CYBRELLA HACKATHON: BIO CODEX

MediChain

Decentralized & Encrypted Medical Records

"Empowering Patients. Securing Data. Saving Lives."

TEAM QUARKS

Aryan Dandotiya	Backend, Frontend, API Development
Saidul H. Chaudhary	Backend, Blockchain Architecture
Divyam K. Choubey	API Integration
Bittu Shah	Deployment & DevOps

SUBMISSION RESOURCES

Round 1: Concept Video

View Design Mockups (Figma)

February 17, 2026

Contents

1 The Problem Landscape

1.1 Problem Statement

The modern healthcare system suffers from **Data Silos**. A patient's medical history is fragmented across different hospitals, clinics, and labs. This fragmentation leads to three critical failures:

1. **Lack of Ownership:** Patients do not own their data; centralized institutions do.
2. **Interoperability Issues:** Transferring records between hospitals is manual, slow, and error-prone.
3. **Security Vulnerabilities:** Centralized databases are prime targets for ransomware attacks.

1.2 Target Audience

- **Primary:** Patients requiring chronic care or ownership of their history.
- **Secondary:** Doctors requiring instant, verifiable patient history.

2 Proposed Solution & USP

2.1 Solution Overview

MediChain is a **Hybrid Decentralized Application (DApp)** utilizing a secure "Lock-and-Key" architecture:

The Vault (Storage)

Encrypted medical files are stored on **IPFS (Pinata)**.

The Lock (Encryption)

Files are encrypted via **AES-256** before upload.

The Key (Access)

The Ethereum Blockchain acts as the access manager. Only the patient's private key can authorize a doctor to decrypt the file.

Figure 1 *– **Current State:** Fragmented Data & Zero Patient Control*

2.2 Unique Selling Proposition

Why MediChain?

- **Patient Sovereignty:** Access is granted and revoked solely via smart contracts.
- **Tamper-Proof Verification:** Files are hashed upon upload. If a single byte changes, the hash mismatches, flagging the file as compromised.

Figure 2 – ***Comparison:*** *Centralized Vulnerability vs. Decentralized Security*

3 Technical Architecture & Stack

3.1 System Workflow

The data flow ensures privacy by design:

Step 1: Input: User selects a file (PDF/Image).

Step 2: Processing: Python Service encrypts file (AES-256).

Step 3: Storage: Encrypted blob uploaded to IPFS. CID Returned.

Step 4: Blockchain: CID and File Hash stored on Sepolia Smart Contract.

Step 5: Output: System verifies on-chain hash before decryption.

3.2 Technology Stack

→ **Frontend:** React.js, Vite, Tailwind CSS, Ethers.js.

→ **Backend:** Django REST Framework, Python FastAPI (Encryption).

→ **Blockchain:** Solidity, Hardhat, Sepolia Testnet.

→ **Storage:** IPFS (Pinata Cloud).

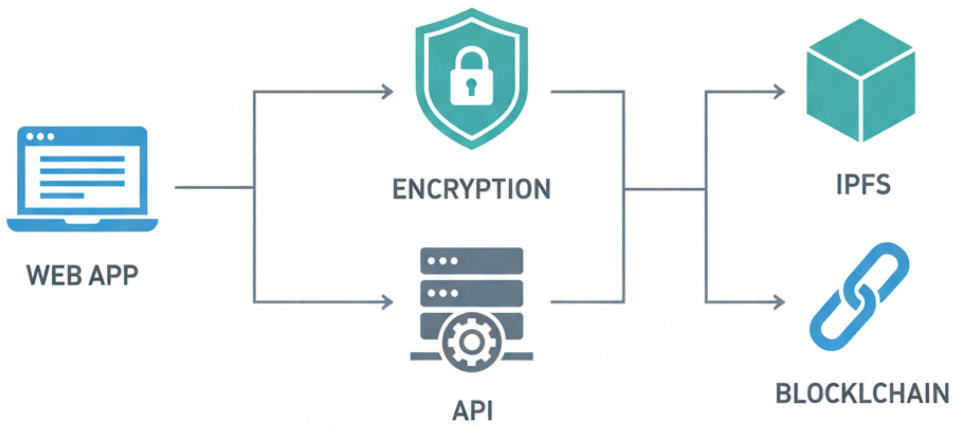

***Figure 3** – **System Architecture:** Frontend → Encryption → IPFS → Blockchain*

4 Key Features & Functionalities

Feature 1: Cryptographic Integrity Check The "Verify" button fetches the file, re-hashes it locally, and compares it with the immutable blockchain record.

Feature 2: Role-Based Dashboards Distinct User Interfaces for Patients (Grant/Revoke Access) and Doctors (Upload/Verify).

Feature 3: Zero-Knowledge Privacy The platform admins cannot view user data; only the private key holder can.

***Figure 4** – **Working Prototype:** Dashboard, Verification Success, and MetaMask Integration.*

5 Implementation Roadmap

Phase 5: **Qualifying Round (Feb 11–17):** Core Architecture setup, Smart Contract deployment (Local), AES Encryption logic.

Phase 5: **Final Excellence Round (Feb 19–25):** Sepolia Testnet deployment, Frontend Wallet integration, UI Polish, Final Demo.

6 Impact & Sustainability

6.1 Social Impact

Reduces medical errors caused by missing history and eliminates redundant testing costs for patients.

6.2 Scalability

High scalability due to off-chain storage (IPFS). Blockchain is used only for lightweight pointers, keeping gas costs minimal.

7 Future Scope: AI Integration

Transforming MediChain from a storage solution to an intelligent assistant.

1. **"Vital-Sync" Summarizer:** In emergencies, doctors cannot read 50 pages. We will integrate GenAI to scan decrypted records and generate a **one-page summary** (Allergies, Blood Type, Conditions).
2. **"Medi-Bot" Assistant:** A RAG-based chatbot allowing patients to ask, *"Can I take Ibuprofen?"* The AI checks history for interactions (e.g., *"No, you are on blood thinners"*).

8 References

[1] **Ethereum Documentation** (ERC Standards).
https://ethereum.org/developers/docs/

[2] **Pinata API Documentation** (IPFS).
https://docs.pinata.cloud/api-reference/introduction

[3] **PyCryptodome** (AES-256 Implementation).
https://pycryptodome.readthedocs.io/en/latest/src/cipher/aes.html

[4] **Hardhat Development Environment**.
https://hardhat.org/docs/getting-started