IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 9, September 2025

DOI: 10.17148/IJIREEICE.2025.13904

QUESTION PAPER LEAKAGE PROTECTION USING BLOCKCHAIN

Dr. Tejashwini Y[1], Mohammed Farzanula[2], Nikhil S[3], Shahid A Khan[4]

Assistant Professor, E&CE, BIET, Davangere, Karnataka India[1]

Student, UG, E&CE, BIET, Davangere, Karnataka India[234]

Abstract: Question paper leakage has emerged as a serious challenge in modern educational systems, undermining the credibility of examinations, creating unfair academic outcomes, and eroding public trust in assessment processes. Traditional paper distribution methods, often centralized and lacking robust safeguards, are highly susceptible to insider threats, data tampering, and unauthorized access. To address these vulnerabilities, this project proposes a blockchain-based solution that leverages decentralization, immutability, cryptographic encryption, and smart contracts to securely manage question papers. The system ensures that all papers are encrypted, stored in a tamper-proof distributed ledger, and made accessible only through time-locked, identity-verified mechanisms. Every access attempt and distribution event is recorded immutably, providing full traceability and eliminating the possibility of undetected leaks. By combining strong security mechanisms with transparent audit trails, the proposed platform not only prevents question paper leakage but also restores examination credibility, minimizes manual handling, and offers a scalable framework for secure academic operations.

Keywords: Blockchain, decentralization, immutability, data integrity, tamper proof.

I. INTRODUCTION

The recurring issue of question paper leakage has compromised examinations and eroded public trust in educational institutions. Traditional systems, which rely on centralized storage and distribution, are vulnerable to hacking, unauthorized access, and human error. This project proposes a blockchain-based solution to ensure the confidentiality, authenticity, and traceability of question papers. Leveraging blockchain's decentralized ledger, cryptographic security, and smart contracts, the system provides a tamper-proof and transparent framework for managing examination materials. By automating access control, time-based releases, and logging all actions for auditability, the proposed prototype aims to secure the entire lifecycle of question papers—from generation and encryption to controlled distribution—thus minimizing the risk of leakage. This approach promises to enhance the fairness and credibility of academic evaluations.

II. OBJECTIVES

To develop a system to prevent question paper leaks and ensure secure, decentralized, and tamperproof paper distribution. using blockchain technology.

To design a system which can generate question papers using AI and track access and modifications through smart contracts and audit logs.

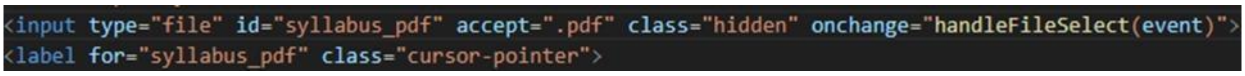
III. METHODOLOGY

This project creates and safely stores university-level question papers using blockchain and artificial intelligence. To ensure efficiency and clarity, the entire procedure is broken down into discrete stages:

A. *Introduction*

1. *Syllabus Upload and Extraction*

The interface enables academics and admin staff to upload PDF syllabuses, which are captured using JavaScript and parsed by a client-side library for question paper generation.



```
<input type="file" id="syllabus_pdf" accept=".pdf" class="hidden" onchange="handleFileSelect(event)">
<label for="syllabus_pdf" class="cursor-pointer">
```

Figure 1 Syllabus upload and Extraction

DOI: 10.17148/IJIREEICE.2025.13904

The frontend allows faculty or admin staff to upload a syllabus in PDF format via an HTML input form. JavaScript captures and parses the file using a client-side library to extract readable text, which is then used as input for the question paper generation process.

2 *Text Preprocessing*

The extracted syllabus content is often unstructured and cluttered. A preprocessing script cleans it by removing special characters, extra spaces, and formatting it into organized topic lists suitable for input to a natural language generation (NLG) model.

3 *Question Paper Generation using Cohere API*

The cleaned syllabus is sent to the Cohere API, an advanced NLP model, which generates a structured question paper with short, long, and descriptive questions aligned to university exam standards.

4 *Display of Generation Logs*

Real-time logs track the question generation process, including API status, completed steps, and the final output. This transparency helps users understand and trust the paper generation workflow.

5 *Storing on Blockchain using Polygon Amoy Testnet*

After generation, users can store the question paper on the Polygon Amoy Testnet via a Solidity smart contract. The contract securely stores the paper using test MATIC tokens, enabling safe and cost-free testing of blockchain functionality while preventing tampering and unauthorized access.

6 *Blockchain Confirmation and Access*

After upload, the system returns a transaction hash as proof of secure storage. This unique ID ensures the paper's integrity and authenticity, as it remains immutable and verifiable on the blockchain at any time.

B. *How Cohere API Works*

The Cohere API is an advanced Natural Language Processing (NLP) tool that helps developers create high-quality text from basic inputs. In this project, it serves a crucial function by transforming syllabus content into well-structured, university-level question papers. Here's a detailed explanation of its role:

```
COHERE_API_KEY=QR4u5tETUkGb0sRQXXq3Iywscem3jRraol4wIj2D
```

Figure 2 Cohere API Key

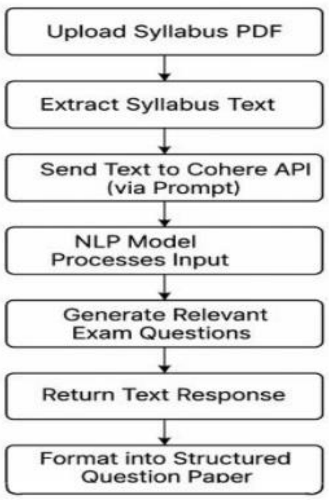

Upload Syllabus PDF
Extract Syllabus Text
Send Text to Cohere API (via Prompt)
NLP Model Processes Input
Generate Relevant Exam Questions
Return Text Response
Format into Structured Question Paper

Figure 3 Cohere API Flow Chart

1. *Input: Receiving Syllabus or Topics*

The system extracts and cleans syllabus content from a PDF or direct input, then structures the topics. These are sent to the Cohere API via a POST request with a prompt like: "Generate a university-level question paper based on the following topics: [Topic List], including short, long, and descriptive questions."

