الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة سعيدة د. مولاي الطاهر

كلية الرياضيات و الإعلام الآلي و الاتصالات السلكية و اللاسلكية

قسم: الإعلام الآلي

Mémoire de Master en informatique

Spécialité : Réseaux Informatique et Systèmes Répartis

RISR

T h è m e

Blockchain-based Decentralized Security For Internet of Things Applications

- **Présenté par :**

MERABET yamina

NASRI anes

- **Dirigé par :**

Dr. TALEB fadia

Année universitaire 2025-2026

ملخص

يُعاني قطاع الرعاية الصحية الحديث من تشتت السجلات الطبية وانعزالها، مما يُشكل مخاطر جسيمة على خصوصية المرضى وأمن البيانات وكفاءة تقديم الرعاية. وتُعد قواعد البيانات المركزية التقليدية عُرضةً للاختراقات ونقاط الضعف. يقترح نظام **HEALIX** (سجل الصحة وتبادل المعلومات) نظامًا شاملاً لامركزيًا لإدارة بيانات الرعاية الصحية، يستفيد من تقنية **BlockChain** والعقود الذكية. من خلال دمج تقنية السجلات الموزعة **(DLT)** والأمان التشفيري وبنية صارمة للتحكم في الوصول القائم على الأدوار **(RBAC)** يضمن نظام **HEALIX** سلامة السجلات الصحية الإلكترونية **(EHR)** وعدم قابليتها للتغيير وسهولة الوصول إليها. ويتمتع المرضى بملكية كاملة لبياناتهم الطبية والتحكم بها، مع منح الوصول تلقائيًا لمقدمي الرعاية الصحية المُعتمدين. علاوة على ذلك، يُتيح دمج أجهزة إنترنت الأشياء مراقبة صحية آلية وفورية، مما يُسد الفجوة بين الرعاية الوقائية وتخزين البيانات بشكل آمن وشفاف.

Abstract

The modern healthcare industry struggles with fragmented, siloed medical records, posing significant risks to patient privacy, data security, and efficient care delivery. Traditional centralized databases are highly vulnerable to breaches and single points of failure. HEALIX (Health Ledger & Information eXchange) proposes a comprehensive, decentralized healthcare data management system leveraging Blockchain technology and Smart Contracts.

By integrating distributed ledger technology (DLT), cryptographic security, and a strict role-based access control (RBAC) architecture, HEALIX ensures the integrity, immutability, and accessibility of Electronic Health Records (EHR). Patients are granted full ownership and control over their medical data, autonomously granting access to authorized healthcare providers. Furthermore, the integration of IoT devices allows for automated, real-time health monitoring, bridging the gap between preventive care and secure, transparent data storage.

Résumé

Le secteur moderne de la santé est confronté à des dossiers médicaux fragmentés et cloisonnés, ce qui entraîne des risques importants pour la confidentialité des patients, la sécurité des données et l'efficacité de la prise en charge médicale. Les bases de données centralisées traditionnelles sont particulièrement vulnérables aux violations de sécurité et aux points de défaillance uniques. HEALIX (Health Ledger & Information eXchange) propose un système complet et décentralisé de gestion des données de santé, basé sur la technologie blockchain et les contrats intelligents.

En intégrant la technologie de registre distribué (DLT), la cryptographie et une architecture stricte de contrôle d'accès basé sur les rôles (RBAC), HEALIX garantit l'intégrité, l'immutabilité et l'accessibilité des dossiers médicaux électroniques (DME). Les patients disposent d'un contrôle total et d'une pleine propriété de leurs données médicales, leur permettant d'accorder de manière autonome l'accès aux professionnels de santé autorisés. De plus, l'intégration des dispositifs IoT permet une surveillance automatisée et en temps réel de la santé, comblant ainsi l'écart entre les soins préventifs et un stockage des données sécurisé et transparent.

Dedications & Acknowledgments

We wish to express our deepest gratitude to our supervisor, **Dr. Fadia Taleb**, for her invaluable guidance, patient mentorship, and insightful feedback throughout this research. Her expertise helped shape the HEALIX project from a concept into a tangible platform.

We are also grateful to the faculty and staff at Dr. Moulay Tahar's University for providing the resources, facilities, and academic environment that made this work possible.

We also thank the jury members, who will agree to read and evaluate this work.

We would like to dedicate this work to our parents and families, whose unwavering belief, sacrifice, and endless encouragement have been the foundation of every achievement in our lives.

And to all those who believe that technology, when guided by empathy, can heal.

Table of Contents

List of tables

List of Figures & Listings

General Introduction

The modern healthcare industry faces significant challenges in data management, security, and interoperability. Patient records are often siloed across various hospitals and clinics, making it difficult for healthcare providers to access a comprehensive medical history. Furthermore, centralized database systems are vulnerable to data breaches, unauthorized modifications, and single points of failure, which compromise patient privacy and life-saving information.

HEALIX (Health Ledger & Information eXchange) is a decentralized, blockchain-based healthcare platform designed to solve these critical issues. By leveraging the power of distributed ledger technology (DLT), smart contracts, and cryptographic security, HEALIX provides a secure, transparent, and immutable medical record management system with built-in role-based access control.

The core vision of HEALIX is to empower patients with true ownership of their health data while enabling seamless, secure collaboration between healthcare providers. By shifting from traditional centralized storage to a decentralized architecture, HEALIX ensures that medical data is tamper-proof, permanently available, and fully auditable.

HEALIX represents a paradigm shift in healthcare data management, moving towards a future where medical records are secure by default, patients are in control, and healthcare providers can deliver better, data-driven care without compromising privacy.

The thesis is organized as follows:

- The first chapter details the fundamental concepts of the blockchain technology.
- The second chapter explores the benefits and applications of blockchain technology in healthcare and IoMT as well as discussing the challenges and limitations.
- The third chapter will be purely the design of our proposed system.
- The fourth chapter will focus on the technical aspects of our system detailing the tools used and their implementation.

Chapter 1

Blockchain Technology

1.1 Introduction

In recent years, blockchain technology has emerged as one of the most transformative innovations across various industries. Originally conceptualized as the underlying architecture for digital currencies, its potential has rapidly expanded beyond the financial sector. Today, blockchain is recognized as a foundational technology capable of revolutionizing supply chains, identity management, and, crucially, healthcare data management systems like HEALIX. By providing a decentralized, trustless, and secure framework, blockchain addresses many of the inherent vulnerabilities found in traditional client-server models.

1.2 Definition

A “**blockchain”** is a decentralized, distributed, and public digital ledger that is used to record transactions across many computers. The design ensures that any involved record cannot be altered retroactively without the alteration of all subsequent blocks. In simpler terms, it is a specialized type of database that stores data in discrete packages called "blocks", which are chronologically chained together using cryptographic hashes. This creates an append-only, immutable history of data.[4]

1.3 Comparison Between Databases and Blockchain Technology

While both traditional databases and blockchains store information, their architecture, administration, and trust models differ significantly.

Table 1.1: comparison between traditional databases and blockchain[4]

Feature	Traditional Database (RDBMS/NoSQL)	Blockchain Technology
Architecture	Centralized (Client-Server model).	Decentralized (Peer-to-Peer).
Authority & Control	Managed by a central administrator or organization.	Distributed consensus; no single point of failure or control.
Data Mutability	Data can be easily updated, modified, or deleted (CRUD operations).	Immutable (Append-only); once recorded, data cannot be changed.
Transparency	Restricted to authorized users; heavily siloed (isolated).	High transparency; all participants share a copy of the ledger.
Performance	High throughput, low latency.	Generally slower due to necessary consensus mechanisms.
Trust model	Trust is placed in the central authority.	Trustless; trust is derived from cryptographic proofs and consensus.

1.4 History

The evolution of blockchain can be summarized in several key phases:

1991 - The Early Concepts: Researchers Stuart Haber and W. Scott Stornetta introduced the concept of a cryptographically secured chain of blocks to timestamp digital documents, preventing them from being backdated or tampered with.[1, 4]

2008 - The Birth of Bitcoin: An anonymous individual or group known as Satoshi Nakamoto published the Bitcoin white paper, detailing a purely peer-to-peer version of electronic cash. It successfully solved the "double-spending" problem without a central authority.[29]

2009 - The Genesis Block: The Bitcoin network launched, mining the first block (the Genesis Block).[1, 30]

2014 - Ethereum and Smart Contracts: Vitalik Buterin and his team launched Ethereum. While Bitcoin was solely a payment network, Ethereum introduced "**smart- contracts"** programmable, self-executing contracts residing on the blockchain. This opened the door for decentralized applications (dApps), including healthcare platforms.[4]

Present Day - Enterprise & Industry Adoption: Blockchain is being adopted for enterprise solutions worldwide. Frameworks are currently being built to handle complex logistics, secure voting, decentralized finance (DeFi), and secure electronic health records (EHR).[4]

1.5 Functionality

The functionality of a blockchain network relies on the seamless interaction of several critical components:

- *Peer-to-Peer (P2P) Network:* Instead of a central server, the network consists of interconnected nodes (computers). Every node maintains a full or partial copy of the blockchain.
- *Consensus Mechanisms:* Before a new block is added to the chain, the nodes must agree on its validity. Mechanisms like Proof of Work (PoW), Proof of Stake (PoS), and Proof of Authority (PoA) are used to achieve this agreement securely.

- *Cryptography:* Public-key cryptography is used to secure identities and transactions. Users possess private keys (for signing transactions) and public keys (serving as their addresses, like in MetaMask).
- *Smart Contracts:* These are self-executing programs stored on the blockchain. In systems like HEALIX, smart contracts autonomously manage user roles and enforce access control without human intervention.

1.6 Structure

The structure of a blockchain is essentially a linked list of sequential blocks. Every block in the chain is strictly connected to the previous one via cryptography, forming a continuous historical record from the genesis block to the very latest block. If someone tries to change the data in a past block, the link is broken, invalidating the rest of the chain.

1.6.1 The Layers of Blockchain Architecture

Table 1.2: blockchain architecture derived from M. Rifat Hossain et al's work in [4] and Jianqi Mu et al. [30]

Layer	Primary Function	Key Components	Developer Focus
Network Layer	Peer-to-peer communication	Nodes, Gossip Protocols, message propagation	Node configuration, network topology
Data Layer	Transaction and block storage	Blocks, Merkle trees, chain structure	Data modeling, storage optimization
Consensus Layer	Agreement among nodes	PoW, PoS, PBFT, DPoS	Validator setup, finality understanding
Cryptography Layer	Security and verification	Hash functions, digital signatures, encryption	Key management, signature verification
Smart Contract Layer	Programmable logic	Contract code, state variables, events	Solidity, Rust, contract security
Execution Layer	Transaction processing	Virtual machines, gas calculations, state transitions	Gas optimization, execution efficiency
Application Layer	User interaction	dApps, wallets, APIs, SDKs	Frontend integration, UX design

1.6.1.1 Network Layer

This foundational layer governs the peer-to-peer (P2P) topology and node discovery protocols essential for decentralization. In a medical context, this layer dictates who is allowed to connect to the network.[30, 4].

When it comes to gossip protocols, Some networks use a DHT (Distributed Hash Table), a type of decentralized distributed system, for node discovery. In such a system, nodes maintain a table containing information about other nodes in the network. For instance, Ethereum uses a custom protocol called Kademlia for peer discovery which is a type of DHT.[4]

The peer-to-peer architecture means there is no central server to attack or shut down. Full nodes or “Miners” maintain a complete copy of the ledger (e.g., hospital data centers, regulatory auditors), ensuring data redundancy and independent verification.

Miners are designated to bundle transactions into new blocks and add them to the chain. They perform the computationally heavy lifting required for consensus.[30, 4]

Light nodes (e.g., patient mobile devices or clinic workstations) download only block headers and rely on full nodes for transaction verification, preserving local storage.

Node discovery typically utilizes a Domain Name System (DNS) seed list or hardcoded bootstrap nodes to establish initial connections. Critically, for healthcare, this layer must often be combined with a VPN or Transport Layer Security (TLS) configurations to ensure the P2P traffic remains encrypted in transit and compliant with HIPAA (Health Insurance Portability and Accountability)'s transmission security requirements, even if the underlying blockchain is permissioned.[30, 4]

1.6.1.2 Data Layer

The data layer defines how information gets organized and stored on the blockchain. This is where blocks, transactions, and Merkle trees come into play, as shown in figures 1.1 and 1.2 bellow detailing the aforementioned components.

- The Block:

The container for data. A typical block consists of a block header and a block body containing the transaction data (such as a new medical record hash).[4]

- Header:

It is used to identify the particular block in the entire blockchain. A block header is hashed periodically by miners by changing the nonce value as part of normal mining activity, also, three sets of block metadata are contained in the block header.[4]

- The Hash:

Every block is processed through a cryptographic algorithm (like SHA-256) to produce a unique, fixed-length string of characters known as the block's hash. This acts like a digital fingerprint for the block.[4]

- Previous Hash:

Every new block includes the hash of the immediately preceding block. This is the "chain" in blockchain. If a malicious actor tries to alter data in an older block, its hash changes. Because the subsequent block holds the *“old”* hash, the link is broken, and the entire network immediately rejects the tampering attempt.[4]

- Timestamp and Nonce:

Blocks include timestamps showing exactly when they were created. A "nonce" (number used once) is also included in systems utilizing Proof of Work to help miners generate a valid block hash.[4]

- Merkle Root:

It is a type of data structure frame of different blocks of data. A Merkle root stores all the transactions in a block by producing a digital fingerprint of the entire transaction. It allows the users to verify whether a transaction can be included in a block or not.[4]

This layer defines the immutable structure of the medical record trail. Information is batched into **blocks**, each containing a **header** and a list of transactions. The header encapsulates the **Previous Block Hash** (creating the cryptographic chain of custody vital for audit trails), a **Timestamp** (establishing the exact chronology of a clinical event), a **Nonce** (used in mining), and the **Merkle Root**. The Merkle Root is particularly efficient in medicine; it allows a patient to prove a specific lab result is included in a block without revealing the rest of their medical history contained in that same block. Transactions (e.g., *Patient A accessed Record X*) are hashed pairwise up the Merkle Tree, ensuring any tampering with a single data point instantly changes the root hash, breaking the chain and alerting the system to data corruption. [30, 4, 45]

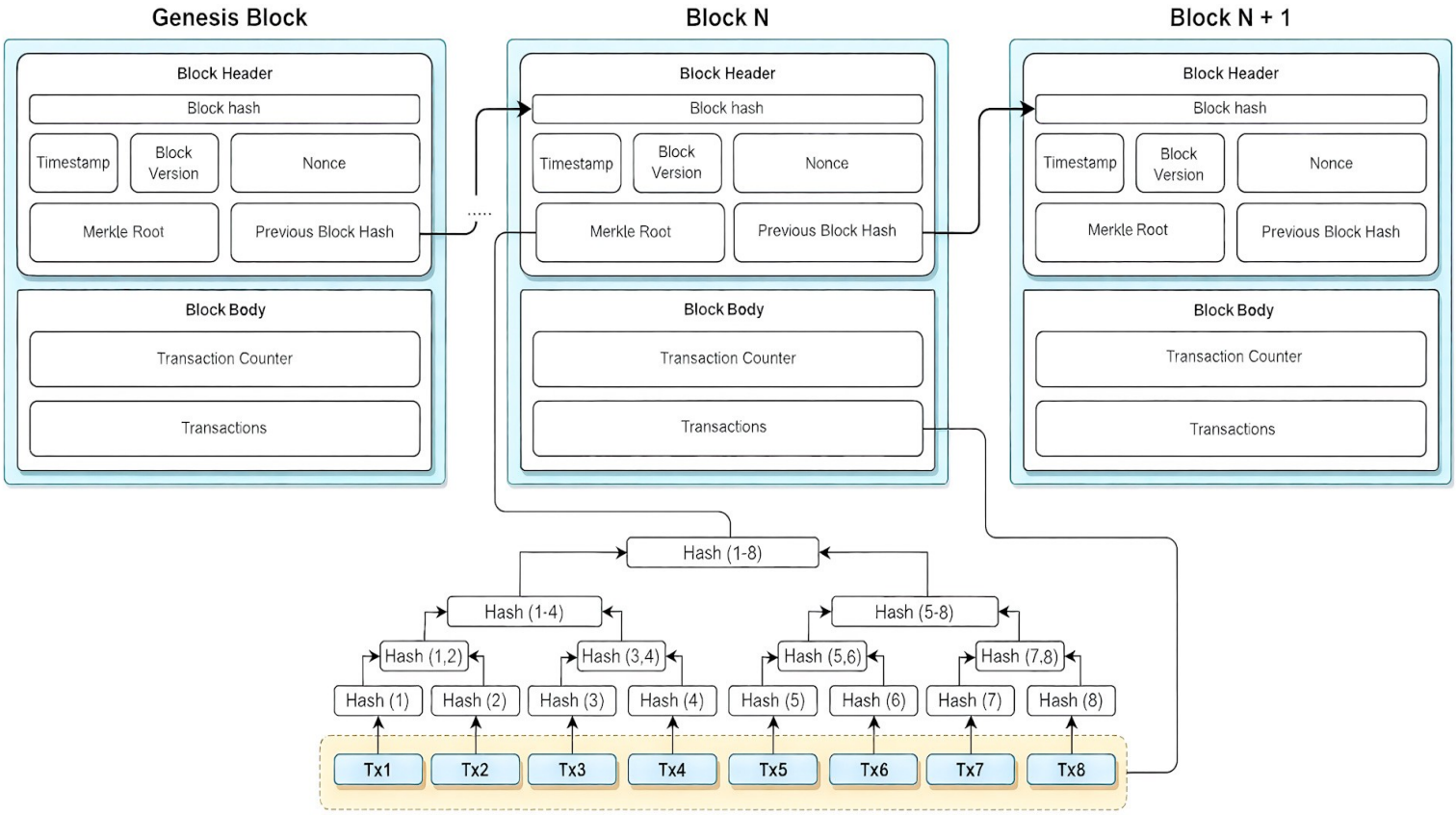

Figure 1.1 Block structure with block sequence of a blockchain.[4]

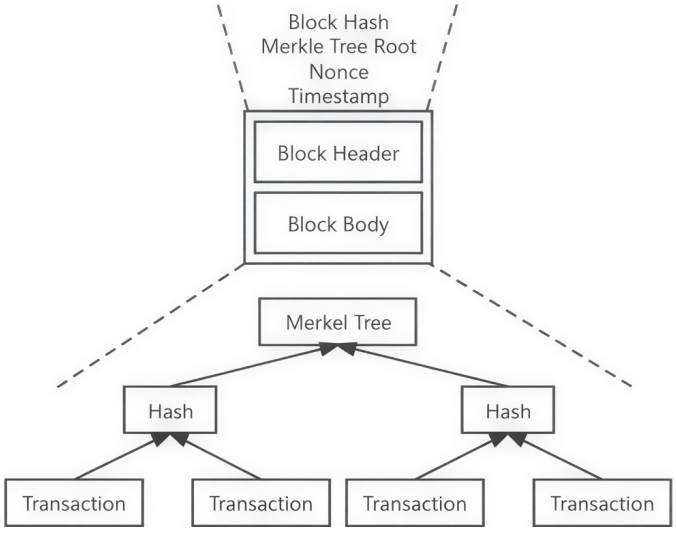

Figure 1.2 Data Structure of Block. [30]

1.6.1.3 Consensus Layer

Consensus mechanisms solve the fundamental problem of getting thousands of independent computers to agree on a single version of truth.

"The purpose of the consensus mechanism is to obtain an agreement on the impact of a transaction on a ledger update and to validate the transaction."[30]

Table 1.3: consensus mechanisms compiled from [4, 45]

Consensus Mechanism	Description
Proof of Work (PoW)	Miners solve complex mathematical problems to validate and add blocks to the blockchain. The first to solve the problem is rewarded. This mechanism is used in Bitcoin and Ethereum (before Ethereum 2.0).
Proof of Stake (PoS)	Validators stake cryptocurrency as collateral, with the chance to create new blocks based on the amount staked. It is more energy-efficient than PoW. Commonly used in Ethereum 2.0.

Consensus Mechanism	Description
Delegated Proof of Stake (DPoS)	Token holders (stakeholders) vote for validators (delegates) who are responsible for validating transactions and creating blocks.This mechanism improves scalability and transaction speed
Proof of Authority (PoA)	Pre-approved validators sign blocks, Used in private or permissioned blockchains. It relies on the reputation of participants to validate transactions
PBFT (Practical Byzantine Fault Tolerance)	PBFT is a deterministic consensus mechanism designed for permissioned blockchain networks where the identities of validators are known. It enables the network to reach agreement on a single state of the ledger even in the presence of malicious (Byzantine) nodes, provided that fewer than one-third of all nodes are faulty.

1.6.1.4 Cryptography Layer

The cryptographic layer is the security backbone of a blockchain, ensuring data integrity, authenticity, and non-repudiation without reliance on a central authority.

1.6.1.4.1 Public Key Cryptography in Practice

Blockchain technology relies on public-key cryptography also known as asymmetric cryptography to handle encryption, decryption, and digital signatures to ensure data security and message authenticity. Users create a key pair comprising a public key (used for encrypting messages) and a private key (kept secret for decrypting messages encrypted with the public key).[4]

1.6.1.4.2 Symmetric Cryptography

Uses a single key for both encrypting and decrypting data. It is fast but requires secure key sharing among parties. (e.g, AES-256 Encryption).

1.6.1.4.3 Asymmetric Cryptography

A user's Public Key acts as their pseudonymous medical ID, while the Private Key is used to cryptographically sign a consent form or an access request.

1.6.1.4.4 Hashing & Hashing Functions

Hashing is a process that converts an input of any length into a fixed-size string of characters. Data integrity relies on SHA-256 (Bitcoin-derived chains) or Keccak-256 (Ethereum-derived). However, medical data on-chain is never stored as plaintext. Only cryptographic hashes of the medical records are stored; the actual imaging files or clinical notes reside in the Storage Layer (see below). Hash functions are irreversible (one-way).[4]

For example: The phrase “Hello World” produces this SHA-256 hash:

a591a6d40bf420404a011733cfb7b190d62c65bf0bcda32b57b277d9ad9f146e

Now change just one letter to lowercase “w” in “world”:

64ec88ca00b268e5ba1a35678a1b5316d212f4f366b2477232534a8aeca37f3c

The two outputs look completely unrelated. This is the avalanche effect in action. An attacker cannot make small adjustments to data and hope for predictable changes in the hash.

1.6.1.4.5 Digital Signatures

Utilizing asymmetric cryptography, a sender cryptographically "signs" a transaction using their private key. Anyone else on the network can use the sender's public key to verify that the transaction indeed came from them and hasn't been tampered with.[4]

Table 1.4: comparing main hashing functions.[4]

Blockchain	Hash Algorithms	Hash/Signature Size
Bitcoin, Ethereum, Litecoin, Zcash, Ripple, NXT, Monero, Byteball.	SHA-256	256 bits
Ethereum	Ethash (Keccak-256, Keccak-512)	256 / 512 bits
Litecoin	Scrypt	256 bits
Monero	RandomX	256 bits
IOTA	Keccak-384	384 bits

1.6.1.5 Smart Contract Layer

Smart contracts are self-executing scripts deployed on the blockchain that automate healthcare workflows. Smart contracts transformed blockchain from a simple ledger into a programmable platform. These programs live on the blockchain and run automatically when predetermined conditions are met. No intermediary can stop them, modify them, or censor specific users.[41, 42, 4]

Programming languages like Solidity, which is used on the Ethereum blockchain, are often used to make smart contracts. The code sets the rules and requirements for the deal, and once it is on the blockchain, it can't be changed. The trait of immutability makes sure that the terms of the contract can't be changed without the agreement of all parties. Once the smart contract code is written then it is compiled into bytecode, a machine-readable form of the code. This bytecode is then executed on the blockchain's virtual machine, a crucial component of the blockchain protocol responsible for handling smart contract execution.[4]

The virtual machine ensures that the smart contracts' actions are transparently and securely executed across the decentralized network.

This vital interplay between programming languages, bytecode, and the virtual machine paves the way for efficient and automated agreement enforcement mechanisms within blockchain ecosystems, making it an integral aspect of this transformative technology.[4]

1.6.1.5.1 Advantages

Removes intermediaries, increases execution speed, reduces administrative costs, and ensures precise execution without human error or manipulation, enforce trustless logic (e.g., "Release MRI (magnetic resonance imaging) result to Dr. Smith *only if* patient consent token is active and insurance eligibility is verified").[4, 42]

1.6.1.5.2 Disadvantages

Code is immutable once deployed; a bug in a smart contract governing emergency access could be catastrophic. Furthermore, due to the oracle problem [32], the contract cannot natively verify real-world events (e.g., "Patient is unconscious"), it relies on an external trusted data feed. This layer replaces manual back-office processes for claims adjudication and consent management.

1.6.1.6 Execution Layer

This layer defines the state transition mechanics and updates the blockchain state. This is where the smart contract code actually runs, gas gets calculated, and state transitions happen.

Transaction: In this context, a transaction is not a payment but a state change request (e.g., *Update_Access_Permission* or *Log_Audit_Event*). a transaction consists typically of a recipient address, a sender address, and a value.[4]

Two primary execution models dominate the landscape: UTXO and account-based systems.

1.6.1.6.1 UTXO Model: Bitcoin's Approach

Bitcoin uses the Unspent Transaction Output (UTXO) model. Instead of tracking account balances, it treats state as discrete coins/tokens, which is excellent for asset tracking but inefficient for complex clinical logic.[4, 37, 45]

1.6.1.6.2 Account-Based Model: Ethereum's Design

Ethereum uses an account-based model similar to traditional banking. Each account has a balance, and transactions adjust these balances directly. It maintains a global state trie, mapping patient addresses directly to their current access control list. This model is far more intuitive for managing evolving patient data permissions.[37]

"The Account-Based approach involves the modification of balances associated with individual user accounts through transactions."[4]

1.6.1.6.3 Gas and Execution Costs

The Ethereum Virtual Machine (EVM) is introduced by Ethereum, a Turing-complete, stack-based virtual environment. This is where smart contracts are executed, often written in Solidity.[4, 43]

Every computation/operation in the EVM costs "gas" to prevent infinite loops and spam. In a medical private/permissioned network, gas is often set to zero or abstracted away via pre-paid infrastructure costs so the end-user (doctor/nurse) never sees a fee when saving a note.

Understanding gas optimization is crucial for developers. Inefficient code does not just waste money, it limits the contract's usability. During network congestion, poorly optimized contracts become prohibitively expensive to use. Techniques like loop optimization proposed by Tamara.B et al. [43], and the Code Mining method proposed by Avik.B et al. [44], and efficient data structures can reduce costs significantly. [41, 43]

1.6.1.7 Storage Layer

This is the most critical architectural decision for medical platforms due to the Right to Erasure (GDPR - The General Data Protection Regulation) and Volume of Imaging Data.

1.6.1.7.1 On-Chain Storage

Only cryptographic pointers (Content Identifiers - CIDs) and audit logs reside here. Storing an MRI (Magnetic Resonance Imaging) scan (200MB+) on-chain is technically impossible and legally problematic (immutability violates deletion rights).[53]

1.6.1.7.2 Off-Chain Storage

Data resides on encrypted IPFS - The InterPlanetary File System, hospital PACS (Picture Archiving and Communication Systems), or secure cloud buckets, Decentralized storage solutions like IPFS and Arweave complement blockchain by hosting files off-chain while storing only content hashes on-chain.

