Decentralized Student Result Storage and verification System using Blockchain and Cloud Computing

Problem Statement:

Educational institutions frequently encounter challenges in maintaining secure, transparent, and easily accessible repositories for student academic records. Traditional systems often present vulnerabilities regarding data integrity, lack efficient mechanisms for student access and management of their documents, and can be ineffective for administrators responsible for uploading and organizing these files. This can lead to concerns about the authenticity and immutability of academic achievements, as well as inefficiencies in administrative workflows related to record management.

Proposed Solution:

To address these limitations, this project proposes the development of a decentralized Student Result Storage System. This innovative solution leverages the security and immutability of blockchain technology, specifically the Ethereum platform, integrated with the scalability and accessibility of cloud computing services. The system will establish distinct user roles for administrators, who will securely upload and manage student records, and for students, who will have secure view-only access with the added convenience of automated downloads to their linked Google Drive accounts. Furthermore, the implementation of an intelligent file suggestion feature during the administrator upload process will streamline data entry and improve overall efficiency. This approach aims to enhance the security, transparency, and accessibility of student academic records while optimizing administrative processes.

The Problem: Data Loss & Verification Gaps

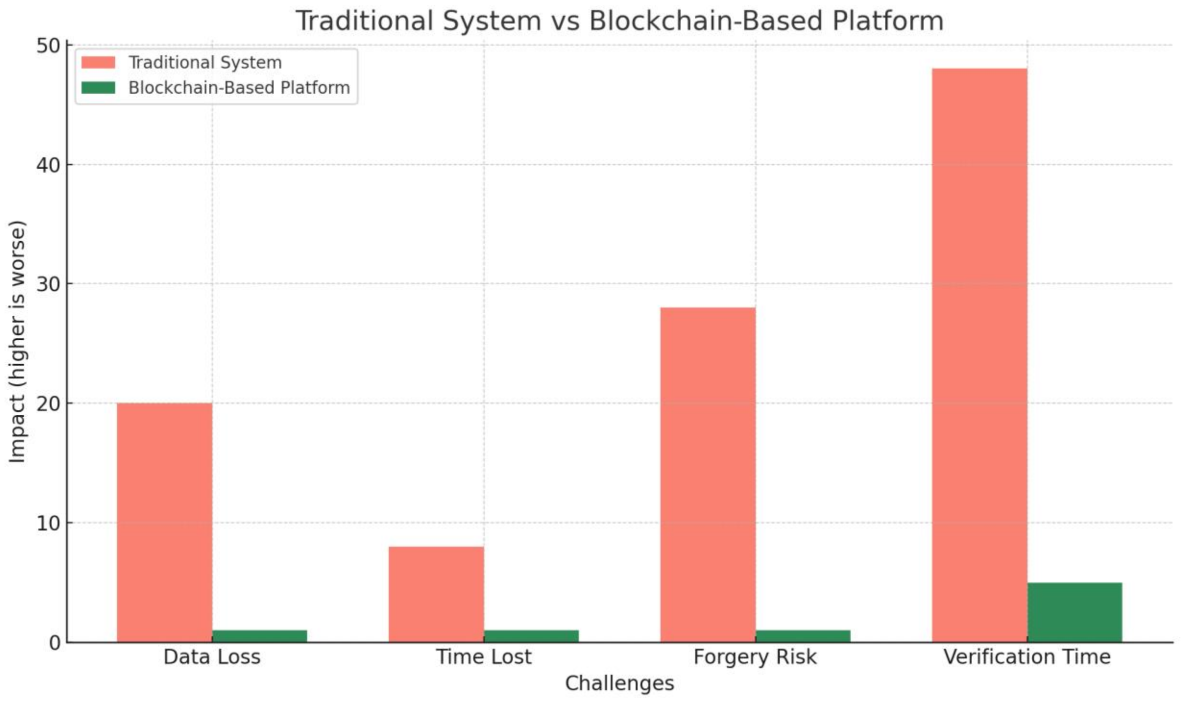
Estimated Academic Data Loss & Access Issues

- 10–20% of students report difficulties retrieving key academic documents (transcripts, degrees, mark sheets) during critical times like job applications or admissions.
- In India alone, with over 3.5 million graduates annually, this means 350,000–700,000 students may face document loss or inaccessibility each year.
- Average time lost per student: 3–10 hours searching for or requesting duplicate documents.