2. *Processing: NLP-Based Question Generation*

Once the prompt is submitted, Cohere's LLM processes it using deep learning to generate relevant, coherent questions. It leverages its academic knowledge to ensure the questions follow standard exam formats and produce natural, human-like content suitable for educational use.

3. *Output: Structured Question Paper Generation*

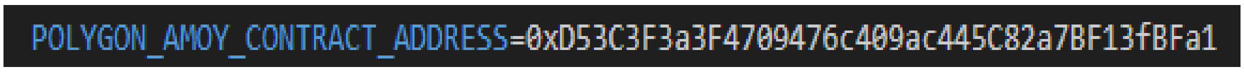
The API returns the questions in plain text, grouped by type (e.g., short answer, long-form). The frontend formats this into a clean, printable question paper, which can be reviewed, edited, and uploaded to the blockchain for secure storage.

C. How Polygon Amoy Testnet Works

The Polygon Amoy Testnet serves as a testing environment for applications designed for the Polygon blockchain. It mimics the main network's functionality but uses test tokens, allowing developers to safely build and test dApps without incurring real costs. In this project, it is used to securely store generated question papers on the blockchain.

1. *Simulated Version of the Polygon Blockchain*

The Amoy Testnet is a public testing platform that replicates the functionality of the main Polygon network. It offers the same infrastructure, smart contract execution, and operational rules but is intended exclusively for testing. This enables developers to simulate real blockchain deployments without any financial risk.



```
POLYGON_AMOY_CONTRACT_ADDRESS=0xD53C3F3a3F4709476c409ac445C82a7BF13fBFa1
```

Figure 4 Polygon Amoy

2. *Use of Test MATIC Tokens*

Transactions on the Amoy Testnet use test MATIC tokens, which are valueless and meant for development purposes. Developers can obtain these tokens from official Polygon faucets to cover gas fees when deploying or interacting with smart contracts. This allows for extensive testing without any real expenses.

3. *Safe Environment for Smart Contract Deployment*

The Amoy Testnet allows developers to write, deploy, and test smart contracts built with Solidity. It verifies contract logic, identifies errors, and ensures smooth interaction between components before mainnet deployment. Early detection of issues during testing helps save time and avoid critical failures later.

4. *Usage in our Project*

In our project, the generated question papers are securely stored on the Amoy Testnet. After the AI (via Cohere) creates the paper, it is hashed and uploaded to the blockchain through a smart contract. This process guarantees the paper is tamper-resistant, timestamped, and publicly verifiable—even in a testing environment. While currently deployed on the testnet, the setup closely mirrors how it would operate on the live Polygon network.

D. *Smart Contracts*

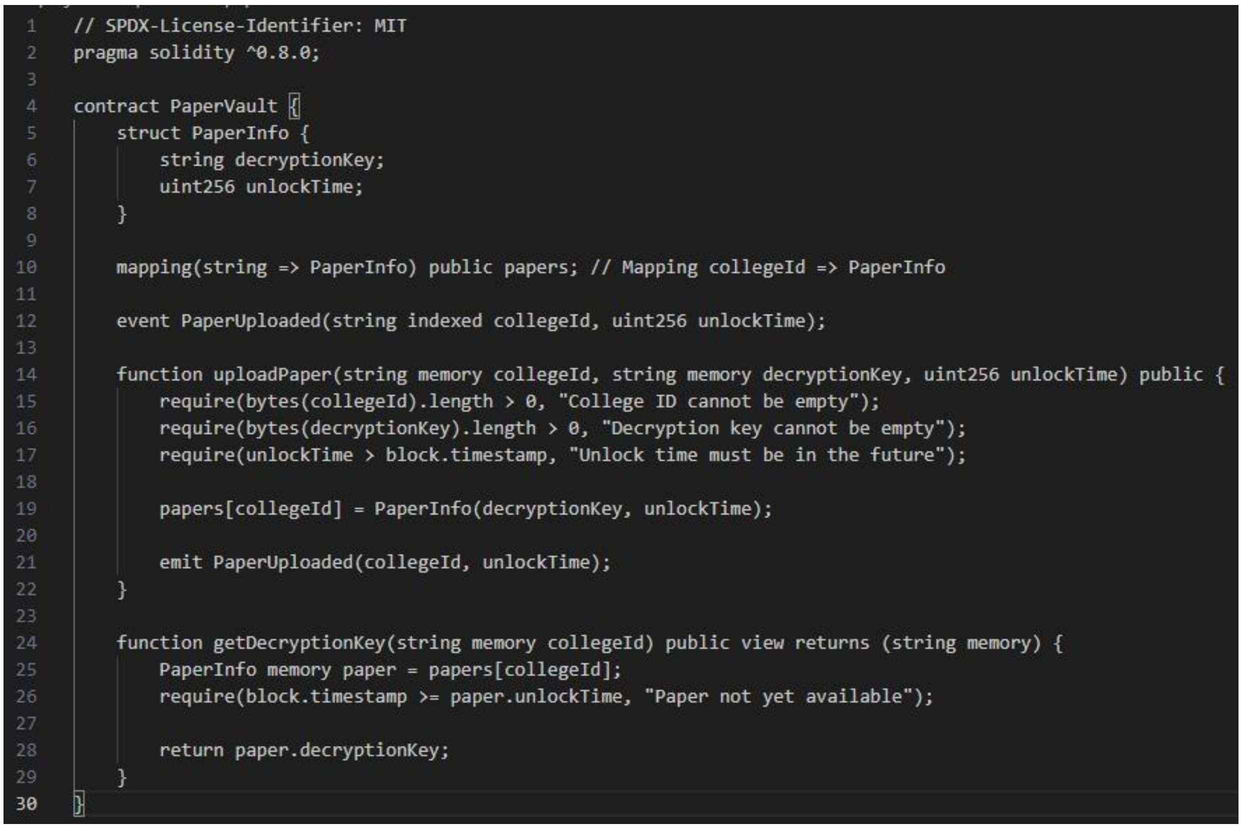
Smart contracts are self-operating code stored on the blockchain that automatically execute predefined tasks when certain conditions are fulfilled. In our project, they play a key role in securely managing, storing, and controlling access to the generated question papers, effectively linking the AI-generated content with the blockchain infrastructure.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 9, September 2025