"Off-chain, or Layer 2, aims to alleviate the burden on the blockchain by handling data storage in external environments while maintaining a secure and verifiable connection to the main chain".[53]

1.6.1.7.3 On-chain vs. Off-chain Storage

To maintain compliance and preserve network efficiency, heavy clinical files (MRIs, lengthy charts, genomic data) are stored “off-chain” such as on secure HIPAA-compliant (Health Insurance Portability and Accountability) cloud servers or distributed networks like IPFS. The blockchain itself (on-chain) only stores lightweight cryptographic hashes and access control pointers. This Hybrid approach ensures performance by balancing cost and scalability while keeping the best of both worlds. [53]

If a patient exercises their right to be forgotten, the off-chain data is deleted or the patient's access token is destroyed. While the hash remains on the blockchain indefinitely, it becomes mathematically useless without the source data.[4]

Table 1.5: comparison between on-chain storage vs off-chain[53]

Feature	On-Chain Storage	Off-Chain Storage
Data Types	Hashes, Smart Contracts, DIDs, Metadata.	High-res images (MRIs), PDFs, full database records.
Cost & Scalability	Exceptionally high cost, very limited size.	Low cost, highly scalable for large payload sizes.
Privacy Compliance	Challenging (due to permanence).	Easy (can delete/encrypt actual data upon request).

1.6.1.8 Application Layer

The application layer is what users actually interact with, the user interface for clinicians and patients. Wallets (software managing private keys), exchanges, and NFT marketplaces all live here. This layer translates blockchain complexity into intuitive interfaces that non-technical users can navigate.[30]

In a well-designed medical platform, the user should be unaware they are using blockchain; they simply click "Share Record," and the wallet silently signs the transaction in the background (Transparency).

1.6.2 Types of Blockchain Architecture

1.6.2.1 Public Blockchains

"A public blockchain is a decentralized digital ledger. It enables transparent recording and verification of transactions across a distributed network of computers."[4] Completely open and permissionless. Anyone can join, read, write, and participate in consensus, While offering maximum trustlessness, they are categorically unsuitable for storing any Protected Health Information (PHI) directly due to the inability to delete data under GDPR/HIPAA and the risk of exposing metadata to global adversaries.[4]

1.6.2.2 Private Blockchain

Permissioned networks controlled by a single organization. Access and consensus participation are heavily restricted, It offers high transaction throughput and control but reintroduces a central point of failure and trust. It is effectively a distributed database with cryptographic audit trails but lacks decentralized censorship resistance. [4]

1.6.2.3 Hybrid/Consortium Blockchains

Combine features of both. The gold standard for medical platforms, Multiple Independent stakeholders jointly operate validator nodes, No single entity controls the network, yet the group collectively can enforce privacy regulations and remove illegal content if legally compelled, satisfying both decentralization goals and real-world legal liability requirements.
A central organization may maintain administrative control and privacy while keeping the ledger partially public for transparency or utilizing a public chain for ultimate settlement.[4]

1.6.2.4 Layer 2 Solutions

Layer 2 solutions are protocols or frameworks that operate on top of an existing Layer 1 blockchain, like Ethereum, to improve scalability and transaction efficiency. They inherit the base security of Layer 1 but use external mechanisms to enhance performance without overloading the main chain. Layer 1 refers to the base blockchain, responsible for transaction validation and consensus. Ethereum's Layer 1 uses Proof of Stake (PoS) to secure transactions.

In contrast, Layer 2 networks offload transactional work while periodically communicating with Layer 1 to maintain data integrity and trust. Unlike Layer 1 upgrades, which require consensus changes and network-wide coordination, Layer 2 enhancements are less disruptive and faster to implement.[24, 31, 4, 53]

Table 1.6: comparison of the different types of blockchains architecture.[4, 24, 31]

Architecture Type	Access Control	Best For	Examples
Public Blockchain	Anyone can participate	Cryptocurrencies, DeFi, NFTs	Bitcoin, Ethereum, Solana
Private Blockchain	Single organization controls access	Internal record keeping, supply chain	Hyperledger Fabric
Consortium Blockchain	Multiple organizations share control	Banking, healthcare, trade finance	R3 Corda, Quorum
Layer 2 Solutions	Inherits from base layer	Scaling exisiting blockchains	Polygon,Optimism,Arbitrum

1.7 Advantages and Disadvantages of Blockchains

1.7.1 Advantages

- Decentralization: Blockchain operates on a peer-to-peer network with no central server or single controlling authority, eliminating single points of failure and making the system resilient against targeted attacks, censorship, or infrastructure outages. [4]
- Immutability: Once confirmed, data cannot be altered or deleted without network consensus, guaranteeing tamper-proof audit trails for medical records and access logs. [4, 53]
- Transparency: All permitted participants share an identical, auditable ledger, enabling independent verification without relying on a central intermediary.[4]
- Enhanced Security: Multi-layered protection via cryptographic hashing, digital signatures, and distributed consensus reduces the attack surface compared to centralized databases.[53]
- Patient-Mediated Exchange: Cryptographic keys and smart contracts give patients direct, granular control over who accesses their data and when, shifting power from institutions to individuals.

1.7.2 Disadvantages

- Scalability Issues: Strict consensus mechanisms can bottleneck transaction throughput, causing delays. As well as storage challenges, Blockchain is for *metadata*, not *megabytes*. It cannot store large clinical files.[53]

- Energy Consumption: Specifically, Proof-of-Work consensus demands massive computational power as well as high storage requirements as the time goes on (overhead), the perception of high environmental cost persists.[4, 53, 30]

- Complexity: Harder to implement, understand, and integrate seamlessly with legacy centralized systems.[52]

- Immutability Risks and Data Conflict: If incorrect data (or a vulnerable smart contract) is written to the chain, it is very difficult to rectify. Permanent on-chain data conflicts directly with GDPR (General Data Protection Regulation) Article 17 (Right to Erasure).[35]

1.8 Usage of Blockchains

Beyond cryptocurrencies, blockchain finds application across multiple sectors:

Healthcare: Secures electronic health records, tracks pharmaceutical supply chains, and integrates IoT devices for real-time monitoring (e.g., HEALIX).

Supply Chain: Delivers real-time, end-to-end traceability of goods from origin to consumer, reducing fraud and improving logistics transparency.

Finance and DeFi: Enables fast cross-border payments, decentralized lending, and automated tokenized trading without central intermediaries [4].

Identity Verification: Powers self-sovereign identity solutions that give users control over personal data, preventing fraud and data harvesting [34].

Real Estate: Tokenizes property assets for fractional ownership and enables transparent, near-instant title transfers.[4]

1.9 Blockchains Today and Cryptocurrencies

Cryptocurrencies serve as the native economic incentive layer for many major public blockchains, rewarding nodes for securing the network.

1.9.1 Bitcoin

The pioneer. A decentralized store of value and digital currency, often referred to as "digital gold." It prioritizes maximal security, immutability, and decentralization over transaction speed.

1.9.2 Ethereum

A revolution in utility, shifting the focus from a simple payment ledger to a global, decentralized computer via smart contracts. It powers the vast majority of decentralized finance, NFTs, and Web3 applications today.

1.9.3 Hyperledger Fabric

An enterprise-focused, permissioned blockchain framework hosted by the Linux Foundation. Unlike public networks like Bitcoin or Ethereum, it targets enterprise adoption using private channels, customizable consensus models, and does not rely on a native cryptocurrency.

1.10 Conclusion

Blockchain technology represents a pivotal evolution in how we store data, transfer value, and establish trust in digital ecosystems. By shifting away from centralized authorities to distributed, cryptographically secured networks, blockchain ensures enhanced privacy, security, and true data immutability.

While challenges like scalability and integration complexity remain, the continuous adoption of advanced consensus models (like Proof of Stake) and varied network types (Public, Private, Hybrid) guarantees its increasing significance in fields critical to our infrastructure, such as the digital healthcare landscape

Chapter 2

The Applications of Blockchain Technology in Healthcare

2.1 introduction

Blockchain's core attributes as immutable audit trails, decentralized management, and cryptographic data provenance present significant potential for healthcare.

Anticipated improvements include streamlined medical record management, automated insurance adjudication, and enhanced integrity in clinical research. Central to this promise is the shift from institution-driven to patient-centered interoperability. Unlike traditional silos where hospitals control data, blockchain enables data subjects (patients) to possess, operate, and authorize access to their medical information. Through smart contract-based rules, patients can grant granular, time-limited permissions to specific researchers or providers.[10]

This model aligns with GDPR rights by establishing technical enforcement of consent. However, due to the volume and sensitivity of Protected Health Information (PHI), blockchain layers store primarily cryptographic pointers and metadata tags, not the raw medical files themselves, though critical exceptions like drug allergy alerts may reside on-chain for emergency accessibility.[10]

Key adoption challenges remain, including scalability, governance, and cross-institutional data standardization.

2.2 Applications of Blockchain in Healthcare

Traditional healthcare processes suffer from inefficiencies due to manual, siloed data management. Blockchain addresses these friction points by introducing automation and trust-minimized data sharing. As noted by the PwC (PricewaterhouseCoopers: a multinational professional services network that provides consulting services to healthcare organizations) Health Research Institute[46], blockchain adoption is advancing in pharmaceutical supply chains to provide auditable transaction histories [9].

The technology's primary advantage lies in decentralized data management, enabling stakeholders (hospitals, regulators, and patients) to share information without a central intermediary [8]. This facilitates patient-centered care by allowing individuals to securely share personal medical information with doctors while maintaining robust compliance monitoring and privacy safeguards [5]. Key benefits driving adoption include data immutability, system robustness, and improved data integrity [8].

While the future outlook is promising for accelerating digital transformation and disrupting service provision, blockchain in healthcare remains an emergent technology. Implementation requires measured, incremental adoption to mitigate risks associated with scalability and patient safety [6].

2.2.1 clinical research

The shift toward real-world data capture in clinical research necessitates patient-centered technologies that ensure data authenticity and secure access. Blockchain offers a distributed ledger solution providing transparency, trust, and robust security. However, blockchain-based systems must navigate complex state and federal research regulations concerning human subject protections and data privacy. Currently, a significant gap exists between technological development and regulatory clarity, as policymakers have yet to provide explicit compliance guidelines for blockchain stakeholders [11].

2.2.2 Drug discovery and Pharmaceutical research

High costs and the need for accelerated innovation drive pharmaceutical companies toward competitive collaboration. Blockchain facilitates trusted information transfer among multiple parties, providing immutable records and time-stamping for robust digital proof of Intellectual Property (IP) [10]. Beyond collaborative research, blockchain improves internal processes by enabling effective tracking of clinical trial data, consent management, and adverse event monitoring. When research is outsourced, blockchain assures data integrity and outcome validation, mitigating the risk of misrepresented results through a transparent, open ecosystem [10].

2.2.3 Clinical Trials

Blockchain addresses critical inefficiencies in clinical trial management, including patient recruitment, data integrity, and regulatory auditing. Initiatives such as the IEEE Standards Association forum highlight the technology's potential to expedite drug development and provide auditors with transparent frameworks for legal and ethical validation. By managing consent, data collection, and trial outcomes in a trustful manner [14], blockchain reduces budget and timeline overruns. Solutions like BlockRX utilize Advanced Digital Ledger Technology (ADLT) to interconnect siloed parties across the pharmaceutical spectrum, from drug discovery to device manufacturing [10, 15].

2.2.4 Medical Fraud Detection

The healthcare supply chain's complexity and direct impact on patient well-being make it uniquely vulnerable to fraud [16]. Blockchain introduces a secure platform to counter this by enhancing data transparency and product traceability. Because records are validated and updated exclusively via smart contracts, the ledger becomes highly resistant to manipulation. This provides an immutable audit trail that eliminates the "holes" inherent in multi-party logistics, ensuring the authenticity of pharmaceuticals and medical devices from manufacturer to patient [10].

Table 2.1:Blockchain benefits and use cases to improve medical record management. An excerpt from [8].

Blockchain: Key Benefit	Biomedical/Health Care Use Case: Improved Medical Record Management
Decentralized Management	Patient-managed health care records: “[Patient] becomes the platform, owning and controlling access to their healthcare data. This removes all obstacles to patients acquiring copies of their healthcare records or transferring them to another healthcare provider.”[20]
Immutable Audit Trail	Unalterable patient records: “The data are stored in the private blockchain cloud. Blockchain may guarantee medical data cannot be changed by anybody including physicians and patients himself/herself internally and natively.”[21]
Data Provenance	Source-verifiable medical records:“Records are signed by source, allows legitimacy of records to be verified(and false records to be plausibly denied).”[22]

Robustness/Availability	Reduced risk of patient record keeping: “Because data is stored on a decentralized network, there is no single institution that can be robbed or hacked to obtain a large number of patient records.”[20]
Security/Privacy	Increased safety of medical records: “Data is encrypted in the blockchain and can only be decrypted with the patient’s private key. Even if the network is infiltrated by a malicious party, there is no practical way to read patient data.”[20]

2.3 Challenges and Limitations of Blockchain in Healthcare

While blockchain technology offers transformative solutions for data security, patient-centric records, and Interoperability in healthcare, its implementation is not without significant hurdles. Transitioning from traditional, centralized databases to a distributed ledger architecture presents several technical, regulatory, and adoption-related challenges. [40]

Table 2.2: comparison between traditional systems and blockchain.[4, 40, 33, 52]

Feature	Traditional Centralized Systems	Blockchain Systems
Architecture	Single point of control (Centralized).	Distributed Ledger (Decentralized).
Scalability	High throughput (tens of thousands TPS).	Lower throughput, prone to congestion.
Storage	Highly scalable, suitable for large files.	High cost for large files, limited on-chain capacity.
Immutability	Data can be easily modified or deleted.	Data is append-only, irreversible by design.
Security	Vulnerable to single point of failure.	High cryptographic security, resistant to tampering.
Interoperability	Fragmented, Proprietary silos.	Still fractured, but standardizing towards openness.
Cost / Setup	Established, predictable infrastructure costs.	High initial setup and transition costs.

2.3.1 Scalability and Transaction Throughput

Public blockchains struggle to process the massive transactional throughput demanded by healthcare networks, such as continuous IoMT data streams, leading to latency and high fees. Healthcare architectures often rely on complex Layer 2 solutions or private chains to overcome scalability limitations.[40, 38, 36, 3]

2.3.2 Storage Constraints and Data Payload Limits

Storing large medical payloads like MRI scans directly on-chain is technically unfeasible and excessively expensive due to ledger replication across all nodes. Healthcare blockchains must utilize hybrid off-chain storage solutions like IPFS, storing only cryptographic hashes on the ledger to reduce data burdens.[53]

2.3.3 The "Right to be Forgotten" and Regulatory Friction

Blockchain's absolute immutability conflicts with privacy regulations like GDPR, which mandate the ability to delete personal data upon request. To remain compliant, systems must ensure no raw personally identifiable information (PII) touches the blockchain, relying instead on anonymized data pointers.[29]

2.3.4 Interoperability and Standardization

The blockchain ecosystem is fragmented with hundreds of competing protocols, varying consensus mechanisms, and a lack of universal healthcare data standards. Seamless interoperability requires complex cross-chain communication and industry-wide agreement on medical data ontologies like HL7 (Health Level 7) FHIR (Fast Healthcare Interoperability Resources).[23, 29, 40]

2.3.5 High Implementation Costs and the Skills Gap

Transitioning to decentralized infrastructure is capital-intensive, requiring complete overhauls of legacy databases and significant hardware updates. This is exacerbated by a severe shortage of specialized developers capable of navigating the complex intersection of cryptography and healthcare informatics.[52]

2.4 Blockchain as a Solution for Healthcare Data Management

2.4.1 Electronic Health Records (EHR) & Personal Health Records (PHR)

Current healthcare infrastructure relies on hospital-centric Electronic Health Records (EHRs), where institutions serve as primary data custodians. Blockchain enables a paradigm shift toward patient-centric Personal Health Records (PHRs), wherein patients hold cryptographic keys to a unified, portable health profile spanning multiple providers. Early work by Azaria et al. [26] established foundational requirements for blockchain-based EHR systems (authentication, confidentiality, accountability, and secure data sharing) while providing patients with immutable access logs across disparate providers.

Subsequent research has expanded upon this foundation. Yang et al. [27] introduce a permissioned blockchain architecture designed to enhance data integrity, secure access control, and interoperability among healthcare providers. By integrating blockchain technology with pre-existing EHR databases, this solution makes use of providers' SQL-based data systems while enabling safe, auditable access and modification logs. The suggested approach is designed to guarantee the transparent and regulated access, modification, and sharing of EHRs. All read, write, and deletion operations performed on patient records are monitored and recorded by the system [28].

2.4.2 Data Integrity and Auditability

Blockchain's core property of immutability ensures data integrity. Once a medical transaction or a hash of a medical record is anchored, it is cryptographically secured across thousands of nodes. This guarantees that a patient's medical history, lab results, or prescription records cannot be maliciously altered or accidentally overwritten by database administrators. It provides a flawless, tamper-proof audit trail for medical compliance.[46, 47]

2.4.3 Consent Management

Blockchain ledgers revolutionize consent management through dynamic smart contracts. Patients can precisely dictate who gets access to their data and under what conditions. A patient can grant a specialist a time-bound access key to a specific MRI scan while completely withholding psychiatric records. This consent can be effortlessly updated or revoked in real-time, instantly dissolving the provider's access keys across the network.[7]

Table 2.3: comparison between traditional EHR and blockchain-based PHR [28]

Feature	Hospital-centric EHR	Blockchain-based PHR
Data Custodian	The Hospital / Healthcare Provider.	The Patient (via cryptographic keys).
Interoperability	Siloed, difficult to share across networks.	Universally accessible via standardized formats.
Consent Control	Paper-based or Broad Blanket Consent.	Granular, Real-time digital Smart Contracts.
Audit Trails	Maintained internally by IT staff.	Immutable, public/consortium ledger verifiable.

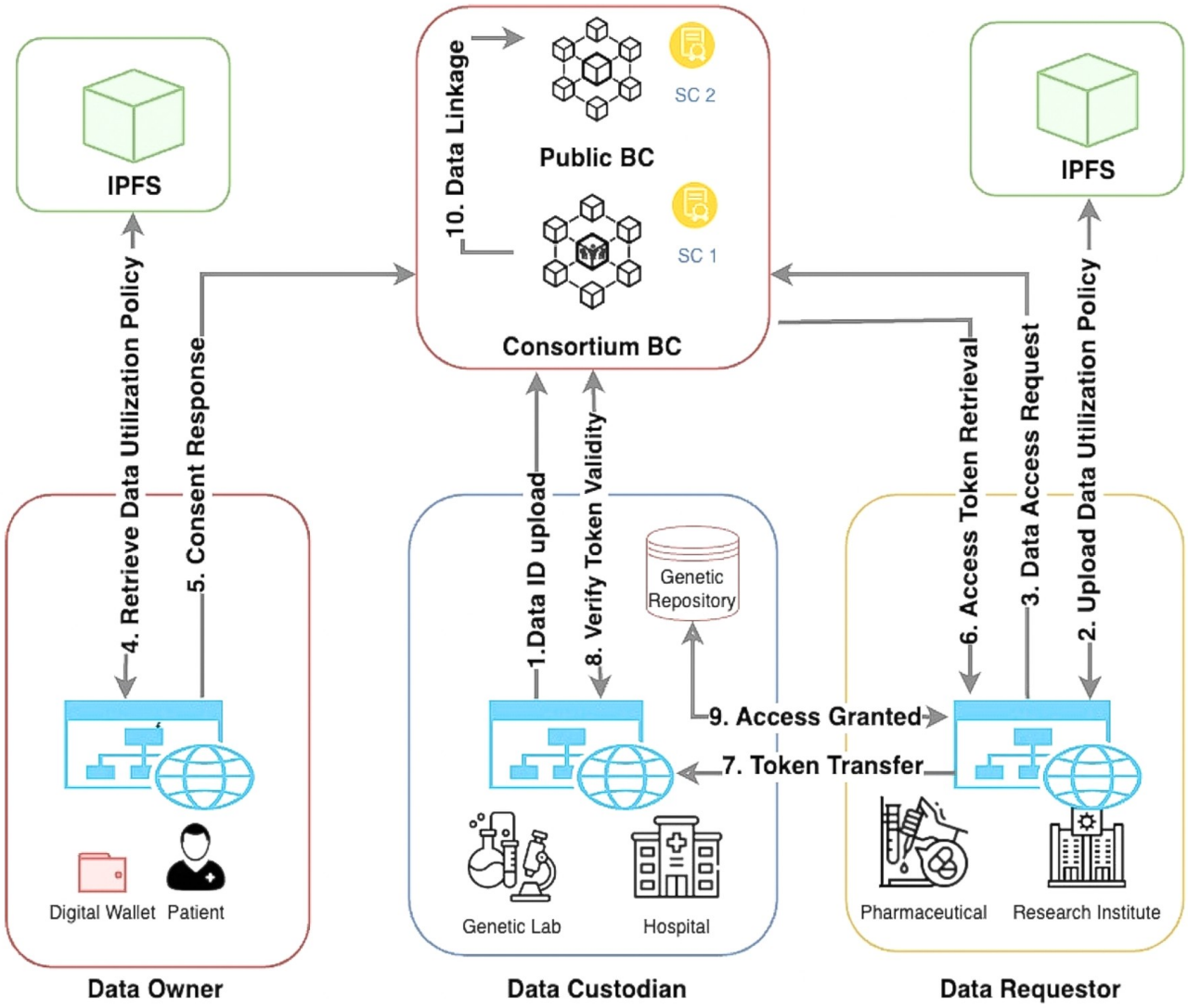

Figure 2.1:Blockchain-based Data Consent Flow, SecureConsent System Architecture.[7]

2.4.4 Case Study & Contextual Example: HEALIX

The “HEALIX” project (Health Ledger & Information eXchange) perfectly encapsulates these elements. HEALIX leverages a decentralized architecture to connect SuperAdmins, Hospitals, Doctors, and Patients globally. Patients are registered on HEALIX to securely manage their PHRs, maintaining absolute *sovereignty over their data*. When a doctor requests a patient's historical records, HEALIX validates the patient’s authorization through blockchain-based smart contract permissions and emergency access policies. Once authorization is confirmed, The platform securely retrieves the encrypted off-chain medical data referenced by the blockchain pointer and decrypts it using protected cryptographic key management mechanisms, ensuring confidentiality, auditability, and dynamic consent enforcement.

2.5 Fundamentals of the Internet of Medical Things (IoMT)

2.5.1 Introduction

The Internet of Things (IoT) represents the pervasive interconnection of uniquely identifiable smart objects and devices, extending connectivity beyond traditional machine-to-machine (M2M) scenarios to enable automation across diverse sectors [17]. Healthcare stands as a primary application domain for IoT, promising remote patient monitoring, chronic disease management, and improved medication adherence [19]. The derived subdomain, the Internet of Medical Things (IoMT), comprises low-energy, wirelessly enabled sensors and smart devices (including diagnostic equipment, smart beds, and wearables) that continuously gather and analyze patient biometric data such as heart rate, oxygen saturation, and sleep patterns [18].

From a provider perspective, IoMT integration is expected to reduce operational costs, minimize device downtime through remote provisioning, and optimize resource scheduling. For patients, it enhances quality of life through continuous, non-invasive monitoring [18]. These distributed sensor networks, however, generate vast streams of sensitive biometric data, creating inherent security and data integrity challenges that subsequent architectural layers (specifically blockchain integration) aim to address.[19]

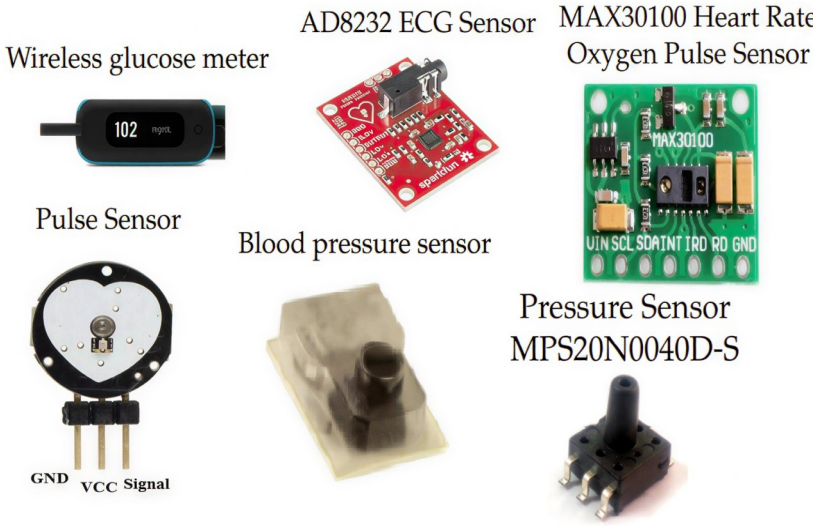

Figure 2.2: some examples of medical IoT devices.[33]

For the interested reader see document [33] pages 13 and 14 for full technical specifications.

2.5.2 Data Generation

IoMT endpoints are prolific data generators, characterized heavily by the "Three Vs": Volume, Velocity, and Variety.[25]

Volume: Millions of connected devices generate petabytes of health data globally on a monthly basis.[36]

Velocity: Data is captured continuously in real-time. Where traditional checks might happen once a month, IoMT devices like heart rate monitors can stream data with sub-second latency. [36]

Variety: The data formats are incredibly diverse, ranging from simple structured numerical measurements (like blood pressure metrics) to complex unstructured datasets (unprocessed ECG waves, sleep cycle analytics, or imaging telemetry).[36]

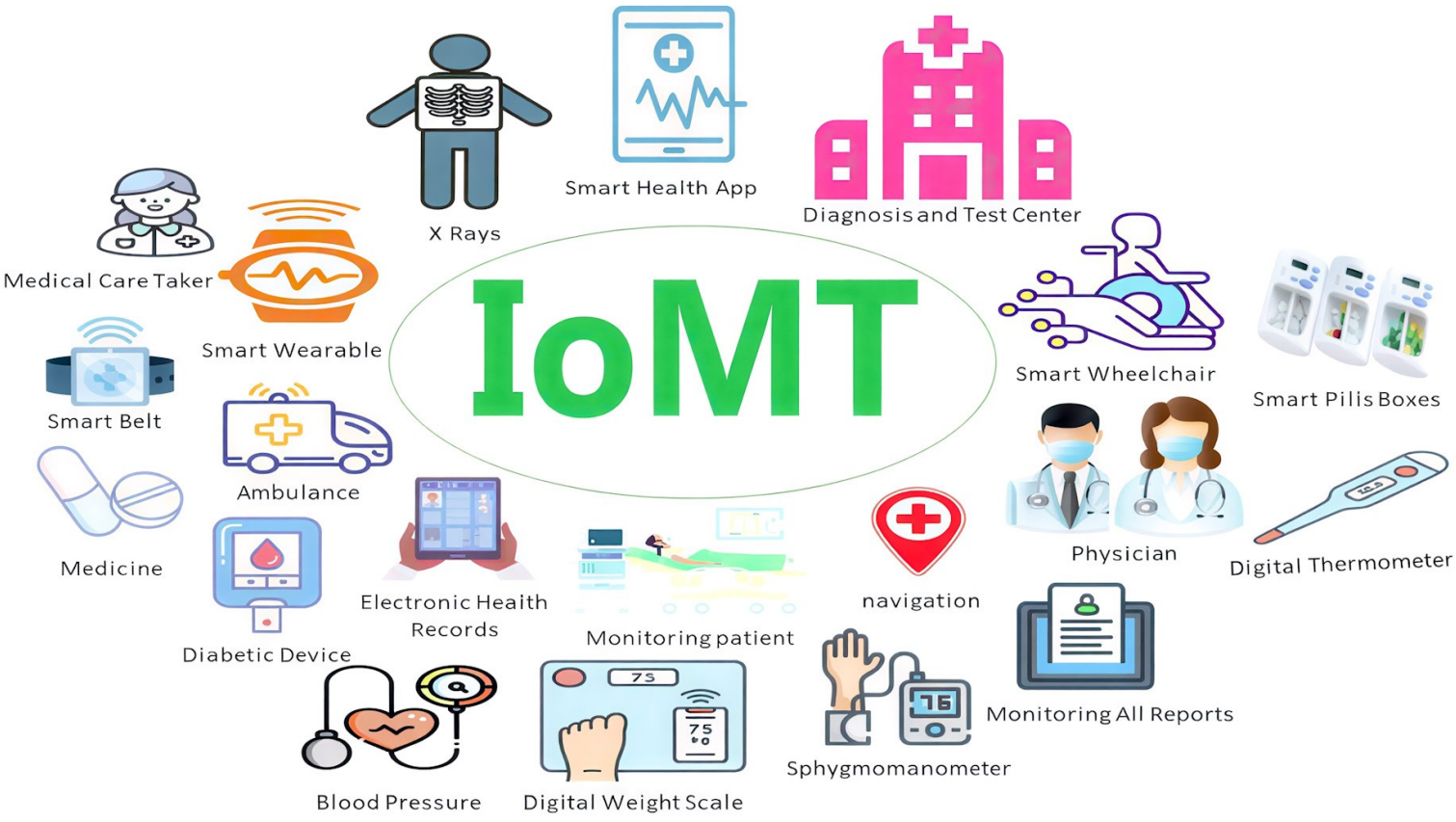

Figure 2.3: The IoMT Ecosystem

Table 2.4:Traditional medical devices vs connected IoMT devices, Adapted from Osama et al.[33]

Feature	Traditional Medical Device	Connected IoMT Device
Data Collection	Episodic, Manual recording	Continuous, Automated real-time
Connectivity	Standalone, Offline	Connected via wi-fi, Bluetooth, 5G, or RFID
Security Risk	Physical Tampering	Remote Hacking, DDoS, Spoofing, MITM attacks
Patient Autonomy	Relies on clinical visits	Empowers proactive, remote self-management

2.6 Wireless Body Area Network (WBAN)

"WBANs are interconnected devices mounted on, in, or around the human body. Medically implanted and wearable sensors are wildly popular in the context of E-Healthcare."[33]

Table 2.5: specifications of WBANs [33]

Feature	Specification and Description
Reliability	Must exceed standard sensor networks due to the critical nature of medical data. Ensures consistent, accurate transmission from sensors to destinations without loss or corruption.
Energy	Devices are battery-constrained. While external sensor batteries may be replaceable, implanted sensors face significant power limitations and replacement complexity, demanding ultra-low-power design.
Human-Centric	Sensors are attached directly to or implanted within the human body. Designs must prioritize biocompatibility, safety, and non-invasive acceptance by sensitive body tissue.

