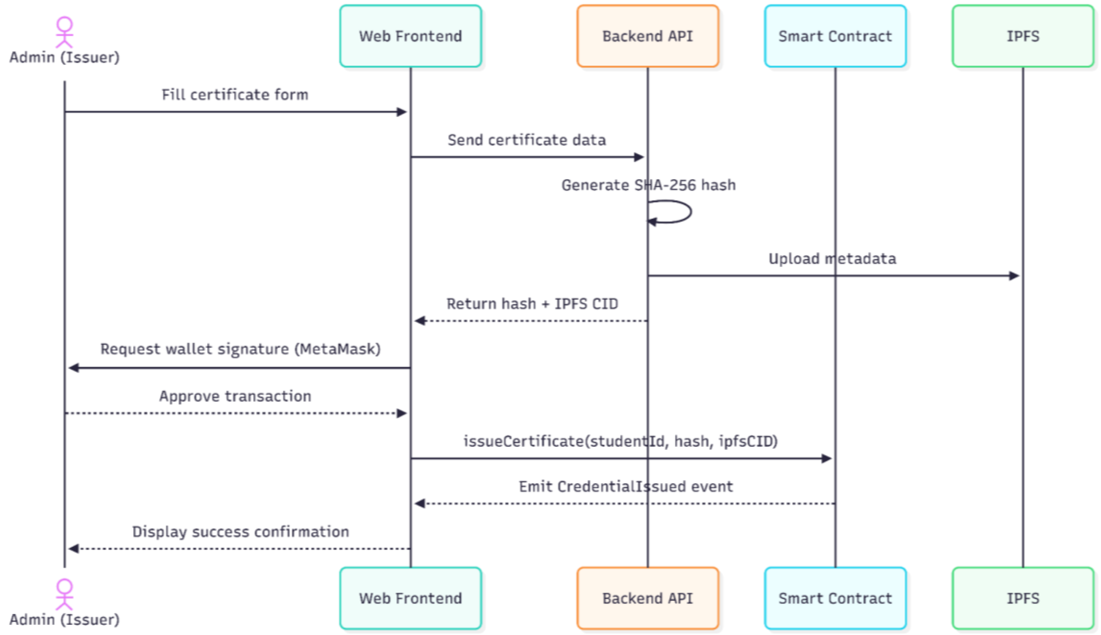
DESIGN AND ARCHITECTUR

4.1 Entity Relationship Diagram (ERD)

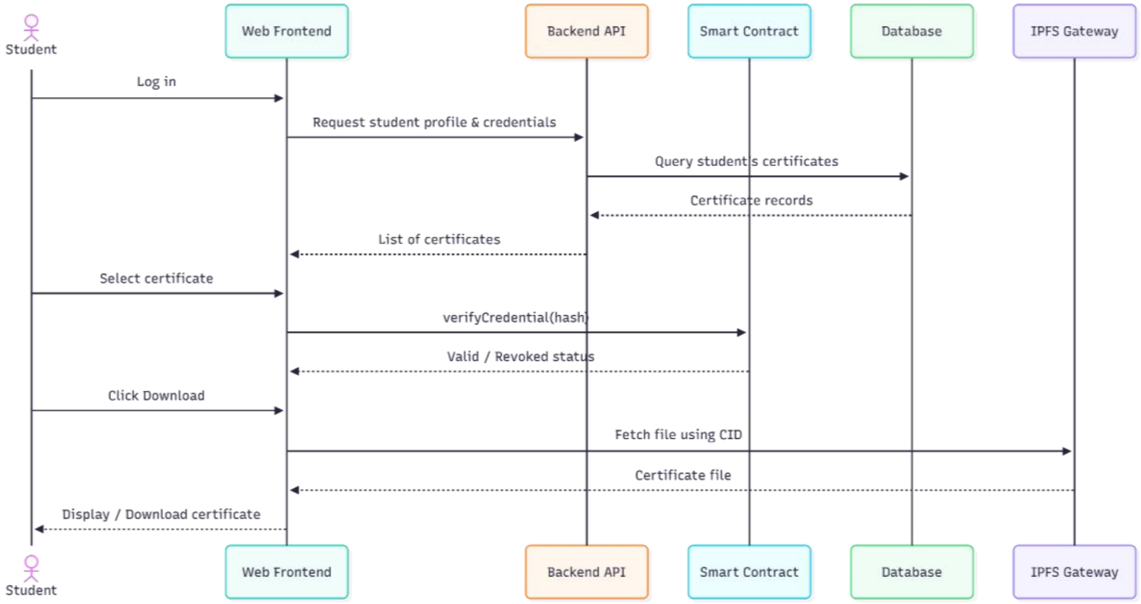


