Core Chain: A Platform for Secure Collaborative Research on Rare Diseases

Date:26th Nov'25

Supervisor: Dr. Ghulam abbas
Co-Supervisor: Mr. ahsan shah

Group Members:

Dua-e-Zahra	2022151
Aiza Azeem	2022077
Saad	2022509
Mahad ali	2022262

Revision History:

Revision History	***Date***	***Comments***
1.00	21st Nov'25	Changes suggested by the co-supervisor
2.00	24th Nov'25	Changes suggested by the supervisor

Document Approval:

The following document has been accepted and approved by the following:

Signature	***Date***	***Name***

List of Contents

1 INTRODUCTION

This document specifies the software requirements for CoreChain; a secure, collaborative research platform designed to enable distributed medical AI model training without centralizing sensitive patient data.

1.1. PURPOSE

The platform focuses on the Tuberculosis (TB) Chest X-ray analysis use case, utilizing publicly available datasets (Shenzhen and Montgomery) to demonstrate its functionality. This SRS defines the system's functional and nonfunctional requirements, serving as a reference for developers, researchers, administrators, and stakeholders involved in the system's design, implementation, and evaluation. It encompasses all components of Core Chain, including the distributed model training framework, secure data coordination layer, user dashboards, and supporting modules for adaptive learning and continuous model improvement.

1.2. INTENDED AUDIENCE

1) **Developers and System Engineers**: to understand implementation level details and system architecture.
2) **Healthcare Researchers**: to interpret how the platform facilitates collaborative AI model training while ensuring data privacy.
3) **Administrators and Project Managers**: to monitor system objectives, constraints, and deployment requirements.
4) **Testers and Evaluators**: to validate that the system meets functional and performance goals.

1.3. PRODUCT SCOPE

Core Chain aims to overcome the challenges of fragmented datasets, privacy regulations, and limited trust among healthcare institutions by providing a secure and collaborative platform for medical research. It enables decentralized model training through privacy-preserving methods, ensuring that patient data remains within institutional boundaries while maintaining compliance with data protection laws. A blockchain layer built on Hyperledger Fabric ensures immutable logging, transparent contribution tracking, and automated incentives via smart contracts. The system combines distributed training, secure record management, and intuitive dashboards for monitoring collaboration and performance. With strong encryption and adaptive model optimization, Core Chain supports continuous improvement as new TB data becomes available, offering a scalable, privacy-preserving, and ethically governed framework for advancing collaborative healthcare research.

Table 1: Terms used in this document and their description

Name	Description
FL	Federated Learning

GDPR	General Data Protection Regulation
HIPAA	Health Insurance Portability and Accountability Act
API	Application Programming Interface

1.4. REFERENCES

1. National Institute of Health, China, 2017. *Shenzhen & Montgomery Chest X-ray Dataset*.
2. Yang, Q., Liu, Y., Chen, T. & Tong, Y., 2019. *Federated Machine Learning: Concept and Applications*. *ACM Transactions on Intelligent Systems and Technology*.
3. Abuzied, H., et al., 2024. *Privacy-Preserving AI Collaboration Using Blockchain Frameworks*. *Cluster Computing Journal*.
4. Linux Foundation, 2024. *Hyperledger Fabric Documentation, Version 2.5*.
5. Owkin, 2024. *Substra: Federated Learning Platform Documentation*. [Online] Available at: https://www.substra.ai [Accessed 22 November 2025].
6. MedCo, 2023. *MedCo Platform Overview*. [Online] Available at: https://medco.epfl.ch [Accessed 22 November 2025].
7. European Union, 2018. *General Data Protection Regulation (GDPR)*.
8. U.S. Department of Health & Human Services, 1996. *Health Insurance Portability and Accountability Act (HIPAA)*.
9. IEEE, 1998. *IEEE Std 830-1998 – Recommended Practice for Software Requirements Specifications*.

2 OVERVIEW

Core Chain is a collaborative AI platform designed for environments where data privacy and institutional autonomy are essential. It enables healthcare organizations to contribute to shared model development while keeping their sensitive datasets local. The system integrates secure training mechanisms, transparent auditability, and intelligent reward management into a unified framework. Core Chain ensures that institutions can participate in model improvement confidently, with clear visibility into contributions, system operations, and overall training progress through an interactive dashboard.

2.1. THE OVERALL DESCRIPTION

The platform operates by allowing each institution to train a model locally on its private data. Instead of transferring raw information, only encrypted model updates are exchanged. These updates are collected, verified, and combined to refine a shared global model. The system maintains a complete, tamper-proof record of training events, enabling traceability for all contributions. Through its web interface, users can view performance metrics, participation statistics, and collaboration activity in real time.

2.2. PRODUCT PERSPECTIVE

Core Chain functions through the integration of several key technologies:

1. **Federated Learning** – Provides the mechanism for distributed training, ensuring data remains within institutional boundaries while still contributing to a global model.
2. **Blockchain (Hyperledger Fabric)** – Acts as the verification and audit layer, recording model updates, enforcing access rules, and executing smart contracts for automated interactions.
3. **Dashboard Interface** – Serves as the control and visualization layer, giving users tools to monitor training cycles, manage collaborations, and track institutional performance.
4. **Continual Learning** – Enables the global model to adapt to new information over time, preserving previously learned knowledge and improving stability as the system evolves.
5. **Reinforcement Learning–Based Incentives** – Dynamically adjusts rewards based on contribution quality and engagement, promoting sustained and meaningful participation.

2.3. PRODUCT FUNCTIONS

Core Chain enables institutions to train AI models locally on medical images while securely exchanging processed updates without transferring raw data. The system aggregates and validates these local updates to refine a shared diagnostic model and maintains verifiable records of all activities, participation, and version history. It provides visual dashboards for monitoring model accuracy, progress, and collaboration metrics, and manages user authentication, authorization, and institutional roles. Additionally, Core Chain supports long-term model improvement through reinforcement learning and continual learning as new data becomes available. Together, these functions create a transparent, privacy-preserving workflow for medical AI research and development.