Feature	Specification and Description
Availability	High availability is essential for critical medical monitoring. Communication links between body sensors and the network sink must be consistently maintained.
Mobility	Networks must support moderate-to-high mobility, adapting to daily human activities. Performance stability is required despite the varying movement patterns of sensor nodes.
Scalability	WBANs are small-scale networks (typically ≤ 15 sensors) dedicated to specific, localized medical monitoring of a single user, contrasting with large-scale Wireless Sensor Networks (WSNs).
Deployment	Characterized by dense, localized sensor placement around a single body. While not constrained by sparse deployment, WBANs are highly susceptible to inter-network interference from nearby WBANs in public spaces.
Topology	No fixed topology constraints. Network configurations dynamically adapt, utilizing star, mesh, hybrid, or cluster topologies based on immediate body movement and communication needs.
Bandwidth	Ranges from hundreds of kbps to a few Mbps, dependent on the short-range technology (Bluetooth, Zigbee, Wi-Fi). Bandwidth selection involves a direct trade-off between data rate and battery consumption.

2.7 Healthcare 4.0

"Healthcare 4.0 refers to integrating digital technologies, data analytics, and AI in healthcare, This includes telehealth, electronic health records, AI-driven diagnostics, and personalized medicine...".[33]

Table 2.6: a summary of Healthcare 4.0 challenges and potential solutions [33, 3, 36]

Challenge	Issues with Healthcare 4.0	Key Solutions
Energy	Medical sensors require up to a ten-year battery life, imposing strict power constraints on communication protocols and network design.	Energy-efficient network algorithms, Edge computing, Quantum computing
Bandwidth	Growing telemedicine demands high-quality video/audio streaming, creating challenges in optimal resource allocation for critical medical traffic.	Edge computing, 5G communications, Device-to-Device (D2D) communication
Network Availability & Reliability	Ultra-high availability and reliability are required but remain unresolved by current infrastructure alone.	5G communications, D2D, Edge computing, Edge intelligence
End-to-End Latency	Certain services demand ultra-low latency (1–5 ms), placing stringent constraints on network architecture and data processing.	Edge computing, Edge intelligence, Software-defined networking (SDN)
Security	Traditional security schemes cannot satisfy the required security level for Healthcare 4.0 nor meet the stringent energy budgets of medical sensors.	Trust management, Blockchain, Quantum encryption
Scalability	Rapid proliferation of connected medical devices requires networks and protocols capable of high scalability.	5G communications, D2D, Edge computing
Massive Data	The vast volume of network traffic generated by numerous devices necessitates robust management methods to prevent congestion and failure.	Big data analytics, D2D, Edge computing, Blockchain

2.7.1 Limitations & Threats in IoMT

The rapid deployment of IoMT brings several critical vulnerabilities:

Device Hijacking and Spoofing: Malicious actors can take control of a device (e.g., an insulin pump) or spoof data streams, potentially leading to lethal medical errors.[33]

Lack of Standardized Device Identity: Unlike computers, many IoMT devices lack standardized credentialing protocols, making it difficult for the network to authenticate whether a heartbeat sensor is legitimately belonging to a specific patient.

Vulnerability to DDoS Attacks: Small IoMT devices have limited processing power. They can be easily co-opted into botnets or targeted by Distributed Denial of Service (DDoS) attacks, rendering vital monitoring offline.

Data Tampering: Data transmitted over wireless networks without robust encryption can be intercepted and altered in transit, destroying the clinical integrity of the metrics.

2.8 Securing IoT Devices with Blockchain

2.8.1 Decentralized Device Identity (DID)

Instead of relying on a central server to manage passwords or certificates, blockchain employs Decentralized Identifiers (DIDs). Each IoMT device is registered on the blockchain with a unique, cryptographically generated public-private key pair upon manufacturing or deployment. This prevents unauthorized devices from spoofing sensor data, as the network immediately rejects any data payload not signed by a recognized DID.[38]

2.8.2 Immutable Event Logs

Security breaches often involve wiping system logs to hide unauthorized access. In a blockchain-IoT hybrid system, sensor data hashes, access requests, and device interactions are recorded directly on the distributed ledger. This creates an immutable event log. Even if an attacker physically compromises a device, they cannot retroactively alter the historical record of its actions on the blockchain.[46, 47, 48, 49]

2.8.3 Edge Computing & Blockchain

Processing data at the "edge" closer to the IoMT devices themselves, drastically reduces latency. Integrating edge gateways with blockchain means that basic filtering and cryptographic hashing can happen locally inside the hospital network before just the verifiable proofs are committed to the distributed ledger. This hybrid synergy accelerates data verification and preserves bandwidth.[3, 33, 36]

2.8.4 Firmware Lifecycle Management

Medical devices frequently require updates to patch vulnerabilities, yet central OTA (Over-The-Air) update servers are prime targets for distribution of malware. By operating firmware updates through the blockchain, manufacturers can publish the cryptographic hash of the new legitimate firmware via a smart contract. The IoMT device will query the blockchain, verify the signature and integrity of the firmware against the immutable ledger, and only install the software if it is a perfect match.[33]

2.9 Privacy and Regulatory Compliance

2.9.1 Navigating Regulations (HIPAA and GDPR)

Blockchain developers face the intricate challenge of aligning immutable ledgers with strict frameworks like the Health Insurance Portability and Accountability Act (HIPAA) and the General Data Protection Regulation (GDPR). The core conflict originates with GDPR's "Right to be Forgotten," as data on a blockchain cannot be erased. Compliance is typically achieved by structurally separating raw identity and health data from the blockchain architecture altogether. [11, 26, 38]

2.9.2 Potential Solutions to Achieve Privacy in Healthcare 4.0

- Implement Strong Access Controls: Enforce strict user authentication mechanisms, including multi-factor authentication, to prevent unauthorized access to sensitive healthcare data [33].
- Encrypt Sensitive Data: Apply robust encryption algorithms to protect patient information from unauthorized access or manipulation during storage and transmission [33].
- Regularly Update and Patch Systems: Maintain up-to-date software, operating systems, and network infrastructure by applying the latest security patches to mitigate known vulnerabilities [33].

- Conduct Thorough Risk Assessments: Perform regular evaluations of potential risks and vulnerabilities within data management systems, analyzing threat impact and implementing appropriate mitigation measures [33].
- Follow Regulatory Frameworks and Compliance Standards: Adhere to regulations such as HIPAA and GDPR to ensure legally compliant data handling. Complement compliance with data anonymization techniques to minimize the risk of re-identification [33, 11, 26].
- Monitor and Detect Security Incidents: Deploy robust intrusion detection systems and security monitoring tools to identify and respond promptly to unauthorized access or suspicious activities [33].
- Establish Data Breach Response Plans: Develop and routinely test incident response plans detailing containment steps, communication protocols, and recovery strategies to minimize the impact of a security breach [33].

2.9.3 Potential Solutions to Ensure Data Consent

- Digital Consent Forms: Implement electronic consent forms that clearly articulate the purpose, risks, and benefits of proposed treatments. These forms enable informed patient decision-making while securely storing consent records [7, 11, 26].
- Two-Factor Authentication: Utilize two-factor authentication methods (such as biometrics or unique codes sent to a patient's mobile device) to verify patient identity prior to consent authorization, ensuring only authorized individuals provide consent [33].
- Consent Management Systems (CMS): Deploy CMS platforms to track and manage patient consent across the entire care continuum. These systems automate consent workflows, monitor consent status, and enforce compliance with legal and regulatory requirements [7, 33].

- Audit Trails: Maintain comprehensive audit trails that document all consent-related activities, including acquisition timestamps, responsible personnel, and any subsequent updates or withdrawals, thereby establishing a transparent and verifiable record [33].
- Education and Communication: Integrate patient portals, interactive interfaces, and educational materials within Healthcare 4.0 systems to inform patients of their rights and foster meaningful engagement in the informed consent process [11, 33, 26].

2.10 Future Trends and Ongoing Challenges

2.10.1 Scalability and Latency

The most persistent bottleneck of integrating IoMT with blockchain is scalability. Real-time monitoring demands high-frequency micro-transactions that base-layer (Layer 1) blockchains like Ethereum cannot natively handle without incurring severe latency and astronomical network fees. The future relies heavily on Layer 2 rollups, state channels, and faster alternative consensus mechanisms specifically tuned for rapid data bursts.[3]

2.10.2 Interoperability

While blockchain attempts to break healthcare data silos, it risks creating its own fragmentation. Hospitals utilizing hyperledger-based localized blockchains might struggle to seamlessly share data with institutions relying on Ethereum Layer 2 solutions. Developing standardized, cross-chain communication protocols and enforcing universal medical data ontologies (like HL7 standards and FHIR) across these diverse networks remains a complex, ongoing challenge.[40]

2.10.3 Integration with AI/ML

The convergence of AI, IoMT, and Blockchain is the ultimate frontier. Artificial Intelligence requires vast amounts of high-quality data to make predictive healthcare diagnoses. Blockchain provides the guarantee that the data streams collected from IoMT devices have not been tampered with. Immutably verified datasets serve as exceptionally trusted training grounds for machine learning algorithms, paving the way for hyper-accurate, automated preventive care.[50, 51]

2.11 Conclusion

The integration of blockchain technology translates the chaos of modern healthcare networks into an orchestration of trust, security, and patient empowerment. By fundamentally shifting from centralized data silos to decentralized, patient-controlled ledgers, blockchain successfully mitigates many of the vulnerabilities introduced by the proliferation of IoMT devices. Through Decentralized Identifiers (DIDs), immutable logging, and self-executing smart contracts, healthcare systems can defend against external hijacking, drastically reduce administrative fraud, and ensure pristine data integrity from the edge device all the way to the specialist's monitor.

Having thoroughly explored the theoretical foundations, sweeping benefits, and recognized limitations of integrating IoMT with distributed ledgers, it is evident that a highly structured technological framework is required to realize this vision. The next chapters will transition from the theoretical landscape to the definitive architecture of the HEALIX ecosystem. It will explicitly map out the technical implementation and conception, detailing the platform's multi-tiered role architecture, the engineering of its core smart contracts, and the specialized workflow that empowers its SuperAdmins, Hospitals, Doctors, and Patients.

Chapter 3

Conceptual Design

3.1 Introduction

The transition from traditional, centralized medical databases to a blockchain-based healthcare ecosystem requires a well-defined conceptual framework. While the theoretical foundations of blockchain technology have been established in the previous chapters, this chapter encapsulates the core conceptual design of the “HEALIX” project. Here, we outline the primary issues plaguing current healthcare systems, the targeted objectives of our solution, the defining roles of all interacting system actors, the high-level global architecture, and the precise role of smart contracts in governing access and data flow within our specific context.

3.2 Problematic

Modern healthcare systems face severe, deeply rooted structural challenges that compromise both patient care and data security. The most prominent problems include:

- Secure Data Sharing:

Figuring out how to seamlessly and securely share sensitive health data with involved parties (doctors, laboratories, insurers) without violating privacy regulations remains a massive hurdle.

- Lack of Patient Control:

In traditional systems, patients have little to no control over who views, modifies, or shares their medical data. They are largely passive entities in the data ownership lifecycle.

- Single Points of Failure:

Centralized databases are prime targets for cyberattacks, ransomware, and unauthorized data breaches, putting millions of sensitive medical records at risk.

3.3 Existing Works

Blockchain-based healthcare systems have evolved from early conceptual models into architectures addressing electronic health record (EHR) management, patient consent, and IoMT integration. The prevailing design pattern is a hybrid architecture combining off-chain medical data storage with on-chain integrity verification and access control through smart contracts.

MedRec (Azaria et al., 2016) [26] is widely regarded as one of the earliest Ethereum-based prototypes for EHR management, developed in collaboration with Beth Israel Deaconess Medical Center. It addressed fragmented access and limited patient control by using smart contracts to manage permissions, data pointers, and audit logs while actual records remained in existing provider databases. MedRec introduced patient-centric access control through Patient–Provider Relationship contracts and established the foundational off-chain storage/on-chain permission pattern, but it did not address IoMT telemetry, emergency access, or public-chain scalability.

SecureConsent (Javed et al., 2024) [7] advances dynamic consent management using blockchain. The system allows patients to grant, revoke, and time-limit access to specific genomic datasets via smart contracts, significantly enhancing transparency and data sovereignty. While effective for fine-grained consent, its scope remains limited to genomic data sharing and does not incorporate IoMT devices, broader clinical workflows, or emergency overrides.

To address scalability and latency limitations, **Mallick et al**. [39] proposed a Blockchain–Fog–IoMT framework for Healthcare 4.0 environments. Their architecture combines fog computing with IPFS-based storage to reduce blockchain overhead and improve real-time responsiveness for connected medical devices. Experimental results demonstrated improvements in latency, storage efficiency, and upload speed. Nevertheless, the framework lacks comprehensive patient-controlled access management and emergency override mechanisms.

Broader **Healthcare 4.0 studies** [33] highlight ongoing challenges related to security, interoperability, privacy, and scalability in decentralised healthcare infrastructures. Existing solutions often focus on isolated domains such as consent management or infrastructure optimisation, while few integrate complete clinical workflows (appointment scheduling, telemetry ingestion, anomaly alerting, emergency access, and longitudinal EHR management). Additionally, tensions between blockchain immutability and regulatory requirements such as GDPR remain unresolved, particularly in public blockchain environments with high transaction costs and metadata exposure.

HEALIX extends existing research by integrating Ethereum smart contracts, IPFS-based off-chain storage, dynamic patient consent, IoMT telemetry monitoring, and emergency break-glass access within a unified proof-of-concept platform. Unlike prior approaches that specialise in either infrastructure scalability or permission management, HEALIX combines EHR management, appointment scheduling, real-time anomaly alerting, and auditable access control into a single architecture, demonstrating a practical and patient-centric blockchain-enabled healthcare workflow.

3.4 Objectives

To address these critical shortcomings, our work, the HEALIX system, is designed with the following core objectives:

- Unified, Patient-Centric Records: By utilizing blockchain, we create a single source of truth for a patient's medical history that can be securely updated and accessed globally, eliminating data silos.
- Decentralized Secure Sharing: Leveraging cryptographic security, we establish a trustless environment where data can be shared among authorized parties safely without relying on a central vulnerability point. Data payloads are stored off-chain (e.g., via IPFS), while immutable references and permissions are managed on-chain.
- Restoring Patient Sovereignty: Our system gives patients absolute ownership over their medical data. They possess the cryptographic keys needed to grant or revoke data access to specific doctors and institutions at will.
- Automated Access Control: We replace slow, human-reliant administrative tasks with automated Smart Contracts that enforce strict Role-Based Access Control (RBAC) in real-time, ensuring that only authorized personnel can read or write medical data.

3.5 System Actors

3.5.1 Human Actors

Super Admin: The ultimate authority and deployer of the HEALIX network. The SuperAdmin does not manage daily medical data but is responsible for the systemic onboarding of verified institutions. They maintain the network's integrity by registering new hospitals and having the power to suspend compromised hospital entities.

Hospital Admin: A User with an administrative role who serves as the primary manager of a specific Hospital. Responsibilities include registering and managing staff (Doctor, Assistant), as well as allocating infrastructure such as Rooms and deploying Devices within them.

Patient: The primary subject of the system. Patients can exist in two states: "*Unclaimed*" or "*Claimed*". Claimed patients securely claim their accounts, manage their own cryptographic identity (Patient Key), and explicitly grant access to their medical records. Unclaimed patients' data is managed implicitly by the hospital via an Escrow Key. Patients can also request appointments.

Doctor: Specialized medical personnel assigned to a Hospital. Doctors process Appointments, create new *RecordMetadata* (uploading actual health data to IPFS), and request access to patient histories. They initiate (OpenSession) and conclude (CloseSession) *AdmissionSessions* for inpatient care and act as the primary *responders* to *AlertRecords* generated by IoT anomaly detection.

Assistant: Administrative support staff within a Hospital, often linked to one or more doctors via an *AssistantDoctor* relationship. Assistants triage and convert *AppointmentRequests* into scheduled Appointments, Register Patients and manage administrative Tasks related to patient visits.

3.5.2 Automated Actors

IoT Device (Device): Hardware sensors assigned to a specific Room. During an active *AdmissionSession*, these devices automatically and continuously submit physiological Readings to the system.

Blockchain Relayer: An asynchronous background mechanism responsible for writing the IPFS CID to the blockchain to guarantee data immutability. It verifies transaction success, links the task to the record, and employs a retry mechanism for failures.

3.6 Global Architecture

The global architecture of the HEALIX system bridges modern web interfaces with decentralized networks. The system does not trust any single actor, Everything is verified (signatures), Encrypted and Anchored (immutability), Each Layer compliments the others.

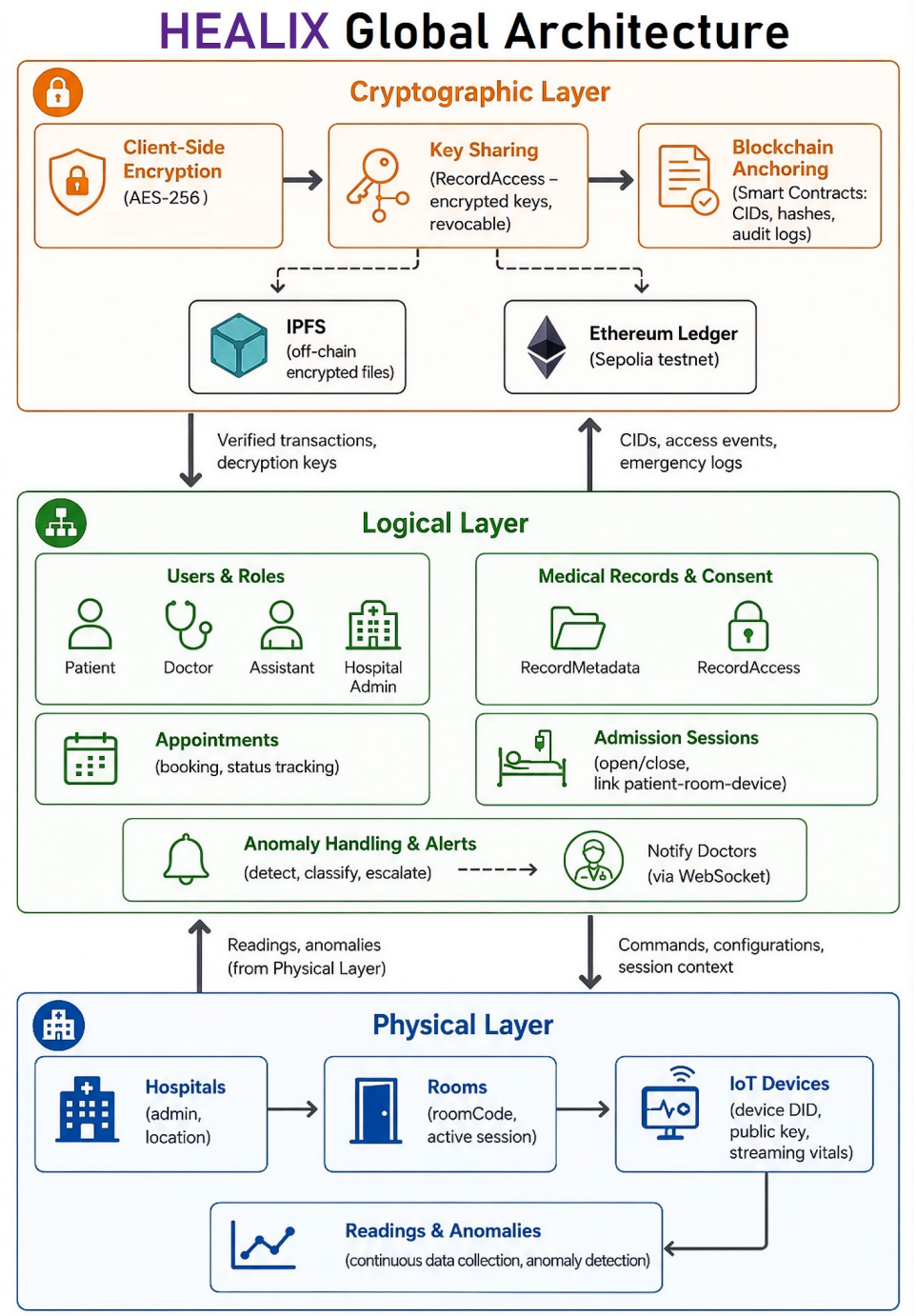

Figure 3.1: Architecture of HEALIX

3.7 Smart Contracts: Overview

3.7.1 User Onboarding & Role Management

Within the “UserRegistry” (which leverages the core rules of “AccessControl”), Hospital Admins act as localized managers. They can register Doctors, Assistants, and Patients, explicitly linking these users to their specific hospital. This prevents cross-hospital data leaks and unauthorized management. Crucially, the “UserRegistry” acts as the definitive source of truth, every other contract in the system constantly queries it to verify a user's role, hospital affiliation, and active status before permitting any actions.

3.7.2 Appointment Scheduling

Patients or Assistants interact with the “AppointmentSystem” to book consultation slots with Doctors. Before confirming a booking, this contract queries the “UserRegistry” to verify that both the patient and the doctor are active and belong to the same hospital. Doctors are responsible for defining their own weekly availability schedules.

The system tracks appointments through their lifecycle (Pending, Confirmed, Completed, or Canceled), ensuring no double-booking occurs and automatically freeing up the doctor's schedule when an appointment is resolved or canceled.

3.7.3 Patient Admission & Telemetry

When a patient is physically admitted, a Hospital Admin uses the “IoTDeviceRegistry” to open an admission session, linking the patient's ID to a specific room containing pre-registered IoT devices. As authorized relay servers or the devices themselves push continuous telemetry data, the contract verifies the active session and anchors the encrypted data hashes to the blockchain.

If the off-chain system detects a health anomaly, it logs a critical alert on-chain. This initiates an acknowledgment workflow where a Doctor must cryptographically sign that they have reviewed the alert, ensuring critical events are never ignored.

3.7.4 Medical Record Management & Consent

Following a consultation or telemetry event, an active Doctor creates a permanent medical record pointer (IPFS CID and Encryption Hash) using the “MedicalRecords” contract. The system enforces strict, granular consent: patients can explicitly grant or revoke read access to specific doctors via their wallets. Alternatively, Hospital Admins can grant access based on implicit consent (e.g., during hospital triage).

In critical emergencies where a patient cannot give consent, doctors can invoke a highly auditable "break glass" protocol (which logs their justification on-chain)to temporarily bypass restrictions and access life-saving medical data.

Ultimately, the smart contracts in our architecture serve as an incorruptible, automated administrator that guarantees every System Actor remains exactly within their defined operational boundaries.

3.8 Workflow Summary

3.8.1 Account Activation Workflow

Assistants pre-register a patient to generate a claim code. The patient then claims their credentials, resulting in the creation of a secure AES Master Key and unique User Identity within the system.

3.8.2 IoT Telemetry Pipeline Workflow

Devices continuously send data through an MQTT broker to the IotService. The service rigorously validates the payload. Invalid payloads or payloads lacking an active session are rejected and logged.

Valid data is persisted, classified, and checked against thresholds. Critical readings trigger the creation of anomalies and alert records, instantly notifying the responsible Doctor via WebSocket and the Assistant via a task notification.

3.8.3 Medical Record Workflow

A Doctor creates a record, encrypting it client-side. The system validates the metadata integrity before the encrypted file is uploaded to IPFS. The resulting CID is registered in the system and synchronized to the blockchain via a reliable Relayer. Access to the record is governed by the patient's claim status (implicit for unclaimed, explicit for claimed). When fetching records, the system audits the request, requiring a token, before the Doctor downloads and decrypts the data locally.

3.8.4 Appointment Workflow

A Patient requests an appointment. The system automatically checks the availability of the requested Doctor. If available, the Assistant is notified to confirm, and the system finalizes the appointment and generates a task. If unavailable, the system immediately returns an error to the Patient.

3.9 Modeling Tools

Several UML (Unified Modeling Language) modeling tools were used to realize the following diagrams, Mermaid [54] is an open-source tool suite that allows the creation of various diagrams such as Sequence, Flowcharts, Mind maps...etc, which has its own syntax to easily convert from code to diagrams. Plantuml [55] is a straight forward design tool that supports multiple UML and non-UML diagrams with its easy to use web-based editor. Other image processing tools were used for minor adjustments such as Photoshop Express [56], GIMP (GNU Image Manipulation Program)[57], and Microsoft Paint for basic Image edits.

3.10 System Diagrams

The following diagrams illustrate the architecture, relationships, and workflows derived from the HEALIX Roles & Permissions models, focusing on user management, data access control, and IoT telemetry integration.

3.10.1 Use Case Diagram

This diagram summarizes the core interactions between the primary actors and the HEALIX system based on the workflows.