4.2 Sequence Diagrams

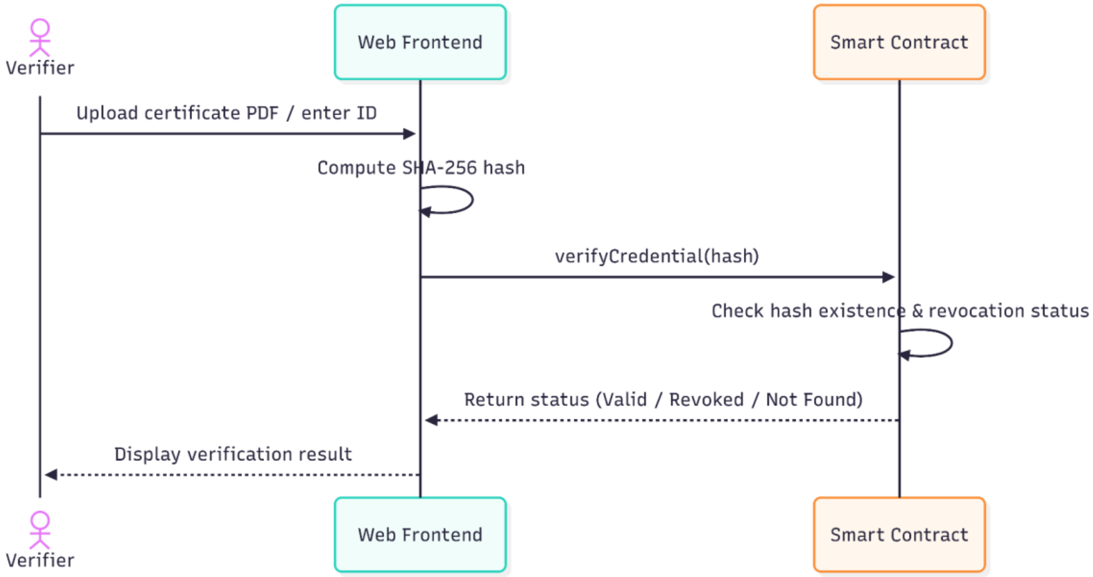
Issuer Sequence:



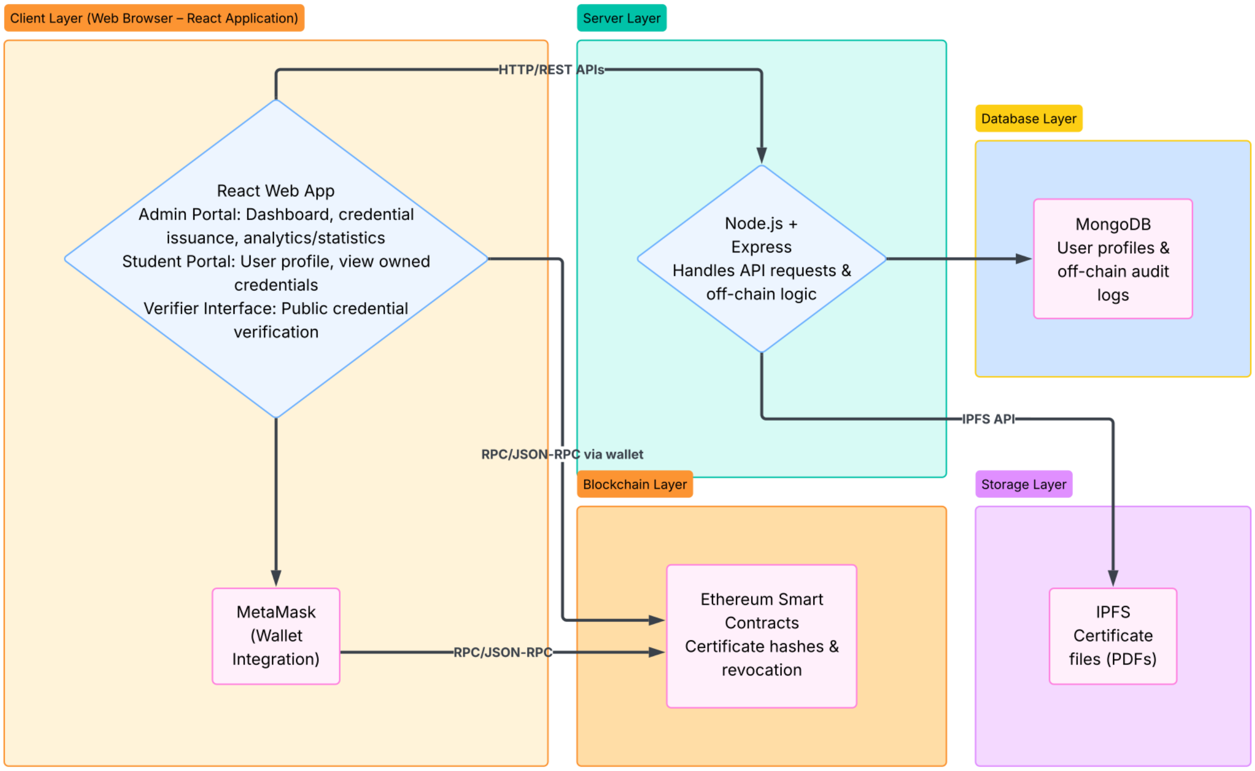
Student Sequence:



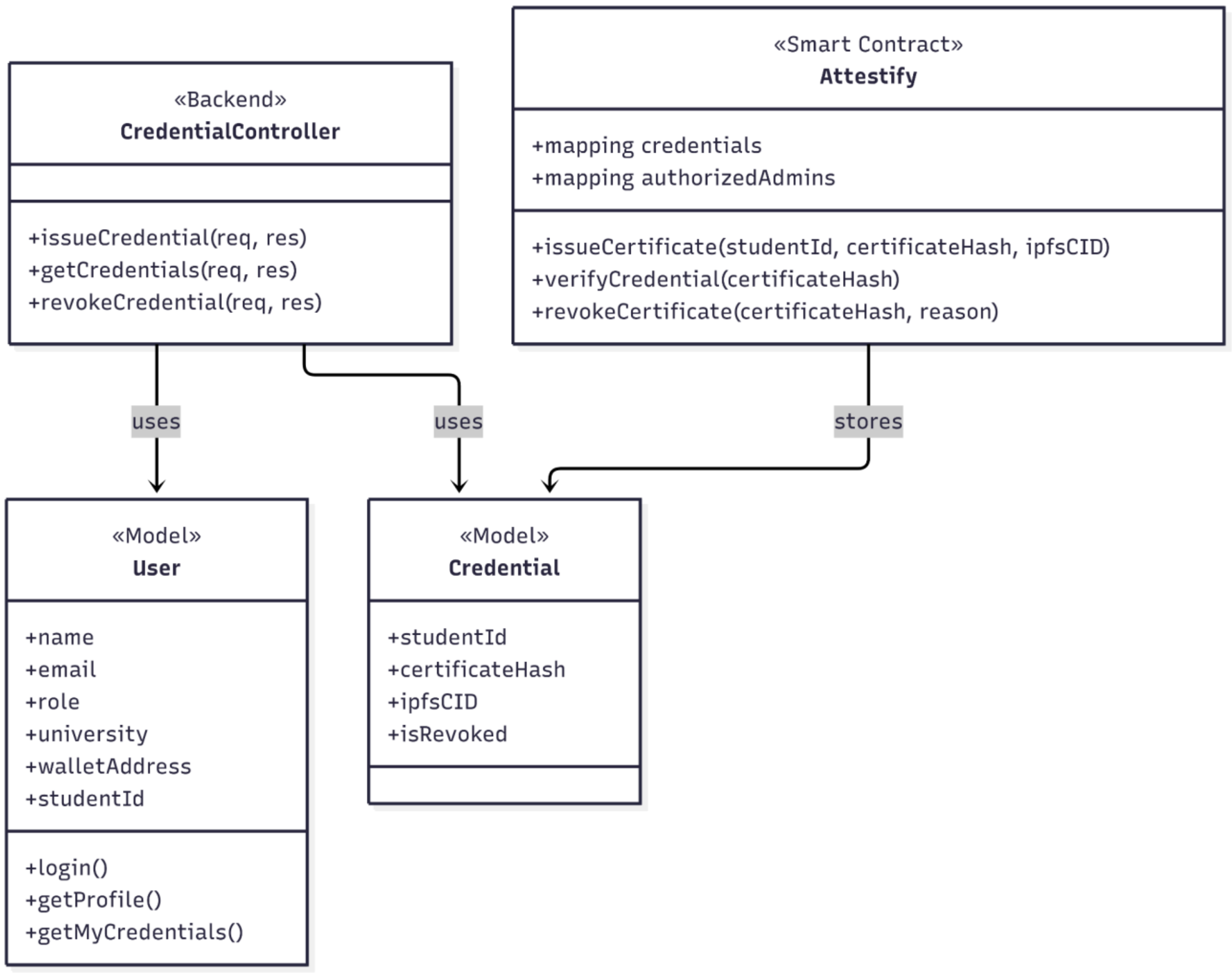
Verifier Sequence:


Verifier
Web Frontend
Smart Contract
Upload certificate PDF / enter ID
Compute SHA-256 hash
verifyCredential(hash)
Check hash existence & revocation status
Return status (Valid / Revoked / Not Found)
Display verification result
Verifier
Web Frontend
Smart Contract