2.4. USER CHARACTERISTICS

Core Chain supports several user classes, each defined by its level of access, responsibilities, and technical expertise:

Table 2: Users in the project

User	Description	Privileges
Administrator	System operator or institutional IT lead	Manage users, nodes, and smart contracts
Institution Partner	Hospitals/research centers	Host Fl nodes, manage local data, Federated Learning

2.5. DESIGN AND IMPLEMENTATION CONSTRAINTS

Development of Core Chain is governed by several constraints:

1. **Regulatory Compliance:** Must adhere to GDPR and HIPAA data protection requirements.
2. **Data Privacy:** Raw medical data cannot leave institutional premises.
3. **Technology Dependence:** Uses specific frameworks (e.g., Hyperledger Fabric, TensorFlow) that define compatible versions and APIs.
4. **Hardware Limitations:** Participating nodes must meet minimum resource thresholds to ensure consistent performance.
5. **Programming Languages:** Python and JavaScript are mandatory for core development.
6. **Deployment Tools:** Containerization through Docker and orchestration with Kubernetes are required for scalability.

2.6. ASSUMPTIONS AND DEPENDENCIES

- Participating institutions have valid research consent and compliant datasets.
- Federated learning SDKs (e.g., Flower) and blockchain frameworks (Hyperledger) are operational.
- Third-party libraries and frameworks such as TensorFlow and Hyperledger Fabric remain actively supported.
- Reliable internet connectivity exists for synchronization between nodes.
- Containerization and orchestration tools (Docker, Kubernetes) are available in the deployment environment.

3 STATE OF THE ART

3.1. LITERATURE REVIEW

Medical research increasingly relies on artificial intelligence (AI) to assist in disease diagnosis and prognosis, yet the sensitivity of healthcare data creates significant challenges for centralized model training. Studies in recent years have shown that Federated Learning (FL**)** offers a promising solution by enabling collaborative model development without transferring raw patient data. Yang et al. (2019) introduced the concept of FL as a distributed paradigm where models are trained locally and aggregated globally, ensuring privacy preservation and compliance with data protection regulations.

Further advancements have combined FL with Blockchain to enhance transparency and trust among collaborating entities. Abuzied et al. (2024) proposed blockchain-based frameworks that record and validate model updates through immutable ledgers, thereby eliminating single points of failure and enabling secure audit trails. Other works have explored optimization strategies that dynamically adjust model parameters based on feedback, improving convergence and accuracy across heterogeneous datasets.

In the medical imaging domain, several studies have demonstrated the feasibility of using FL for chest X-ray classification tasks, including Tuberculosis detection**.** Research utilizing the Shenzhen and Montgomery datasets has validated the effectiveness of deep learning models such as CNNs and DenseNets for TB screening. However, most prior systems relied on centralized training, limiting their scalability and compliance with privacy laws. These findings underscore the need for a decentralized yet verifiable platform that facilitates cross-institutional medical AI collaboration—an objective that Core Chain directly addresses.

3.2. EXISTING SYSTEMS

- **Flower**

It provides scalability, supports multiple machine learning frameworks such as TensorFlow and PyTorch, and allows simulation of large-scale heterogeneous environments. However, despite its flexibility, Flower lacks native mechanisms for privacy preservation and does not provide transparent or auditable contribution logging. Core Chain improves upon this by integrating encryption techniques such as homomorphic encryption and coupling them with blockchain-based audit trails to ensure security and accountability. (Yang et al., 2019).

- **FLoBC**

This framework combines federated learning with blockchain to create a distributed and verifiable training environment. While it successfully demonstrates the potential of integrating these two technologies, it remains in a prototype stage and lacks enforcement of domain-specific policies, particularly those needed in the healthcare sector. Core Chain extends this idea by incorporating smart contracts specifically designed to enforce medical data-use policies and ethical compliance for rare disease research. (Abuzied et al., 2024).

- **Substra**

The Substra framework, developed by Owkin, is a production-ready platform for large-scale federated learning deployments. It offers traceability and supports orchestration through Kubernetes, making it suitable for industrial environments. However, its complex setup requirements make it less accessible to smaller institutions, and it does not provide built-in incentive mechanisms. Core Chain simplifies deployment through Docker and Helm charts, enabling one-click installation on hospital servers, and introduces token-based reward automation to encourage collaboration. (Linux Foundation, 2024).

- **MedCo**

It focuses on secure multi-party analytics using homomorphic encryption and supports operations such as regression and survival analysis. While it provides strong privacy guarantees and post-quantum security, MedCo is limited to statistical analysis and does not support full federated learning pipelines. Core Chain builds upon MedCo's secure computing foundation by incorporating end-to-end federated learning workflows and adding blockchain-based incentive and provenance tracking features. (Yang et al., 2019; Abuzied et al., 2024).

LIMITATIONS OF EXISTING SYSTEMS:

- **Data Privacy Concerns:** Many existing systems require partial data sharing for validation or central aggregation, which can compromise patient confidentiality and breach healthcare data protection laws.
- **Lack of Transparency:** Frameworks often lack immutable audit trails, making it difficult to trace contributions, verify updates, or ensure accountability among participants.
- **No Incentive Mechanism:** Most current platforms do not reward active collaboration or contribution, discouraging institutions from consistent participation.
- **Static Learning Process:** Many systems are unable to incorporate new data without retraining from scratch, resulting in inefficiency and limited adaptability over time.
- **Limited Medical Domain Focus:** Most existing frameworks are general-purpose and lack dedicated support for healthcare-specific needs such as regulatory compliance, ethical oversight, and imaging workflows.

WHAT ARE WE OFFERING?

