RV COLLEGE OF ENGINEERING®
BENGALURU–560059

(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Integrated Hospital Resource Management System (IHORMS)

Mini Project Report

Submitted by

CHIRAG H **1RV24AI402**

KUSHAL S GOWDA **1RV23AI048**

in partial fulfilment for the requirement of 5th Semester

DBMS Laboratory (CD252IA)

Under the Guidance of

Prof. Harshitha V
Assistant Professor
Department of AIML
RV College of Engineering
Bengaluru–59

Dr. Vijayalakshmi M.N.
Associate Professor
Department of AIML
RV College of Engineering
Bengaluru–59

Academic Year

2025–2026

RV College of Engineering®
Mysore Road, RV Vidyaniketan Post,Bengaluru - 560059, Karnataka, India

CERTIFICATE

Certified that the project work titled **Integrated Hospital Resource Management System (IHORMS)** is carried out by **CHIRAG H (1RV24AI402)** and **KUSHAL S GOWDA (1RV23AI048)**, who are bonafide students of RV College of Engineering, Bengaluru, in partial fulfillment of the curriculum requirement of 5th Semester Database Management Systems Laboratory Mini Project during the academic year 2025-2026.

It is certified that all corrections/suggestions indicated for the Internal Assessment have been incorporated in the report. The report has been approved as it satisfies the academic requirements in all respect laboratory mini-project work prescribed by the institution.

Signature of Faculty **Signature of Head of the Department**

External Viva

Name of Examiners	**Signature with Date**
1.	
2.	

Contents

Chapter 1

INTRODUCTION

The healthcare industry faces unprecedented challenges in managing patient data, coordinating care across multiple departments, and ensuring seamless communication between medical professionals. Traditional hospital management systems operate in isolated silos, creating fragmentation in patient care, inefficient resource allocation, and compromised data security. Small and medium-sized hospital chains struggle particularly with multi-branch coordination, leading to duplicate patient records, inconsistent treatment protocols, and inability to leverage organizational insights.

The Integrated Hospital Resource Management System (IHORMS) addresses these critical gaps through a unified, multi-tenant architecture designed specifically for hospital organizations operating across multiple physical locations. Unlike conventional single-location systems, IHORMS provides centralized oversight while maintaining branch-level operational autonomy, enabling seamless patient transfers, unified medical histories, and organization-wide analytics.

IHORMS revolutionizes healthcare delivery through intelligent automation and role-based workflows. At its core lies a sophisticated machine learning engine that analyzes patient symptoms to recommend appropriate specialists, reducing misdiagnosis and improving triage efficiency. Real-time telemetry monitoring with automated critical alerts ensures immediate response to patient deterioration, while comprehensive billing integration with insurance workflows streamlines financial operations.

The system implements a hierarchical role structure spanning from platform super administrators managing multiple hospital organizations, to organization administrators overseeing multi-branch networks, down to branch-level staff including doctors, nurses, receptionists, pharmacists, and patients themselves. Each role receives a customized interface with precisely scoped permissions, ensuring data privacy compliance while maximizing operational efficiency.

1.1 Objective

1.1.1 Primary Objectives

Multi-Tenant Hospital Management Platform: To design and implement a multi-tenant hospital management platform supporting multiple organizations with isolated data boundaries, enabling Software-as-a-Service deployment while ensuring strict HIPAA compliance and patient data privacy.

Intelligent Doctor Recommendation System: To develop an intelligent doctor recommendation system using machine learning algorithms that analyze patient symptoms and medical history to suggest appropriate specialists, reducing triage errors and improving treatment outcomes.

Comprehensive Clinical Workflow Management: To create a comprehensive clinical workflow management system encompassing appointment scheduling, medical documentation, admission processes, discharge approvals, and prescription management with full audit trails for regulatory compliance.

Real-Time Patient Telemetry Monitoring: To implement real-time patient telemetry monitoring with automated critical alert generation when vital signs exceed safe thresholds, enabling immediate medical intervention for deteriorating patients.

Advanced Billing and Insurance Integration: To integrate advanced billing and insurance claim processing with itemized billing, discount management, payment tracking, and automated insurance submission workflows.

1.1.2 Secondary Objectives

Centralized Organization-Level Analytics: To establish centralized organization-level analytics providing hospital administrators with revenue trends, appointment volumes, staff utilization metrics, and predictive insights for strategic planning.

Comprehensive Pharmacy Inventory Management: To implement comprehensive pharmacy inventory management with stock tracking, expiry monitoring, low-stock alerts, and predictive demand forecasting using time-series analysis.

Patient Self-Service Portals: To develop patient self-service portals enabling appointment booking, medical history viewing, bill payment, and insurance claim submission without requiring staff intervention.

Robust Data Security and Access Control: To ensure robust data security and access control through role-based authentication, session management, comprehensive audit logging, and patient access tracking for HIPAA compliance.