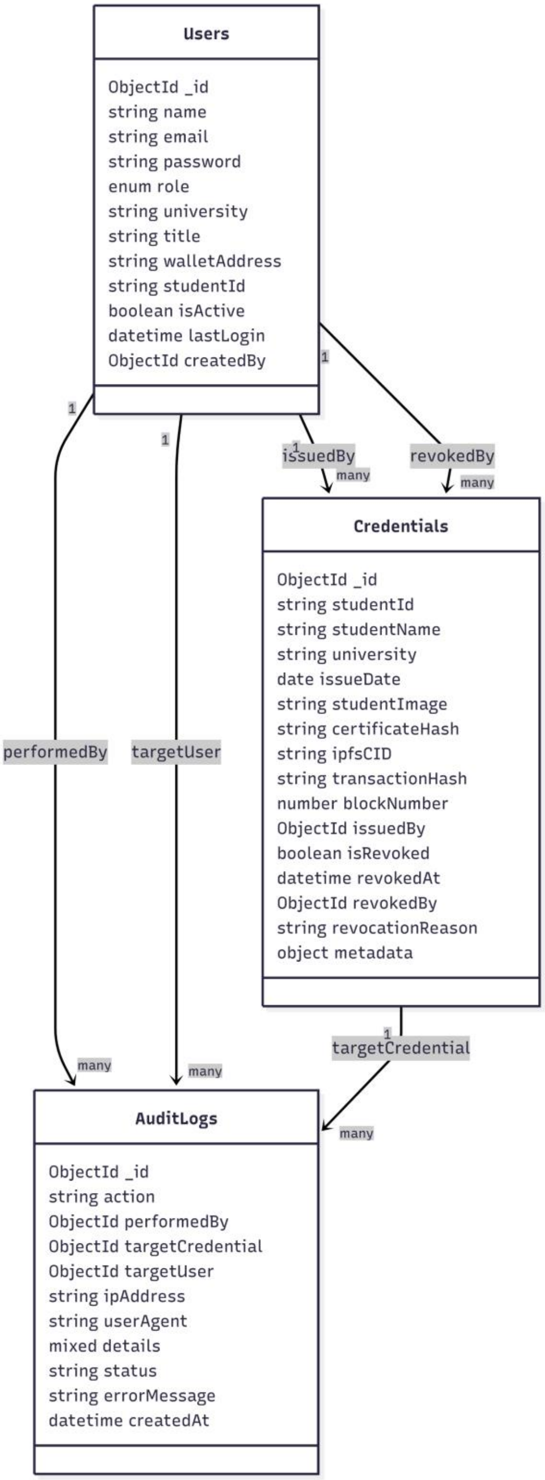
4.3 Architecture Design


Client Layer (Web Browser – React Application)
Server Layer
Database Layer
HTTP/REST APIs
React Web App
Admin Portal: Dashboard, credential issuance, analytics/statistics
Student Portal: User profile, view owned credentials
Verifier Interface: Public credential verification
Node.js + Express
Handles API requests & off-chain logic
MongoDB
User profiles & off-chain audit logs
IPFS API
RPC/JSON-RPC via wallet
Blockchain Layer
Storage Layer
MetaMask (Wallet Integration)
RPC/JSON-RPC
Ethereum Smart Contracts
Certificate hashes & revocation
IPFS
Certificate files (PDFs)

