# **Project Proposal Guidelines**

### **Format and Due**

* **Length:** 2 pages (≈ 600–800 words) + **one diagram**.
* **File name:** docs/PROPOSAL.md in your group repo.
* **Due:** End of Week 5

### **Grading (10 pts)**

* **Clarity and Feasibility (3):** Clear problem, scoped MVP, realistic plan.
* **Research Alignment (2):** Cites which prior work/theme you build on.
* **Milestones (2):** Specific week-by-week outcomes through Week 14.
* **Architecture and Risks (2):** Diagram + top risks with mitigations.
* **Roles and Logistics (1):** Who does what; meeting cadence; repo link.

## **Required Sections (use these exact headings)**

1. **Title and One-Line Value Proposition**

*“Blockchain-Backed IoT Data Verification System for Smart Farming”*

Value Proposition: an immutable and verifiable record for agricultural sensor data, ensuring transparency and trust in environmental monitoring using blockchain technology.

1. **Problem and Stakeholders (2–4 sentences)**

Who hurts today? What’s broken? Who benefits if you solve it (end-users, operators, auditors, regulators)?

Current agricultural systems may lack a trustworthy yet simple mechanism for ensuring the integrity of environmental data. Some of these data points may be temperature, humidity, and PH levels. Without a tamper-proof log of this data, it can be altered or lost, which can hurt farmers who need that valuable information for operations of even audits of their yearly profits or net gain. It benefits farmers by providing a reliable data record, which will benefit consumers by increasing the transparency between farmers and the consumer, which in turn increases trust in the food supply chain.

1. **Research Alignment (1–2 sentences)**

Name **which theme** you’re extending:

* 1. **IoT + blockchain** (signed telemetry, gateway verification, on-chain attestations)

We will be extending IoT devices with the blockchain into farmers and their data they need, such as data logging.

1. **Platform and Rationale (pick one)**
   1. **Ethereum (Hardhat local, optional testnet)** (public EVM, event-driven UI, wallets, ERC-standards).

Briefly justify why your choice fits your problem (privacy, policy, ecosystem, tooling).

Etherum is well-suited for our project due to the smart contract capabilities with Solidity and the ecosystem of dev tools like Hardhat, Web3.js, and MetaMask. For the system focused on a transparent, immutable public log of data, Ethereum’s architecture fits the problem and project perfectly. If we use Hardhat or a local testing environment, it can allow for low cost development and testing as well.

1. **MVP Features (must-have) + Stretch**

List 2–4 **MVP** features you will demo by Week 10, and 1–2 **Stretch** targets if time allows.

Simulated IoT Data Generation: A python or node.js script that generates fake sensor data (temperature, humidity, or pH) and it’ll simulate IoT devices

On-Chain Data Hashing: A backend service hashes the sensor data using SHA-256 and stores the resulting hash along with metadata (such as timestamps) on the Ethereum blockchain.

Basic Data Logging Smart Contract: A Solidity smart contract to receive and log data hashes from authorized sources

Blockchain Interaction: A script using either web3.js or ethers.js to interact with the deployed smart contract on local or testnet blockchains.

Stretch targets:

Role-based access control: enhance the smart contract to include RBAC, which ensures only registered farmers can write data and manipulate it.

Data Visualization Dashboard: A basic web app for data visualization, using either React or Flask.

1. **Architecture Sketch (diagram)**

Show **smart contracts/chaincode**, **backend/gateway**, **UI**, **data sources** (and where identity/VCs sit). Label which events/transactions flow on-chain.

A diagram of a software process

AI-generated content may be incorrect.

1. **Security and Privacy Requirements**

One paragraph on how you’ll protect data and operations (e.g., VC-gated write APIs, endorsement policy, private data collections, input validation, rate limiting, DP noise in analytics).

The primary goal is to ensure the data integrity. We can achieve this by hashing the sensor data before committing it to the blockchain, since it’s immutable. Doing this can create a tamper-proof audit trail. To protect operations, the system will implement access control within the smart contract. One of our stretch goals includes role based access where only authorized farmers and manipulate and change the data. All the data written to the chain will be hashes of the original data, not the raw data itself, which provides an additional layer of privacy.

1. **Milestones (Weeks 6–14)**

Use week labels and concrete deliverables (e.g., “W7: vertical slice UI→contract state change”, “W10: threat model + 3 mitigations implemented”, “W12: LLM-assisted audit or analytics dashboard”).

* **W6:** Env up; skeleton contract/chaincode; first unit test; vertical slice planned.
* **W7:** Vertical slice demo (UI/CLI → contract → state change/event).
* **W8:** Feature 1 complete (Data simulation and hashing) + events consumed by client.
* **W9:** Feature 2 complete (On-chain data logging via smart contract) + basic authZ (owner-only write access).
* **W10:** Threat model; 3 mitigations implemented (e.g., input validation in contract, access control modifiers).
* **W11:** ≥10 tests for the smart contract; metrics captured (e.g., transaction latency).
* **W12:** LLM assist (audit/test) or analytics dashboard for visualizing data.
* **W13:** Freeze & polish; doc pass (README/runbook).
* **W14:** Dry-run poster/pitch; fix feedback.