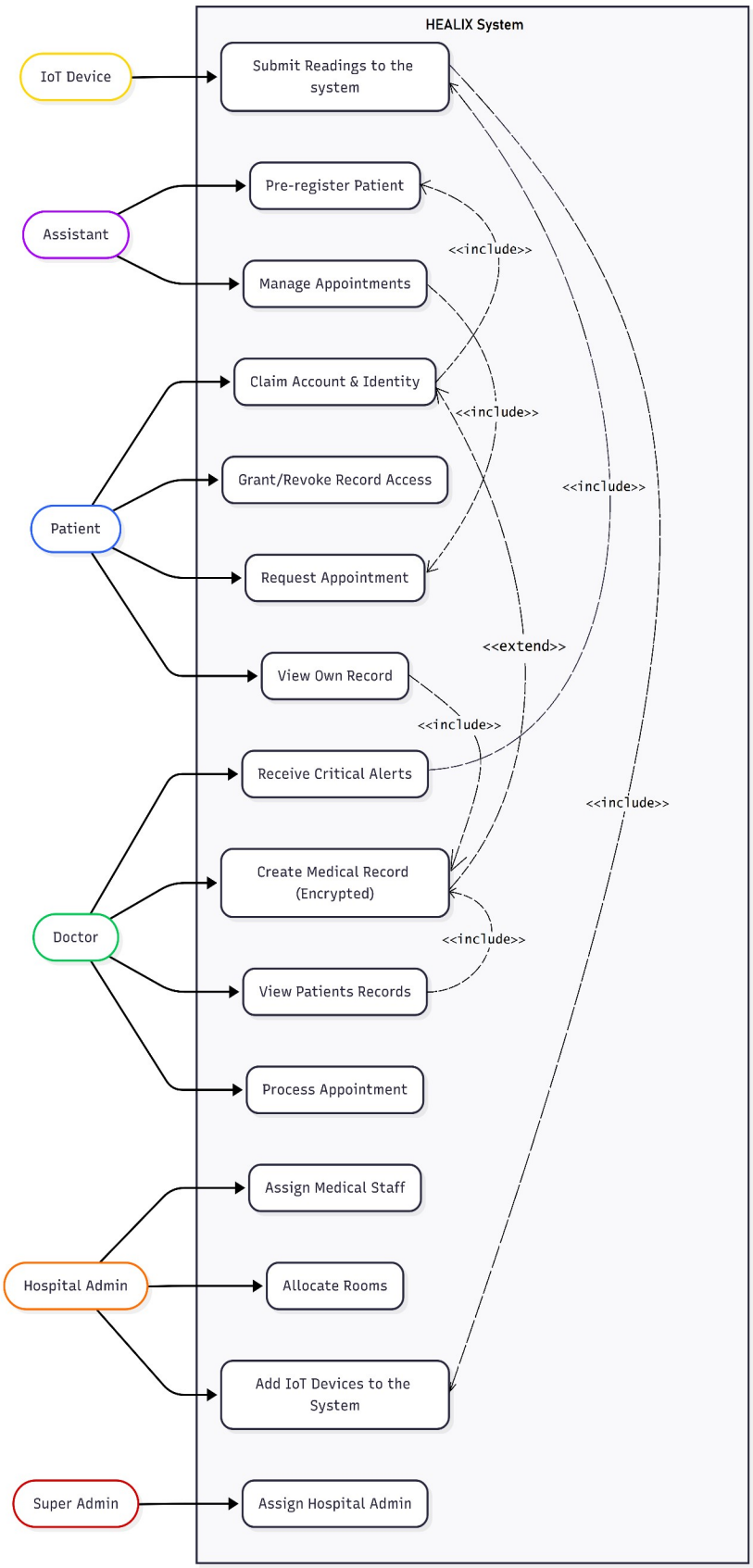

Figure 3.2: Use-Case Diagram of HEALIX

3.10.2 Class Diagram

This diagram represents the logical structure of the system, focusing on the main business entities (classes), their attributes, and their high-level relationships.

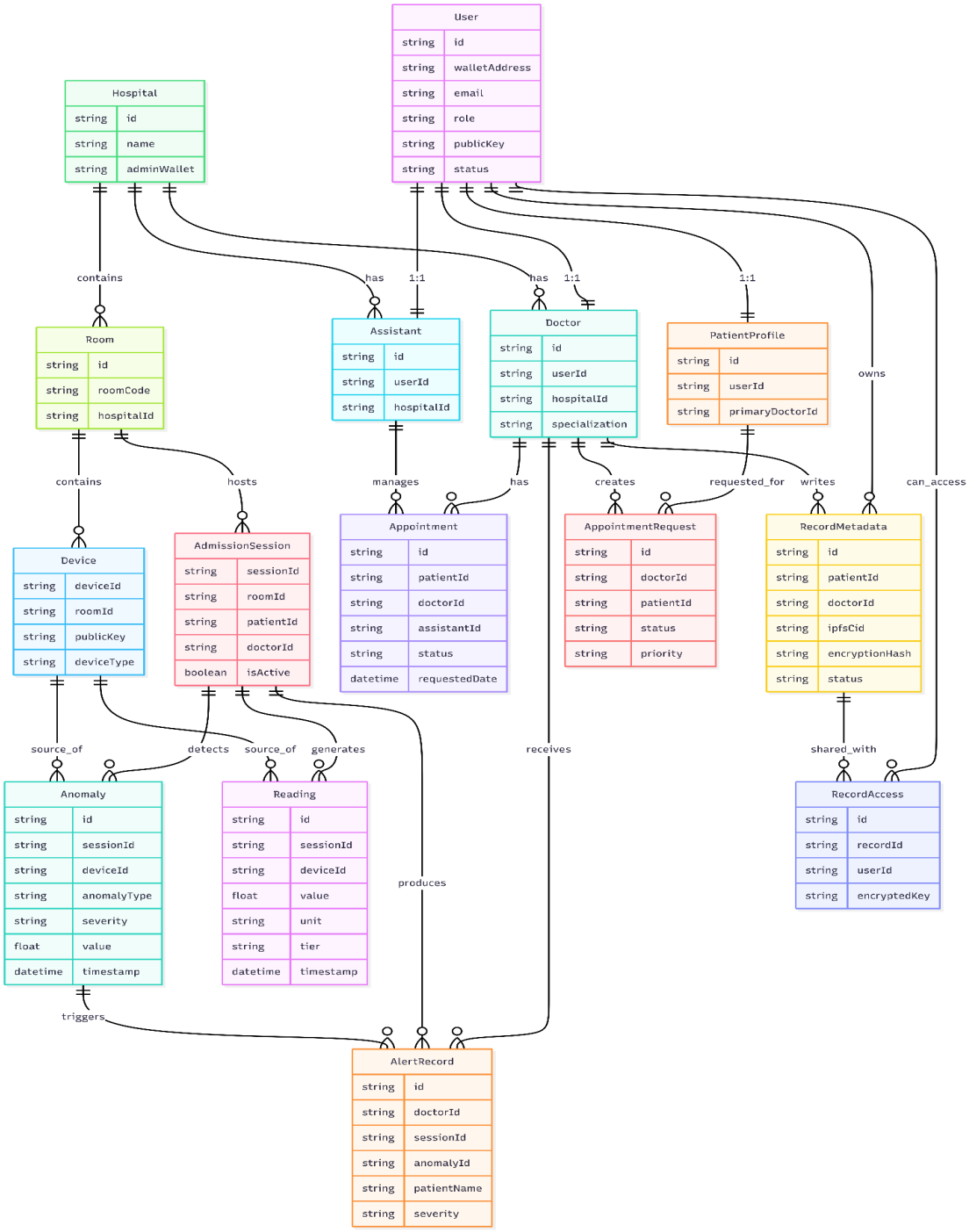

Figure 3.3: Domain Model, Conceptual Class Diagram

3.10.3 Entity-Relationship Diagram

This diagram translates the conceptual model into a real relational database structure ready for implementation with a DBMS like PostgreSQL.

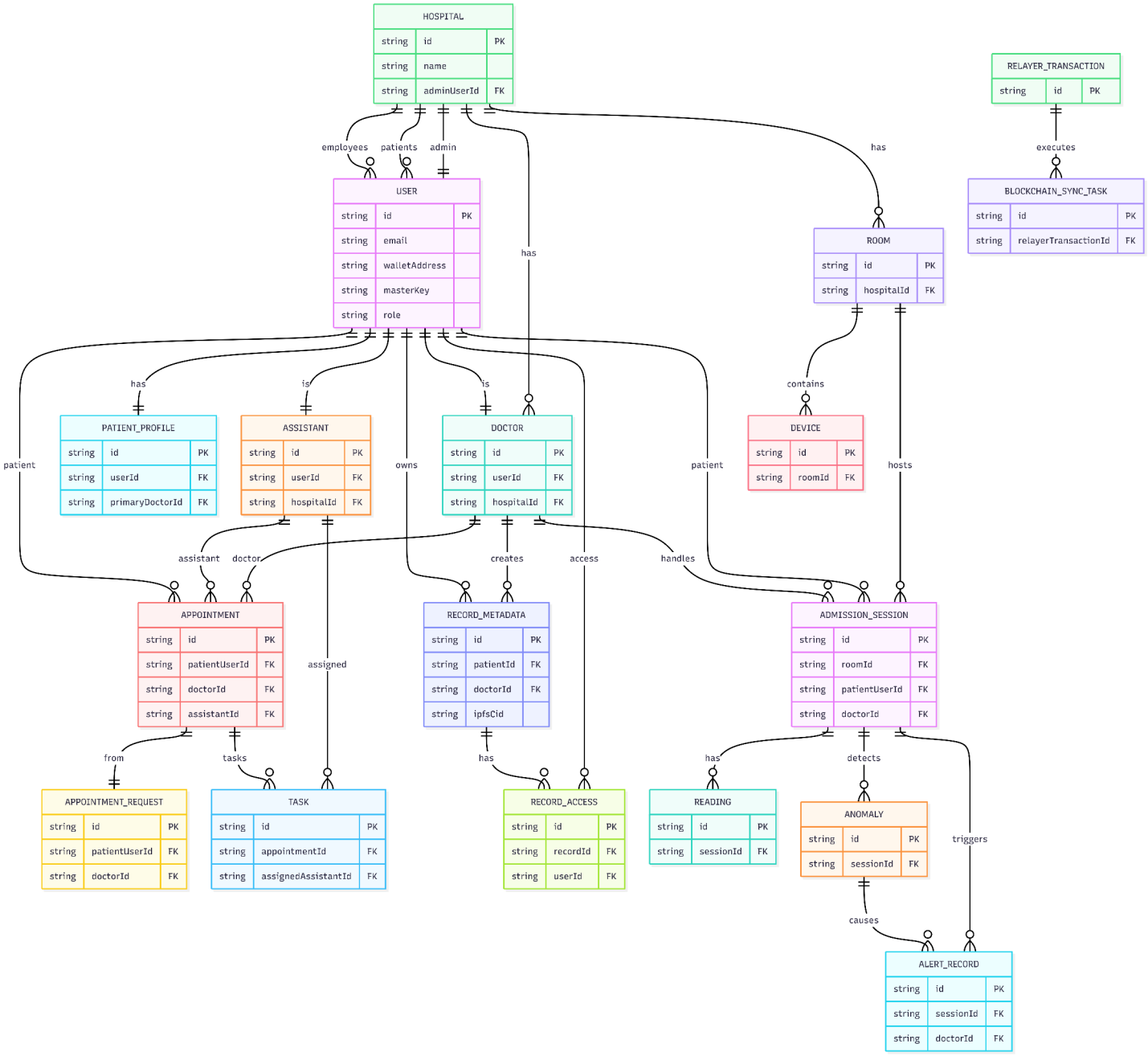

Figure 3.4: Database Schema, Entity-Relationship Diagram

3.10.4 Sequence Diagrams

3.10.4.1 Sequence Diagram for Patient Registration / Account Claim.

This flow illustrates how a patient is pre-registered by hospital staff and subsequently claims their account to generate their cryptographic identity.

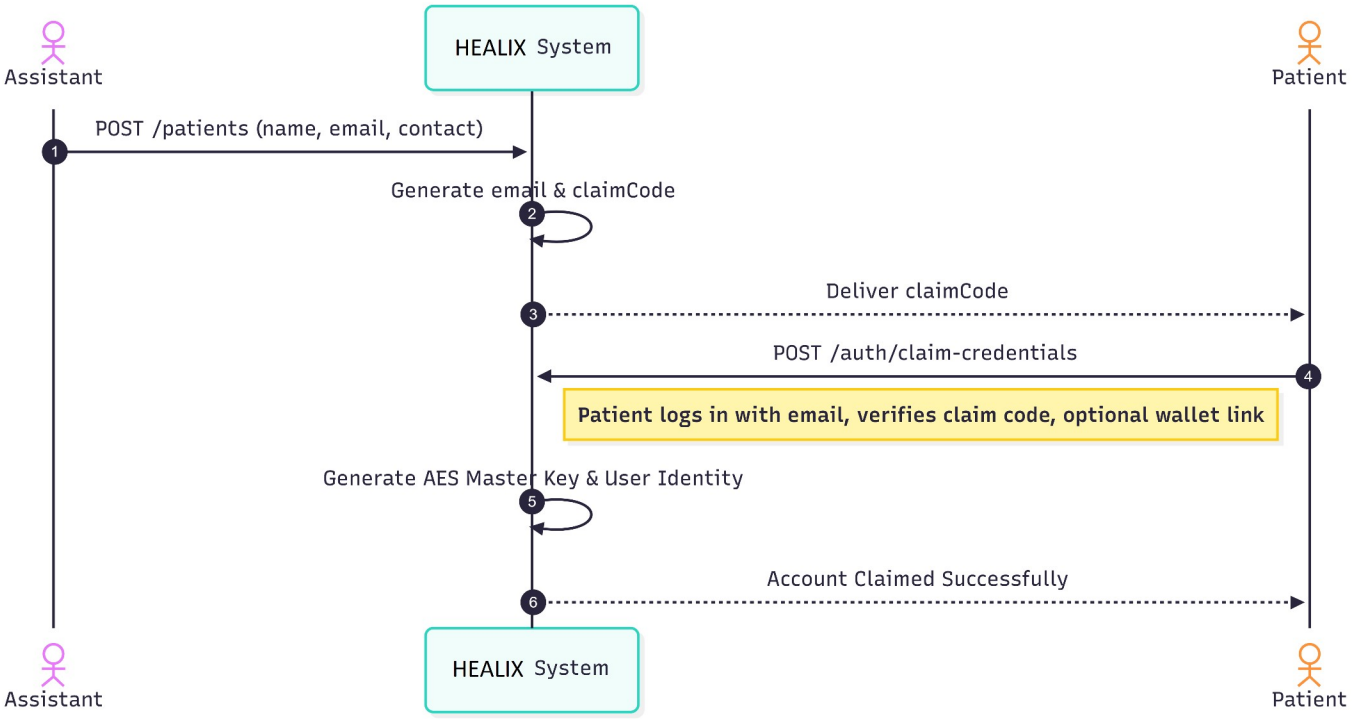

Figure 3.5: Sequence diagram For patient account claim

3.10.4.2 Sequence Diagram for the Appointment process

This flow details how a patient requests an appointment and how it is processed by the hospital's assistant.

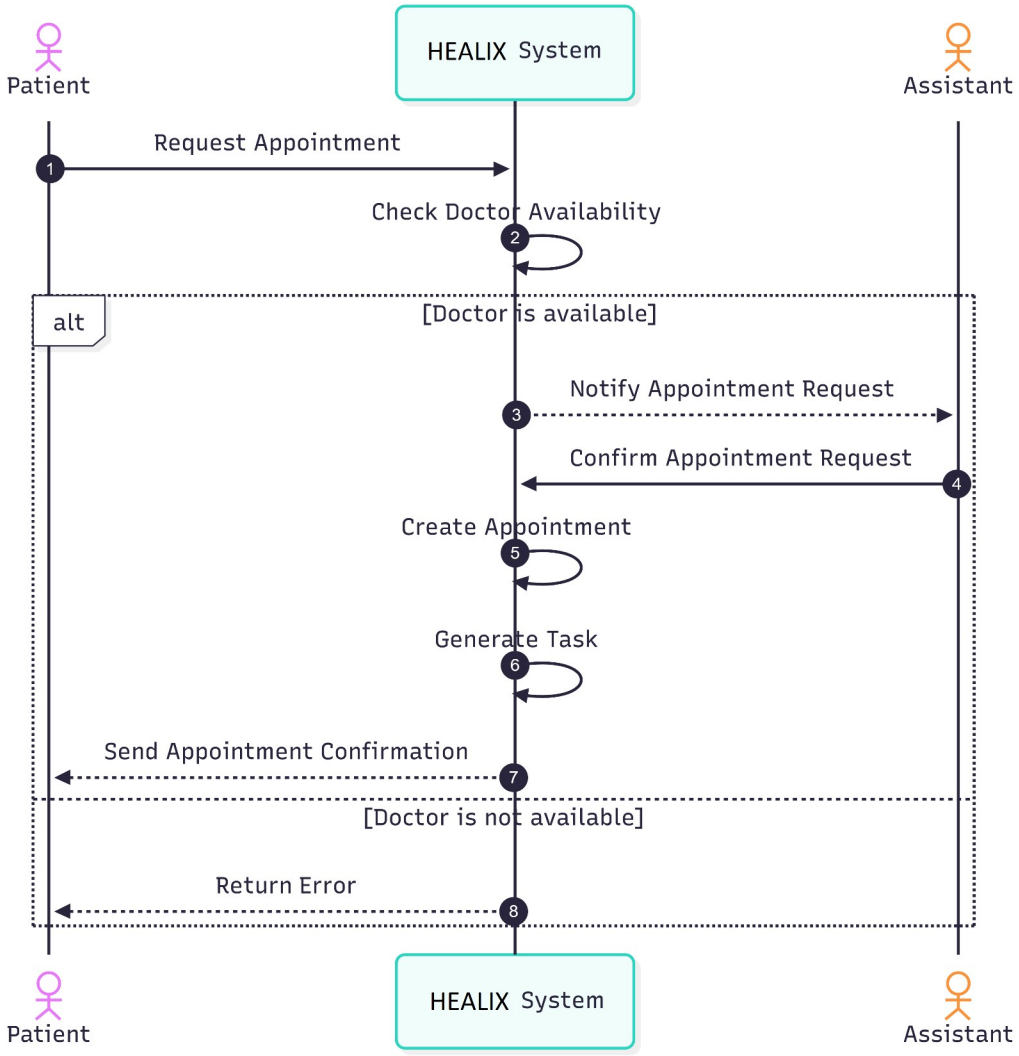

Figure 3.6: Sequence diagram for appointments

3.10.4.3 Sequence Diagram of the IoT Telemetry Pipeline

This flow shows the continuous processing of data from IoT devices, including verification, contextual resolution, and alerting.

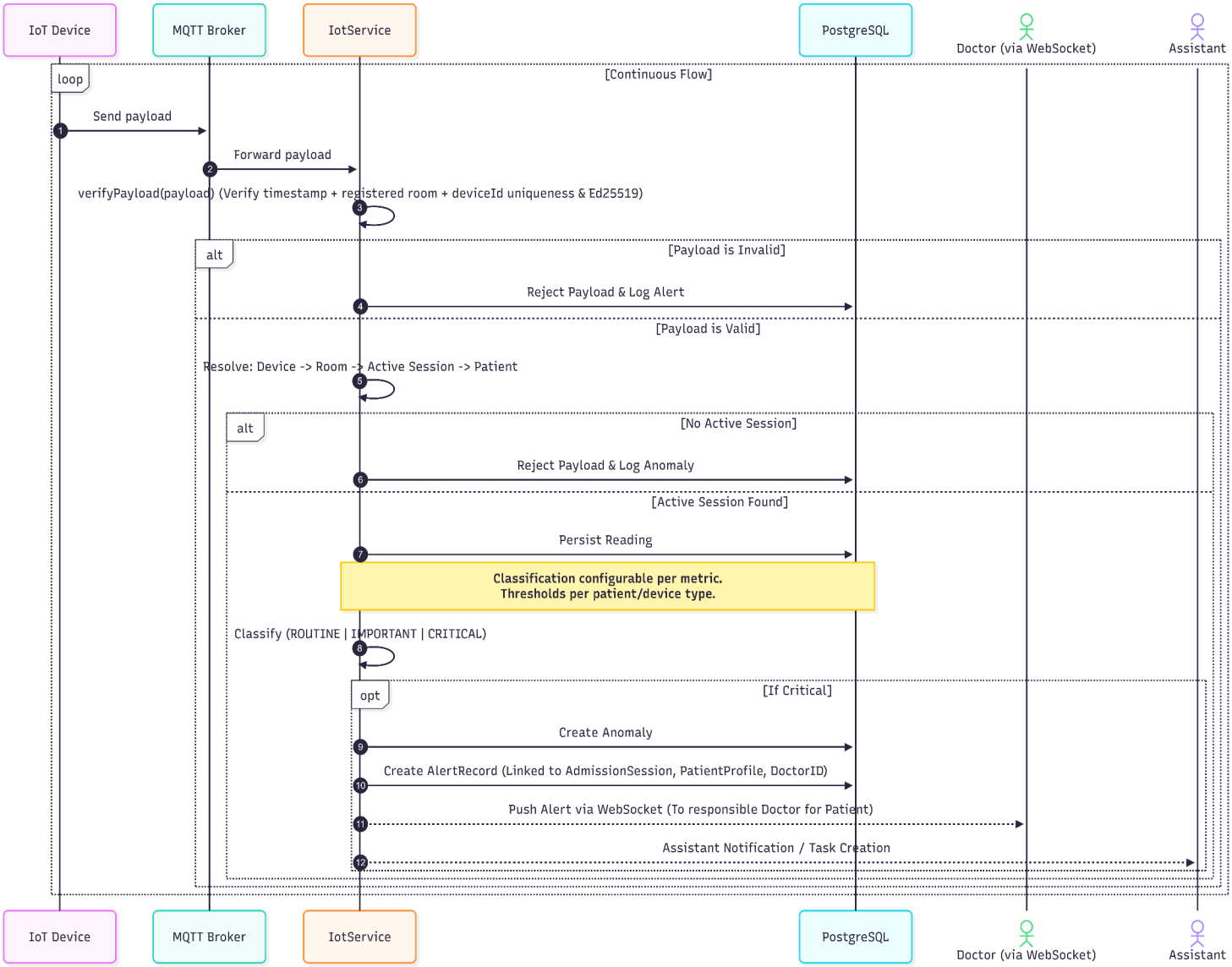

Figure 3.7: sequence diagram for the IoT pipeline

3.10.4.4 Sequence Diagram for the Medical Record Flow

This flow demonstrates the end-to-end lifecycle (Creation, Access, and Retrieval) of a medical record, emphasizing the cryptographic security model and patient-centric access control.

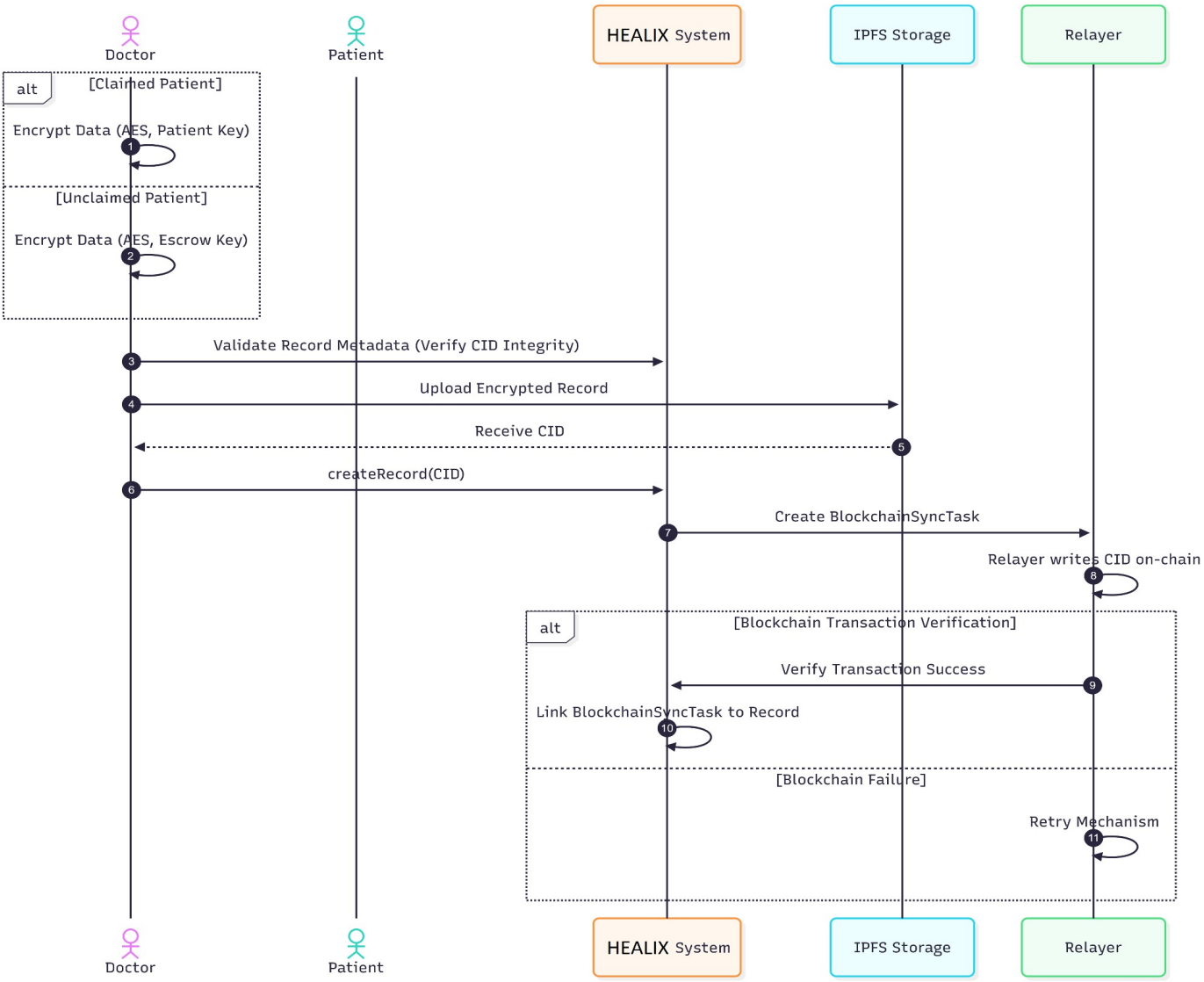

Figure 3.8: Sequence Diagram demonstrating Record Creation & Storage Flow and Blockchain Synchronization & Verification

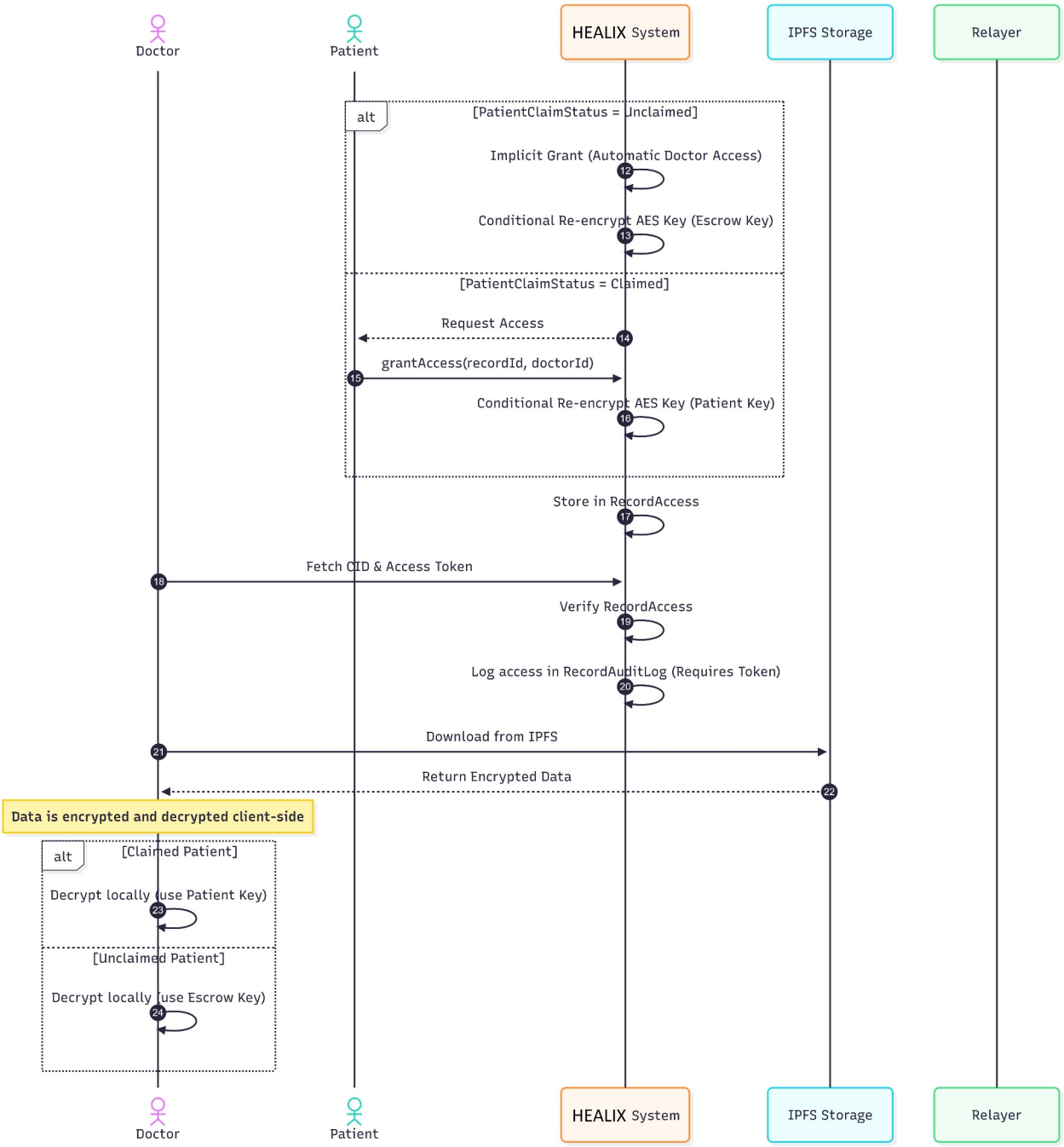

Figure 3.9: Sequence Diagram Demonstrating Access Control & Key Management, Record Retrieval & Decryption

3.11 Conclusion

The conceptual design of the HEALIX platform fundamentally reimagines healthcare data management by merging the immutable trust of blockchain technology with the agility of modern backend architectures and real-time IoT integration. By establishing distinct, role-based actors (Patients, Doctors, Administrators, and IoT Devices) and meticulously mapping their interactions through rigorous use-case and sequence workflows, the system is conceived not merely as a static data repository, but as a dynamic, deeply secure ecosystem.