DOI: 10.17148/IJIREEICE.2025.13904



```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract PaperVault {
    struct PaperInfo {
        string decryptionKey;
        uint256 unlockTime;
    }

    mapping(string => PaperInfo) public papers; // Mapping collegeId => PaperInfo

    event PaperUploaded(string indexed collegeId, uint256 unlockTime);

    function uploadPaper(string memory collegeId, string memory decryptionKey, uint256 unlockTime) public {
        require(bytes(collegeId).length > 0, "College ID cannot be empty");
        require(bytes(decryptionKey).length > 0, "Decryption key cannot be empty");
        require(unlockTime > block.timestamp, "Unlock time must be in the future");

        papers[collegeId] = PaperInfo(decryptionKey, unlockTime);

        emit PaperUploaded(collegeId, unlockTime);
    }

    function getDecryptionKey(string memory collegeId) public view returns (string memory) {
        PaperInfo memory paper = papers[collegeId];
        require(block.timestamp >= paper.unlockTime, "Paper not yet available");

        return paper.decryptionKey;
    }
}
```

Figure 5 Smart Contract

1. *Written in Solidity*

Solidity is the main programming language used for developing smart contracts on Ethereum-compatible platforms such as Polygon. It enables the definition of specific rules and logic for handling and storing question papers. Once the Solidity code is compiled and deployed to the blockchain, it becomes permanent and cannot be altered.

2. *Defines the Upload, Storage, and Retrieval Logic*

The smart contract includes functions for:
Uploading: After a question paper is generated, its hash and metadata (e.g., timestamp, subject) are recorded on the blockchain.
Storage: This information is stored as immutable transactions on the ledger.
Retrieval: Authorized users can access details like the paper hash and upload time to confirm authenticity and integrity.

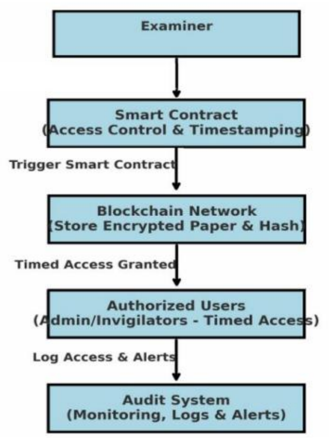

Examiner
Smart Contract (Access Control & Timestamping)
Trigger Smart Contract
Blockchain Network [Store Encrypted Paper & Hash]
Timed Access Granted
Authorized Users (Admin/Invigilators - Timed Access)
Log Access & Alerts
Audit System (Monitoring, Logs & Alerts)

Figure 6 Smart Contract Flow Chart

3. *Deployed on the Polygon Amoy Testnet for Safe Testing*

The smart contract was deployed on the Polygon Amoy Testnet, which replicates the main Polygon network while using test MATIC tokens. This environment enabled us to test the full workflow—including question paper generation, blockchain storage, and data retrieval—without incurring real costs or risks. The successful testnet deployment serves as a reliable foundation for future migration to the mainnet in a production setting.

E. *Frontend UI Features*
The frontend of our project is built to offer a seamless and user-friendly interface for managing question paper generation and blockchain uploads. Developed using modern web technologies and styled with Tailwind CSS, it emphasizes clarity, responsiveness, and ease of use.

1. *File Upload Interface for Syllabus*

Users can upload syllabus PDFs using a simple drag-and-drop or click-to-upload interface. The system validates file type and size to ensure only appropriate PDFs are accepted, enhancing usability and reducing errors. Once uploaded, the syllabus content initiates the AI-driven question paper generation process. Users also receive visual feedback, including the file name and options to remove or replace the file before continuing.