Certificate Fraud & Verification Failures

- 28% of job applicants in India submit fake or unverifiable degrees (AuthBridge 2020).

- Massive forged certificate rackets have exposed over 50,000 fake degrees sold via fake university websites (Times of India).

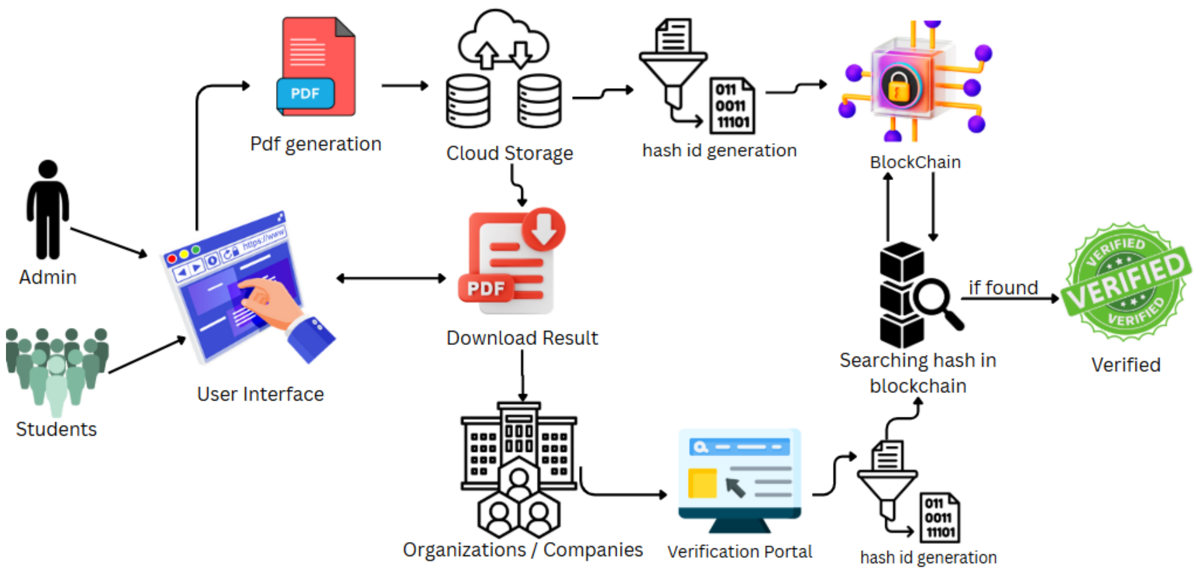


- Institutions and employers struggle to verify documents quickly and reliably.

Our Solution: Blockchain-Powered Academic Record Platform

- Key Benefits Delivered by the Platform
- 100% data integrity via blockchain (Ethereum)
- Up to 80% faster document retrieval and sharing
- Reduces forgery risk by 95%+ with immutable, verifiable credentials
- Saves hours of student time during applications
- Builds trust with employers and institutions via secure, transparent records

System Architecture



This flowchart illustrates the process of uploading, storing, accessing, and verifying student results using a system that integrates cloud storage and blockchain technology. Let's break down each step:

1. **Admin & Students:** The flowchart starts with two types of users:
   - **Admin:** Represents the administrator of the system, likely responsible for uploading and managing student results.
   - **Students:** Represent the students who will access and download their results.
2. **User Interface:** Both the admin and students interact with the system through a user interface (likely a web application). The hand clicking on the screen suggests interaction.
3. **Admin Action -> PDF Generation:** The arrow from the "Admin" to "User Interface" and then to "Pdf generation" indicates that the administrator initiates the process by uploading student results, which are then converted or exist in PDF format.
4. **PDF -> Cloud Storage:** The generated PDF files of student results are then stored in "Cloud Storage." This signifies the use of a cloud-based storage solution (like AWS S3, Google Cloud Storage, etc.) to hold the actual result documents.
5. **Cloud Storage -> Download Result (Student Action):** Students, through the "User Interface," can access and "Download Result" (the PDF file) from the "Cloud

Storage." The arrow going directly from "Cloud Storage" to "Download Result" and then to "Organizations / Companies" via the "Verification Portal" suggests a direct download path for authorized users.