The strategic architectural division between on-chain cryptographic anchoring and off-chain high-throughput processing ensures that sensitive medical records and critical health telemetry remain both fully verifiable and immediately accessible.

Ultimately, this foundational conceptual blueprint guarantees that as HEALIX transitions from theoretical design to practical implementation, it remains steadfastly aligned with its core objectives: granting patient data ownership, streamlining clinical workflows, and enforcing uncompromised data integrity.

Chapter 4

Implementation: Tools and Technology Stack

4.1 Introduction

This chapter details the technical implementation of the proposed HEALIX platform. The architecture follows a modular, service-oriented design spanning from the blockchain settlement layer to the IoT sensor edge. The technology selections outlined below were governed by the healthcare-specific constraints identified in Chapter 2: namely, high availability for IoMT data streams, cryptographic security for Protected Health Information (PHI), and off-chain scalability for large medical files.

4.2 Core Programming Languages and Tools

The platform is polyglot, utilizing languages best suited for specific runtime environments.

4.2.1 TypeScript / JavaScript

Definition: A strongly-typed superset of JavaScript that compiles to plain JavaScript, enabling static type-checking and advanced object-oriented patterns.[58]

Usage in Implementation: Serves as the primary language across the entire full-stack ecosystem. TypeScript is used in the NestJS backend for building type-safe APIs, in the React frontend for component state management, and in Hardhat scripts for writing blockchain deployment tests. This unified language reduces context-switching overhead during development.

Figure 4.1: TypeScript & JavaScript logos

4.2.2 Solidity (v0.8.24)

Definition: A high-level, object-oriented language created specifically for implementing smart contracts on Ethereum Virtual Machine (EVM) compatible blockchains.[59]

Usage in Implementation: Used to author the core smart contracts that govern the platform's logic. Version 0.8.24 includes built-in overflow checks, which are critical for preventing arithmetic vulnerabilities in contracts handling patient consent tokens and audit log references.

Figure 4.2: Solidity Logo

4.2.3 Python (v3)

Definition: An interpreted, high-level general-purpose programming language known for its extensive libraries in data science and hardware interfacing.[60]

Usage in Implementation: Exclusively deployed within the Fog/IoT Simulation Layer. It powers the node simulators that generate mock biometric data streams and executes lightweight machine learning models for anomaly detection (e.g., flagging irregular heart rate patterns before writing to the blockchain).

Figure 4.3: Python Logo

Installation: Python 3, pip, and virtual environments

```
> sudo apt install -y python3 python3-pip python3-venv
```

Verification:

```
> python3 --version   //Expected: Python 3.10+
> pip3 --version
```

4.2.4 Node.js

Definition: An open-source, cross-platform JavaScript runtime environment built on Chrome's V8 engine. It executes JavaScript code outside a web browser, enabling server-side scripting.[61]

Usage in Implementation: Serves as the foundational runtime environment for the entire backend and tooling ecosystem. The NestJS framework, Hardhat scripts, Prisma migrations, and Web3 libraries all execute within Node.js. Its event-driven, non-blocking I/O model is essential for handling the high-concurrency requirements of real-time IoMT data streams and WebSocket connections without performance degradation.

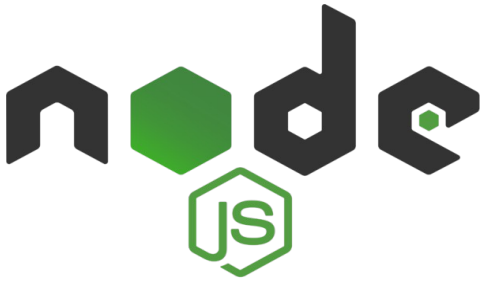

Figure 4.4: NodeJs Logo

Installation: Node.js (v22.x) and npm

```
> curl -fsSL https://deb.nodesource.com/setup_22.x | sudo -E bash -
> sudo apt install -y nodejs
```

Verification:

```
> node --version        //Expected: V22.16.0
> npm --version         //Expected: 10.9.2
```

4.3 Blockchain and Web3 Integration Layer

This layer is responsible for on-chain logic execution and connectivity between the off-chain application and the Ethereum/Sepolia network.

4.3.1 Hardhat

Definition: A comprehensive development environment and task runner for Ethereum software. It facilitates compiling, deploying, testing, and debugging EVM code locally. [62]

Usage in Implementation: Used as the primary smart contract workbench. It manages the local hardhat node for simulation, runs unit tests for the access control contracts, and executes deployment scripts to the Sepolia testnet.[63]

Figure 4.5: Hardhat & Sepolia Testnet logos

4.3.2 Ethers.js (v6.16.0) & Web3.py (v6.15.1)

Definition: *Ethers.js* is a compact JavaScript library for interacting with the Ethereum Blockchain, *Web3.py* is its Python counterpart.[64, 65]

Usage in Implementation: *Ethers.js* is integrated into the NestJS backend and React frontend to sign transactions and read contract state. *Web3.py* is utilized by the Fog Nodes to interact with the blockchain natively from a Python environment, allowing the simulation layer to write data hashes on-chain directly.

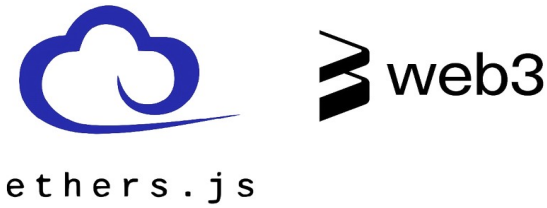

Figure 4.6: Ethers.js & Web3 Logos

4.3.3 OpenZeppelin Contracts

Definition: A library of secure, community-vetted, and reusable smart contracts for implementing standards like ERC-20, ERC-721, and access control mechanisms.[66]

Usage in Implementation: Inherited in the platform's smart contracts to implement Role-Based Access Control (RBAC) and Ownable patterns. This ensures that only authorized hospital nodes can validate blocks (Proof-of-Authority simulation) and that contract upgrades are secure.

Figure 4.7: OpenZeppelin Logo

4.3.4 Web3Auth

Definition: An authentication infrastructure that provides a seamless Web3 login experience using social logins (Google, Email) to generate and manage cryptographic key pairs. [67]
Usage in Implementation: Integrated into the frontend to abstract away private key management for non-technical users (patients/doctors). This is crucial for user adoption, as it eliminates the need for users to store seed phrases while maintaining self-custody of their medical access credentials.

4.4 Backend Architecture and Data Storage

This layer handles business logic, off-chain data persistence, and secure storage of large medical files.

4.4.1 NestJS (v11)

Definition: A progressive Node.js framework for building scalable and maintainable server-side applications using TypeScript. It enforces a modular architecture (Modules, Controllers, Providers).[68]

Usage in Implementation: Serves as the core orchestration backbone. It exposes RESTful APIs (Representational State Transfer Application Programming Interface) for the frontend, manages WebSocket gateways for real-time alerts, and integrates all other services (Prisma, Redis, MQTT) via dependency injection modules.

Figure 4.8: NestJS Logo

4.4.2 Prisma ORM (v7.3.0) & PostgreSQL

Definition: *Prisma* is a next-generation Object-Relational Mapping (ORM) tool. *PostgreSQL* is a powerful open-source relational database.[69, 70]

Usage in Implementation: Used to model and store off-chain metadata. This includes user profiles (non-sensitive), device registration data, and mapping tables that link patient IDs to IPFS Content Identifiers (CIDs). PostgreSQL ensures ACID (Atomicity, Consistency, Isolation, and Durability) compliance for audit logs that do not require blockchain-level immutability.

Figure 4.9: PrismaORM & PostgreSQL Logos

installation: PostgreSQL 15 & Start Service

```
> sudo apt install -y postgresql postgresql-contrib
> sudo systemctl start postgresql
> sudo systemctl enable postgresql
```

4.4.3 IPFS & Pinata

Definition: The InterPlanetary File System (IPFS) is a peer-to-peer hypermedia protocol for storing data in a decentralized manner. Pinata provides a pinning service ensuring IPFS files remain accessible.[71, 72]

Usage in Implementation: All large medical files (MRI scans, X-ray images) are encrypted and stored off-chain on IPFS via the Pinata SDK. The backend receives the IPFS CID and stores only this 46-byte hash reference on the blockchain. This solves the scalability and GDPR compliance issues associated with on-chain storage.

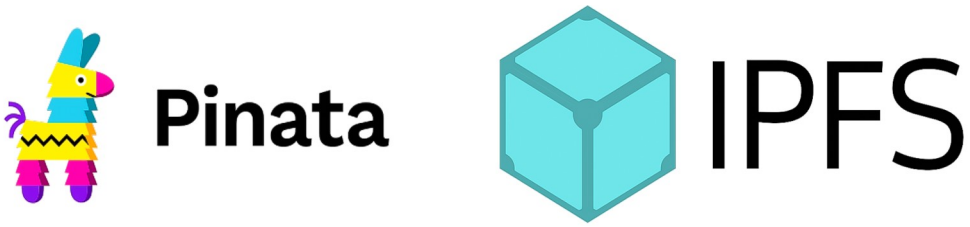

Figure 4.10: Pinata & IPFS Logos

4.4.4 Socket.io & MQTT

Definition: Socket.io enables real-time, bidirectional event-based communication between browser and server. *MQTT* is a lightweight messaging protocol optimized for high-latency or low-bandwidth networks common in IoT.[73, 74]

Usage in Implementation: Socket.io pushes instant notifications to the React frontend when a new patient record is shared. *MQTT* acts as the bridge listening to the Fog/IoT Layer, ingesting streams of biometric data from simulated wearable sensors.

Figure 4.11: Socket.io & MQTT Logos

Installation: Mosquitto MQTT Broker & Start Service

```
> sudo apt install -y mosquitto mosquitto-clients
> sudo systemctl start mosquitto
> sudo systemctl enable mosquitto
```

4.4.5 BullMQ / Redis

Definition: BullMQ is a Node.js library that implements a fast and robust queue system built on top of Redis that helps in resolving many modern age micro-services architectures (in-memory data store) for handling asynchronous background tasks.[75, 76]

Usage in Implementation: Manages computationally intensive background jobs. For example, when a new file is uploaded, a BullMQ job is queued to handle file encryption and IPFS upload without blocking the main HTTP thread.

Figure 4.12: Redis & BullMQ Logos

4.4.6 Security & Auth Libraries

JWT/Passport: Manages stateless authentication for the REST API.[77]

Crypto-js / Eth-crypto: Used for encrypting data client-side before it ever touches the server and for verifying cryptographic signatures generated by the patient's Web3Auth wallet.[78, 79]

4.5 Frontend User Interface

The presentation layer designed for clinicians and patients.

4.5.1 React (v18.3.1) & Vite (v5.2.0)

Definition: *React* is a declarative JavaScript library for building component-based user interfaces. *Vite* is a modern build tool that offers extremely fast Hot Module Replacement (HMR).[80, 81]

Usage in Implementation: *React* structures the UI into reusable components (e.g., PatientDashboard).

Vite provides the development environment and optimized production bundles, with custom plugins to polyfill Node.js crypto modules needed for Web3 wallet interactions.

Figure 4.13: React & Vite Logos

4.5.2 Tailwind CSS & Recharts

Definition: *Tailwind* is a utility-first CSS framework. *Recharts* is a charting library built on D3 and React.[82, 83]

Usage in Implementation: *Tailwind* ensures a responsive, accessible design compliant with healthcare UX standards. *Recharts* visualizes the real-time biometric data streams received via MQTT/Socket.io, rendering live heart rate and blood pressure trends on the patient dashboard.

Figure 4.14: Tailwind CSS & Recharts Logos

4.5.3 React Hook Form & i18next

Definition: *React Hook Form* manages form state and validation efficiently. *I18next* is an internationalization framework, it detects the user language and loads the translations.[84, 85]

Usage in Implementation: Used for creating complex medical intake forms with minimal re-renders. *i18next* prepares the platform for multi-region deployment, supporting multiple languages for patient-facing consent forms.

Figure 4.15: i18next & React Hook Form Logos

4.6 Project Dependencies Installation

These commands are used to install the necessary Node.js packages for the various platform components.

```
// 1. Root Directory (Hardhat & Smart Contracts)
> cd /home/anesx/HEALIX-project
> npm install

// 2. Backend Directory (NestJS & Prisma)
> cd /home/anesx/HEALIX-project/backend
> npm install

// 3. Frontend Directory (React & Vite)
> cd /home/anesx/HEALIX-project/frontend
> npm install
```

4.7 Database Initialization

Commands to create the database and run the Prisma migrations.

```
// Create the PostgreSQL database
> sudo -u postgres psql -c "CREATE DATABASE HEALIX_db;"

// Generate Prisma Client and Push Schema
> cd /home/anesx/HEALIX-project/backend
> npx prisma generate
> npx prisma db push

// Optional: Seed the database with initial data
> npx prisma db seed
```

4.8 IoT Simulation and Fog Computing Layer

This layer simulates the edge environment where data is generated and pre-processed.

4.8.1 Scikit-learn & NumPy

Definition: *Scikit-learn* is a machine learning library for Python. *NumPy* provides support for large, multi-dimensional arrays and matrices.[86, 87]

Usage in Implementation: This combination simulates Edge AI on a Fog Node. The script runs a lightweight anomaly detection model (e.g., Isolation Forest) on incoming vitals to detect anomalies *before* pushing data to the cloud. This reduces bandwidth and ensures only validated or emergency-alert data triggers a blockchain transaction.

Figure 4.17: Scikit-learn & NumPy Logos

4.8.2 Eclipse Paho MQTT

Definition: An MQTT client implementation in Python.[88]

Usage in Implementation: Used to publish the simulated sensor readings (heart rate, SpO2) to the central MQTT broker, mimicking the behavior of a real WBAN gateway.

Figure 4.18: Eclipse Paho Logo

4.9 Python Environment Setup

The IoT simulation and fog layers require dedicated Python virtual environments.

```
// IoT Simulation Virtual Environment Setup
> cd /home/anesx/HEALIX-project/iot_simulation
> python3 -m venv .venv
> source .venv/bin/activate
> pip install -r requirements.txt
> deactivate

// Fog Layer Virtual Environment Setup
> cd /home/anesx/HEALIX-project/fog-layer
> python3 -m venv .venv
> source .venv/bin/activate
> pip install -r requirements.txt
> deactivate
```

4.10 Development Operations and Quality Assurance

4.10.1 Docker Compose

Definition: A tool for defining and running multi-container Docker applications.[89]

Usage in Implementation: Defines the local development environment. A single docker-compose up command spins up the PostgreSQL database, Redis cache, and Mosquitto MQTT broker. This ensures all developers and researchers can reproduce the exact same environment required for running the integration tests.

Installation:

```
> sudo apt install -y docker.io docker-compose
> sudo systemctl start docker
> sudo systemctl enable docker
> sudo usermod -aG docker $USER
```

4.10.2 ESLint, Prettier & Jest

Definition: *ESLint* statically analyzes code for errors. *Prettier* enforces consistent formatting. *Jest* is a JavaScript testing framework.[90, 91, 92]

Usage in Implementation: Integrated into the CI/CD pipeline. *Jest* runs unit tests on the NestJS services and smart contract interaction logic to verify that consent management and data anchoring functions perform as expected before deployment.

Figure 4.19: ESLint, Prettier, Jest Logos

4.10.3 Vite Node Polyfills

Definition: A plugin that provides browser-compatible implementations of Node.js core modules.[93]

Usage in Implementation: Essential for the React frontend to function correctly. It polyfills modules like Buffer and crypto, allowing the ethers.js library to perform cryptographic signing and hashing within the browser environment without breaking the Vite build process.

4.11 Implementation Technology Summary Table

Table 4.1: A summary table of all the tools used

Category	Tool / Library	Version / Reference	Primary Implementation Role
Core Languages & Tools	NodeJs	Latest LTS	Server-side JavaScript runtime for backend and tooling.
	TypeScript / JavaScript	Latest	Full-stack development (Backend API, Frontend UI, Blockchain Scripts).
	Solidity	v0.8.24	Smart contract logic for consent and audit trails.
	Python	v3	Fog/IoT simulation and ML anomaly detection.
Blockchain & Web3	Hardhat	@nomicfoundation	Smart contract development, testing, and deployment pipeline.
	Ethers.js	v6.16.0	Blockchain interaction library for Node.js and Browser.
	Web3.py	v6.15.1	Blockchain interaction library for Python Fog Nodes.
	OpenZeppelin	Community Vetted	Secure base contracts for Access Control (RBAC).
	Web3Auth	@web3auth/modal	User-friendly wallet onboarding and key management.
Backend & Data	NestJS	v11	Scalable server-side application framework.
	Prisma ORM	v7.3.0	Type-safe database schema management and querying.
	PostgreSQL	v8.17.2	Relational storage for off-chain metadata and user profiles.
	IPFS / Pinata	@pinata/sdk	Decentralized, off-chain storage for encrypted medical files.

	Socket.io	@nestjs/platform	Real-time bidirectional communication for alerts.
	MQTT	v5.15.0	Lightweight messaging protocol for IoT sensor data ingestion.
	BullMQ / Redis	@nestjs/bull	Asynchronous background job processing (file uploads).
Frontend	React	v18.3.1	Core UI component library.
	Vite	v5.2.0	Build tool and development server with HMR.
	Tailwind CSS	v3.4.18	Utility-first styling for responsive healthcare UI.
	Recharts	Latest	Data visualization for biometric trend charts.
	i18next	react-i18next	Multi-language support for global deployment readiness.
IoT/Fog Simulation	Scikit-learn	Latest	Lightweight ML for edge-based anomaly detection.
	Eclipse Paho MQTT	paho-mqtt	MQTT client for publishing simulated biometric data.
	NumPy	Latest	Numerical computation supporting ML models.
DevOps & QA	Docker Compose	docker-compose.yml	Local infrastructure orchestration (DB, Redis, Broker).
	Jest	Latest	Unit and end-to-end testing framework.
	ESLint / Prettier	Latest	Code quality enforcement and consistent formatting.
	Vite Node Polyfills	Plugin	Enabling Node.js core modules (crypto) in the browser.

4.12 Startup Commands

The core platform requires multiple services to be running simultaneously. PostgreSQL and Mosquitto need to be running before starting the following services.

```
// 1. Start Hardhat Blockchain Node (Terminal 1)
> cd /home/anesx/HEALIX-project
> npx hardhat node

// 2. Deploy Smart Contracts / Bootstrap (Terminal 2)
// Run only after the Hardhat node is up
> cd /home/anesx/HEALIX-project
> npx hardhat run scripts/bootstrap.cjs --network localhost

// 3. Start Backend (Terminal 3)
> cd /home/anesx/HEALIX-project/backend
> npm run start:dev

// 4. Start Frontend (Terminal 4)
// Clear Vite cache first if necessary: rm -rf node_modules/.vite
> cd /home/anesx/HEALIX-project/frontend
> npm run dev
```

```
// 5. Start IoT Simulation (Terminal 5 – Optional)
> cd /home/anesx/HEALIX-project/iot_simulation
> source .venv/bin/activate
python3 simulator.py

// 6. Start Fog Layer (Terminal 6 – Optional)
> cd /home/anesx/HEALIX-project/fog-layer
> source .venv/bin/activate
> python3 fog_node.py
```

4.13 Smart Contracts: Technical Details and Code Snippets

This section outlines the primary smart contracts powering the HEALIX ecosystem, detailing their core functionality and most important methods.

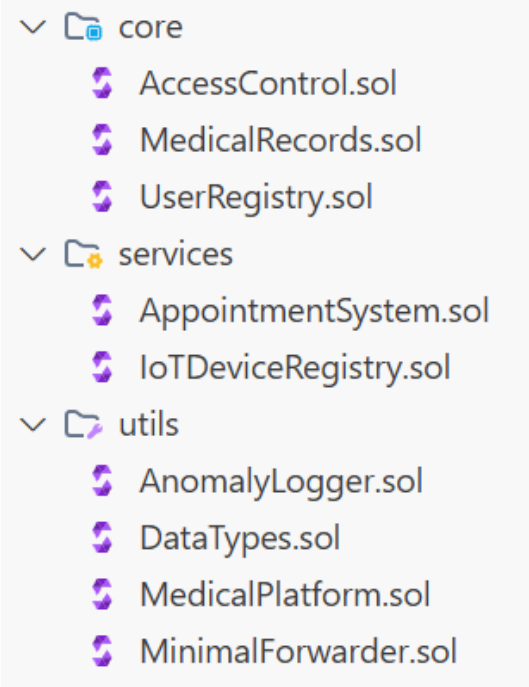

Figure 4.20: Solidity Smart Contracts and their placement in our system

4.13.1 AccessControl.sol

The AccessControl abstract contract serves as a centralized authorization layer, decoupling permission logic from business operations. This separation reduces redundancy across contracts and ensures consistent enforcement of role-based constraints. By leveraging standardized components from OpenZeppelin, the design avoids custom security implementations, minimizing the risk of access control vulnerabilities.

```
/**
 * @title MediChainAccessControl
 * @dev Role-based access control using OpenZeppelin's AccessControl.
 * Defines roles for the MediChain platform: DEFAULT_ADMIN, HOSPITAL_ADMIN, DOCTOR, PATIENT.
 */
abstract contract MediChainAccessControl is AccessControl {
    // --- Role Definitions ---
    bytes32 public constant HOSPITAL_ADMIN_ROLE =
        keccak256("HOSPITAL_ADMIN_ROLE");
    bytes32 public constant DOCTOR_ROLE = keccak256("DOCTOR_ROLE");
    bytes32 public constant PATIENT_ROLE = keccak256("PATIENT_ROLE");
    bytes32 public constant ASSISTANT_ROLE = keccak256("ASSISTANT_ROLE");
    bytes32 public constant IOT_DEVICE_ROLE = keccak256("IOT_DEVICE_ROLE");
```

Listing 4.1: Role hierarchy in AccessControl.sol contract

```
function _linkDoctorToHospital(
    address doctor,
    address hospitalAdmin
) internal {
    doctorToHospital[doctor] = hospitalAdmin;
    emit DoctorLinkedToHospital(doctor, hospitalAdmin, block.timestamp);
}
```

Listing 4.2: Doctor–hospital association mechanism

```
function _setUserActiveStatus(address user, bool active) internal {
    isUserActive[user] = active;
    emit UserStatusChanged(user, active, _msgSender(), block.timestamp);
}
```

Listing 4.3: Function Used internally for Global user state enforcement

4.13.2 MedicalRecords.sol

The “MedicalRecords” contract governs the immutable registration of patient health data references, associated metadata, and cryptographic access permissions. Rather than storing raw medical data on-chain, the design anchors IPFS content identifiers while enforcing patient-centric access control policies. The contract enables controlled data sharing, dynamic permission management, and emergency override capabilities, ensuring both privacy and availability in critical scenarios.

```
function createMedicalRecord(
    bytes32 recordId,
    bytes32 patientIdHash,
    string calldata ipfsCid,
    bytes32 encryptionHash
) external onlyDoctor {
    _createRecord(recordId, patientIdHash, ipfsCid, encryptionHash);
}
```

Listing 4.4: Create Medical Record function in MedicalRecords.sol

```
function grantAccess(
    address doctor
) external onlyPatient onlyActiveUser(doctor) {
    if (userRegistry.userRoles(doctor) != DataTypes.UserRole.DOCTOR)
        revert NotAuthorized();
    address sender = _msgSender();
    if (accessPermissions[sender][doctor]) revert AccessAlreadyGranted();

    accessPermissions[sender][doctor] = true;
    totalAccessGrants++;

    emit AccessGranted(sender, doctor, sender, block.timestamp);
}
```

Listing 4.5:Grant Access Function in MedicalRecords.sol

```
    function revokeAccess(address doctor) external onlyPatient {
        address sender = _msgSender();
        if (!accessPermissions[sender][doctor]) revert AccessNotGranted();

        accessPermissions[sender][doctor] = false;

        emit AccessRevoked(sender, doctor, sender, block.timestamp);
    }
```

Listing 4.6: Revoke Access Function in MedicalRecords.sol

```
function breakGlass(
    address patient,
    string calldata reason
) external onlyDoctor {
    if (bytes(reason).length == 0) revert EmergencyReasonRequired();
    if (bytes(reason).length > 256) revert ReasonTooLong();

    address sender = _msgSender();
    address doctorHospital = userRegistry.userHospital(sender);
    if (doctorHospital == address(0)) revert DoctorNotLinkedToHospital(); // Doctor must be in a hospital

    address patientHospital = userRegistry.userHospital(patient);
    // If patient is linked to a hospital, doctor must be in the same one for break-glass access.
    if (
        patientHospital != address(0) && patientHospital != doctorHospital
    ) {
        revert DifferentHospitalAccessDenied();
    }

    hasActiveBreakGlass[patient][sender] = true;
    breakGlassHistory[patient][sender].push(
        BreakGlassAccess({
            used: true,
            timestamp: block.timestamp,
            reason: reason
        })
    );

    emit BreakGlassActivated(sender, patient, block.timestamp, reason);
}
```

Listing 4.7: Break Glass Emergency Function in MedicalRecords.sol

4.13.3 UserRegistry.sol

The “UserRegistry” contract manages participant onboarding, identity binding, and role attribution within the system. It establishes verifiable on-chain identities for medical staff and provides a secure mechanism to associate real-world patients with blockchain accounts. This design bridges off-chain identity verification with on-chain authorization, ensuring that all interactions originate from authenticated and properly classified entities.

```
// --- Account Claiming ---
function claimAccount(bytes32 idHash, string calldata code) external {
    if (idToWallet[idHash] == address(0)) revert UserNotFound();
    address placeholder = idToWallet[idHash];
    if (users[placeholder].isClaimed) revert AccountAlreadyClaimed();
    address sender = _msgSender();
    if (users[sender].wallet != address(0))
        revert WalletAlreadyRegistered();
    if (keccak256(abi.encodePacked(code)) != activationCodeHashes[idHash])
        revert InvalidActivationCode();

    // Transfer data from placeholder to real wallet
    DataTypes.User storage user = users[placeholder];
    user.wallet = sender;
    user.isClaimed = true;
    users[sender] = users[placeholder];
    delete users[placeholder];

    idToWallet[idHash] = sender;
    _grantRole(PATIENT_ROLE, sender);
    _setUserActiveStatus(sender, true);

    // Update hospital users list if applicable
    address admin = userHospital[sender];
    if (admin != address(0)) {
        address[] storage hUsers = hospitalUsers[admin];
        for (uint i = 0; i < hUsers.length; i++) {
            if (hUsers[i] == placeholder) {
                hUsers[i] = sender;
                break;
            }
        }
    }
    // Update allUsers array
    for (uint i = 0; i < allUsers.length; i++) {
        if (allUsers[i] == placeholder) {
            allUsers[i] = sender;
            break;
        }
    }
    emit AccountClaimed(sender, idHash, block.timestamp);
}
```

Listing 4.8: Account Claiming Function from UserRegistry.sol

4.13.4 AppointmentSystem.sol

The “AppointmentSystem” contract coordinates the scheduling lifecycle of patient–doctor consultations. It maintains the state transitions of appointments, enabling booking, modification, and completion tracking in a transparent and tamper-resistant manner. By encoding scheduling logic on-chain, the system ensures consistency and auditability across all interactions between patients and healthcare providers.