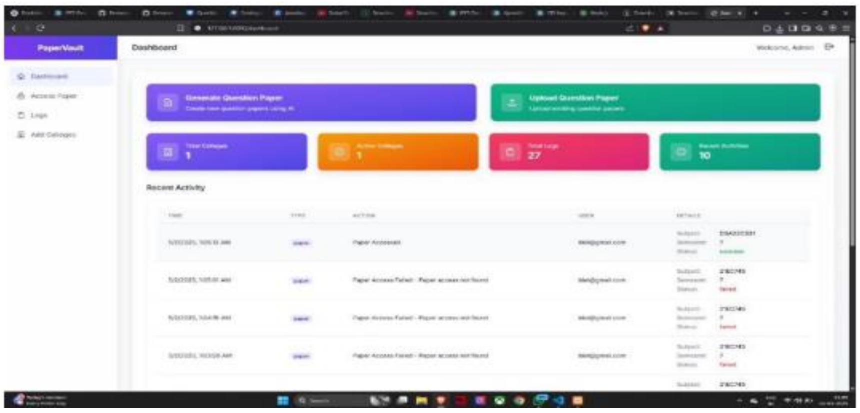


Figure 7 Dashboard

2. Generation Logs Display

The UI features a dedicated section for displaying real-time logs during the question paper generation process. This allows users to monitor the AI's progress, providing transparency and assurance that the system is functioning correctly. The logs are shown in a scrollable box with a monospace font for easy readability and debugging.

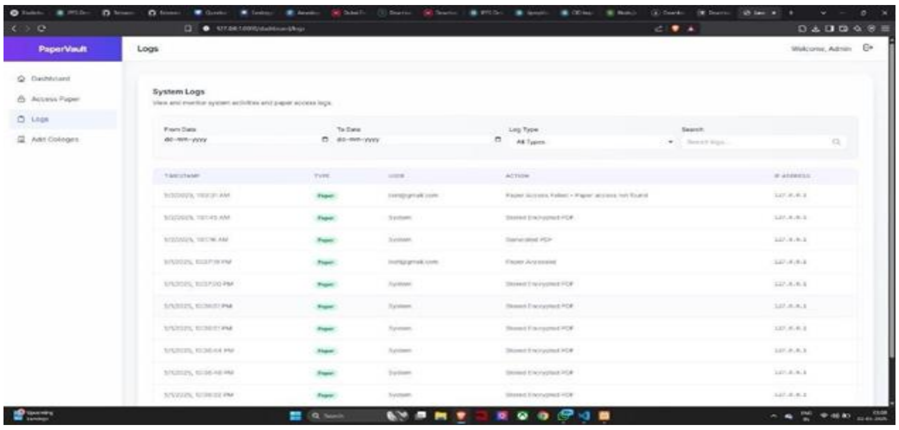


Figure 8 System Logs

3. *Blockchain Upload Modal for Final Submission*
Once the question paper is generated, users can upload it to the blockchain through a streamlined modal popup. This modal guides them through the final submission step, confirms the upload, and displays the transaction status. It offers clear feedback on the success of the storage process, reinforcing trust in the system's security and data permanence.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 9, September 2025

DOI: 10.17148/IJIREEICE.2025.13904

4. *Responsive and User-Friendly Layout Using Tailwind CSS*

The frontend is styled using Tailwind CSS, a utility-first framework that enables fast and visually clean design. Its responsive layout adjusts seamlessly across desktops, tablets, and mobile devices. Thoughtful use of typography, spacing, and color contrast ensures readability and accessibility, while interactive elements like buttons and input fields feature hover and focus effects to enhance user experience.

IV. SOFTWARE DESCRIPTION

The software is designed to protect the integrity and confidentiality of question paper management in educational institutions by leveraging Blockchain technology, Artificial Intelligence (AI) for question generation, cryptographic encryption, and smart contracts. The goal is to develop a system that can securely generate, store, control, and release question papers in a tamper-proof and auditable manner.

1. *System Architecture Overview*

The system architecture is modular and consists of the following components:
User Interface (Frontend): A web application where examiners, administrators, and other users interact with the system.
Backend Application Server: Handles business logic, file management, and AI communication.
AI Integration Layer: Connects with Cohere API for automatic question generation based on uploaded syllabi.
Encryption Engine: Encrypts question papers before uploading to blockchain.
Blockchain Layer: Stores encrypted question papers and manages access control using smart contracts.
Smart Contract Module: Implements access timing, role-based permissions, and data integrity enforcement.

2. *Technology Stack*

Layer/Module - Technology/Tool Used
Frontend - Laravel (PHP framework), Tailwind CSS
Backend - Laravel Blade, PHP
Encryption - AES (CBC mode), Base64 Conversion (JavaScript)
AI Integration - Cohere API
Blockchain - Ethereum-compatible platform
Smart Contracts - Solidity (version ^0.8.0), MIT License
Deployment Environment - Localhost or web server (development mode)
Programming Languages - PHP, HTML, CSS, JavaScript, Solidity
Data Exchange Format - JSON, Base64, PDF

3. *Detailed Module Description*

3.1 User Interface (Frontend)
Built using Laravel Blade templates and styled with Tailwind CSS, the UI is designed to be responsive and user-friendly.

- Login Page: Secure email/password authentication form with icons and input validation.
- Syllabus Upload Interface: Allows uploading of syllabus in PDF format. Triggers the AI question generator.
- Animated Backgrounds: Glowing, non-intrusive animated elements for modern visual aesthetics.
- Upload Modals and Logs: Modal pop-ups for uploading question papers to blockchain and logs to track the process.

3.2 *Question Generation Module (AI Integration)*

- Input: PDF file containing the syllabus.
- Process: The syllabus is parsed and divided into modules. Each module is sent to the Cohere AI API which generates sample questions.
- Output: A set of 10-mark questions (limited to 2 per module in demo mode).

Limitations: The demo version of Cohere only allows limited generation. No AI filtering or validation — human review is recommended.