Core Chain enables secure, privacy-preserving collaboration for training diagnostic models without sharing raw data. It ensures transparency through verifiable logs, supports continuous learning from new data, automates model validation and aggregation, and provides intuitive dashboards for researchers to monitor performance and contributions efficiently.

4 USER/SYSTEM REQUIREMENTS

This section outlines all functional and technical requirements necessary for users and system components to interact effectively with Core Chain.

4.1. External Interface Requirements

These requirements describe how the platform connects with users, hardware, software, databases, and communication channels.

4.1.1 User Interface

The Core Chain platform provides a web-based interface accessible via standard browsers to all users. Key characteristics include:

- Researcher Dashboard: Displays model accuracy, updates, and collaboration metrics.
- Admin Portal: For user management, auditing, and institution onboarding.
- Clean UI using React/Tailwind with secure login (OAuth2.0 + biometrics).

4.1.2 Hardware Interface

- Institutional servers (GPU-enabled preferred) or low-resource edge devices.

- Containers deployed via Docker/Kubernetes.
- HTTPS/TLS for secure data transmission; RESTful APIs for control commands and monitoring.
- Local nodes perform model training and send processed updates to the central aggregator.

4.1.3 Software Interface

- **Backend:** Python (FL) + Node.js (smart contracts).
- **Blockchain:** Hyperledger Fabric.
- **Database:** On-chain ledger + off-chain metadata.
- **APIs:** RESTful for model and metadata exchange.

4.1.4 Database Interface

- Blockchain serves as immutable storage for model metadata and transactions.
- Encrypted off-chain database stores performance and participant data.

4.1.5 Communication Interface

- TLS 1.3 for all communications.
- Peer-to-peer encrypted FL network using gRPC.

5 FUNCTIONAL REQUIREMENTS

FR- 1 Local Model Training
FR- 2 Encrypted Data Transmission
FR- 3 Blockchain Logging
FR- 4 Smart Contracts
FR- 5 User Management
FR- 6 Real-time Dashboard
FR- 7 Audit and Traceability
FR- 8 Token Incentivization
FR- 9 Deployment Scalability

5.1. Functional Requirements with Traceability information

FR- 1: Local Model Training

The system shall enable hospitals and institutions to train AI models locally without sharing or transferring raw patient data.

Requirement ID	FR-1		**Requirement Type**		Functional			**Use Case #**		UC-1	
Status	***New***	☑	***Agreed-to***	☑	***Baselined***		***Rejected***				
Parent Requirement #	None										
Description	The system shall allow participating hospitals and research institutions to train AI models locally on their own datasets without transferring raw patient data.										
Rationale	Ensures privacy preservation and compliance with medical data protection regulations while enabling collaborative model training.										
Source	Project Scope Document				**Source Document**		Project Scope Document				
Acceptance/Fit Criteria	Local nodes must successfully complete at least one training round and contribute encrypted model updates to the global model.										
Dependencies	Requires installed federated learning framework (e.g., Flower) and compatible local datasets.										
Priority	***Essential***	☑	***Conditional***		***Optional***						
Change History	Initial version										

FR- 2: Encrypted Data Transmission

The system shall encrypt all model updates using homomorphic encryption before transmission.

Requirement ID	FR-2		**Requirement Type**		Functional			**Use Case #**		UC-2	
Status	***New***	☑	***Agreed-to***	☑	***Baselined***		***Rejected***				
Parent Requirement #	None										
Description	The system shall encrypt all model gradients and parameters using homomorphic encryption before transmission.										
Rationale	Prevents data leakage and maintains confidentiality of patient information during model aggregation.										
Source	Project Scope Document				**Source Document**		Project Scope Document				

Acceptance/Fit Criteria	Encrypted gradients cannot be decrypted without private keys; model aggregation must complete successfully with encrypted inputs.						
Dependencies	Requires cryptographic key generation and secure key storage modules.						
Priority	***Essential***	☑	***Conditional***		***Optional***		
Change History	Initial version						

FR- 3: Blockchain Logging

The system shall log all model updates and transactions immutably on the blockchain.

Requirement ID	FR-3		**Requirement Type**		Functional		**Use Case #**		UC-3	
Status	***New***	☑	***Agreed-to***	☑	***Baselined***		***Rejected***			
Parent Requirement #	FR- 1									
Description	The system shall record every model update, contribution, and transaction immutably on a blockchain ledger.									
Rationale	Provides transparency, auditability, and traceability of all research contributions.									
Source	Project Scope Document				**Source Document**		Project Scope Document			
Acceptance/Fit Criteria	Each update generates a unique blockchain transaction hash viewable through the dashboard.									
Dependencies	Requires a functional Hyperledger Fabric network and consensus among nodes.									
Priority	***Essential***	☑	***Conditional***		***Optional***					
Change History	Initial version									

FR- 4: Smart Contracts

The system shall use smart contracts to automate rewards, validate updates, and enforce policies.

Requirement ID	FR-4		**Requirement Type**		Functional		**Use Case #**		UC-4	
Status	***New***	☑	***Agreed-to***	☑	***Baselined***		***Rejected***			
Parent Requirement #	FR- 3									

Description	The system shall use smart contracts to automate reward distribution, validate model updates, and enforce collaboration policies.						
Rationale	Ensures fair and automatic incentivization, reduces manual oversight, and maintains policy compliance.						
Source	Project Scope Document		**Source Document**	Project Scope Document			
Acceptance/Fit Criteria	Smart contracts execute automatically upon valid contribution confirmation and update the contributor's balance.						
Dependencies	Requires blockchain integration and token incentive mechanism.						
Priority	***Essential***	☑	***Conditional***		***Optional***		
Change History	Initial version						

FR- 5: User Management

The system shall enforce role-based access control and secure authentication for all users.