4.13.5 IoTDeviceRegistry.sol

The “IoTDeviceRegistry” contract oversees the registration, association, and operational state of medical IoT devices within healthcare environments. It links devices to physical contexts such as rooms and patient sessions, while anchoring telemetry data hashes and detected anomalies to the blockchain. This approach enables verifiable tracking of device activity and ensures the integrity of high-frequency medical data streams without incurring excessive on-chain storage costs.

```
function recordReading(
    bytes32 sessionId,
    bytes32 readingHash,
    string calldata ipfsHash
) external onlyAuthorizedRelay {
    require(sessions[sessionId].isActive, "IoT: session not active");

    sessionReadings[sessionId].push(
        DeviceReading({
            sessionId: sessionId,
            readingHash: readingHash,
            ipfsHash: ipfsHash,
            timestamp: block.timestamp,
            device: msg.sender
        })
    );

    if (devices[msg.sender].deviceWallet != address(0)) {
        devices[msg.sender].lastUpdate = block.timestamp;
    }

    emit ReadingRecorded(msg.sender, sessionId, readingHash);
}
```

Listing 4.9: Record Readings Mechanism for high-frequency Telemetry

4.13.6 AnomalyLogger

The “AnomalyLogger” contract functions as a dedicated, immutable logging layer for critical health-related events. It records alerts generated by monitoring systems with associated severity levels, ensuring that significant incidents are permanently auditable and cannot be altered or suppressed. This separation of concerns isolates critical event tracking from general data flows, improving reliability and traceability.

```
function logAnomaly(
    address deviceAddress,
    address patientAddress,
    string calldata anomalyType,
    string calldata severity
) external returns (uint256) {
    _anomalyCounter++;
    uint256 newId = _anomalyCounter;

    anomalies[newId] = AnomalyEvent({
        id: newId,
        deviceAddress: deviceAddress,
        patientAddress: patientAddress,
        anomalyType: anomalyType,
        severity: severity,
        timestamp: block.timestamp,
        dataHash: bytes32(0) // Placeholder for future IPFS hash
    });

    patientAnomalies[patientAddress].push(newId);

    emit AnomalyDetected(newId, patientAddress, anomalyType, block.timestamp);

    return newId;
}
```

Listing 4.10: IoT data anchoring and anomaly detection mechanisms

4.13.7 MedicalPlatform.sol

The “MedicalPlatform” contract acts as a unifying coordination layer that integrates the system’s modular components. It facilitates interaction between specialized contracts and provides aggregated access to distributed data structures. This abstraction simplifies external interaction with the platform while preserving the modularity and separation of responsibilities across the underlying architecture.

```
/**
 * @notice Initialize the platform with all contract addresses
 * @dev This should be called once after all contracts are deployed
 */
function initialize(
    address _userRegistry,
    address _medicalRecords,
    address _appointmentSystem,
    address _iotDeviceRegistry
) external onlyOwner {
    require(!isPlatformActive, "Platform already initialized");
    require(_userRegistry != address(0), "Invalid UserRegistry address");
    require(
        _medicalRecords != address(0),
        "Invalid MedicalRecords address"
    );
    require(
        _appointmentSystem != address(0),
        "Invalid AppointmentSystem address"
    );
    require(
        _iotDeviceRegistry != address(0),
        "Invalid IoTDeviceRegistry address"
    );
    userRegistry = UserRegistry(_userRegistry);
    medicalRecords = MedicalRecords(_medicalRecords);
    appointmentSystem = AppointmentSystem(_appointmentSystem);
    iotDeviceRegistry = IoTDeviceRegistry(_iotDeviceRegistry);
    isPlatformActive = true;
    emit PlatformDeployed(
        _userRegistry,
        _medicalRecords,
        _appointmentSystem,
        _iotDeviceRegistry,
        block.timestamp
    );
}
```

Listing 4.11: Cross-contract initialization and coordination mechanism

4.14 Test Environment

This project was tested on 2 machines supporting 2 different Operating Systems:

Table 4.2: Test Devices Specifications

Specifications	Machine 1	Machine 2
Operating System	Linux – Ubuntu	Windows – 10 Home
Device Model	Dell Latitude 5490	LENOVO Legion 7 16ACHg6
Processor	Intel® Core™ i5-8250U @ 1.60 GHz (4 Cores / 8 Threads)	AMD Ryzen 7 5800H @ 3.20Ghz Boost Max 4.45Ghz (8 Cores / 16 Threads)
Graphical Unit	Intel UHD Graphics 620 (Integrated)	NVIDIA GeForce RTX 3080 Laptop (Dedicated) 16GB
Memory Configuration	2 × 8 GB DDR4 SODIMM	2 × 8 GB DDR4-3200MHz SDRAM
Storage Type and Capacity	SK hynix PC401 NVMe SSD 256GB	Samsung NVMe Gen4 PCIe M.2 SSD 1TB

4.15 The Interfaces of The System

In this section, some of our system's interfaces will be presented, starting with the home page then the users Dashboards.

4.15.1 The Home Page

This represents the main page of our system, it offers the users (admins, patients, healthcare professionals) the possibility to access each of their own space to enter the information and to proceed to their corresponding pages (Dashboards).

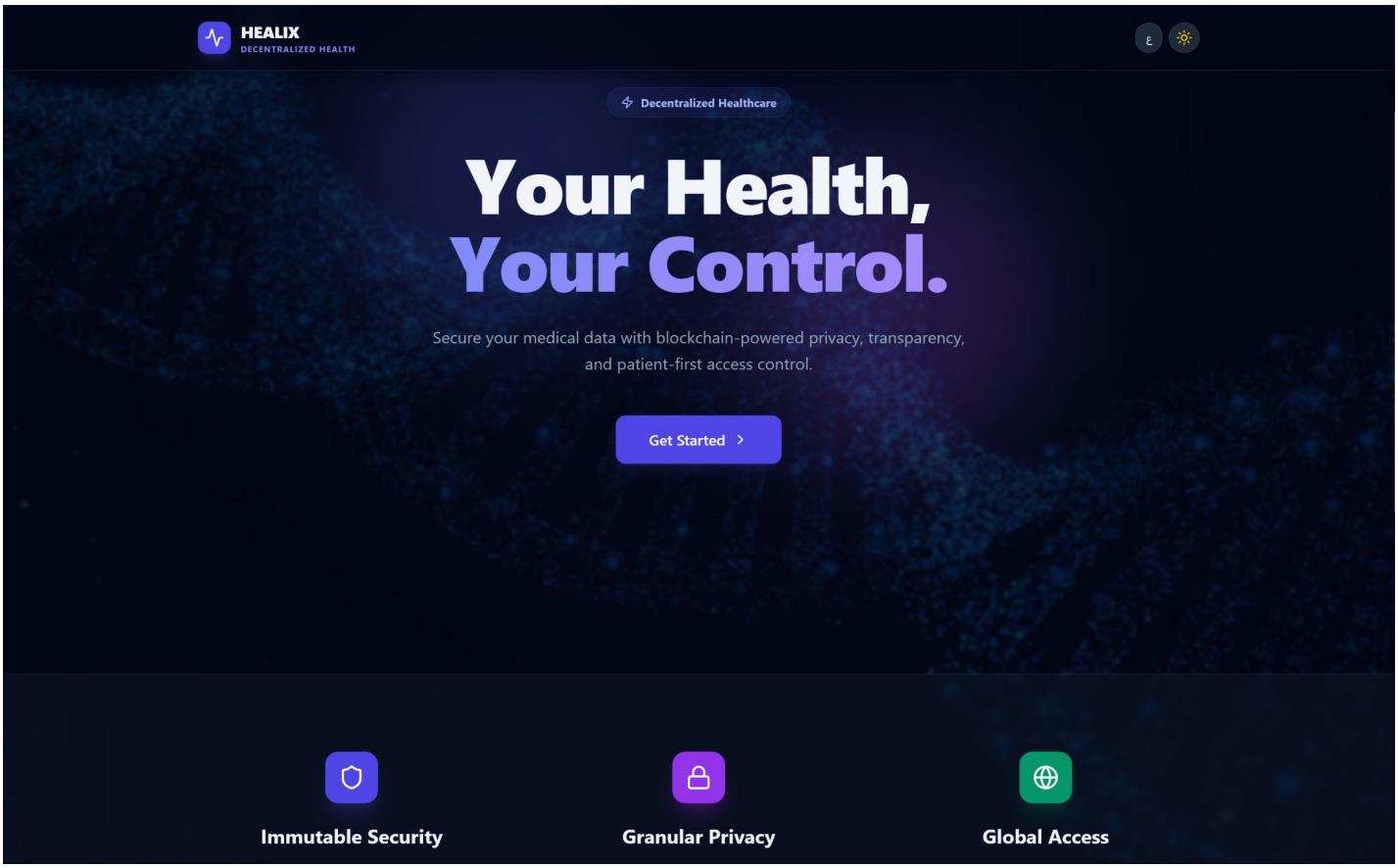

Figure 4.21: Home page of HEALIX

4.15.2 Super Admin Page

The Super Admin Dashboard is how the super admin registers new hospitals (and assigning Hospital Admins) or manages existing ones or suspend problematic ones (Decommissioned, Unavailable, Canceled...etc), as discussed in the Actors section in Chapter 3.

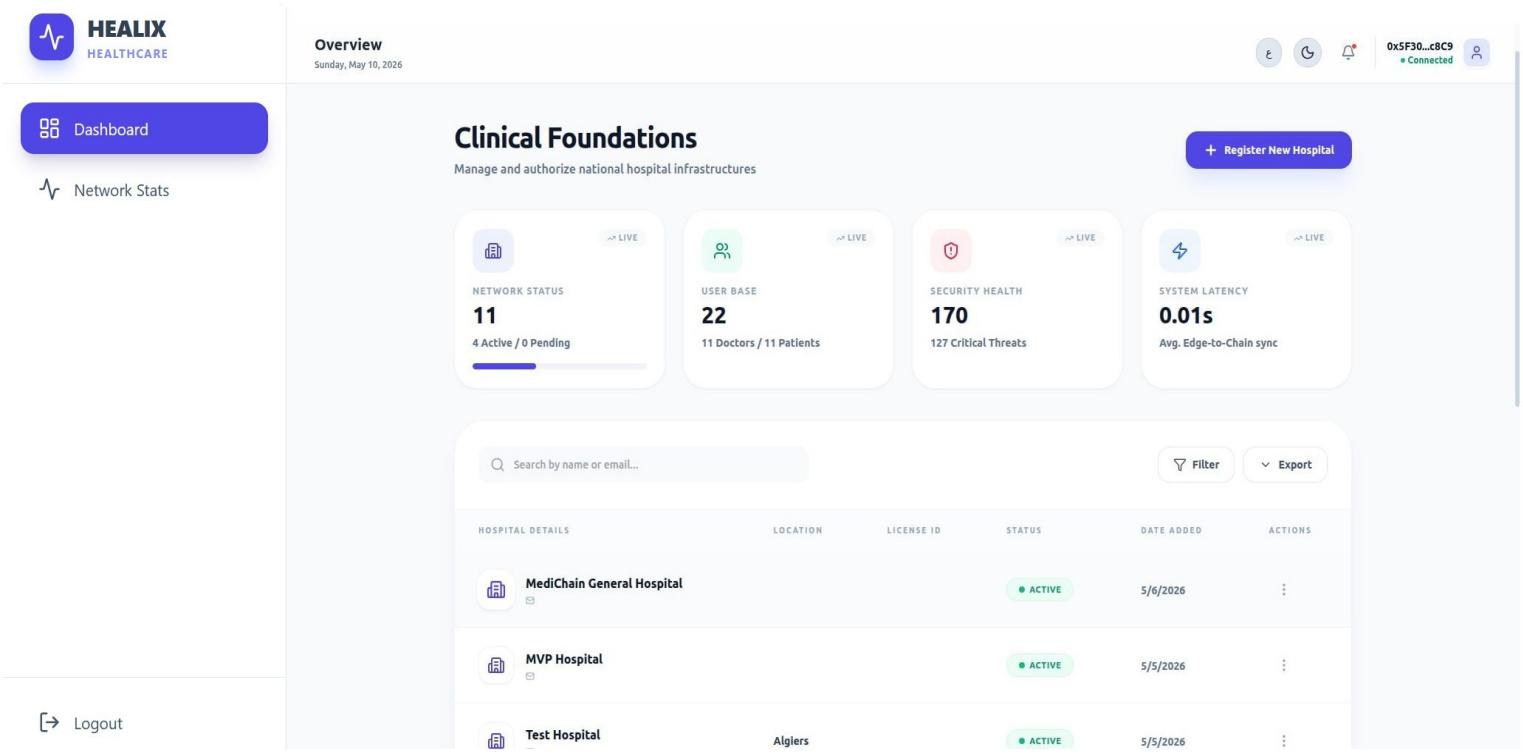

Figure 4.22: Super Admin Dashboard

4.15.3 The Hospital Admin Page

This is where the designated Hospital Admin manages their specific hospital respectively through the dashboard via Staff invitations, and Room setups by registering new rooms and the corresponding IoT devices and adding them to the system.

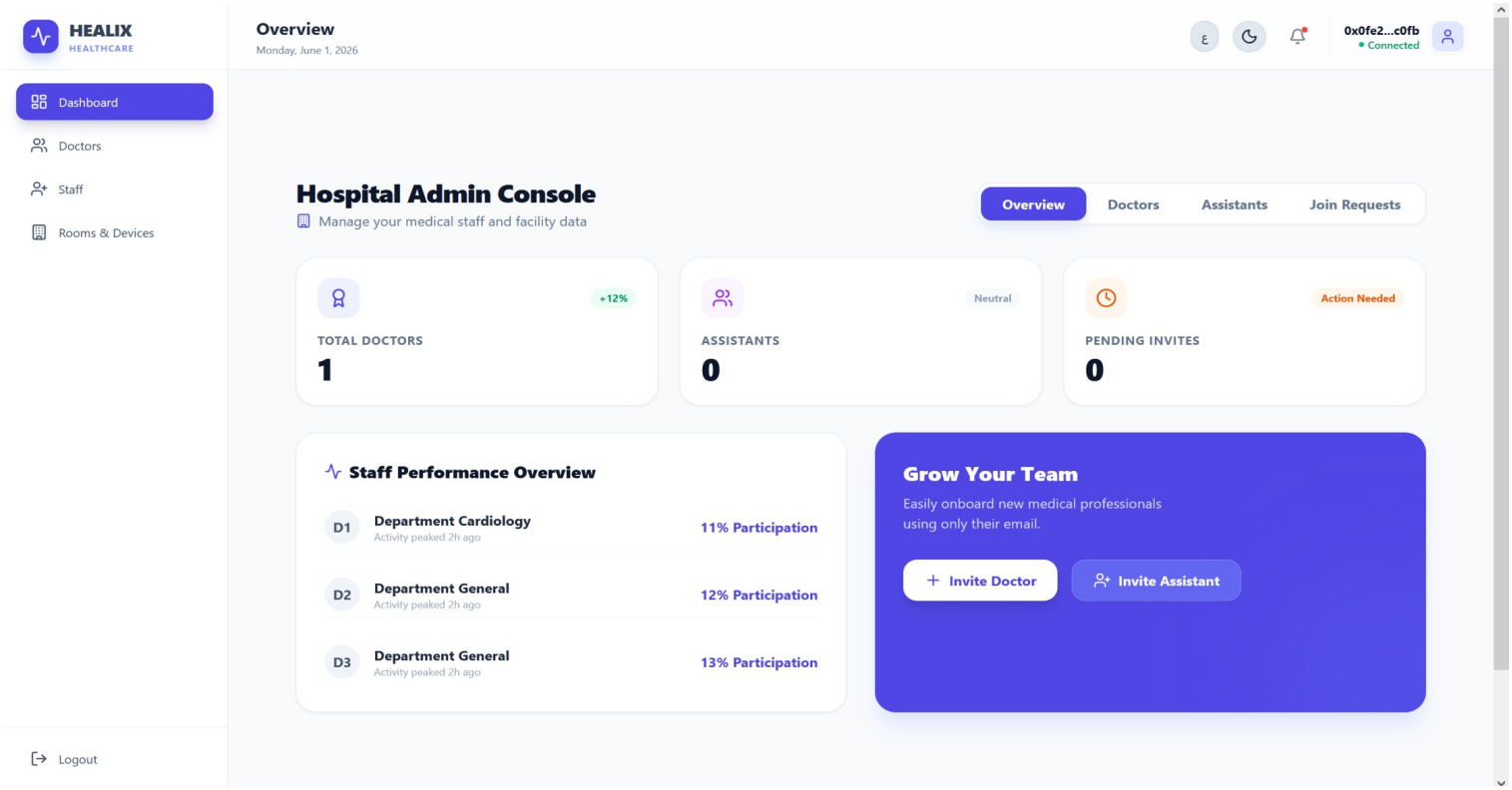

Figure 4.23: Hospital Admin Dashboard

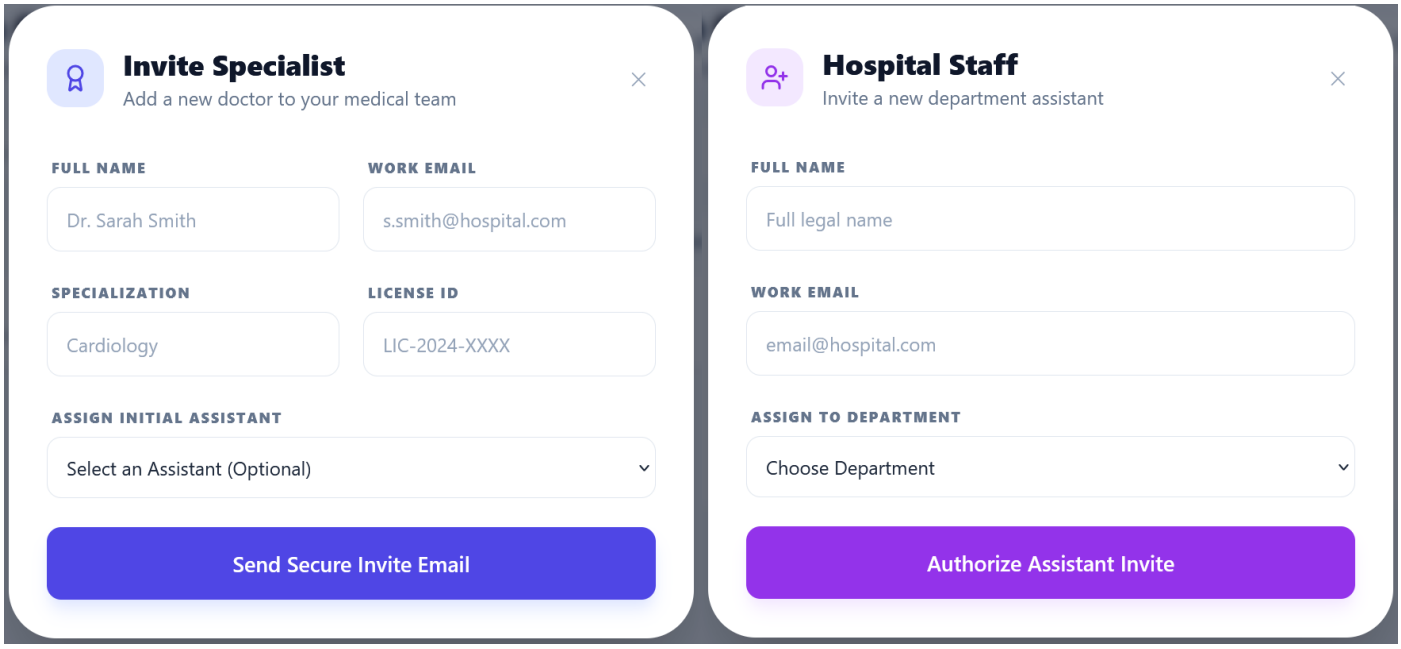

Figure 4.24: Assigning Doctors and Assistants to a specific hospital

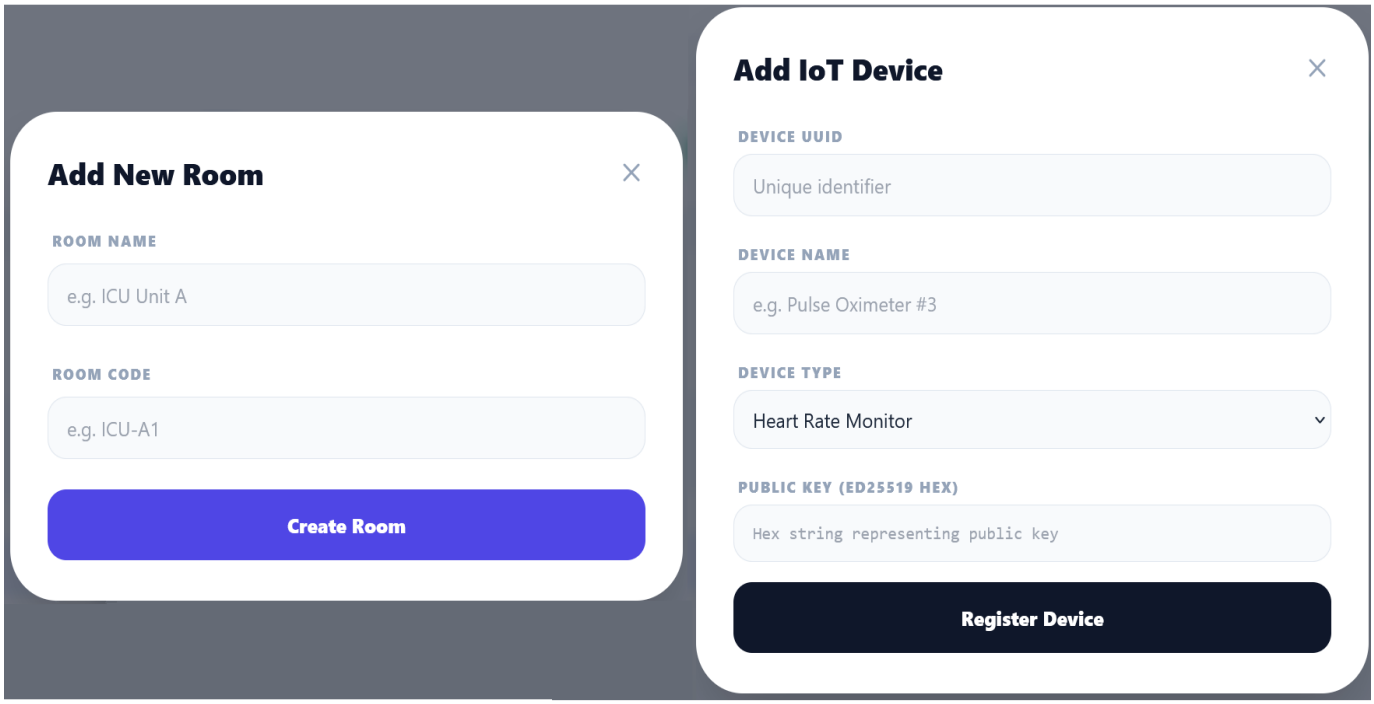

Figure 4.25: Adding new Rooms and IoT Devices

4.15.4 The Doctors Dashboard

This is where a Doctor can view their patients and their Records, Manage their own appointments and observe patient's vitals through IoT Devices data streams for any anomalies.