3.3 *Encryption Module*

To ensure the security of question papers before uploading them to the blockchain:

- PDF to Base64: The PDF file is read via FileReader and converted to a Base64 string.
- AES Encryption: Uses AES in CBC (Cipher Block Chaining) mode. Requires a secret key and Initialization Vector (IV). Encryption is done client-side using JavaScript. Keys are held only in the browser memory (demo limitation).
- Security Note: For production, encryption must be moved to the server-side to protect the secret key.

3.4 *Blockchain Upload and Storage*

- Upload Modal: Provides a form to enter College ID, decryption key, and unlock time.
- Blockchain Storage: Encrypted Base64 question paper is uploaded to the blockchain. Smart contract stores the data against the College ID.
- Logging: Upload logs are maintained to track uploads and access attempts.

3.5 *Smart Contract (Paper Vault. solution)*

A smart contract in Solidity that governs the secure storage and timed access of question papers. This MIT-licensed smart contract, built with Solidity ^0.8.0, securely stores decryption keys and unlock times using a struct and maps them to college IDs. The uploadPaper function validates inputs, stores encrypted data, and emits a PaperUploaded event. The getDecryptionKey function allows access only after the unlock time, preventing early retrieval and ensuring secure distribution.

3.6 *Access Control and Timing Mechanism*

Smart Contract Locking: Ensures question papers are inaccessible before the set unlock time and prevents unauthorized access with strict identity checks.
Audit Logs: Immutable logs record every paper upload and access attempt and helps in post-exam audit and tamper investigation.

3.7 *Security Features*

1. AES Encryption - Ensures confidentiality of uploaded PDFs
2. Time-Locked Smart Contracts - Decryption keys accessible only after a specified timestamp
3. Blockchain Immutability - Once uploaded, papers cannot be modified or deleted
4. Transparent Audit Logs - Every action is logged for accountability
5. Role-Based Access (Planned) - Paper access limited to authorized roles (examiners, admins)

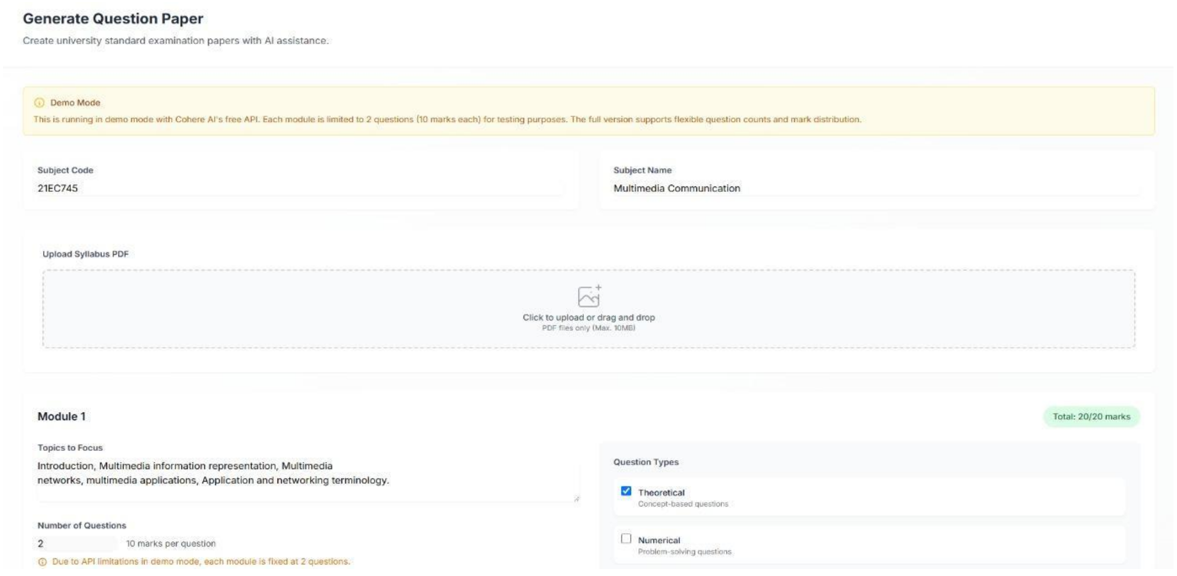
V. RESULTS

1. Outcomes

The system parsed uploaded syllabi into modules, generated relevant questions using Cohere's API, and securely encrypted them with AES-CBC. Encrypted papers were uploaded to the blockchain via a smart contract, which stored metadata and unlock times. Access to decryption keys was restricted until the unlock time, ensuring secure and traceable paper handling.

1.1 **Syllabus Upload and Parsing**

The system successfully allowed the upload of PDF syllabus files and automatically parsed them into separate modules, preparing the content for AI-based processing. Figure 9 represents the frontend of the Syllabus Upload interface, question choices and difficulty levels for the question paper to be generated before turning it into a prompt.



Generate Question Paper

Create university standard examination papers with AI assistance.

Demo Mode

This is running in demo mode with Cohere AI's free API. Each module is limited to 2 questions (10 marks each) for testing purposes. The full version supports flexible question counts and mark distribution.

Subject Code: 21EC745

Subject Name: Multimedia Communication

Upload Syllabus PDF

Click to upload or drag and drop
PDF files only (Max. 10MB)

Module 1 — Total: 20/20 marks

Topics to Focus: Introduction, Multimedia information representation, Multimedia networks, multimedia applications, Application and networking terminology.