Requirement ID	FR-5		**Requirement Type**		Functional		**Use Case #**		UC-5	
Status	***New***	☑	***Agreed-to***	☑	***Baselined***		***Rejected***			
Parent Requirement #	None									
Description	The system shall implement role-based access control (RBAC) and secure authentication for researchers and administrators.									
Rationale	Protects the system from unauthorized access and ensures proper segregation of privileges.									
Source	Project Scope Document				**Source Document**		Project Scope Document			
Acceptance/Fit Criteria	Each role (Admin, Researcher, Institution) has restricted permissions; login requires OAuth2.0 and biometric verification.									
Dependencies	Depends on user management module and authentication API integration.									
Priority	***Essential***	☑	***Conditional***		***Optional***					
Change History	Initial version									

FR- 6: Dashboard Visualization

The system shall provide a researcher dashboard showing model performance and contributions in real time.

Requirement ID	FR-6		**Requirement Type**		Functional		**Use Case #**		UC-5		
Status	***New***	☑	***Agreed-to***	☑	***Baselined***		***Rejected***				
Parent Requirement #	FR- 1										
Description	The system shall provide a researcher dashboard that visualizes model performance, accuracy trends, and contribution statistics in real time.										
Rationale	Enables participants to monitor progress, evaluate model quality, and encourage collaboration.										
Source	Project Scope Document				**Source Document**		Project Scope Document				
Acceptance/Fit Criteria	Dashboard displays updated metrics (accuracy, loss, contribution count) after every global aggregation round.										
Dependencies	Requires backend data pipeline and frontend visualization modules.										
Priority	***Essential***	☑	***Conditional***		***Optional***						
Change History	Initial version										

FR- 7: Audit Trail

The system shall keep audit logs of all user actions, model updates, and contract executions.

Requirement ID	FR-7		**Requirement Type**		Functional		**Use Case #**		UC-5		
Status	***New***	☑	***Agreed-to***	☑	***Baselined***		***Rejected***				
Parent Requirement #	FR-3										
Description	The system shall maintain complete audit logs of all user actions, model updates, and smart contract executions.										
Rationale	Ensures accountability and provides an audit trail for regulatory compliance.										
Source	Project Scope Document				**Source Document**		Project Scope Document				
Acceptance/Fit Criteria	Logs must record timestamp, user ID, transaction hash, and action type, retrievable by admins on request.										

Dependencies	Requires blockchain and off-chain database integration.						
Priority	***Essential***	☑	***Conditional***		***Optional***		
Change History	Initial version						

FR- 8: Reward System

The system shall reward researchers with tokens based on their verified contributions.

Requirement ID	FR-8		**Requirement Type**		Functional		**Use Case #**		UC-5	
Status	***New***	☑	***Agreed-to***	☑	***Baselined***		***Rejected***			
Parent Requirement #	FR-4									
Description	The system shall implement a token-based incentive mechanism to reward researchers proportionally to their verified contributions.									
Rationale	Motivates institutions to participate actively in federated training and data sharing.									
Source	Project Scope Document				**Source Document**		Project Scope Document			
Acceptance/Fit Criteria	Contributors automatically receive tokens via smart contract once their update is validated on the blockchain.									
Dependencies	Depends on functional blockchain wallet and reward management modules.									
Priority	***Essential***	☑	***Conditional***		***Optional***					
Change History	Initial version									

FR- 9: Deployment Scalability

The system shall support scalable deployment via Docker and Kubernetes.

Requirement ID	FR-10		**Requirement Type**		Functional		**Use Case #**		UC-5	
Status	***New***	☑	***Agreed-to***	☑	***Baselined***		***Rejected***			
Parent Requirement #	FR-1									
Description	The system shall support modular and scalable deployment using Docker containers and Kubernetes orchestration.									

Rationale	Facilitates easy setup across multiple institutions and ensures high availability under varying workloads.						
Source	Project Scope Document		**Source Document**		Project Scope Document		
Acceptance/Fit Criteria	The system must deploy successfully across at least three institutional nodes using container orchestration.						
Dependencies	Requires Kubernetes cluster and Docker images for all system components.						
Priority	***Essential***	☑	***Conditional***		***Optional***		
Change History	Initial version						

6 NONFUNCTIONAL REQUIREMENTS & SOFTWARE SYSTEM ATTRIBUTES

6.1. Performance Requirements

The system shall deliver fast and efficient performance under all operational conditions. Each model aggregation round must complete within 10 seconds, while blockchain transactions should be recorded and verified within 15 seconds. The platform must support at least 2 institutional nodes operating concurrently, and the dashboard should load metrics within 10 seconds of request. Resource scaling must activate automatically when CPU usage exceeds 75%, and encryption overhead must not exceed 10% of the total training time. In the event of a system failure, recovery should be completed within two aggregation rounds to ensure minimal disruption.

6.2. Security Requirements

Security is paramount for the Core Chain platform. All data and model updates shall be encrypted. Multi-factor authentication is required for all administrative and researcher accounts, and the system must comply with GDPR and HIPAA regulations. Blockchain technology will provide immutable logging of all actions and model updates, while access keys must be securely stored and restricted to authorized users only.

6.3. Usability Requirements

The platform must be user-friendly and accessible. The dashboard should offer a simple and intuitive interface suitable for non-technical users, with real-time progress indicators during model training and aggregation. Built-in tooltips and help sections will guide users through all features, and the interface must comply with accessibility standards to ensure inclusive use by all participants.

6.4. Maintainability Requirements

Core Chain should be maintainable with minimal effort. The system will follow a modular microservices architecture, and the source code must adhere to standard documentation and style guidelines. Version control using Git shall be implemented to track code changes, manage collaboration, and maintain code integrity. Continuous Integration and Continuous Deployment (CI/CD) pipelines will automate testing and deployment processes. All module updates must maintain backward compatibility to prevent disruptions in existing functionality.