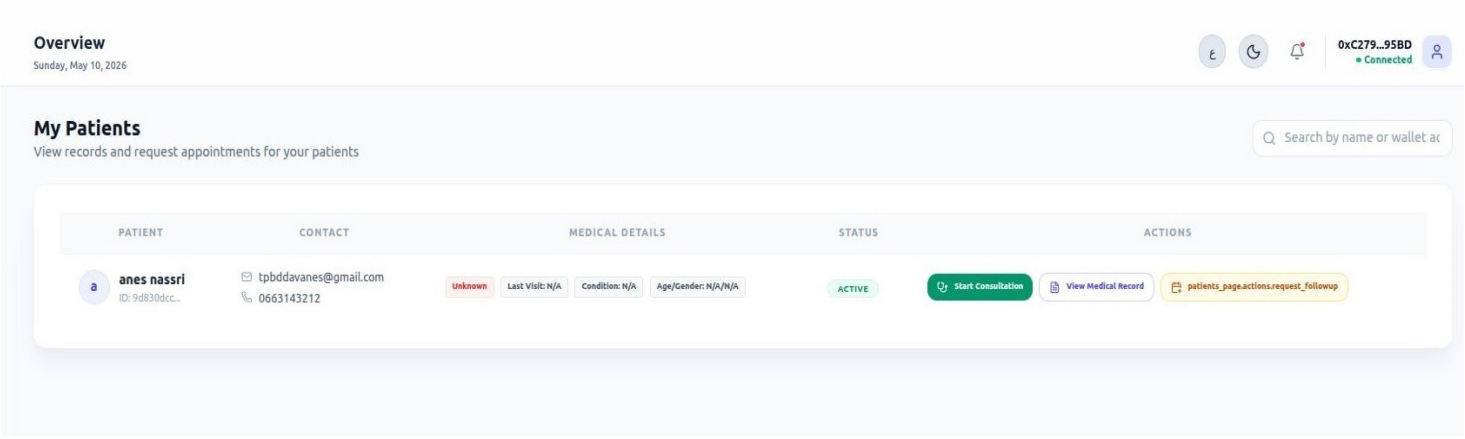

Figure 4.26: Patients List Overview

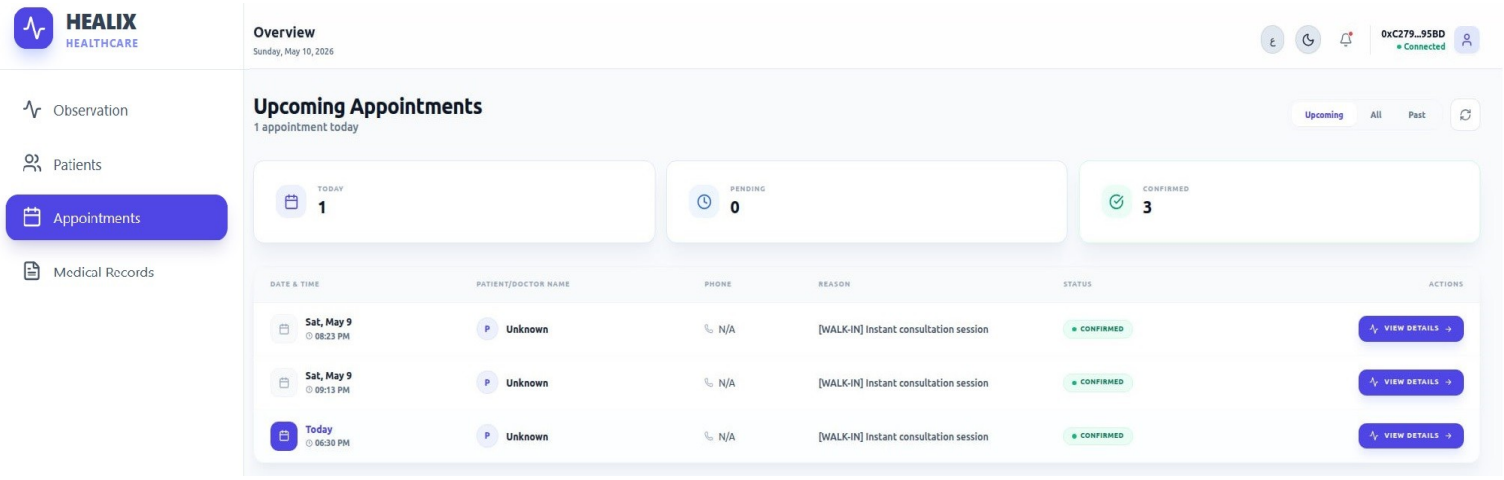

Figure 4.27: Overview of the Appointments

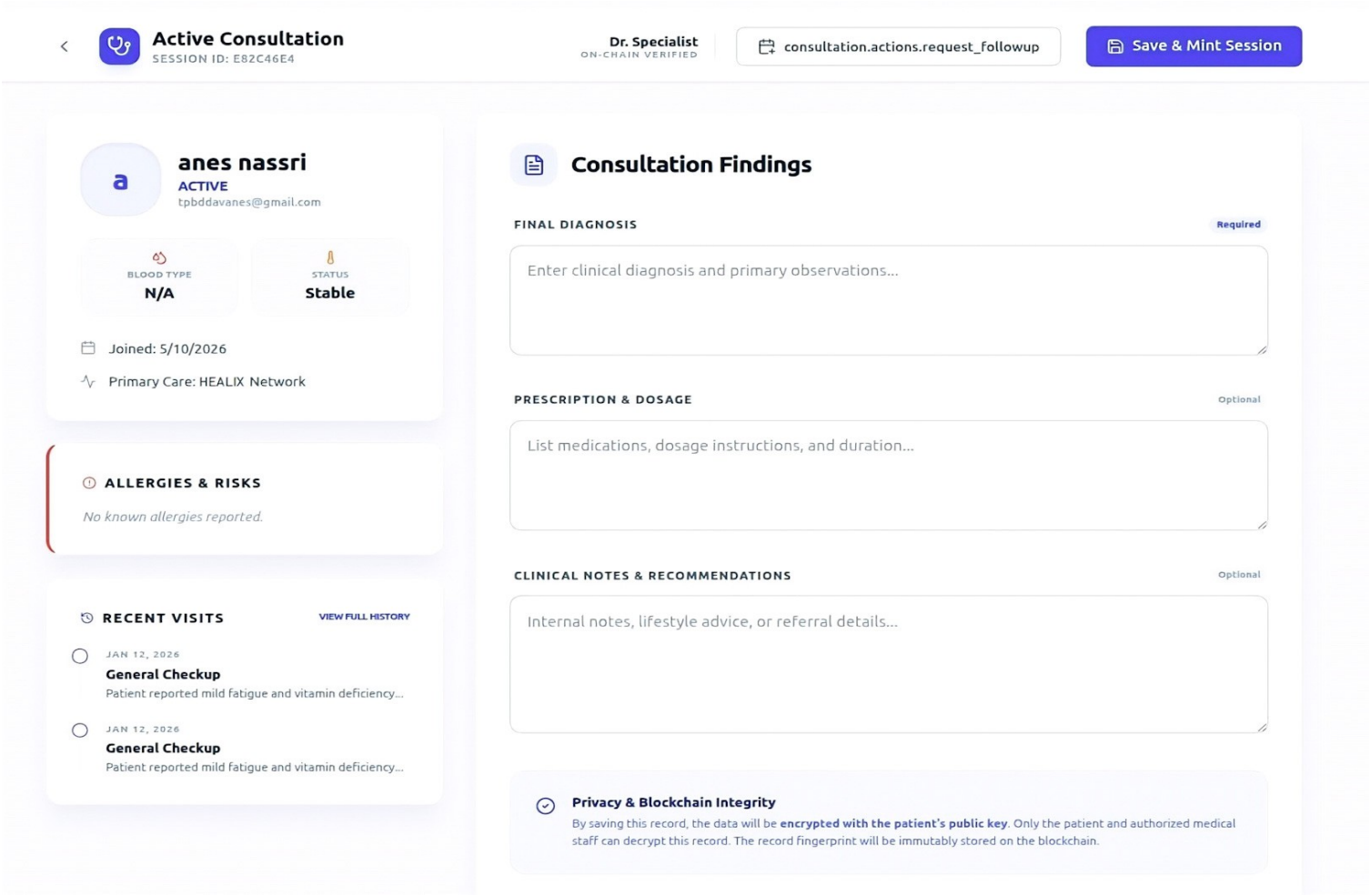

Figure 4.28: Appointment and Session Consultation

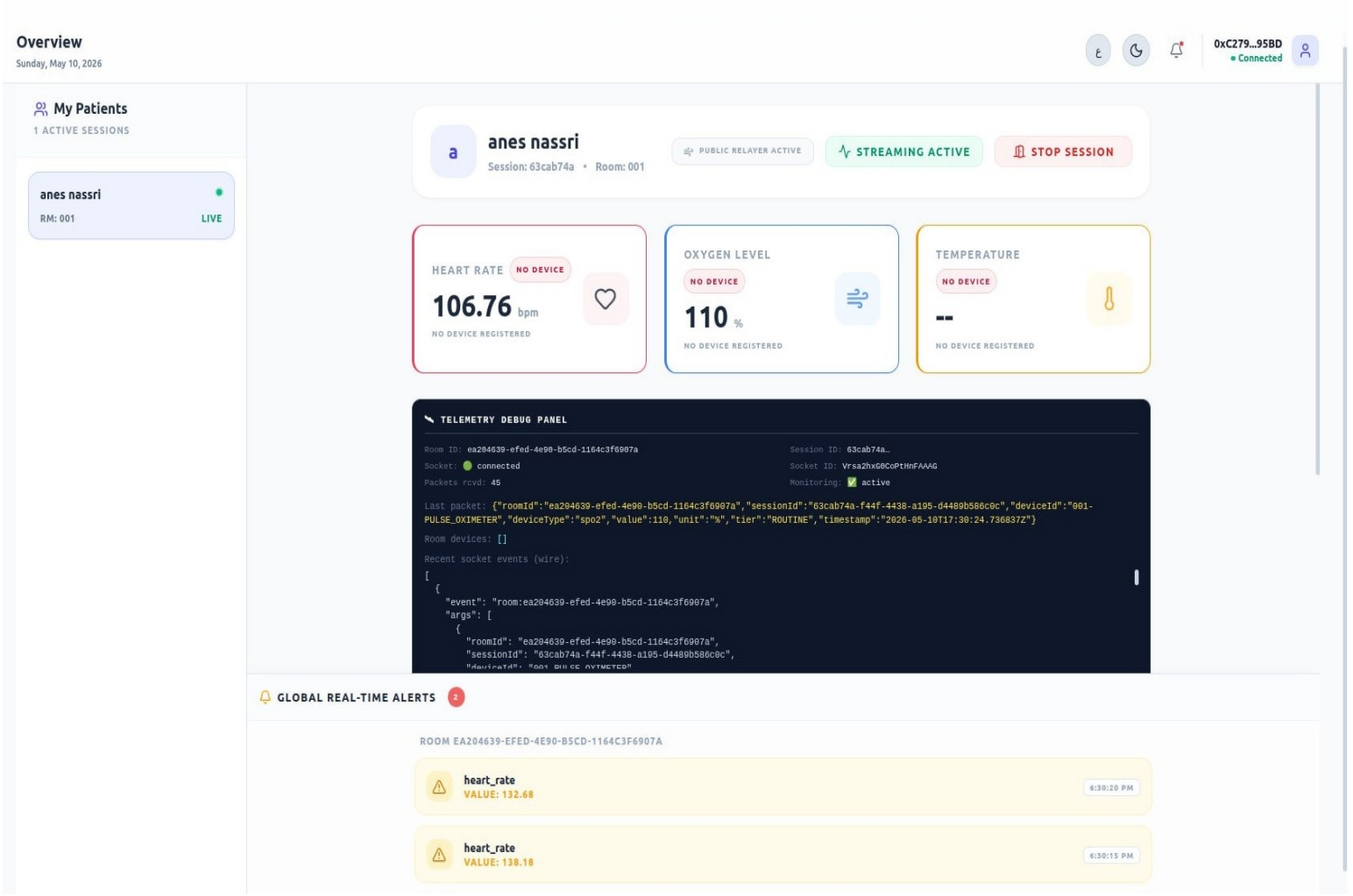

Figure 4.29: IoT Live-Feed monitoring and anomaly alerts

4.15.5 The Patients Dashboard

From the Dashboard, The patients can view their own medical records and manage access privileges, View their appointments and a summary Overview of their status and activities.

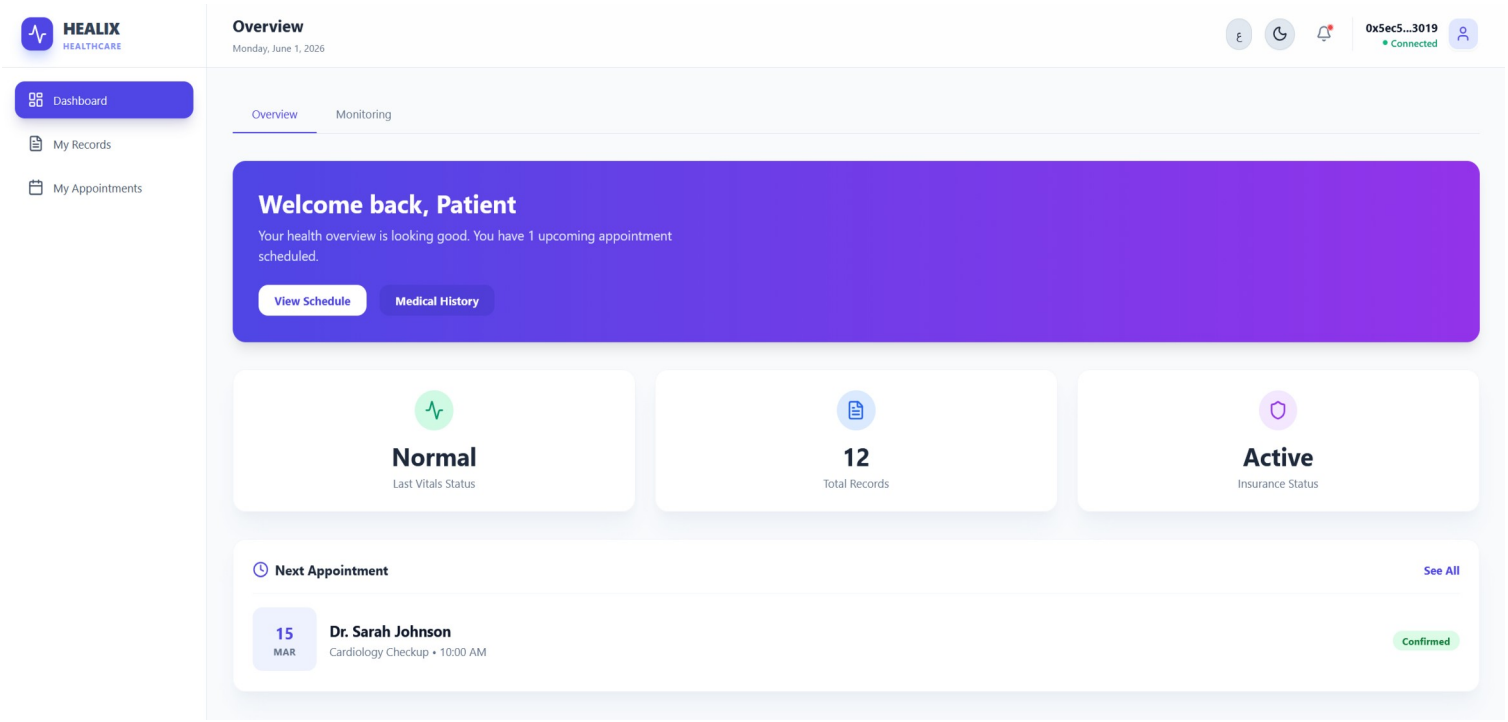

Figure 4.30: Overview of the patient's dashboard

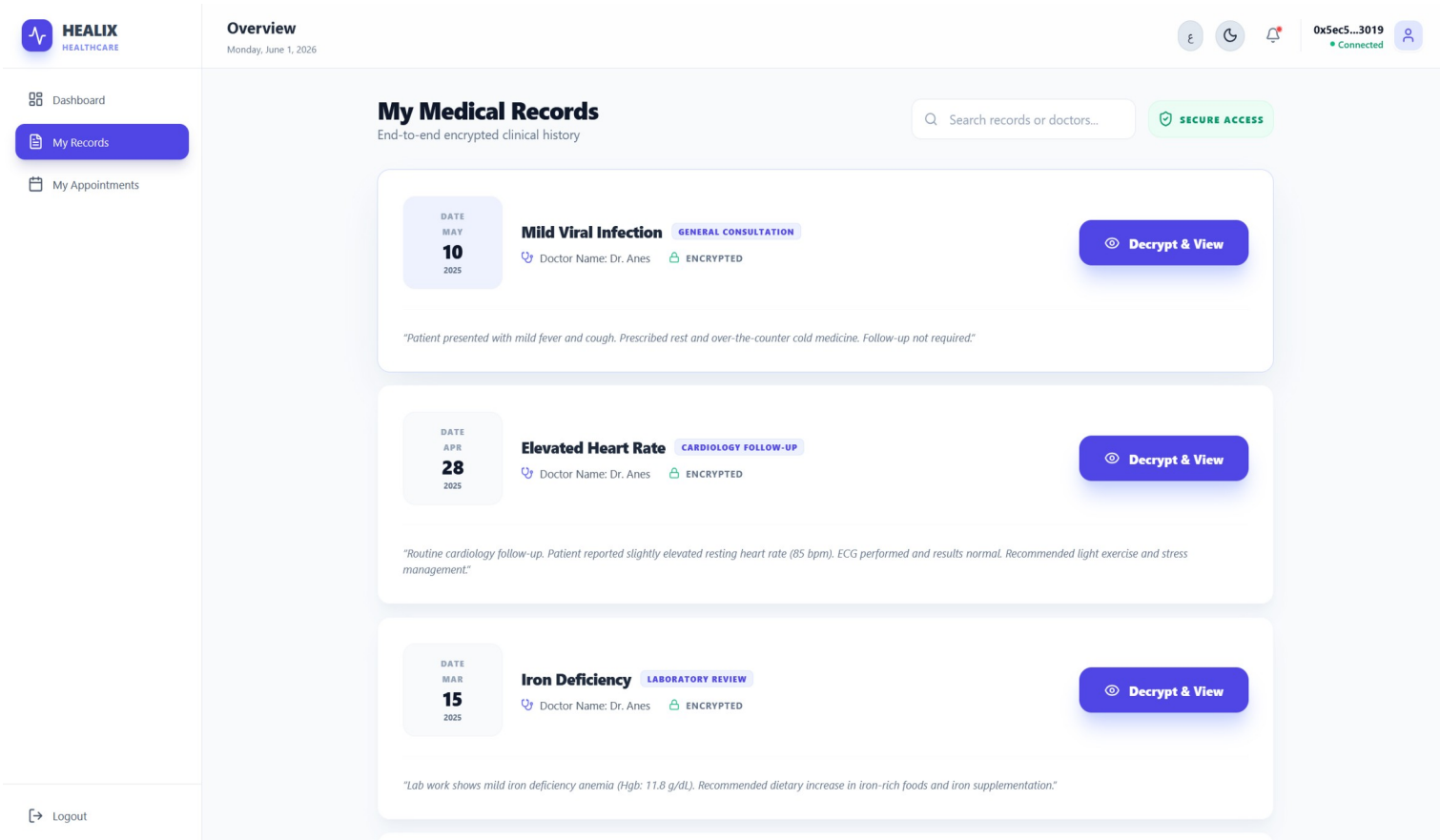

Figure 4.31: Patient's own records interface

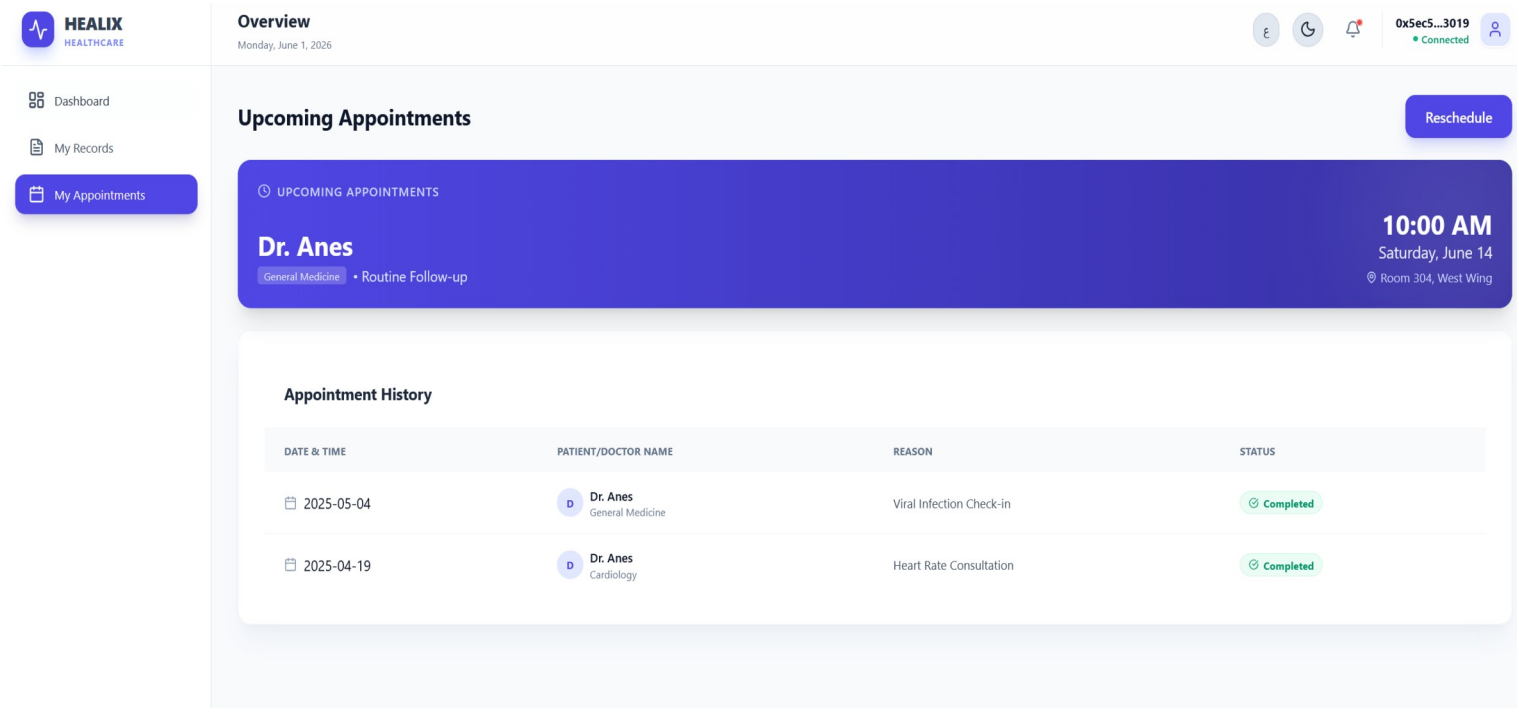

Figure 4.32: Patient's own Appointments overview

4.15.6 The Assistant's Dashboard

From the dashboard, The assistant can get an overview of their assignments as well as manage their tasks and appointments, they can also register patients and check doctor's availability using the built-in calendar manager as seen in the figures bellow.