Number of Questions: 2 — 10 marks per question

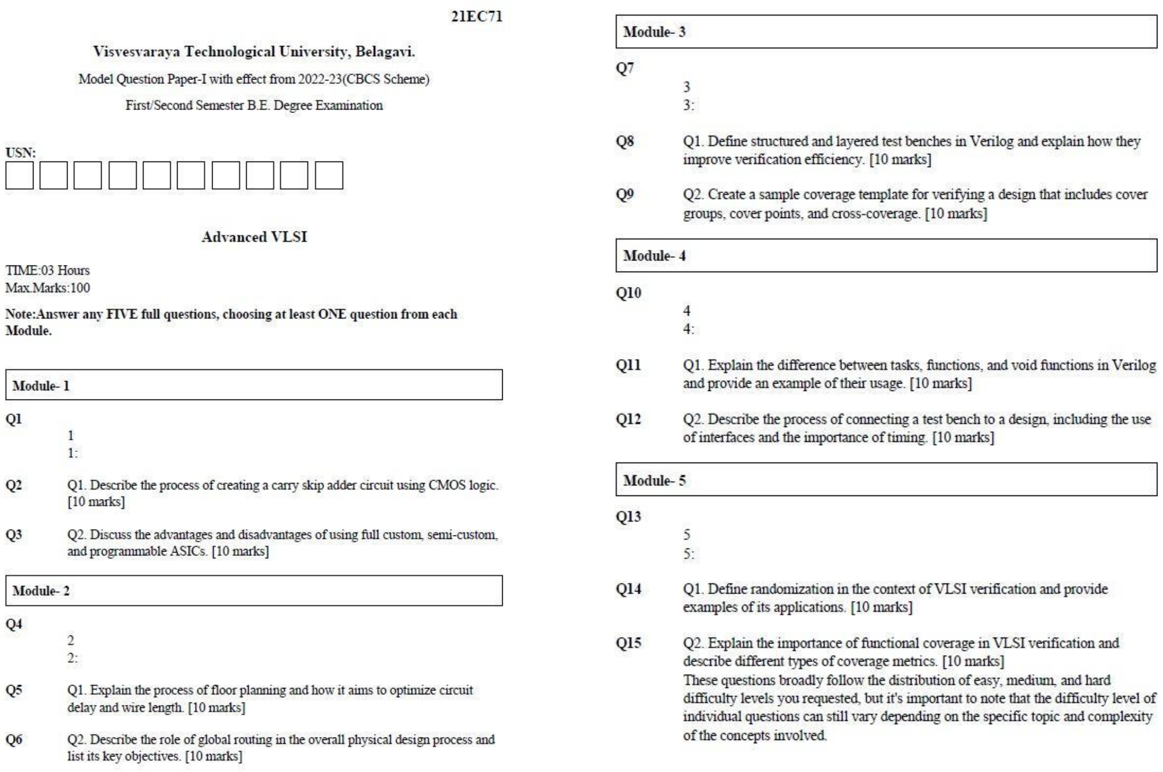
Due to API limitations in demo mode, each module is fixed at 2 questions.

Question Types
[✓ Theoretical] Concept-based questions
Numerical — Problem-solving questions

Figure 9 Syllabus Upload

1.2 AI-Generated Question Papers (Cohere API)

The platform was integrated with Cohere's demo API to generate examination questions by using prompts generated by using the syllabus and other inputs provided by the users. For each module identified from the syllabus, the system produced two relevant 10-mark questions. These questions accurately reflected the context and topics outlined in the uploaded syllabus content. Figure 10 represents the AI-Generated Question Papers.



21EC71

Visvesvaraya Technological University, Belagavi.

Model Question Paper-I with effect from 2022-23(CBCS Scheme)

First/Second Semester B.E. Degree Examination

USN:

Advanced VLSI

TIME:03 Hours
Max.Marks:100

Note:Answer any FIVE full questions, choosing at least ONE question from each Module.

Module- 1

Q1
1
1:

Q2 Q1. Describe the process of creating a carry skip adder circuit using CMOS logic. [10 marks]

Q3 Q2. Discuss the advantages and disadvantages of using full custom, semi-custom, and programmable ASICs. [10 marks]

Module- 2

Q4
2
2:

Q5 Q1. Explain the process of floor planning and how it aims to optimize circuit delay and wire length. [10 marks]

Q6 Q2. Describe the role of global routing in the overall physical design process and list its key objectives. [10 marks]

Module- 3

Q7
3
3:

Q8 Q1. Define structured and layered test benches in Verilog and explain how they improve verification efficiency. [10 marks]

Q9 Q2. Create a sample coverage template for verifying a design that includes cover groups, cover points, and cross-coverage. [10 marks]

Module- 4

Q10
4
4:

Q11 Q1. Explain the difference between tasks, functions, and void functions in Verilog and provide an example of their usage. [10 marks]

Q12 Q2. Describe the process of connecting a test bench to a design, including the use of interfaces and the importance of timing. [10 marks]

Module- 5

Q13
5
5:

Q14 Q1. Define randomization in the context of VLSI verification and provide examples of its applications. [10 marks]

Q15 Q2. Explain the importance of functional coverage in VLSI verification and describe different types of coverage metrics. [10 marks]
These questions broadly follow the distribution of easy, medium, and hard difficulty levels you requested, but it's important to note that the difficulty level of individual questions can still vary depending on the specific topic and complexity of the concepts involved.