6.5. Compliance Requirements

The platform must adhere to all relevant regulations and institutional policies. GDPR, HIPAA, and internal ethics policies shall be followed rigorously. Audit logs must be readily available for compliance and ethical reviews, and regular security and data integrity audits shall be performed to ensure continued adherence to these standards.

6.6. Portability Requirements

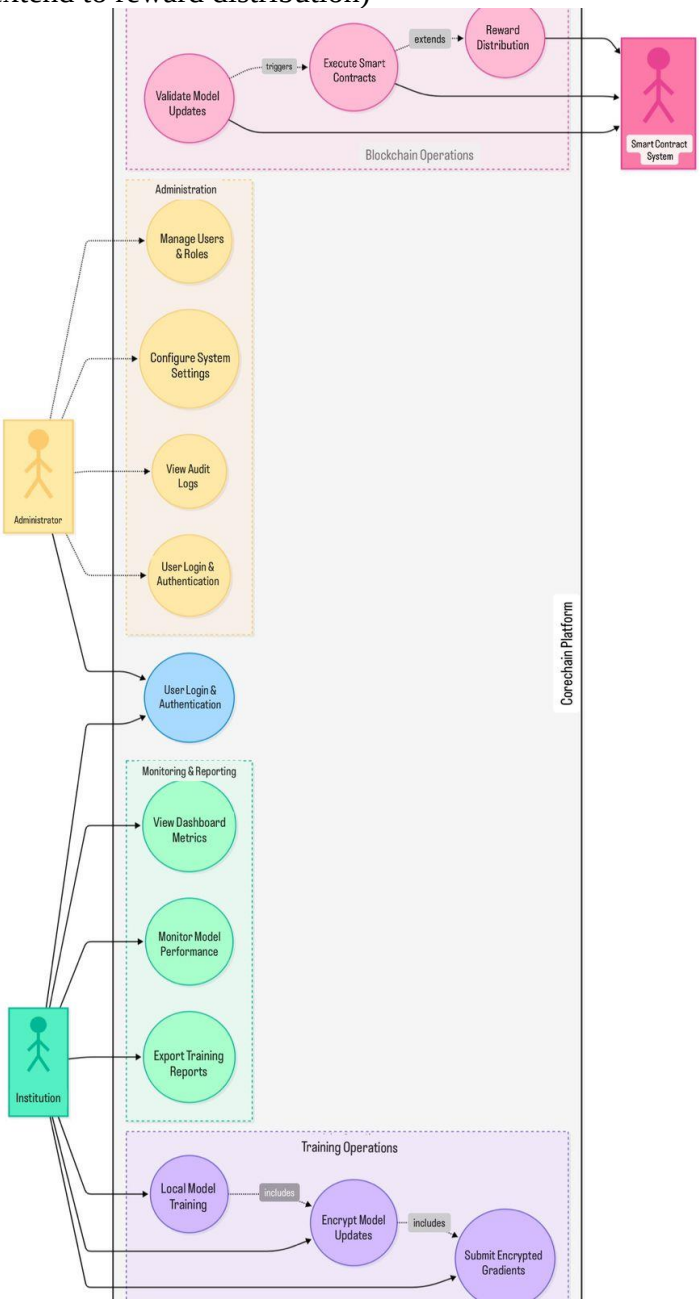
The platform should be highly portable and easy to deploy. Core Chain will be containerized using Docker and deployable via Kubernetes. It must run on both Linux and Windows servers, and the dashboard should be accessible from all modern web browsers and devices. Additionally, the system must integrate seamlessly with institutional APIs through REST interfaces to support interoperability across diverse environments.

7 Project Design/Architecture

7.1. 4+1 ARCHITECTURE VIEW MODEL

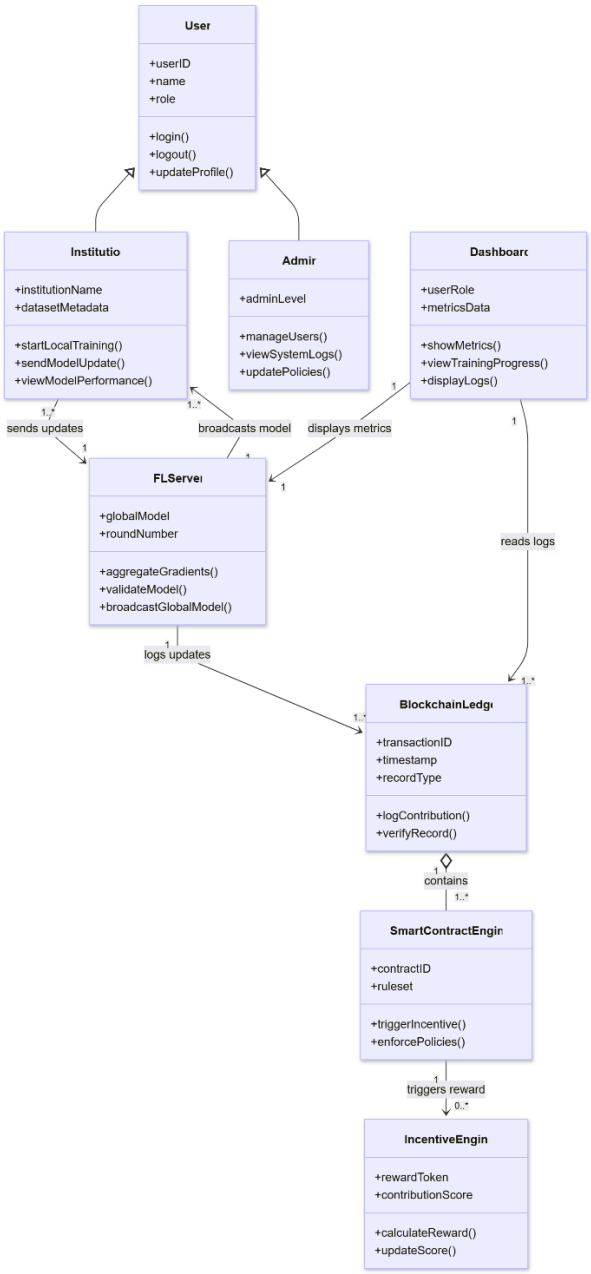
7.1.1 Use Case View:

The Use Case View presents the system's functionality from an external perspective. Three primary actors interact with CoreChain: Institution Partners who train models and monitor performance, Administrators who manage the system, and the Smart Contract System that automates blockchain operations. The diagram shows 14 core use cases organized into functional groups. Relationships between use cases are indicated using UML stereotypes: `<includes>` shows mandatory dependencies (e.g., training includes encryption), while `<extends>` shows optional enhancements (e.g., smart contracts extend to reward distribution)



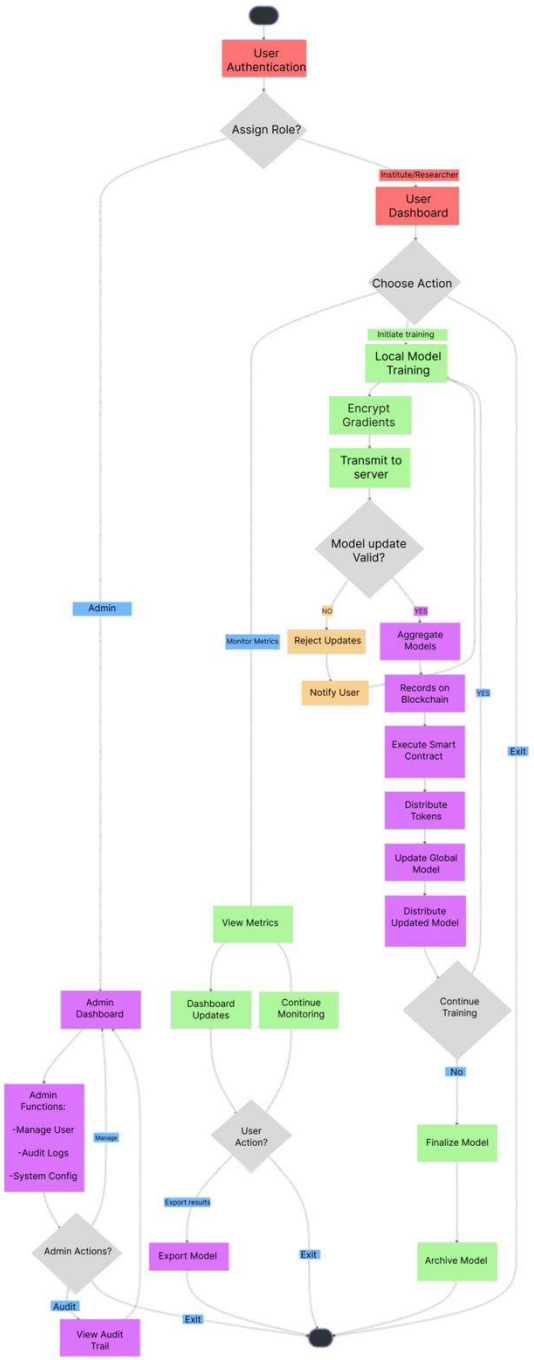
7.1.2 Logical View

This class diagram models a federated learning system where Institutions and Admins inherit core authentication and profile features from the User class. Institutions perform local training and send model updates to the FLServer, which aggregates, validates, and redistributes the global model. Users view training progress and system insights through the Dashboard, which also accesses logs stored in the BlockchainLedger. The ledger maintains immutable records and contains SmartContractEngine components that automatically enforce rules and trigger incentives. These incentives are processed by the IncentiveEngine, which calculates and updates reward scores. Overall, the diagram shows how training, logging, and incentive mechanisms integrate to create a secure and transparent federated learning workflow.



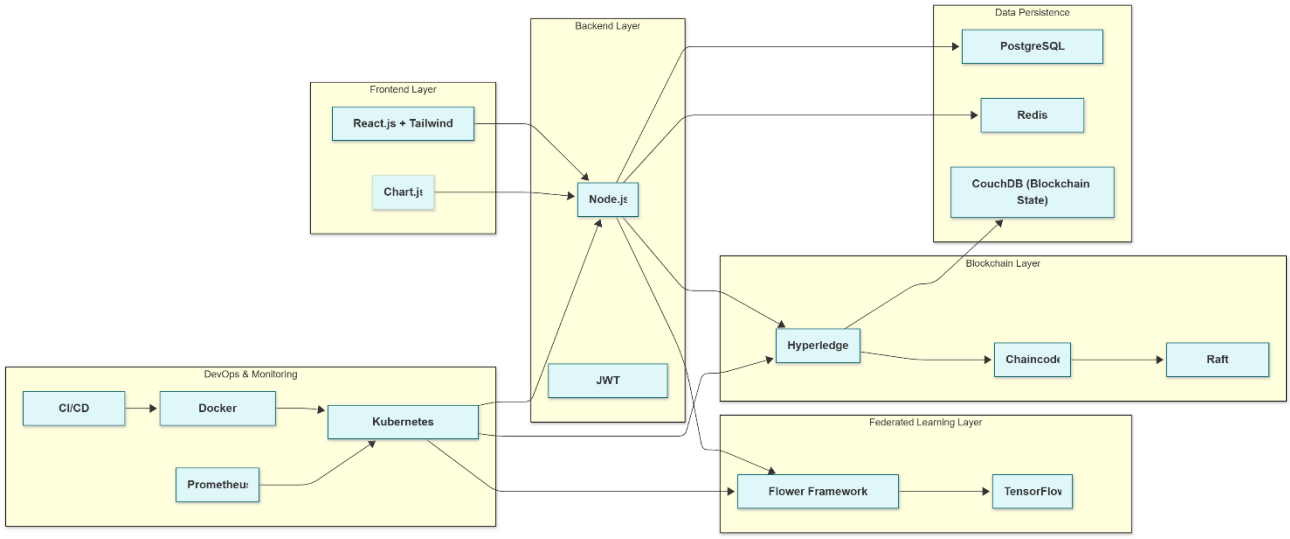
7.1.3 Process View:

The Process View depicts CoreChain's runtime behavior during a complete federated learning cycle. The workflow begins with user authentication and role assignment, branching into three paths: Admin, Researcher, or Institution. Institution partners initiate local training, encrypt gradients, and submit updates. The system validates these updates; valid submissions trigger blockchain logging and smart contract execution for reward distribution, while invalid updates are rejected with notification. The aggregated global model is distributed back to institutions, completing one training round. This iterative process continues until the model converges.



