**Blockchain Commit Lifecycle**

The Blockchain Commit Lifecycle is a critical component of the traceability platform's integrity framework. It ensures that every key event-ranging from cultivation milestones to inventory transfers and regulatory submissions-is captured immutably, verifiably, and transparently. This lifecycle guarantees the **non-repudiation, tamper-evidence, and historical traceability** of all activities within the regulated supply chain of Saint Lucia.

**1. Event Trigger and Data Generation**

The lifecycle begins when a system or user-initiated action occurs, such as:

* Registering a plant batch
* Uploading a lab result
* Conducting a product transfer or sale
* Destroying cannabis waste

These events are captured via the mobile or web application and routed through the appropriate microservice (e.g., Cultivation, Lab, Inventory), ensuring domain-level data segregation and processing integrity.

**2. Data Hashing and Transformation**

Once the data is validated by the backend service (built using Node.js and NestJS), the system creates a **SHA-256 hash** of the event payload. This cryptographic hash acts as a **fingerprint** of the event, ensuring that even the slightest modification of the underlying data would generate a completely different hash.

The original event data is securely stored in a **PostgreSQL database** (augmented by TimescaleDB for temporal data), while the hash becomes the reference token for the audit and immutability process.

**3. Ledger Submission**

The system submits the hash-along with minimal identifying metadata such as timestamp, entity ID, and action type-to an **immutable ledger**, either:

* **Amazon QLDB** (for the MVP), which offers serverless, cryptographically verifiable journaling, or
* **Hyperledger Fabric** (for future scalability or sovereign deployments), which supports permissioned distributed ledger technology.

Each transaction in the ledger is **cryptographically chained** to the previous transaction using Merkle Tree structures or internal digest chaining, ensuring **append-only behavior** and preventing any alteration or deletion of previous records.

**4. Anchor Storage and Indexing**

Once committed, the ledger returns a **proof-of-write hash** or transaction ID, which is stored back in the primary system database alongside the original event record. This creates a **bi-directional link** between operational data and its blockchain representation, enabling both internal audits and third-party verification.

This step ensures:

* Traceability of event → hash → ledger entry
* Reproducibility: a verifier can rehash the event and compare it to the ledger
* Legal defensibility in case of audits, disputes, or regulatory inquiries

**5. Auditability and Verification**

Using visual tools like Kibana or Metabase, regulators and auditors can explore:

* A chronological trail of events
* Ledger entries with hash references
* Anomalies such as duplicate hashes, unauthorized deletions, or inconsistent timestamps

Blockchain entries are **tamper-evident by design**, and any manipulation attempt will result in broken hash chains, immediately flagging integrity violations.

**6. Benefits for Regulators and Stakeholders**

* **Transparency**: All lifecycle actions are observable via secure dashboards
* **Trust**: Immutable blockchain anchoring eliminates suspicion of manual overrides
* **Compliance**: Supports international regulatory standards such as GxP, HIPAA, and GDPR
* **Automation**: Ensures audit logs and regulatory submissions are backed by verifiable cryptographic proof

**Conclusion**

The Blockchain Commit Lifecycle provides the cryptographic backbone of the traceability system. By incorporating **hash chaining, tamper-evidence, and blockchain anchoring**, the architecture guarantees that every regulated action is:

* **Traceable**
* **Immutable**
* **Verifiable**

This positions Saint Lucia's traceability infrastructure as a **trusted, modern, and defensible system**, designed not only to meet current compliance demands but also to anticipate future regulatory evolution.