Figure 10 AI Generated Question Papers

1.3 Blockchain Upload and Logging

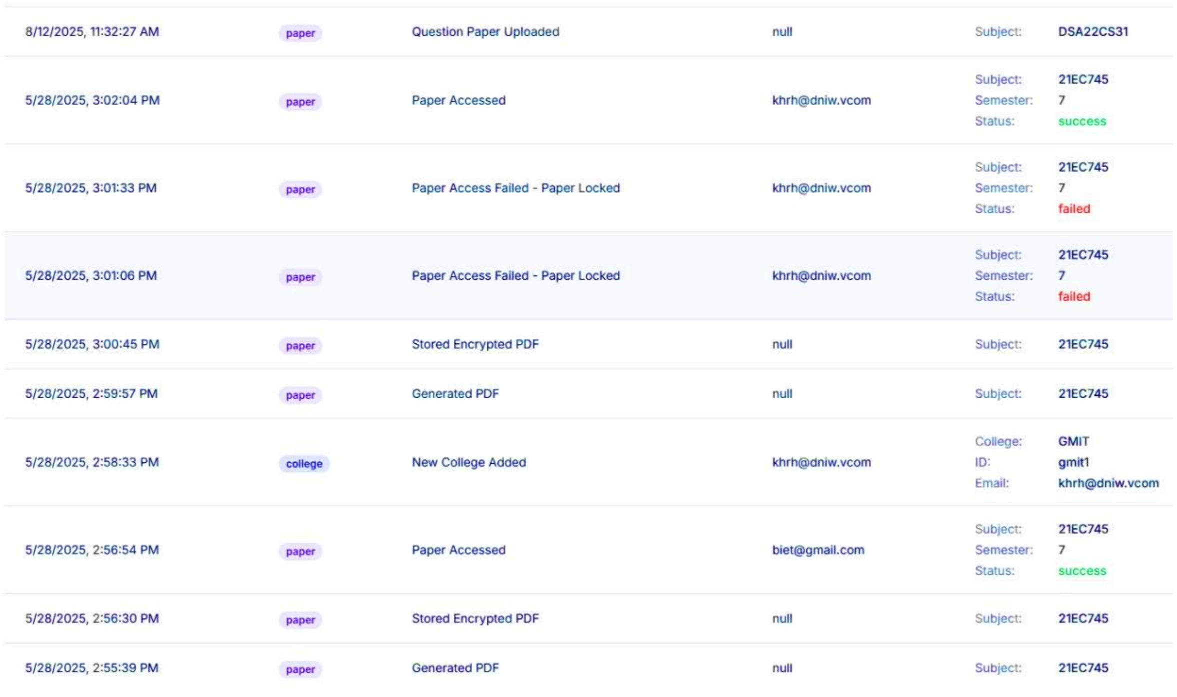
The encrypted papers were uploaded to the blockchain via a custom smart contract. This contract securely stored the related metadata, including the unlock time and decryption key, alongside the encrypted data. Each upload was recorded in the blockchain's event logs, enabling complete traceability of all transactions.

ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⪤ Peer-reviewed & Refereed journal ⪤ Vol. 13, Issue 9, September 2025

DOI: 10.17148/IJIREEICE.2025.13904

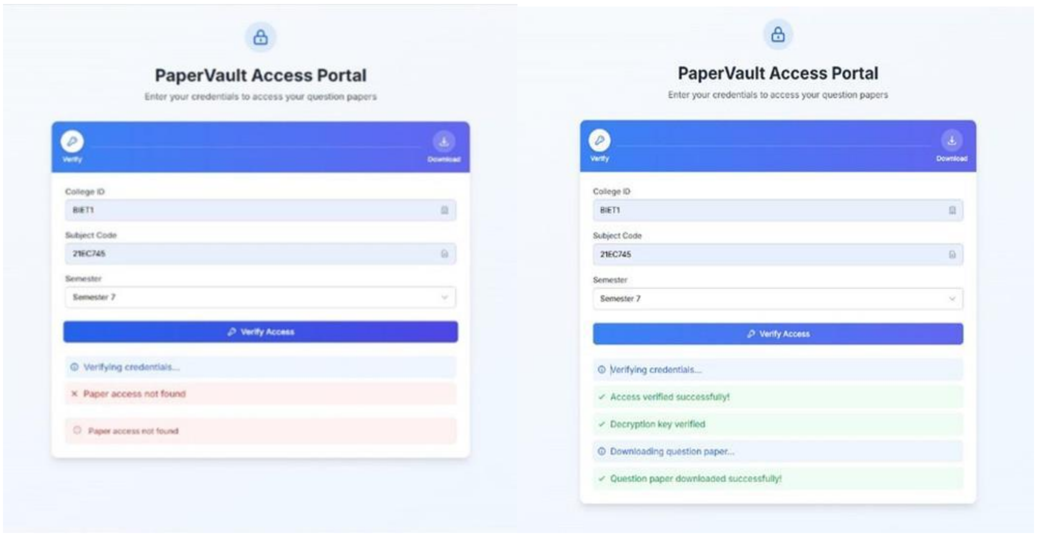


8/12/2025, 11:32:27 AM	paper	Question Paper Uploaded	null	Subject: DSA22CS31
5/28/2025, 3:02:04 PM	paper	Paper Accessed	khrh@dniw.vcom	Subject: 21EC745 Semester: 7 Status: success
5/28/2025, 3:01:33 PM	paper	Paper Access Failed - Paper Locked	khrh@dniw.vcom	Subject: 21EC745 Semester: 7 Status: failed
5/28/2025, 3:01:06 PM	paper	Paper Access Failed - Paper Locked	khrh@dniw.vcom	Subject: 21EC745 Semester: 7 Status: failed
5/28/2025, 3:00:45 PM	paper	Stored Encrypted PDF	null	Subject: 21EC745
5/28/2025, 2:59:57 PM	paper	Generated PDF	null	Subject: 21EC745
5/28/2025, 2:58:33 PM	college	New College Added	khrh@dniw.vcom	College: GMIT ID: gmit1 Email: khrh@dniw.vcom
5/28/2025, 2:56:54 PM	paper	Paper Accessed	biet@gmail.com	Subject: 21EC745 Semester: 7 Status: success
5/28/2025, 2:56:30 PM	paper	Stored Encrypted PDF	null	Subject: 21EC745
5/28/2025, 2:55:39 PM	paper	Generated PDF	null	Subject: 21EC745

Figure 11 Recorded Logs

1.4 Timed Access Control

Access to the decryption key was strictly restricted until the preset unlock time, as enforced by the smart contract. Any unauthorized attempts to retrieve the key before this time were blocked. Once the unlock time was reached, the keys were successfully retrieved and used to decrypt the stored question papers.