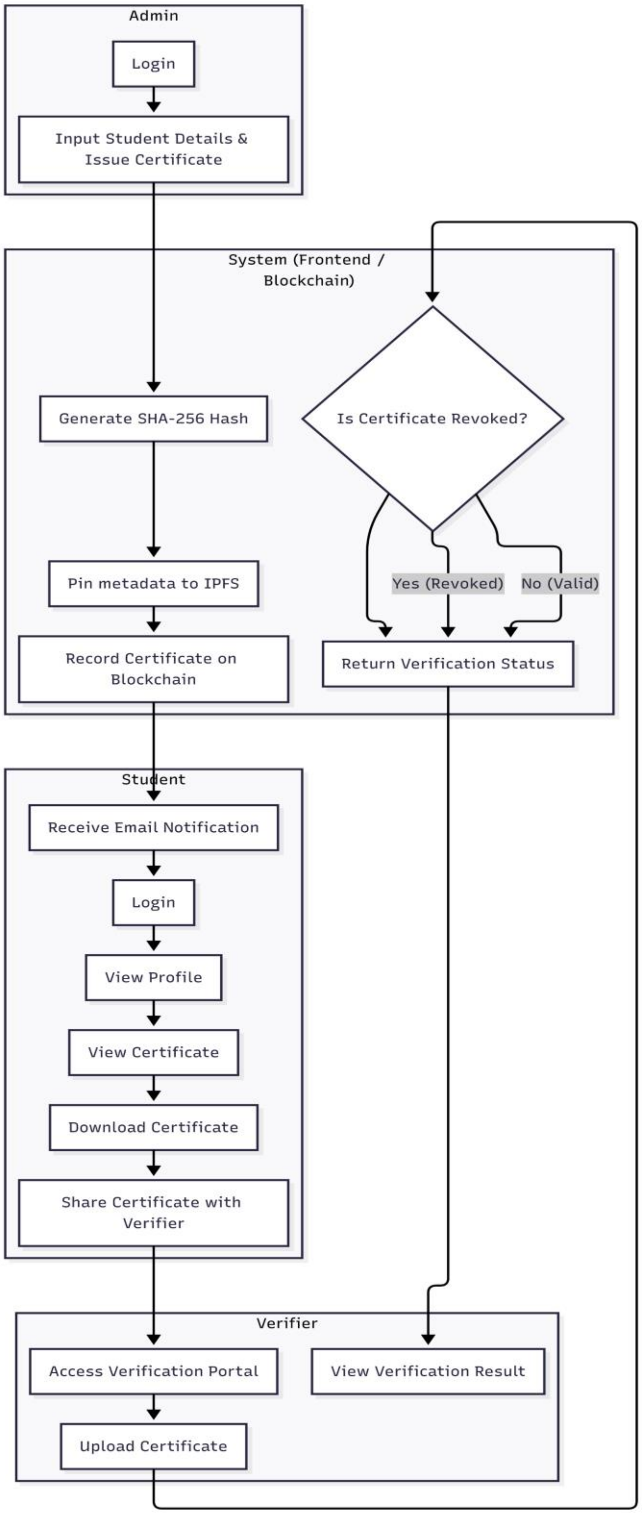
4.4 Class Diagram



4.5 Database Design (NoSQL)

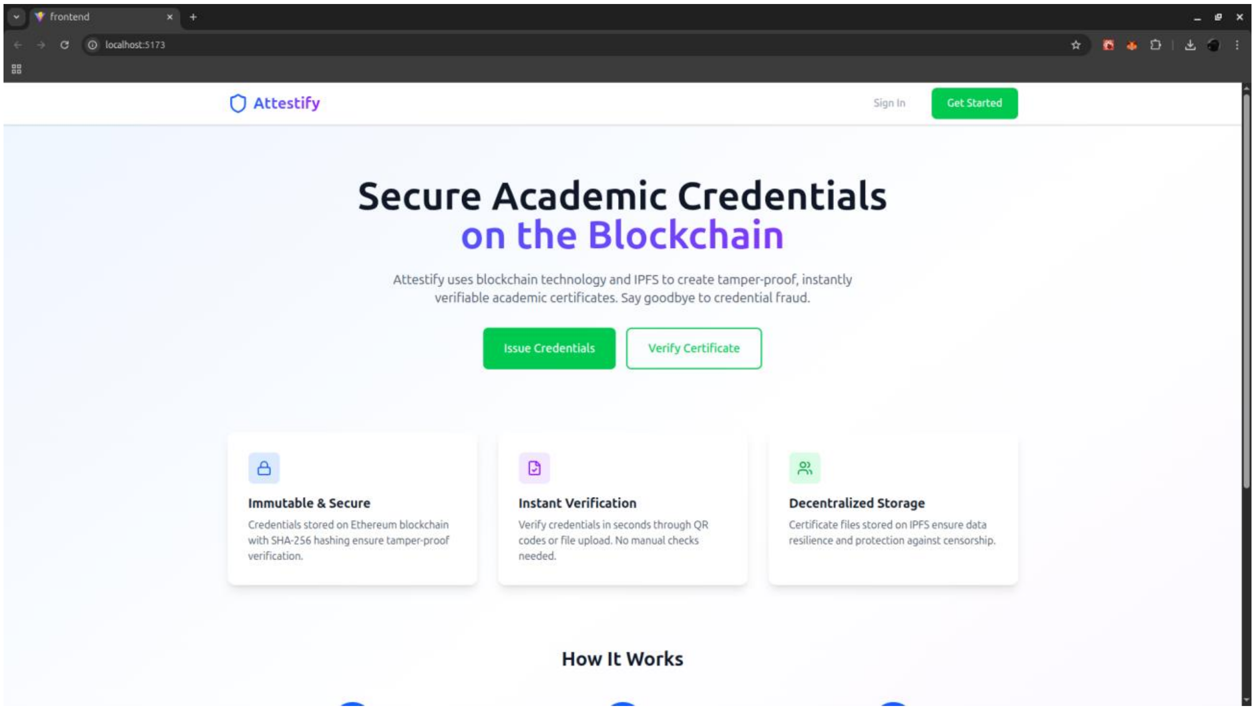


4.6 Activity Diagram


Admin
Login
Input Student Details & Issue Certificate
System (Frontend / Blockchain)
Generate SHA-256 Hash
Is Certificate Revoked?
Pin metadata to IPFS
Yes (Revoked)
No (Valid)
Record Certificate on Blockchain
Return Verification Status
Student
Receive Email Notification
Login
View Profile
View Certificate
Download Certificate
Share Certificate with Verifier
Verifier
Access Verification Portal
View Verification Result
Upload Certificate

CHAPTER 5

Interface Design



Secure Academic Credentials on the Blockchain

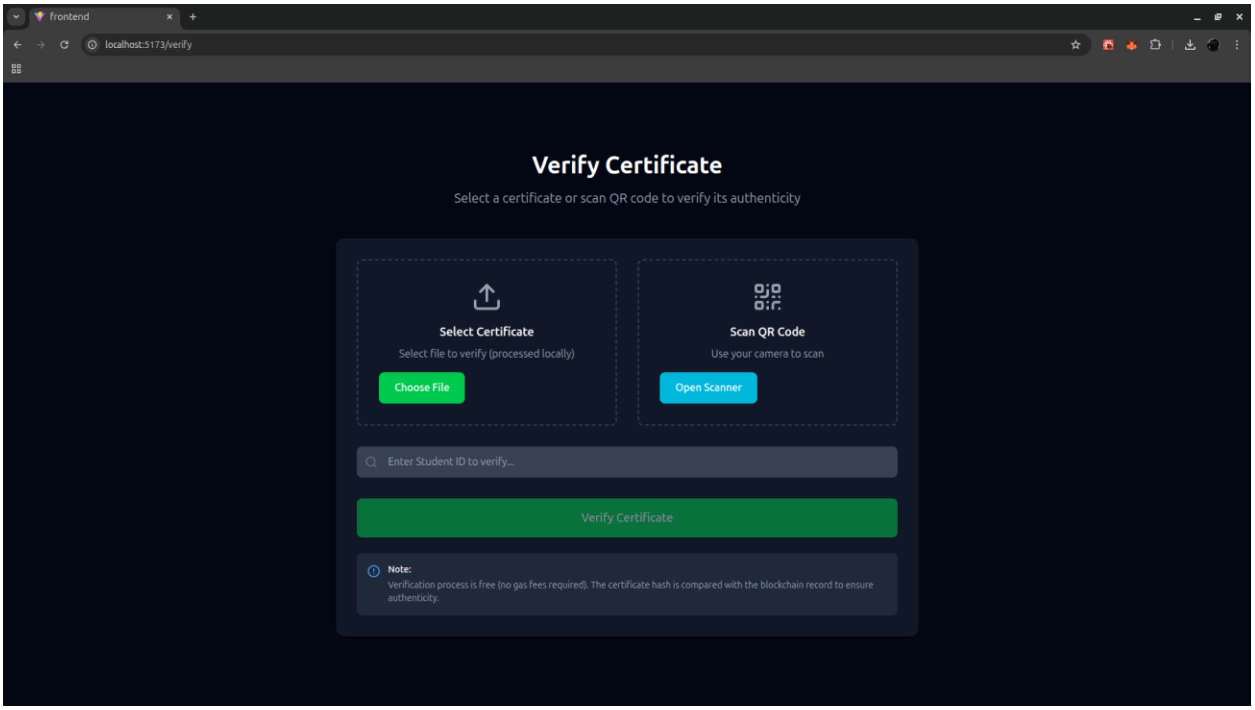
Attestify uses blockchain technology and IPFS to create tamper-proof, instantly verifiable academic certificates. Say goodbye to credential fraud.