6. **Cloud Storage -> Hash ID Generation:** Simultaneously, after the PDF is stored in the cloud, the system generates a unique "hash id" for that specific PDF file. A hash function takes the content of the file and produces a fixed-size string (the hash) that acts as a digital fingerprint. Any change to the original PDF will result in a different hash.
7. **Hash ID Generation -> Blockchain:** The generated "hash id" is then recorded on the "Blockchain." This is the crucial step where the integrity of the result is secured. By storing the hash on a blockchain, which is immutable and transparent, any future attempts to alter the result can be detected by recalculating the hash and comparing it to the one on the blockchain.
8. **Organizations / Companies -> Verification Portal:** External entities like "Organizations / Companies" that need to verify the authenticity of a student's result can do so through a "Verification Portal."
9. **Verification Portal -> Hash ID Generation:** The organization/company, through the verification portal, would likely upload the student's result document (presumably a PDF they received). The system then generates a "hash id" for this uploaded document.
10. **Hash ID Generation -> Searching hash in blockchain:** The newly generated hash id from the uploaded document is then used to search for a matching hash within the "Blockchain."
11. **Searching hash in blockchain -> Verified (if found):**
    - **If found:** If a matching hash is found on the blockchain, it strongly indicates that the document being verified is the original, unaltered result that was initially uploaded by the administrator. The "VERIFIED" stamp signifies a successful verification.
    - **If not found (implied):** If the hash is not found on the blockchain, it would suggest that the document being verified is either not an official result or has been tampered with since it was originally recorded. This outcome isn't explicitly shown with a negative indicator but is the logical consequence of a failed search.

Tools and Technologies

- **Cloud service :AWS**

  Why AWS EC2 is Great for Beginners in Cloud Storage?

  Amazon EC2 offers an easy-to-use, scalable, and secure platform for deploying virtual servers, making it ideal for beginners working on cloud storage projects.

Key Highlights:

- Free Tier Access: 750 hours/month with t2.micro/t3.micro to get started at no cost.
- Customizable VMs: Full control over OS, memory, and storage types (EBS, instance store).
- Integrated Storage: Seamless use of Amazon EBS for block storage and S3 for object storage.
- Scalable & Secure: Auto-scaling, load balancing, IAM roles, and security groups included.

Use Cases:

- Hosting file upload/download services
- Managing digital certificates or academic records
- Running backend servers with reliable storage

- **BLOCKCHAIN: Ethereum Platform**: Ethereum is an open-source, public blockchain platform that enables the creation and execution of smart contracts. Smart contracts are self-executing contracts with the terms of the agreement directly written into code. They automate processes and enforce rules without the need for intermediaries. Ethereum uses its own cryptocurrency called Ether (ETH) to pay for transaction fees, known as "gas." The Ethereum Virtual Machine (EVM) is the runtime environment for executing smart contracts on the Ethereum network. Ethereum's robust smart contract capabilities make it well-suited for defining the logic and rules for our student result storage system.

- **HASH GENERATION: Algorithm : SHA-256 (Secure Hash Algorithm 256-bit)**
  SHA-256 is a secure cryptographic hash function that generates a unique 256-bit hash for any input. In our project, it's used to create a digital fingerprint of PDF files, ensuring they haven't been altered or tampered with.

Why SHA-256?

- Widely used in secure systems like blockchain and digital signatures
- Produces unique, irreversible hashes ideal for file integrity checks
- No known vulnerabilities, offering a good balance of speed and security

NPM Package Used: js-sha256

We use the `js-sha256` package for efficient hashing in Node.js. It's lightweight, has no external dependencies, works in both Node.js and browsers, and performs well with binary data like PDFs.

- **For Handling Uploaded PDFs (Backend):**
  1. Backend Framework's File Handling Capabilities: Most web frameworks (Flask, Django, Express, Spring Boot, etc.) have built-in ways to handle file uploads received via multipart/form-data.

  2. Cloud Storage SDK:The SDK for your chosen cloud provider (e.g., boto3 for AWS S3 with Python, Google Cloud Client Libraries for Python/Node.js, Azure Storage SDK) will be essential for interacting with your cloud storage to save the uploaded PDF files.
- **For PDF Generation (if the admin uploads raw data that needs to be converted to PDF):**

  Backend Libraries (for programmatic PDF creation):Node.js: pdfmake, html-pdf, puppeteer (can generate PDFs from HTML).

- **User-Interface**
- **frontend :**react

- **Backend:**node.js

WORK DISTRIBUTION

NAMES	WORK ALLOTTED
Vinayak	Frontend (React)
Snehal	Backend (Node.js)
Suraj	Block-chain (Ethereum)
Srushti	Cloud computing(AWS)