UnCodemy

Blockchain Final Year Project Ideas PDF

A good blockchain final year project uses a smart contract only where shared, tamper-evident records add real value, such as certificates, voting, supply chains or land records. Build it in Solidity with Hardhat, deploy on the Sepolia testnet, connect a React front end through MetaMask, and store large files on IPFS. Never use real money.

Key takeaways

- Explain why a blockchain is needed; examiners often ask "why not a normal database?"
- Deploy on a testnet such as Sepolia; testnet ETH is free from faucets.
- Use OpenZeppelin contracts for tokens and access control.
- Write unit tests for every contract function and show gas costs.
- The PDF gives plans and steps, not finished source code.

Who this PDF is for

This PDF is for final-year B.Tech and MCA students building decentralised apps. Most plans use the Ethereum toolchain (Solidity, Hardhat, ethers.js, MetaMask, IPFS); two use Hyperledger Fabric for permissioned networks. Each plan lists contracts, front-end and back-end modules, and viva points, so a team of 3 or 4 can split contracts, testing, front end and documentation.

How to choose your project

- **Need for trust**: pick problems where several parties must share records they cannot quietly change.
- **Public or permissioned?** Ethereum testnets for open apps; Hyperledger Fabric for business networks.
- **Keep data off-chain** when large or personal; store only hashes on-chain.
- **Plan security reviews**: reentrancy, access control and integer checks.

Standard level projects

1. Academic Certificate Verification

- **Problem:** Let employers verify degree certificates instantly.
- **Network and tools:** Ethereum Sepolia testnet, IPFS.
- **Tech stack:** Solidity, Hardhat, OpenZeppelin AccessControl, React, ethers.js.
- **Main modules:** Issuer role, certificate hash registry, IPFS upload, public verify page, revoke function.

- **Steps:** Hash each certificate PDF, store the hash on-chain, keep the file on IPFS, verify by re-hashing.
- **Show in the viva:** Why only the hash goes on-chain.
- **Extensions:** Add QR codes on printed certificates.

2. Decentralised Voting System

- **Problem:** Run a college election with transparent, tamper-evident counts.
- **Network and tools:** Local Hardhat network, then Sepolia.
- **Tech stack:** Solidity, Hardhat, React, MetaMask.
- **Main modules:** Voter registration, candidate list, voting window, one-vote rule, live results.
- **Steps:** Whitelist voters, open and close voting by time, block double voting, show results.
- **Show in the viva:** How the contract stops double voting, and the privacy limits of public votes.
- **Extensions:** Add commit-reveal voting for secrecy.

3. Crowdfunding Platform

- **Problem:** Release funds only when backers approve milestones.
- **Network and tools:** Sepolia testnet.
- **Tech stack:** Solidity, Hardhat, Next.js, ethers.js.
- **Main modules:** Campaign creation, contributions, milestone requests, backer voting, refunds.
- **Steps:** Write the escrow contract, add milestone voting, refund if goal not met, test edge cases.
- **Show in the viva:** Reentrancy protection in the withdraw function.
- **Extensions:** Add an ERC-20 reward token.

4. Land Records Registry

- **Problem:** Record property ownership transfers transparently.
- **Network and tools:** Hyperledger Fabric test network, or Sepolia.
- **Tech stack:** Hyperledger Fabric chaincode (Go or JavaScript) or Solidity, React.
- **Main modules:** Property registration, transfer with approvals, history view, registrar role.
- **Steps:** Model assets, require registrar and both parties to approve transfers, show full history.
- **Show in the viva:** Why a permissioned network suits government records.
- **Extensions:** Add document hashes from IPFS.

Advanced level projects

5. Supply Chain Tracking

- **Problem:** Trace a product from farm or factory to shop.
- **Network and tools:** Hyperledger Fabric test network.
- **Tech stack:** Fabric chaincode, Node.js SDK, React, QR codes.
- **Main modules:** Organisations and roles, batch creation, hand-over events, QR trace page.
- **Steps:** Set up three organisations, record each hand-over, let buyers scan a QR to view history.
- **Show in the viva:** Endorsement policies in Fabric.
- **Extensions:** Add IoT temperature readings for cold-chain goods.

6. NFT Ticketing for Events

- **Problem:** Issue event tickets as NFTs to curb fake tickets.
- **Network and tools:** Sepolia testnet, IPFS for metadata.
- **Tech stack:** Solidity (ERC-721 via OpenZeppelin), Hardhat, React.
- **Main modules:** Minting, metadata, resale cap, gate check-in scanner.
- **Steps:** Mint tickets, store metadata on IPFS, cap resale price in the contract, verify ownership at entry.
- **Show in the viva:** How the resale cap is enforced.
- **Extensions:** Add proof of attendance tokens.

7. Decentralised File Storage with Access Control

- **Problem:** Share files with chosen users without a central server.
- **Network and tools:** IPFS, Sepolia testnet.
- **Tech stack:** Solidity, IPFS, client-side encryption, React.
- **Main modules:** Encrypt and upload, store CID on-chain, grant and revoke access, download and decrypt.
- **Steps:** Encrypt in the browser, upload, store the CID with access lists, share keys with allowed users.
- **Show in the viva:** Why encryption is needed even on IPFS.
- **Extensions:** Add expiring access.