1. **Team and Roles + Logistics**

PM/Scrum • Contract/Chaincode • Backend/API • Frontend/UX • DevOps/Test.

Weekly standup time + comms channel. Repo URL.

Emily: PM/Scrum & Frontend/UX

Jacob: Contract/chaincode

Jason: Backend/API

Madeline: DevOps/Test

Weekly Standup Time/Comms Channel: The weekend prior to due date on Discord

Github: <https://github.com/ehdkq/Blockchain-Main-Project.git>

1. **Top Risks (3) and Mitigations**

*Example:* “Data realism → use synthetic generator and public seed dataset; Policy complexity → start with allow-list; Chaincode bugs → unit tests + event assertions.”

**Risk 1: Smart Contract Vulnerabilities** → **Mitigation:** A bug in the Solidity code could allow unauthorized data logging or fail under certain conditions. We will mitigate this by implementing a comprehensive test suite using Hardhat to cover all contract functions and by using static analysis tools to check for common security vulnerabilities.

**Risk 2: Unrealistic Data Simulation** → **Mitigation:** The simulated sensor data may not accurately reflect real-world agricultural conditions, which could limit the demo's impact. We will research public agricultural data sets to inform our synthetic data generator, ensuring the values and patterns are as realistic as possible for the MVP.

**Risk 3: Blockchain Scalability and Cost** → **Mitigation:** Logging every sensor reading individually to a public blockchain would be slow and expensive. For this project, we will use a local Hardhat testnet to eliminate cost and latency concerns. A potential stretch goal is to explore batching techniques, where a single Merkle root is committed on-chain to represent many readings, drastically reducing transaction volume.

## **What you must have by end of Week 5**

* PROPOSAL.md committed, **public repo** initialized, project board with ≥6 stories, roles set, and timeboxed weekly updates scheduled (Week 6–14).

# **Research-Aligned Project Ideas (pick one and tailor)**

Below are concrete, **scoped** project seeds.

## **1) Privacy-Preserving Healthcare Data Sharing (3-tier)**

**Theme:** Healthcare privacy + custodian repository + local differential privacy.

**Platform:** **Fabric / NeuroBlock** (best for org roles, private data, audit).

**MVP (Weeks 6–10):**

* Patient/Provider registry with role-based actions (create consent, write encounter summary).
* **Custodian gateway** signs and logs access requests; **on-chain audit** event per access.
* Minimal LDP: add Laplace noise to an aggregate (e.g., weekly counts) before logging to chain.

**Stretch (Weeks 11–13):**

* Private Data Collections (PDC) for PHI, hash on-chain; consent revocation flow.
* Simple analytics dashboard (noisy counts) + “right-to-revoke” demo.

**Data:** Synthetic EHR snippets (CSV/JSON).

**Demo metric:** Latency of consent→write; epsilon used for DP; audit trail completeness.

## **2) SSI-FL: Self-Sovereign Identity + Federated Learning**

**Theme:** DIDs/VCs + federated training with DP; on-chain accountability.

**Platform:** **Ethereum (Hardhat local)** to emit identity/session events; Python/Node for FL loop.

**MVP:**

* Issue a VC (mock or JSON VC) to each “site”; **VC-gated start** of FL round logged on-chain.
* 3 “sites” train locally on synthetic text or tabular data; coordinator aggregates.
* Add DP noise to gradients/metrics; log each round’s metrics hash on-chain.

**Stretch:**

* Reward mechanism (ERC-20) for honest rounds; slashing on failed proof (mock).
* Basic attack simulation (label flip) with detection threshold logged.

**Data:** Synthetic ICU notes or tabular vitals (public seeds).

**Demo metric:** F1 score vs rounds; privacy parameter; on-chain event history.

## **3) Trusted Traceability for Cold-Chain Food/Pharma**

**Theme:** Supply-chain anti-counterfeit and provenance.

**Platform:** **Fabric / NeuroBlock** (multi-org roles, endorsements) or **Ethereum** (if consumer-facing).

**MVP:**

* Batch asset lifecycle (manufacture → ship → receive → dispense) with **role enforcement**.
* Each hop emits an **event** containing QR/NFC scan hash + timestamp.
* Consumer verification web page: scan code → show batch path with signatures.

**Stretch:**

* Temperature breach alerts (simulated sensor) → flagged on-chain; quarantine action.
* Aggregated KPIs (on-time %, breach %) with small DP noise.