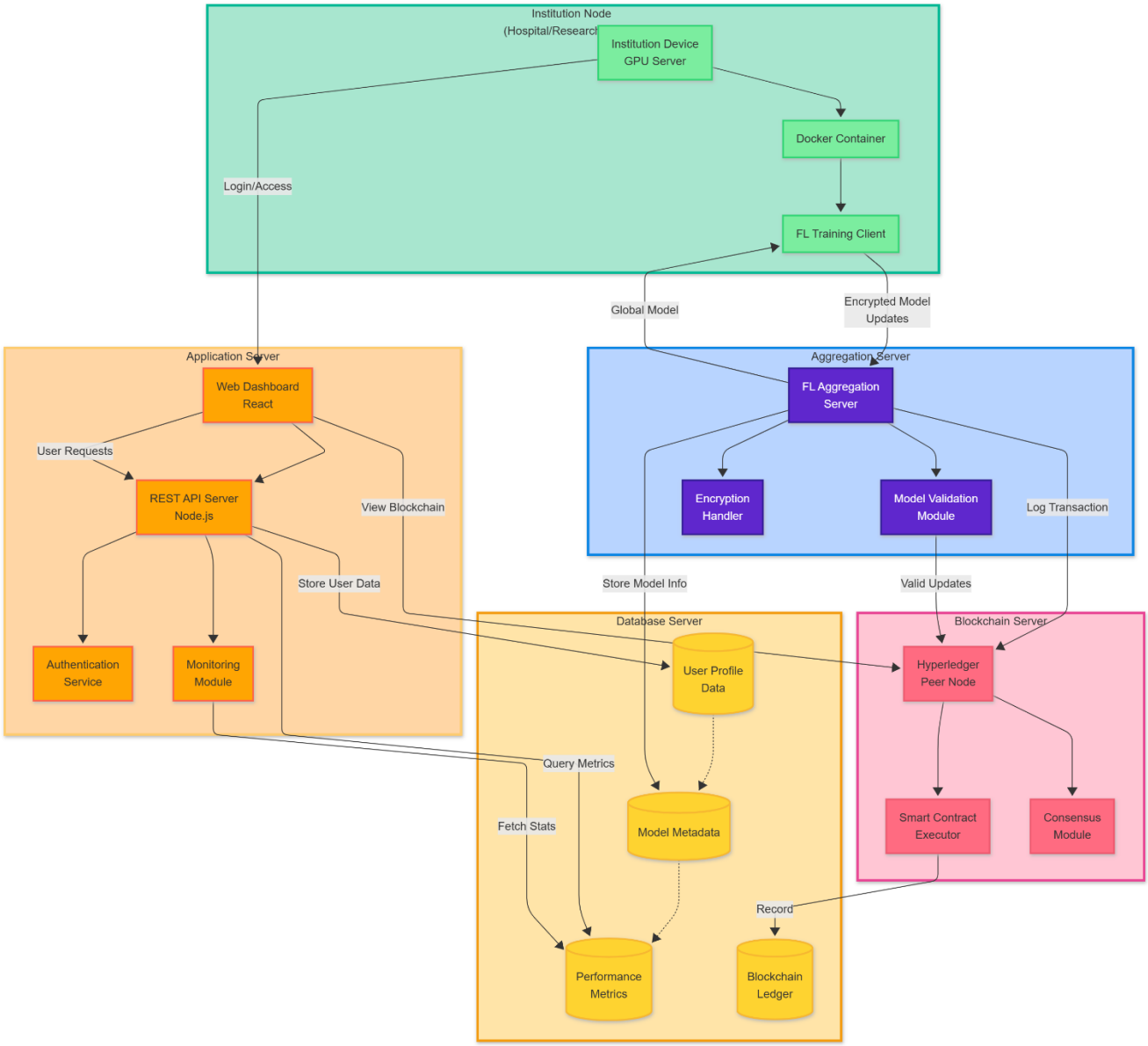
7.1.4 Development View

The Development View outlines CoreChain's technical architecture from a developer's perspective. The frontend utilizes React with Tailwind CSS for responsive UI and Chart.js for visualizations. The backend is built with Node.js and Express, implementing JWT-based authentication. The FL layer leverages the Flower framework with Python for distributed training, supporting TensorFlow and PyTorch. Blockchain operations run on Hyperledger Fabric with chaincode written in Go and Raft consensus. Data persistence uses PostgreSQL for metadata, Redis for caching, and CouchDB for blockchain state. The entire system is containerized with Docker and orchestrated via Kubernetes, with CI/CD pipelines automating deployment and Prometheus/Grafana providing monitoring.