Scalable Database Architecture: To create scalable database architecture supporting millions of medical records with optimized indexing, efficient query performance, and automated backup mechanisms.

1.2 Scope

IHORMS is designed as a production-ready, multi-tenant hospital management platform suitable for deployment across hospital chains ranging from 3-50 branch locations. The system encompasses the complete patient journey from initial registration through appointment scheduling, clin-

ical documentation, admission management, pharmacy services, and final billing with insurance processing.

The platform supports eight distinct user roles with hierarchical access controls: Super Admins managing the platform infrastructure, Organization Admins overseeing hospital chains, Branch Admins managing individual locations, Doctors providing clinical care, Nurses monitoring patient vitals, Receptionists handling front-desk operations, Pharmacy Staff managing inventory, and Patients accessing self-service features.

While IHORMS provides comprehensive operational management and clinical documentation capabilities, it does not extend to specialized medical imaging analysis, laboratory information systems integration, or direct integration with medical devices beyond manual vital sign entry. The system is optimized for PostgreSQL deployment with FastAPI backend architecture, ensuring sub-second query performance even with millions of historical records.

IHORMS adheres to healthcare data privacy standards including HIPAA requirements through encrypted data storage, comprehensive audit trails, role-based access controls, and patient access logging. The system is designed for deployment on standard cloud infrastructure or on-premises servers, with minimal hardware requirements making it accessible to resource-constrained healthcare facilities.

Chapter 2

SOFTWARE REQUIREMENT SPECIFICATION

This chapter details the technical prerequisites and functional specifications necessary for successful deployment and operation of IHORMS, encompassing hardware infrastructure, software dependencies, and comprehensive functional requirements across all user roles.

2.1 Software Requirements

2.1.1 Operating System

IHORMS server supports deployment on Ubuntu Server 20.04 LTS or 22.04 LTS (recommended), CentOS 8 or Rocky Linux 8, Windows Server 2019/2022, or Docker containerized deployment on any host OS.

2.1.2 Backend Technologies

Python 3.11+: Core backend language with async/await support for high-performance API endpoints.

FastAPI 0.109+: Modern, high-performance web framework with automatic OpenAPI documentation, dependency injection, and async request handling.

Uvicorn/Gunicorn: ASGI server for production deployment with worker process management and load balancing.

Pydantic 2.x: Data validation and serialization library ensuring type safety and automatic request/response validation.

2.1.3 Database Technologies

PostgreSQL 14+: Primary relational database providing ACID compliance, advanced indexing, full-text search, and robust transaction management for critical healthcare data.

SQLAlchemy 2.0+: Python ORM providing database abstraction, query generation, connection pooling, and migration support through Alembic.

psycopg2: PostgreSQL adapter for Python enabling efficient database connectivity and query execution.

2.1.4 Authentication and Security

OAuth2 with Password Flow: Industry-standard authentication protocol for secure credential exchange and token-based session management.

JWT (JSON Web Tokens): Stateless authentication tokens containing encrypted user identity, role, and organization claims.

Passlib with bcrypt: Cryptographic password hashing using bcrypt algorithm with configurable work factor for future-proof security.

CORS Middleware: Cross-Origin Resource Sharing configuration enabling secure API access from web frontends.

2.1.5 Frontend Technologies

HTML5: Semantic markup providing accessible, standards-compliant user interfaces.

CSS3: Modern styling with Flexbox/Grid layouts, responsive design, and custom animations.

JavaScript (ES6+): Client-side interactivity, AJAX requests, dynamic DOM manipulation, and local session management.

Chart.js 4.x: Interactive data visualization for billing analytics, appointment trends, and organizational metrics.

2.1.6 Machine Learning Libraries

scikit-learn: Machine learning framework for doctor recommendation system using text similarity algorithms and classification models.

Faker: Synthetic data generation library for populating development/testing databases with realistic hospital records.

pandas: Data manipulation library for processing medical history, generating reports, and preparing analytics datasets.

2.2 Hardware Requirements

2.2.1 Server-Side Requirements

The IHORMS backend infrastructure is optimized for modern cloud or on-premises deployment:

Processor: Intel Xeon or AMD EPYC series with minimum 4 cores, 2.4 GHz clock speed. 8-core configuration recommended for production deployments handling 100+ concurrent users.

Memory (RAM): Minimum 8 GB for development environments, 16-32 GB recommended for production deployments to support concurrent database connections, API requests, and in-memory caching.

Storage: Minimum 50 GB SSD storage for application and database files. Production deployments require 500GB-2TB based on patient volume, with daily incremental backups.

Network: Gigabit Ethernet for local deployment, minimum 100 Mbps internet bandwidth for cloud hosting to ensure responsive API performance across branch locations.

2.2.2 Client-Side Requirements

Client workstations accessing IHORMS web interfaces require:

Processor: Any modern dual-core processor capable of running Chrome, Firefox, or Edge browsers.