**Data:** Synthetic batch manifests + CSV “sensor” streams.

**Demo metric:** % of batches verifiable; alerting on breach; end-to-end scan latency.

## **4) IoT → Gateway → Chain: Signed Telemetry with Alerts**

**Theme:** IoT security; signed telemetry; on-chain attestations and alerts.

**Platform:** **Ethereum (Hardhat local)** for events and quick UI, or **Fabric** if ABAC needed.

**MVP:**

* Device keys; gateway verifies signatures and **commits hash** + metadata on-chain.
* Simple **ABAC** (role claim or allow-list) to restrict who can submit telemetry.
* Alert rule (e.g., out-of-range temp) emits AlertRaised event; UI subscribes to events.

**Stretch:**

* VC-gated write at the gateway (present a VC JWT to submit).
* Batch proof (Merkle root) of daily readings to reduce on-chain cost.

**Data:** Synthetic sensor generator (Node or Python).

**Demo metric:** Alert detection latency; event integrity (signature check pass rate).

## **5) Micro-grants / Treasury for Lab Equipment (DeFi lite)**

**Theme:** Transparent disbursement with approvals; safe Solidity patterns.

**Platform:** **Ethereum (Hardhat local)**.

**MVP:**

* Multi-sig or role-based **Treasury** contract with proposals and approvals.
* Stream or milestone-based disbursement to grantees (ERC-20 “LabCredit”).
* Event-driven dashboard shows proposals, votes, payouts.

**Stretch:**

* Rate limiting and pausing; fuzz tests; static analysis and LLM-assisted audit.

**Demo metric:** Proposal→payout latency; security score from audit checklist.

## **What we expect to see in proposals (per idea)**

* **Why this theme?** (tie to research line above)
* **Why this chain?** (privacy/governance vs openness/tools)
* **MVP demo** scene in one paragraph (what we will see on screen in 60 seconds)
* **1 diagram** labeling contracts/chaincode, gateway, UI, data flow, and where **identity/VC** sits
* **Milestones** (Weeks 6–14) matching our course schedule
* **Top 3 risks** + mitigation (data realism, policy complexity, time, performance)
* **Success metrics** (e.g., auditability, latency, accuracy, privacy epsilon, verification rate)

## **Milestone template (paste into your proposal)**

* **W6:** Env up; skeleton contract/chaincode; first unit test; vertical slice planned.
* **W7:** **Vertical slice** demo (UI/CLI → contract → state change or event).
* **W8:** Feature 1 complete + events consumed by client.
* **W9:** Feature 2 complete + basic authZ (roles/VC).
* **W10:** Threat model; **3 mitigations** implemented.
* **W11:** ≥10 tests; metrics captured (perf or privacy).
* **W12:** LLM assist (audit/test) or analytics dashboard.
* **W13:** Freeze and polish; doc pass (README/runbook).
* **W14:** Dry-run of poster/pitch; fix feedback.

# **Deliverables for Week 5**

### **1. Repo template file**

Add this to each group repo at docs/PROPOSAL\_TEMPLATE.md:

# Project Proposal  
  
## 1. Title & One-Line Value Proposition  
> Example: “Trusted Traceability for Cold-Chain Vaccines — batch-level proof from manufacturer to clinic.”  
  
## 2. Problem & Stakeholders  
Who hurts today? What’s broken? Who benefits if solved?  
  
## 3. Research Alignment  
Theme: [Healthcare privacy | SSI+FL | Supply-chain | IoT+Blockchain | Badges | Treasury]   
Why it connects to instructor’s prior work:  
  
## 4. Platform & Rationale  
Choice: [Fabric/NeuroBlock | Ethereum/Hardhat]   
Why it fits our problem:  
  
## 5. MVP Features + Stretch  
MVP (2–4 features):   
Stretch (1–2 if time):  
  
## 6. Architecture Sketch  
\*(Insert diagram here)\*   
Label contracts/chaincode, backend/gateway, UI, data flow, identity/VCs.  
  
## 7. Security & Privacy Requirements  
One paragraph: how we’ll protect data & operations.  
  
## 8. Milestones (Weeks 6–14)  
- W6: Env up; skeleton contract/chaincode; first unit test; vertical slice planned  
- W7: Vertical slice demo (UI/CLI → contract → state change/event)  
- W8: Feature 1 complete + events consumed  
- W9: Feature 2 complete + basic authZ (roles/VC)  
- W10: Threat model; 3 mitigations implemented  
- W11: ≥10 tests; metrics captured  
- W12: LLM assist (audit/test) or analytics dashboard  
- W13: Freeze & polish; doc pass  
- W14: Dry-run poster/pitch; fix feedback  
  
## 9. Team & Roles + Logistics  
PM/Scrum • Contract/Chaincode • Backend/API • Frontend/UX • DevOps/Test   
Standup time + comms channel:   
Repo URL:   
  
## 10. Top Risks & Mitigations  
- Risk 1 → Mitigation   
- Risk 2 → Mitigation   
- Risk 3 → Mitigation