Immutable & Secure
Credentials stored on Ethereum blockchain with SHA-256 hashing ensure tamper-proof verification.

Instant Verification
Verify credentials in seconds through QR codes or file upload. No manual checks needed.

Decentralized Storage
Certificate files stored on IPFS ensure data resilience and protection against censorship.

How It Works



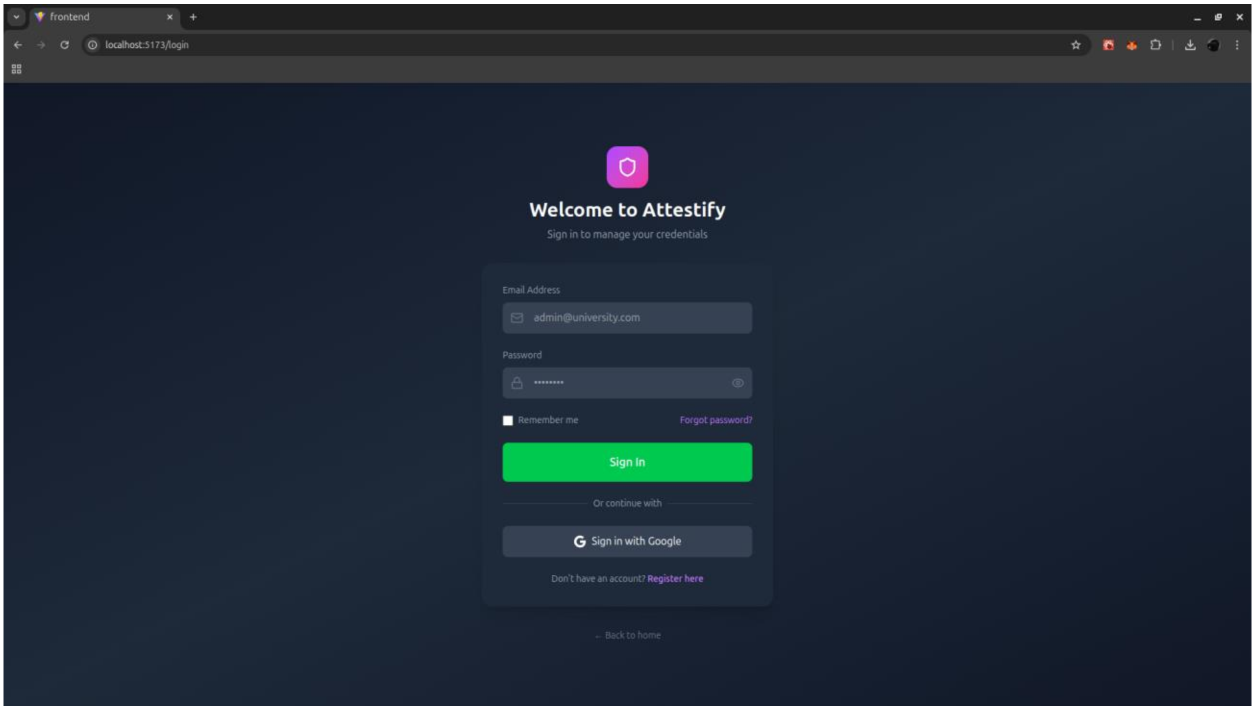
Verify Certificate

Select a certificate or scan QR code to verify its authenticity

Select Certificate
Select file to verify (processed locally)

Scan QR Code
Use your camera to scan

Note:
Verification process is free (no gas fees required). The certificate hash is compared with the blockchain record to ensure authenticity.



frontend

localhost:5173/login

Welcome to Attestify

Sign in to manage your credentials

Email Address

admin@university.com

Password

••••••••

□ Remember me

Forgot password?