8. Medical Record Consent Ledger

- **Problem:** Let patients control which hospitals can see their records.
- **Network and tools:** Hyperledger Fabric or Sepolia; synthetic records only.
- **Tech stack:** Fabric or Solidity, off-chain encrypted storage, React.
- **Main modules:** Patient consent grants, hospital requests, access log, revocation.
- **Steps:** Store consent on-chain, records off-chain encrypted, log every access.
- **Show in the viva:** Why records themselves stay off-chain.
- **Extensions:** Add emergency access with audit.

Research-oriented projects

9. Smart Contract Vulnerability Study

- **Problem:** Find and fix common bugs in sample contracts.
- **Network and tools:** Deliberately vulnerable contracts you write, or public teaching sets such as Ethernaut levels.
- **Tech stack:** Solidity, Hardhat, Slither, Foundry or Hardhat tests.
- **Main modules:** Vulnerable samples, exploit tests, fixed versions, tool comparison.
- **Steps:** Write contracts with reentrancy and access bugs, exploit them in tests, fix, run Slither on both.
- **Show in the viva:** One exploit and its fix, step by step.
- **Extensions:** Compare two analysis tools.

10. Gas Optimisation Benchmark

- **Problem:** Measure how coding choices change gas cost.
- **Network and tools:** Local Hardhat network.
- **Tech stack:** Solidity, Hardhat gas reporter.
- **Main modules:** Baseline contract, optimised versions, gas tables.
- **Steps:** Write versions using storage versus memory, packing and events, compare gas.
- **Show in the viva:** The gas table and why each change helps.
- **Extensions:** Compare on a Layer 2 testnet.

11. Consensus Algorithm Simulator

- **Problem:** Compare proof of work, proof of stake and PBFT in a simulation.
- **Network and tools:** Simulated nodes on your laptop.
- **Tech stack:** Python or Go, matplotlib.
- **Main modules:** Node model, three consensus modes, fault injection, metrics.
- **Steps:** Simulate nodes, inject faulty nodes, measure finality time and message count.
- **Show in the viva:** Fault tolerance limits of each algorithm.
- **Extensions:** Add network delay models.

12. Self-sovereign Identity Prototype

- **Problem:** Let students hold verifiable credentials in a wallet.
- **Network and tools:** Sepolia or a local network.
- **Tech stack:** W3C Verifiable Credentials concepts, DIDs, a JavaScript library, React.
- **Main modules:** Issuer, holder wallet, verifier, revocation registry.
- **Steps:** Issue a signed credential, store it in the wallet, verify signatures and revocation.
- **Show in the viva:** Difference between on-chain registry and off-chain credentials.
- **Extensions:** Add selective disclosure.

Report structure

A final-year report is marked on clarity as much as on code. A structure most universities accept:

1. **Abstract** (150 to 250 words): problem, method, main result.
2. **Introduction:** background, problem statement, objectives and scope.
3. **Literature review:** 8 to 15 papers or existing systems, and the gap your project fills.
4. **Requirements and design:** functional and non-functional requirements, architecture diagram, contract design (state variables, functions, roles), a sequence diagram of one transaction, and on-chain versus off-chain data split.
5. **Implementation:** modules, tools, key algorithms and screenshots.
6. **Testing and results:** unit test results, gas costs per function, testnet contract address and transaction hashes, and security checks.
7. **Conclusion and future scope.**
8. **References** in the format your department asks for (IEEE style is common).

Follow your own university's template if it gives one; it overrides this list.

Common mistakes to avoid

- Using a blockchain where a normal database would do, with no reason given.
- Storing personal data on a public chain.
- Committing private keys or seed phrases to GitHub.
- No tests for contract edge cases.

Frequently asked questions

Which blockchain project is best for a final year?

Certificate verification, voting, crowdfunding with milestones and supply chain tracking. Each has a clear reason to use shared, tamper-evident records.

Do I need real cryptocurrency?

No. Use a local Hardhat network and the Sepolia testnet. Testnet ETH is free from faucets and has no real value.

Ethereum or Hyperledger Fabric?

Ethereum testnets for public apps such as voting or NFTs. Hyperledger Fabric for business networks such as supply chains where members are known.

What do examiners ask in a blockchain viva?

Why a blockchain is needed, how consensus works, what is stored on-chain, gas costs, and how you secured the contract.

Does this PDF include full source code?

No. It gives the problem, data, stack, modules and steps for each project. You write the code yourself, which is what examiners test in the viva.

Is this PDF free?

Yes. Fill in the short form on this page and the PDF downloads right away.

Sources and further reading

- Ethereum developer documentation
- Solidity documentation
- Hardhat documentation
- OpenZeppelin Contracts
- Hyperledger Fabric documentation

About this PDF

Prepared by the Uncodemy training team as practice project ideas. Datasets and tools named are public; check each dataset's licence before use. We update it every January. Corrections: info@uncodemy.com.

More free resources and courses: uncodemy.com | info@uncodemy.com | +91 98183 66550