PaperVault Access Portal

Enter your credentials to access your question papers

College ID: BIET1
Subject Code: 21EC745
Semester: Semester 7

Verifying credentials...
× Paper access not found
Paper access not found

PaperVault Access Portal

Enter your credentials to access your question papers

College ID: BIET1
Subject Code: 21EC745
Semester: Semester 7

Verifying credentials...
✓ Access verified successfully!
✓ Decryption key verified
Downloading question paper...
✓ Question paper downloaded successfully!

Figure 12 Paper access

2. UI and System Behaviour

Feature	Status	Notes
Responsive UI with Tailwind	Working	Animated backgrounds and user-friendly layout
Email/Password Login	Working	Form-based, no 2FA (future work)
Generation Logs	Working	Live log updates in scrollable area
Upload Modal & Feedback	Working	Modal shows status of blockchain uploads
Audit Trail Logging	Partial Demo	Uploads logged, access logs could be extended

Table 1. UV and system behaviour

3. Performance & Security Insights

Security: AES encryption proved effective for confidentiality. Smart contract access control prevented early paper access. Logs on blockchain were immutable and tamper-resistant.

Limitations Noted: Client-side encryption is suitable only for demo/testing. Secret keys and IVs must be secured in a production backend. AI-generated content lacked semantic quality control.

4. Demonstration Results (Demo Mode)

Tasks	**Observed Result**
Syllabus upload	Immediate parsing and AI trigger
Question generation	Within 5–10 seconds per module
Paper encryption and upload	< 2 seconds to convert & encrypt
Blockchain upload and mapping	Completed with smart contract confirmation
Timed unlock test	Blocked before time; successful after time
Decryption with key	Paper restored correctly post-decryption

Table 2. Demonstration Results

5. Summary of achievements

Developed a full-stack prototype integrating: AI question generation, AES encryption, Smart contracts and blockchain. Demonstrated end-to-end flow from syllabus upload to secured timed question paper delivery. Validated the use of blockchain for tamper-proof, transparent paper management. Proved potential for real-world deployment in academic institutions with further enhancements.

VI. CONCLUSION

Question paper leakage continues to undermine the credibility and fairness of academic examinations. This project offers a blockchain-based solution that ensures secure, transparent, and tamper-proof question paper management. By using blockchain's immutability and smart contracts, the system enables time-locked, rule-based access, protecting against unauthorized handling and early leaks. Unlike traditional methods, this decentralized approach provides end-to-end security, full traceability, and automated control. Every step—from generation to distribution—is logged on the blockchain, creating an immutable audit trail. The project also lays the groundwork for future enhancements like biometric authentication, multi-chain support, and large-scale deployment across educational institutions, marking a major step toward secure academic governance.

REFERENCES

[1]. Blockchainbased Competitive Examination System in India Preventing Paper Leaks and Mitigating Frauds, Authors Anush D Kamble, Deepak A Kamble, Publication Date August 2024, Journal International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)
[2]. Secured Question Paper Management System, Authors S. M. Tawhidur Rahman, Md. Nashif Iftekhar, Rashedul Alam, S. M. Raihan Gafur, Dip Nandi, Publication Date 2021, Journal AIUB Journal of Science and Engineering, Volume 20, Issue 1
[3]. BSSSQS A BlockchainBased Smart and Secured Scheme for Question Sharing in the Smart Education System, Authors Anik Islam, Md Fazlul Kader, Soo Young Shin, Publication Date September 2019, Journal Journal of Information and Communication Convergence Engineering (J. Inf. Commun. Converg. Eng.), Volume 17, Issue 3, Pages 174-184
[4]. Leveraging Blockchain Technology to Enhance Exam Security and Prevent Question Paper Leaks at Punjab State Aids Control Society , Authors Ms. Resham Sethi, Ms. Anushri Israni, Mr. Nikhil Uppal, Mr. Shashi Pal, Mr. Vikram R Singh, Dr. Harpeet Kaur, Dr. Karthik Adapa, IAS, Publication Date February 2024, Publication eGov Elets Online (https //egov.eletsonline.com)
[5]. A Study on Blockchain Technology, Authors P. S. G. Aruna Sri and D. Lalitha Bhaskari, Publication Date March 2018, Source International Journal of Engineering & Technology, Vol. 7 (2.7), pp. 418–421, DOI 10.14419/ijet.v7i2.7.10757
[6]. A comprehensive review of blockchain technology Underlying principles and historical background with future challenges, Authors Tripathi Gautami, Ahad Mohd Abdul, Casalino Gabriella, Publication Date December 1, 2023, Publication ScienceDirect (domain of modeling, simulation, applied mathematics and decision sciences)
[7]. Cohere API. (n.d.). Cohere for Text Generation. Retrieved from https //docs.cohere.com
[8]. EthereumFoundation.(n.d.).SolidityDocumentation. Retrieved from https //docs.soliditylang.org