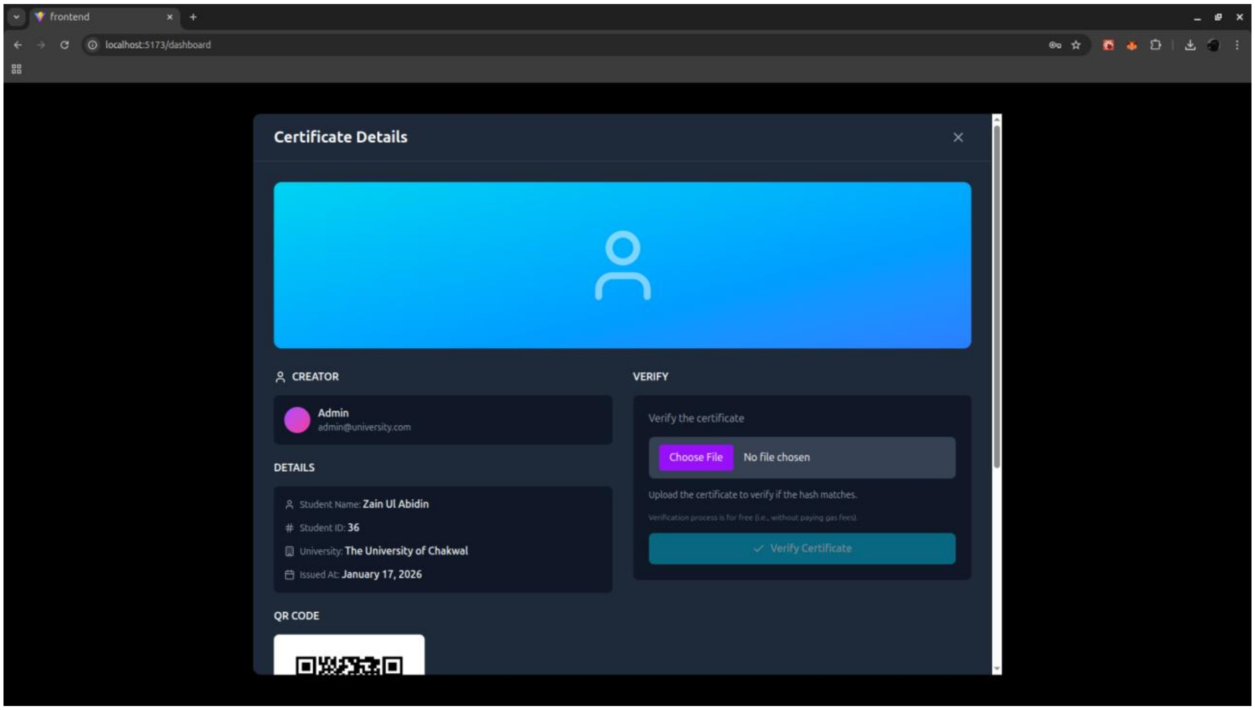
Sign In

Or continue with

Sign in with Google

Don't have an account? Register here

← Back to home



frontend

localhost:5173/dashboard

Certificate Details

CREATOR

Admin
admin@university.com

DETAILS

Student Name: Zain Ul Abidin

Student ID: 36

University: The University of Chakwal

Issued At: January 17, 2026

QR CODE

VERIFY

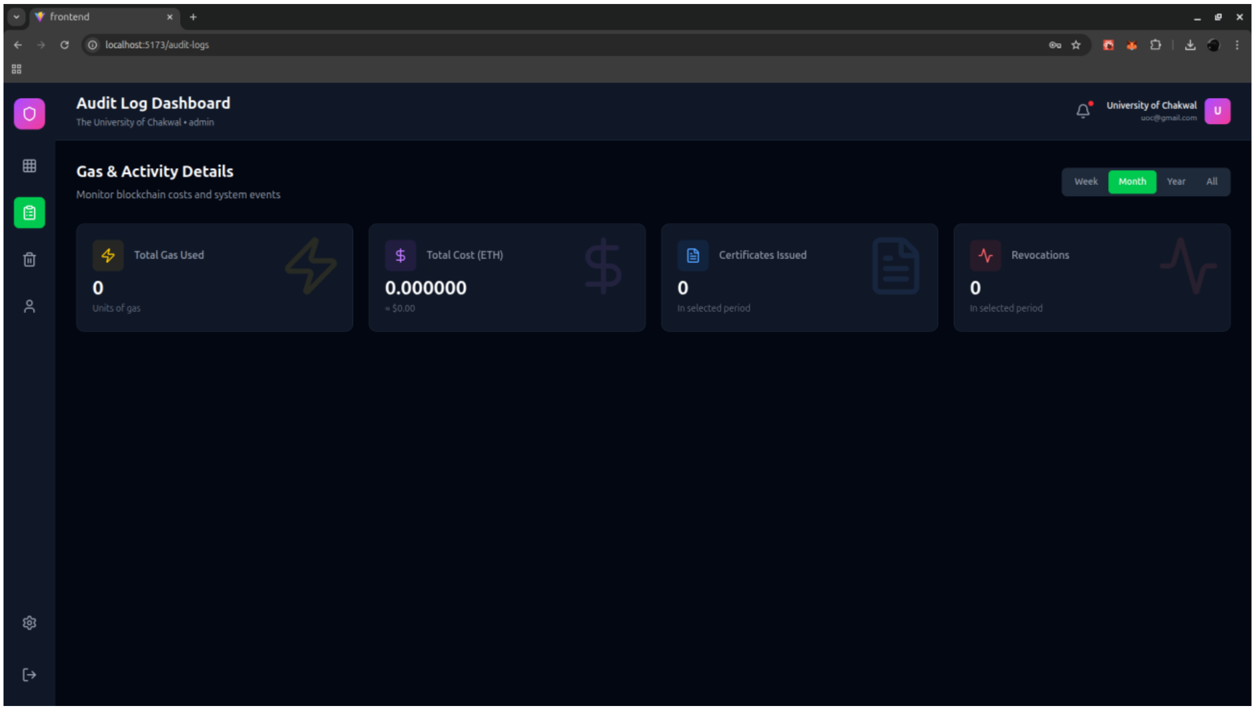
Verify the certificate

Choose File No file chosen

Upload the certificate to verify if the hash matches.

Verification process is for free (i.e., without paying gas fees).