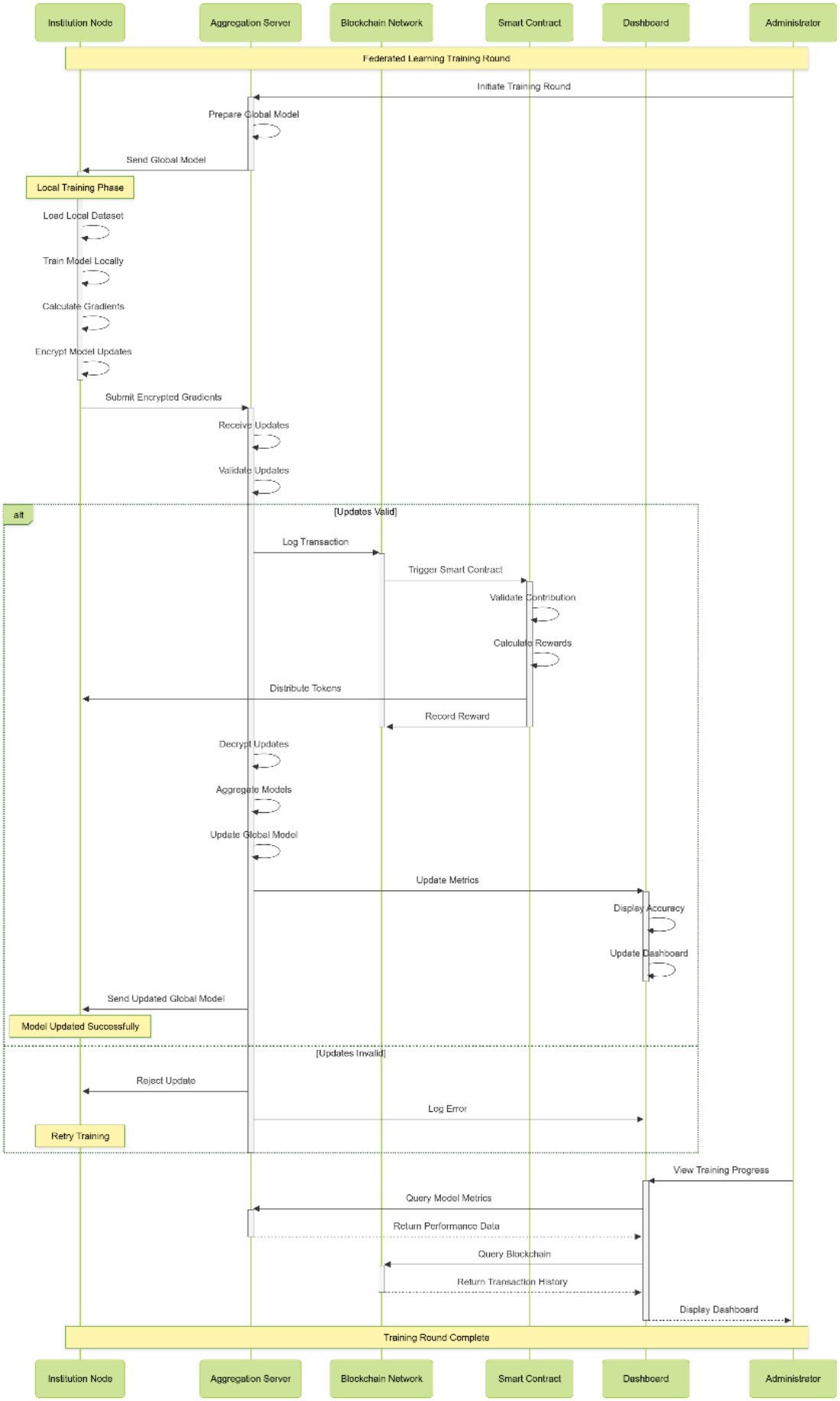
7.1.5 Physical View

The Physical View illustrates CoreChain's deployment topology across distributed infrastructure. Institution nodes consist of GPU-enabled servers running Docker containers that host FL training clients and maintain local patient data. These nodes connect through a load balancer via encrypted channels to a cloud-based Kubernetes cluster hosting the aggregation server, API backend, and web dashboard. The blockchain network comprises three Hyperledger Fabric peer nodes and an orderer node for consensus, ensuring decentralization and fault tolerance. The storage layer separates concerns: PostgreSQL stores user metadata, Redis provides session caching, and CouchDB maintains the blockchain ledger. All communication uses TLS 1.3 encryption, ensuring data security in transit.



7.1.6 Sequence Diagram

The Sequence Diagram traces a complete federated learning training round chronologically. The administrator initiates training, prompting the aggregation server to distribute the global model to institution nodes. Each institution trains locally on its dataset, encrypts the resulting gradients, and submits them to the aggregation server. The server validates these updates; valid submissions trigger blockchain transaction logging and smart contract execution for reward distribution. The smart contract validates contributions, calculates rewards, and distributes tokens to participating institutions. Simultaneously, the aggregation server decrypts and aggregates the model updates, creating a new global model that is redistributed to all institutions. Invalid updates trigger rejection notifications, allowing institutions to retry. Throughout this process, the dashboard queries both the aggregation server and blockchain to display real-time metrics to administrators and researchers.


Institution Node
Aggregation Server
Blockchain Network
Smart Contract
Dashboard
Administrator
Federated Learning Training Round
Initiate Training Round
Prepare Global Model
Send Global Model
Local Training Phase
Load Local Dataset
Train Model Locally
Calculate Gradients
Encrypt Model Updates
Submit Encrypted Gradients
Receive Updates
Validate Updates
alt
[Updates Valid]
Log Transaction
Trigger Smart Contract
Validate Contribution
Calculate Rewards
Distribute Tokens
Record Reward
Decrypt Updates
Aggregate Models
Update Global Model
Update Metrics
Display Accuracy
Update Dashboard
Send Updated Global Model
Model Updated Successfully
[Updates Invalid]
Reject Update
Log Error
Retry Training
View Training Progress
Query Model Metrics
Return Performance Data
Query Blockchain
Return Transaction History
Display Dashboard
Training Round Complete
Institution Node
Aggregation Server
Blockchain Network
Smart Contract
Dashboard
Administrator