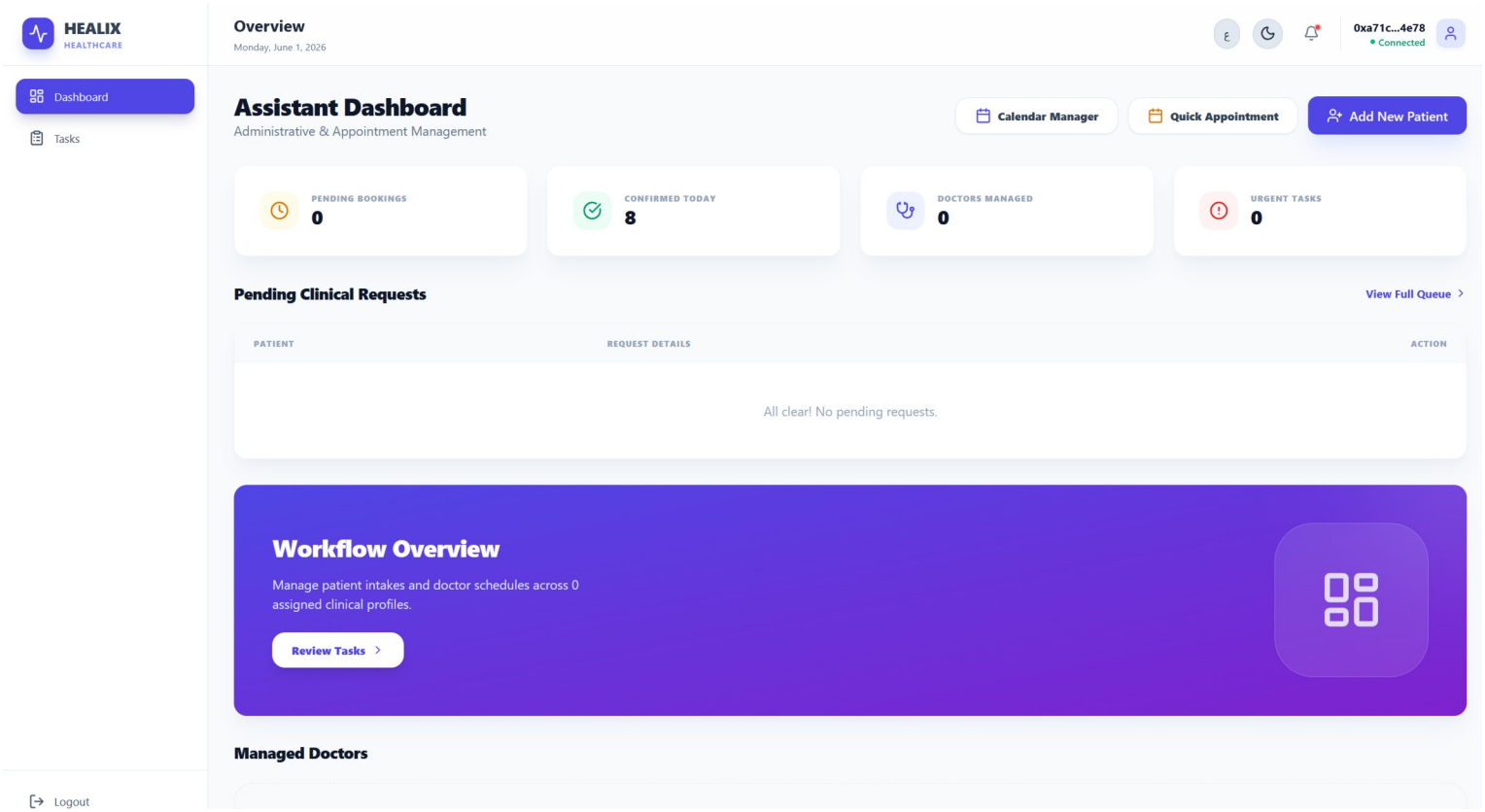

Figure 4.33: The Assistant's Dashboard Overview

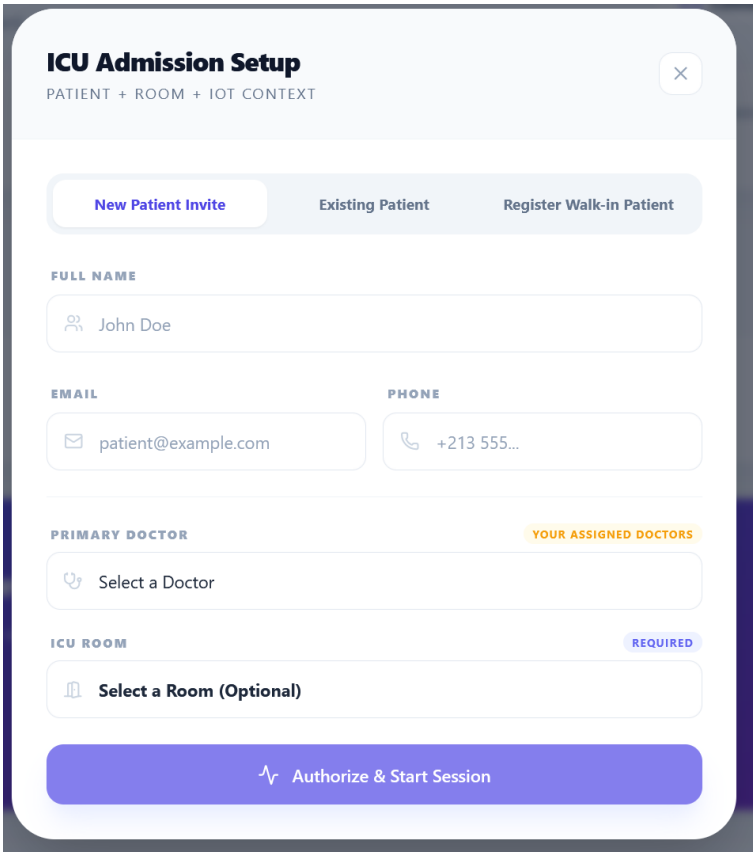

Figure 4.34: Patient Registration and Admission interface

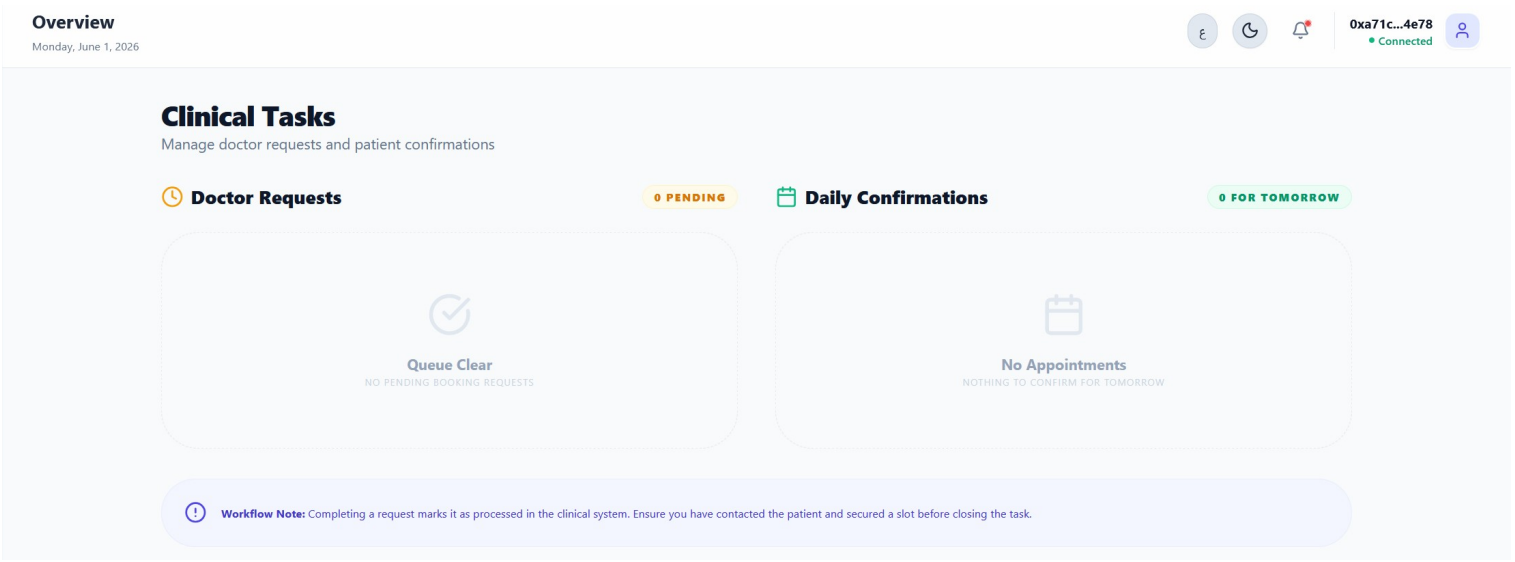

Figure 4.35: Overview of the Assistant's Tasks and appointments

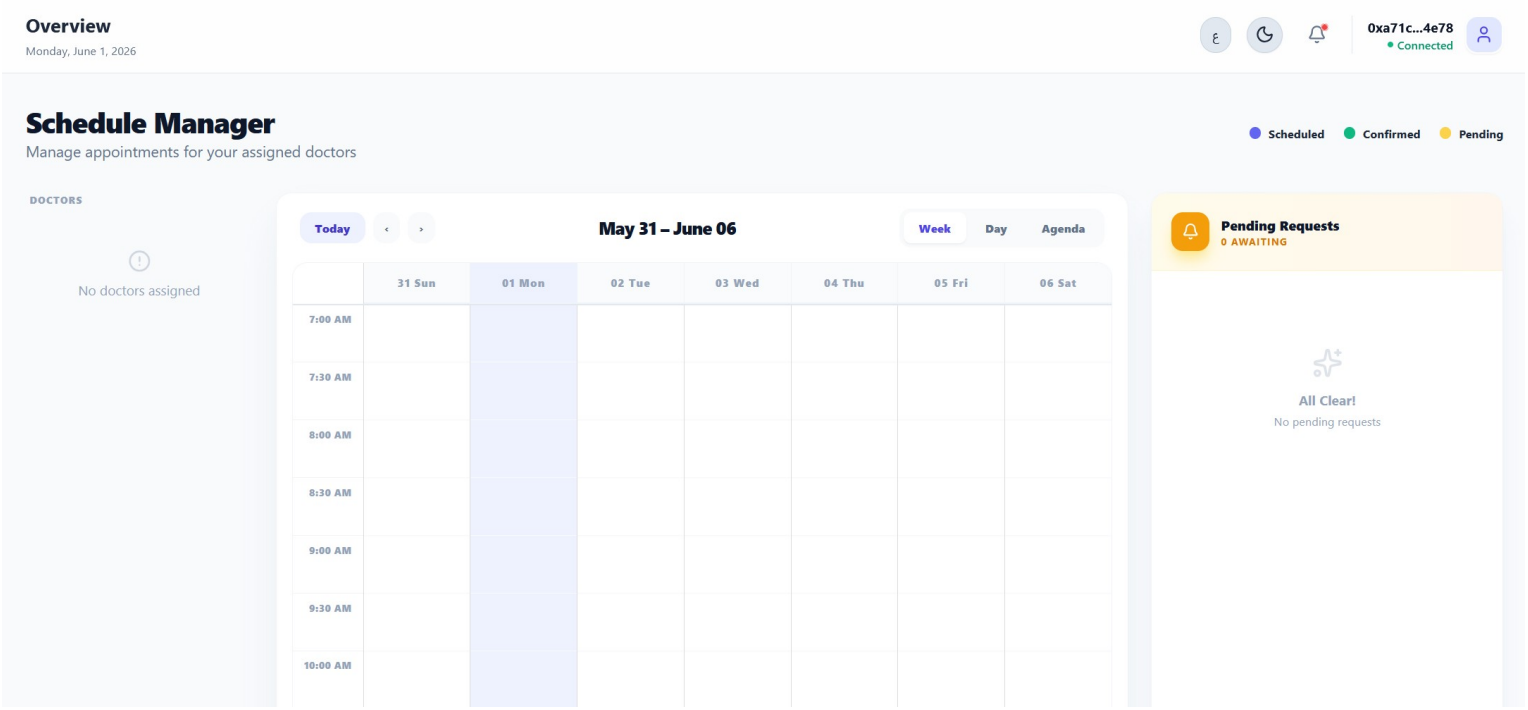

Figure 4.36: Schedule and Calendar manager for the assigned doctor's appointments

4.16 Conclusion

In summary, this chapter has detailed the comprehensive and multi-layered technology stack required to bring the HEALIX ecosystem to life. By strategically combining foundational languages like Solidity and TypeScript with advanced Web3 frameworks such as Hardhat and Ethers.js, the project successfully bridges decentralized, immutable smart contracts with modern web interfaces. The integration of high-performance backend infrastructure (Node.js, Prisma, IPFS) and a responsive, React-based frontend ensures a secure, scalable, and seamless user experience.

Furthermore, the incorporation of Python-driven IoT simulations and machine learning anomaly detection demonstrates the platform's capability to handle real-time, critical medical telemetry.

Together, these technologies (orchestrated by a robust suite of specialized smart contracts) provide the necessary architectural foundation to realize a decentralized, patient-centric healthcare model.

General Conclusion

The modern healthcare industry faces critical, systemic challenges regarding data fragmentation, patient privacy, and infrastructure vulnerability. This dissertation presented HEALIX, a decentralized healthcare platform that leverages blockchain to address data fragmentation, privacy risks, and infrastructure vulnerability. The Ethereum-based proof-of-concept combined smart contracts, IPFS off-chain storage, and strict Role-Based Access Control to give patients cryptographic sovereignty over their records. Integration of simulated IoMT devices enabled real-time monitoring and anomaly detection, demonstrating that blockchain can eliminate single points of failure, guarantee tamper-proof audit trails, and enable trustless collaboration among patients, clinicians, and administrators.

Future Work and Strategic Roadmap Scaling to enterprise healthcare demands a transition from Ethereum to a governed Hyperledger Fabric consortium, delivering higher throughput, lower latency, and granular privacy controls essential for HIPAA/GDPR compliance.

Furthermore, the strategic roadmap for HEALIX includes the integration of several advanced technical features:

- **Zero-Knowledge Proofs (ZKPs):** Implementing ZKPs will allow patients to verify specific health conditions or credentials to third parties without exposing the underlying sensitive medical data, establishing a new standard for patient privacy.
- **AI-Driven Analytics:** Expanding the current anomaly detection models into comprehensive AI-driven diagnostic analytics. This will enable predictive healthcare insights derived securely from the decentralized, anonymized data pool.
- **Real-World IoMT Integration:** Advancing from simulated IoT nodes to the deployment of physical, specialized medical sensors within clinical settings, ensuring secure, high-frequency data anchoring directly from hospital beds to the blockchain.
- **Global Interoperability Standards:** Upgrading the platform to natively support standard healthcare data formats, such as HL7 and FHIR, allowing HEALIX to seamlessly interface and integrate with existing legacy hospital infrastructure.

Ultimately, HEALIX establishes a powerful precedent for the future of decentralized healthcare. By mapping a clear trajectory from a successful Ethereum prototype toward a scalable, governed Hyperledger Fabric enterprise solution, this research lays the groundwork for a secure, patient-centric, and technologically advanced global medical ecosystem.

References

[1] Sherman, Alan T.; Javani, Farid; Zhang, Haibin; Golaszewski, Enis (January 2019). "On the Origins and Variations of Blockchain Technologies". IEEE Security & Privacy. **17** (1): 72–77. https://ieeexplore.ieee.org/document/8674176 https://chaum.com/wp-content/uploads/2022/02/Origins-of-Blockchain-08674176.pdf

[2] hn, J.; Yi, E.; Kim, M. Blockchain Consensus Mechanisms: A Bibliometric Analysis (2014–2024) Using VOSviewer and R Bibliometrix. Information 2024, 15, 644. https://doi.org/10.3390/info151006444

[3] Youness Bentayeb, Kenza Chaoui and Hassan Badir. “Integrating Blockchain and Edge Computing: A Systematic Analysis of Security, Efficiency, and Scalability”. International Journal of Advanced Computer Science and Applications (IJACSA) 16.1 (2025). http://dx.doi.org/10.14569/IJACSA.2025.0160160 https://thesai.org/Publications/ViewPaper?Volume=16&Issue=1&Code=IJACSA&SerialNo=60

[4] M. Rifat Hossain, F. A. Nirob, A. Islam, T. M. Rakin and M. Al-Amin, "A Comprehensive Analysis of Blockchain Technology and Consensus Protocols Across Multilayered Framework," in *IEEE Access*, vol. 12, pp. 63087-63129, 2024, doi: 10.1109/ACCESS.2024.3395536. https://doi.org/10.1109/ACCESS.2024.3395536

[5] lkhateeb, Y.M. (2021) Blockchain Implications in the Management of Patient Complaints in Healthcare. Journal of Information Security, 12, 212-223. https://doi.org/10.4236/jis.2021.123011

[6] Angraal, S., Krumholz, H.M. and Schulz, W.L. (2017) Blockchain Technology: Applications in Health Care. Cardiovascular Quality and Outcomes, 10, e003800. https://doi.org/10.1161/CIRCOUTCOMES.117.003800

[7] I. T. Javed, V. Lemieux and D. A. Regier, "SecureConsent: A Blockchain-Based Dynamic and Secure Consent Management for Genomic Data Sharing," *2024 International Conference on Smart Applications, Communications and Networking (SmartNets)*, Harrisonburg, VA, USA, 2024, pp. 1-7, doi: 10.1109/SmartNets61466.2024.10577693. https://ieeexplore.ieee.org/document/10577693 https://www.researchgate.net/publication/382033079_SecureConsent_A_Blockchain-Based_Dynamic_and_Secure_Consent_Management_for_Genomic_Data_Sharing

[8] Kuo, T.T., Kim, H.E. and Ohno-Machado, L. (2017) Blockchain Distributed Ledger Technologies for Biomedical and Health Care Applications. Journal of the American Medical Informatics Association, 24, 1211-1220. https://doi.org/10.1093/jamia/ocx068

[9] PWC (2018) A Prescription for Blockchain and Healthcare: Reinvent or Be Reinvented. https://www.pwc.com/mx/es/publicaciones/archivo/2018/10/20181010-pwc-mx-a-prescription-for-blockchain-and-healthcare.pdf

[10] Khalifeh Ahmad, 2020, Blockchain Technology and its Implementations in Medical and Healthcare Field, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 09, Issue 09 (September 2020), https://doi.org/10.5281/zenodo.18678396

[11] Charles W, Marler N, Long L and Manion S (2019) Blockchain Compliance by Design: Regulatory Considerations for Blockchain in Clinical Research. Front. Blockchain 2:18. doi: https://doi.org/10.3389/fbloc.2019.00018

[12] Kuo TT, Kim HE, Ohno-Machado L. Blockchain distributed ledger technologies for biomedical and health care applications. J Am Med Inform Assoc. 2017 Nov 1;24(6):1211-1220. doi: 10.1093/jamia/ocx068. PMID: 29016974; PMCID: PMC6080687. https://doi.org/10.1093/jamia/ocx068

[13] Gordon, W.J. and Catalini, C. (2018) Blockchain Technology for Healthcare: Facilitating the Transition to Patient-Driven Interoperability. Computational and Structural Biotechnology Journal, 16, 224-230.https://doi.org/10.1016/j.csbj.2018.06.003

[14] Benchoufi M, Ravaud P. Blockchain technology for improving clinical research quality. Trials. 2017 Jul 19;18(1):335. doi: 10.1186/s13063-017-2035-z. PMID: 28724395; PMCID: PMC5517794. https://doi.org/10.1186/s13063-017-2035-z

[15] Retrieved from BlockRx: The Pharmaceutical Blockchain of Value. https://www:blockrx:com/white-paper/ https://isolve.io/our-solutions/adlt/ https://isolve.io/blockrx/#

[16] Kevin. A.C.; Breeden, E.A.; Davidson, C.; Mackey, T.K. Leveraging Blockchain Technology to Enhance Supply Chain Management in Healthcare: An exploration of challenges and opportunities in the health supply chain. Blockchain Healthcare. https://doi.org/10.30953/bhty.v1.20

[17] J. Höller, V. Tsiatsis, C. Mulligan, S. Karnouskos, S. Avesand, and D. Boyle, From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence. Amsterdam, The Netherlands: Elsevier, 2014. https://www.mforum.ru/arc/iot-book_compressed_MForum.pdf

[18] S. M. R. Islam, D. Kwak, MD. H. Kabir, M. Hossain, and K.-S. Kwak, ‘‘The Internet of Things for health care: A comprehensive survey,’’ IEEE Access, vol. 3, pp. 678–708, 2015. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7113786

[19] Z. Pang, ‘‘Technologies and architectures of the Internet-of-Things (IoT) for health and well-being,’’ M.S. thesis, Dept. Electron. Comput. Syst., KTH-Roy. Inst. Technol., Stockholm, Sweden, Jan. 2013. https://web.archive.org/web/20230121033928/https://www.diva-portal.org/smash/get/diva2:621384/FULLTEXT01.pdf

[20] Ivan D. Moving Toward a Blockchain-based Method for the Secure Storage of Patient Records. ONC/NIST Use of Blockchain for Healthcare and Research Workshop. Gaithersburg, Maryland, United States: ONC/NIST; 2016.https://www.truevaluemetrics.org/DBpdfs/Technology/Blockchain/9-16-drew_ivan_20160804_blockchain_for_healthcare_final.pdf

[21] Yue X, Wang H, Jin D, Li M, Jiang W. Healthcare Data Gateways: Found Healthcare Intelligence on Blockchain with Novel Privacy Risk Control. J Med Syst. 2016 Oct;40(10):218. doi: 10.1007/s10916-016-0574-6. Epub 2016 Aug 26. PMID: 27565509. https://doi.org/10.1007/s10916-016-0574-6

[22] Blough D, Ahamad M, Liu L, Chopra P. MedVault: Ensuring security and privacy for electronic medical records. NSF CyberTrust Principal Journal of the American Medical Informatics Association, 2017, Vol. 24, No. 6 1219 https://www.academia.edu/96894823/CT_T_MedVault_ensuring_security_and_privacy_for_electronic_medical_records

[23] Mark Phillips, Ph.D. Departments of Radiation Oncology, Neurosurgery, Biomedical Informatics University of Washington, Seattle, WA “The Synergy of Ontologies and HL7-FHIR” https://amos3.aapm.org/abstracts/pdf/155-52183-1531640-157500.pdf

[24] Nikolas Sargeant, “A Comprehensive Guide to Layer 2 Solutions: Scaling Ethereum” https://www.cryptowisser.com/guides/layer-2-solutions/

[25] Admond Lee, Data Science Engineer at DataSource.ai “Volume, velocity, and variety: Understanding the three V's of big data” https://www.datasource.ai/en/data-science-articles/volume-velocity-and-variety-understanding-the-three-v-s-of-big-data

[26] A. Azaria, A. Ekblaw, T. Vieira and A. Lippman, "MedRec: Using Blockchain for Medical Data Access and Permission Management," *2016 2nd International Conference on Open and Big Data (OBD)*, Vienna, Austria, 2016, pp. 25-30, doi: 10.1109/OBD.2016.11. https://ieeexplore.ieee.org/document/7573685

https://people.cs.pitt.edu/~babay/courses/cs3551/papers/MedRec.pdf

[27] Z. Yang, K. Yang, L. Lei, K. Zheng and V. C. M. Leung, "Blockchain-Based Decentralized Trust Management in Vehicular Networks," in *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1495-1505, April 2019, doi: 10.1109/JIOT.2018.2836144. https://ieeexplore.ieee.org/document/8358773

https://www.researchgate.net/publication/325132959_Blockchain-Based_Decentralized_Trust_Management_in_Vehicular_Networks

[28] Chentouf, F.Z.; Hassoun, M.E.A.; Bouchkaren, S. Electronic Health Record Systems Based on Blockchain: A Comprehensive Survey. *Appl. Sci.* 2026, *16*, 3768. https://doi.org/10.3390/app16083768 https://www.mdpi.com/2076-3417/16/8/3768

[29] Minango, J.; Carvajal Mora, H.; Zambrano, M.; Orozco Garzón, N.; Pérez, F. Distributed Ledger Technology in Healthcare: Enhancing Governance and Performance in a Decentralized Ecosystem. Technologies 2025, 13, 58. https://doi.org/10.3390/technologies13020058

[30] Jianqi Mu, (16 October 2024) CONF Blockchain Technology-Comprehensive Analysis, Applications, and Emerging Challenges, Proceedings of the 2024 2nd International Conference on Image, Algorithms and Artificial Intelligence (ICIAAI 2024), Atlantis Press, 253 263 2352-538X https://doi.org/10.2991/978-94-6463-540-9_27

https://www.atlantis-press.com/proceedings/iciaai-24/126004125 https://www.atlantis-press.com/article/126004125.pdf

[31] Dvorchuk D., Shpinareva I. Analysis of Blockchain-Technology. *Cybernetics and Computer Technologies*. 2025. **2**. P. 77–87.https://doi.org/10.34229/2707-451X.25.2.7

https://cctech.org.ua/13-vertikalnoe-menyu-en/735-abstract-25-2-7-arte

[32] Duely C, Gambacorta L, Garratt R, Wilkens P.K (7 September 2023) BIS Bulletin No 76, The oracle problem and the future of DeFi https://www.bis.org/publ/bisbull76.pdf

[33] Osama, M.; Ateya, A.A.; Sayed, M.S.; Hammad, M.; Pławiak, P.; Abd El-Latif, A.A.; Elsayed, R.A. Internet of Medical Things and Healthcare 4.0: Trends, Requirements, Challenges, and Research Directions. Sensors 2023, 23, 7435. https://doi.org/10.3390/s23177435

[34] Cocco, L.; Tonelli, R. A Self-Sovereign Identity–Blockchain-Based Model Proposal for Deep Digital Transformation in the Healthcare Sector. Future Internet 2024, 16, 473. https://doi.org/10.3390/fi16120473

[35] Art. 17 GDPR Right to erasure ('right to be forgotten') https://gdpr-info.eu/art-17-gdpr/

[36] Tseng, L.-M.; Chen, P.-F.; Wen, C.-Y. Design of Edge-IoMT Network Architecture with Weight Based Scheduling. Sensors 2023, 23, 8553. https://doi.org/10.3390/s23208553 https://www.mdpi.com/1424-8220/23/20/8553

[37] Y. Wu and L. Kong, A3Verkle Trie: A World State Tree Organization Model Oriented to Active Accounts, Blockchain: Research and Applications,100377, doi: https://doi.org/10.1016/j.bcra.2025.100377

[38] Gobika S. and Vaishnavi N., "Blockchain Based Identity Management System", *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol*, vol. 11, no. 2, pp. 1413–1420, Mar. 2025, doi: 10.32628/CSEIT25112471. https://www.researchgate.net/publication/389967371_Blockchain_Based_Identity_Management_System

[39] Mallick, S.R., Lenka, R.K., Tripathy, P.K. *et al.* A Lightweight, Secure, and Scalable Blockchain-Fog-IoMT Healthcare Framework with IPFS Data Storage for Healthcare 4.0. *SN COMPUT. SCI.* **5**, 198 (2024). https://doi.org/10.1007/s42979-023-02511-8

[40] Jabbar, S., Lloyd, H., Hammoudeh, M. *et al.* Blockchain-enabled supply chain: analysis, challenges, and future directions. *Multimedia Systems* 27, 787–806 (2021). https://doi.org/10.1007/s00530-020-00687-0

[41] T. Hossain, S. Hassan, F. H. Bappy, M. N. Yanhaona, T. S. Zaman and T. Islam, "Bridging Immutability with Flexibility: A Scheme for Secure and Efficient Smart Contract Upgrades," *2025 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, Pisa, Italy, 2025, pp. 1-5, doi: 10.1109/ICBC64466.2025.11114684 https://arxiv.org/pdf/2504.09652

https://ieeexplore.ieee.org/document/11114684

[42] Taherdoost, H. Smart Contracts in Blockchain Technology: A Critical Review. Information 2023, 14, 117. https://doi.org/10.3390/info14020117

[43] Tamara Brandst¨atter, Stefan Schulte, J¨urgen Cito, Michael Borkowski, TU Wien, Austria, Institute of Flight Guidance, German Aerospace Center (DLR), Braunschweig, Germany, “Characterizing Efficiency Optimizations in Solidity Smart Contracts” https://dsg.tuwien.ac.at/team/sschulte/paper/bc2020.pdf

[44] Avik Banerjee, Michael Sober, and Stefan Schulte. 2025. Towards Solidity Smart Contract Efficiency Optimization through Code Mining. In Proceedings of the 40th ACM/SIGAPP Symposium on Applied Computing (SAC '25). Association for Computing Machinery, New York, NY, USA, 348–357. https://doi.org/10.1145/3672608.3707768

[45] Z. Zheng, S. Xie, H. Dai, X. Chen and H. Wang, "An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends," *2017 IEEE International Congress on Big Data (BigData Congress)*, Honolulu, HI, USA, 2017, pp. 557-564, doi:10.1109/BigDataCongress.2017.85. https://ieeexplore.ieee.org/document/8029379

[46] Zhang Y, Ma Z and Meng J (2025) Auditing in the blockchain: a literature review. Front. Blockchain 8:1549729. doi: 10.3389/fbloc.2025.1549729
https://www.frontiersin.org/journals/blockchain/articles/10.3389/fbloc.2025.1549729/full

[47] Ashar Ahmad, Muhammad Saad, Mostafa Bassiouni, and Aziz Mohaisen. 2018. Towards Blockchain-Driven, Secure and Transparent Audit Logs. In Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous '18). Association for Computing Machinery, New York, NY, USA, 443–448. https://doi.org/10.1145/3286978.3286985

[48] Maxwell Nana Ameyaw, CPA, KPMG, USA, Courage Idemudia, Independent Researcher, London, ON, Canada. and Toluwalase Vanessa Iyelolu, Financial analyst, Texas USA. “The role of blockchain in auditing processes: A review and future perspectives”

International Journal of Scientific Research Updates, 2024, 08(01), 037–053. Publication history: Received on 22 May 2024; revised on 28 June 2024; accepted on 01 July 2024

Article DOI: https://doi.org/10.53430/ijsru.2024.8.1.0045
https://orionjournals.com/ijsru/sites/default/files/IJSRU-2024-0045.pdf

[49] Md Shariful Islam and M. Sohel Rahman, Department of CSE, BUET, ECE Building, West Palashi, Dhaka-1205. “LogStamping: A blockchain-based log auditing approach for large-scale systems” https://arxiv.org/pdf/2505.17236

[50] Bathula, A., Gupta, S.K., Merugu, S. *et al.* Blockchain, artificial intelligence, and healthcare: the tripod of future—a narrative review. *Artif Intell Rev* 57, 238 (2024). https://doi.org/10.1007/s10462-024-10873-5

[51] Leon Witt Dept. of Comp. Sci. & Tech. Tsinghua University & Fraunhofer HHI. Armando Teles Fortes Dept. of Comp. Sci. & Tech. Tsinghua University, Kentaroh Toyoda, Member, IEEE Inst. of High Performance Computing A*STAR, Wojciech Samek, Member, IEEE Dept. of Elec. Eng. & Comp. Sci. TU Berlin & Fraunhofer HHI, Dan Li, Member, IEEE Dept. of Comp. Sci. & Tech. Tsinghua University. *"Blockchain and Artificial Intelligence: Synergies and Conflicts"* https://arxiv.org/pdf/2405.13462

[52] S. R. Ali, M. I. Memon, J. Hussain, M. Mohatram and M. Faraz, "Digital Learning Platforms and Blockchain for Entrepreneurial Skill Development: Bridging Gaps in Education and Practice," in *IEEE Access*, vol. 13, pp. 180877-180890, 2025, doi: 10.1109/ACCESS.2025.3622194 https://ieeexplore.ieee.org/document/11205345

[53] M. Boughdiri, T. Abdellatif and C. Ghedira Guegan, "A Systematic Literature Review on Blockchain Storage Scalability," in *IEEE Access*, vol. 13, pp. 102194-102219, 2025, doi: 10.1109/ACCESS.2025.3578451. https://ieeexplore.ieee.org/document/11029213

[54] Mermaid Diagram and Visualization Tool https://mermaid.js.org

[55] PlantUML Modeling Tool https://plantuml.com/

[56] Photoshop Express Image Editing https://helpx.adobe.com/photoshop-express/using/photoshop-express-overview.html

[57] GIMP https://www.gimp.org/downloads/

[58] https://www.typescriptlang.org/docs/

[59] https://docs.soliditylang.org/en/v0.8.24/

[60] https://www.python.org/doc/

[61] https://nodejs.org/docs/latest/api/

[62] https://hardhat.org/docs/getting-started

[63] https://docs.etherscan.io/introduction

[64] https://docs.ethers.org/v5/

[65] https://web3py.readthedocs.io/en/stable/

[66] https://www.openzeppelin.com/solidity-contracts

[67] https://web3auth.io/docs

[68] https://docs.nestjs.com/

[69] https://www.prisma.io/docs

[70] https://www.postgresql.org/docs/

[71] https://docs.ipfs.tech/

[72] https://docs.pinata.cloud/quickstart

[73] https://socket.io/docs/v4/

[74] https://mqtt.org/getting-started/

[75] https://docs.bullmq.io/

[76] https://redis.io/docs/latest/

[77] https://www.passportjs.org/packages/passport-jwt/

[78] https://www.npmjs.com/package/crypto-js

[79] https://www.npmjs.com/package/eth-crypto

[80] https://react.dev/learn/thinking-in-react

[81] https://vite.dev/guide/

[82] https://tailwindcss.com/docs/installation/using-vite

[83] https://recharts.github.io/

[84] https://react-hook-form.com/

[85] https://www.i18next.com/

[86] https://scikit-learn.org/stable/getting_started.html

[87] https://numpy.org/doc/stable/

[88] https://eclipse.dev/paho/

[89] https://docs.docker.com/compose/

[90] https://eslint.org/docs/latest/

[91] https://prettier.io/docs/

[92] https://jestjs.io/

[93] https://www.npmjs.com/package/vite-plugin-node-polyfills