✓ Verify Certificate



Audit Log Dashboard

The University of Chakwal • admin

Gas & Activity Details

Monitor blockchain costs and system events

Total Gas Used

0

Units of gas

Total Cost (ETH)

0.000000

≈ $0.00

Certificates Issued

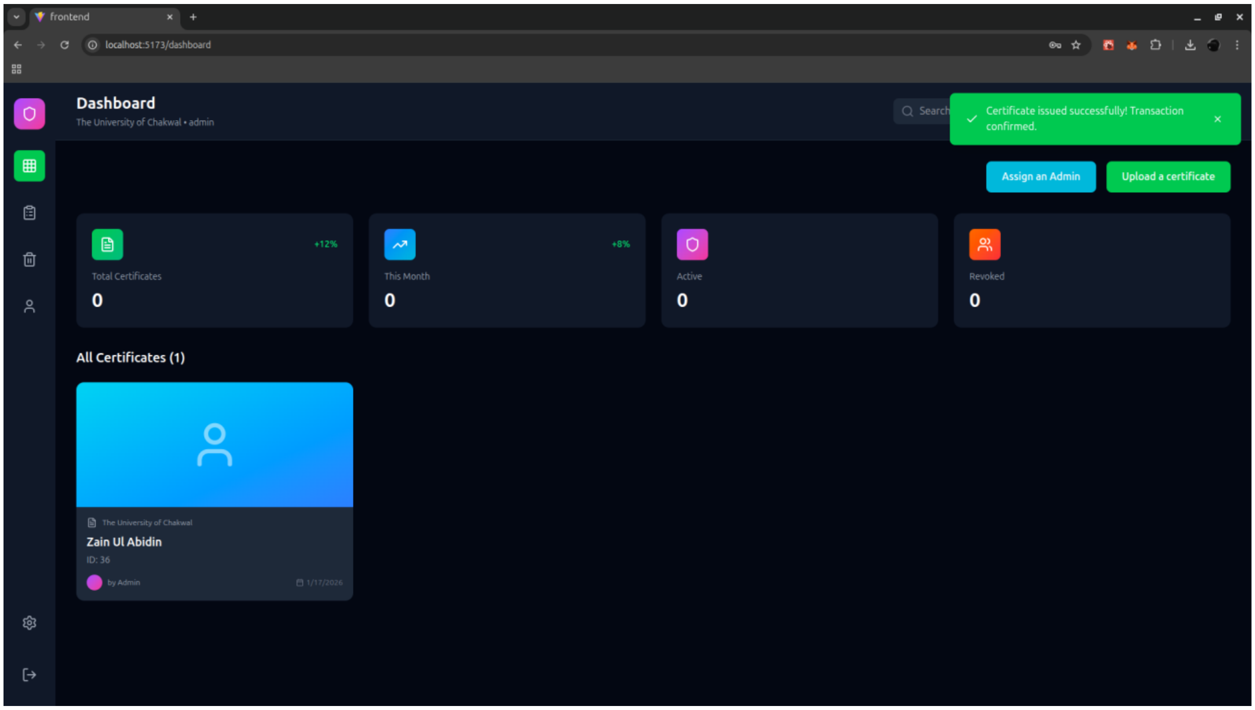
0

In selected period

Revocations

0

In selected period



Dashboard

The University of Chakwal • admin

Certificate issued successfully! Transaction confirmed.

+12%

Total Certificates

0

+8%

This Month

0

Active

0

Revoked

0

All Certificates (1)

The University of Chakwal

Zain Ul Abidin

ID: 36

by Admin

1/17/2026

References

1. **Benet, J. (2014).** *IPFS - Content Addressed, Versioned, P2P File System.* https://ipfs.tech/ (Technical specifications for decentralized file storage).
2. **Buterin, V. (2014).** *A Next-Generation Smart Contract and Decentralized Application Platform.* Ethereum Whitepaper. https://ethereum.org/en/whitepaper/
3. **Ethers.js Documentation. (2023).** *Interacting with the Ethereum Blockchain.* https://docs.ethers.org/ (Guidelines for client-side blockchain interaction).
4. **MongoDB, Inc. (2024).** *Mongoose: Elegant MongoDB Object Modeling for Node.js.* https://mongoosejs.com/docs/ (Guidelines for off-chain metadata schema design).
5. **Antonopoulos, A. M., & Wood, G. (2018).** *Mastering Ethereum: Building Smart Contracts and DApps.* O'Reilly Media. (Foundational concepts for DApp architecture and Solidity development).
6. **Hardhat Documentation. (2024).** *Ethereum Development Environment.* https://hardhat.org/docs (Best practices for testing and deploying on the Sepolia testnet).
7. **OpenZeppelin. (2024).** *OpenZeppelin Contracts Documentation.* https://docs.openzeppelin.com/contracts/ (Security patterns for Ownable and AccessControl modules used in the Attestify smart contract).
8. **Schmidt, P., & Reed, K. (2016).** *Digital Certificates: An Open Standard for Academic and Professional Credentials on the Blockchain.* MIT Media Lab Report. https://www.blockcerts.org/
9. **Nakamoto, S. (2008).** *Bitcoin: A Peer-to-Peer Electronic Cash System.* https://bitcoin.org/bitcoin.pdf (Origin of the distributed ledger technology utilized in this project).
10. **Government Technology Agency of Singapore. (2019).** *OpenCerts: A Blockchain-based Platform for Verifiable Certificates.* https://opencerts.io/ (Framework for region-specific educational credential verification).
11. **Credly. (2023).** *The Digital Credentialing Ecosystem: Modernizing Certification Management.* https://info.credly.com/ (Centralized badge and digital credential standards).
12. **W3C. (2022).** *Decentralized Identifiers (DIDs) v1.0.* https://www.w3.org/TR/did-core/ (Standard for self-sovereign identity referenced in the Future Work section).
13. **Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H. (2017).** *An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends.* IEEE 6th International Congress on Big Data.