Memory (RAM): Minimum 4 GB for smooth browser operation and responsive interface rendering.

Display: 1366x768 minimum resolution, 1920x1080 recommended for optimal dashboard viewing.

Network: Minimum 10 Mbps connection to backend server for real-time updates and responsive page loads.

2.3 Functional Requirements

2.3.1 Multi-Tenant Architecture Requirements

Organization Isolation: Complete data segregation between hospital organizations ensuring Apollo patients cannot access Fortis records, implemented through database-level filtering on every query.

Branch Management: Organization administrators can create, activate, deactivate, and manage multiple branch locations with independent staff assignments and resource allocation.

Standardized ID Generation: Automated generation of globally unique identifiers following format ORG-BRANCH-ENTITY-SEQUENCE, e.g., APL-MUM-P00001 for first patient at Apollo Mumbai.

Cross-Branch Patient Transfer: Patients can book appointments at any branch within their organization with complete medical history visibility.

2.3.2 Authentication and Role-Based Access Control

Unified Login Portal: Single sign-on interface where users authenticate with email and password, with backend automatically routing to role-appropriate dashboard.

JWT Token Security: Upon successful authentication, server issues encrypted JWT containing user ID, role, organization ID, and branch ID, stored client-side for subsequent API requests.

Role-Based Routing: Frontend enforces role-based page access—nurse attempting to access doctor portal receives automatic redirect. Backend API endpoints validate JWT role claims before processing requests.

Session Management: Automatic session expiration after 8 hours or 30 minutes of inactivity, requiring re-authentication for security.

Password Security: Passwords hashed using bcrypt with random salt, minimum 8 characters with complexity requirements, stored securely without plaintext exposure.

2.3.3 Doctor Portal Requirements

Appointment Dashboard: Daily schedule view showing confirmed appointments with patient demographics, chief complaints, and consultation room assignments.

Clinical Documentation: Interface for recording detailed diagnosis, treatment plans, prescriptions, and clinical verdicts with structured data entry.

Patient Admission: One-click admission workflow allowing doctors to admit patients to ICU, general ward, or emergency rooms directly from consultation interface.

Discharge Approval: Review pending discharge requests from nurses, approve/reject with mandatory discharge summary entry, triggering billing finalization.

Medical History Access: Complete patient timeline showing all previous visits, diagnoses, prescriptions, and telemetry readings across all branches.

Patient Search: Advanced search by patient UID, name, phone, or appointment date with access logging for HIPAA compliance.

2.3.4 Nurse Portal Requirements

Ward Management: Real-time view of all admitted patients showing room assignments, admission dates, and responsible doctors.

Telemetry Recording: Structured forms for entering patient vitals including heart rate, blood pressure, temperature, oxygen saturation, and respiratory rate.

Critical Alert System: Automated threshold monitoring triggering immediate visual alerts when vitals exceed safe ranges.

Discharge Request Workflow: Interface for submitting discharge requests to attending physicians with supporting notes, tracked status, and approval history.

Patient Monitoring Dashboard: Timeline view of telemetry readings with graphical trends enabling pattern recognition.

2.3.5 Receptionist Portal Requirements

Patient Registration: Comprehensive form capturing demographics, contact information, emergency contacts, blood group, insurance details with policy numbers and expiry dates.

Appointment Scheduling: Calendar interface for booking appointments with doctor specialty filtering, availability checking, and automatic room assignment.

Smart Doctor Recommendation: ML-powered suggestion system analyzing patient's chief complaint to recommend appropriate specialists with availability status.

Appointment Management: View, reschedule, cancel appointments with automated notification generation.

Billing Interface: Create itemized bills with multiple line items, discount percentages, tax calculations, and support for multiple payment methods.

Patient Search: Quick lookup by name, phone, or patient UID for returning patients.

2.3.6 Pharmacist Portal Requirements

Inventory Dashboard: Real-time stock levels for all medicines with batch tracking, expiry dates, and reorder thresholds.

Stock Management: Update inventory quantities, record new stock receipts with batch numbers and manufacturer details.

Expiry Monitoring: Automated alerts for medicines nearing expiry, enabling proactive stock rotation.

Low Stock Alerts: Visual indicators and notifications when medicine quantities fall below configurable reorder levels.

Time-Series Forecasting: Predictive analytics using historical consumption data to forecast future demand, optimizing procurement decisions.

Prescription Fulfillment: Interface for processing pharmacy orders linked to appointments with automatic inventory deduction.

2.3.7 Patient Portal Requirements

Self-Service Appointment Booking: Patients can book appointments by entering symptoms, receiving AI-powered doctor recommendations, and selecting available time slots.

Medical History Viewer: Comprehensive timeline of all consultations with diagnosis, prescriptions, and vital readings from any branch within the organization.

Bill Management: View all bills with detailed breakdowns, support for online payment, and insurance claim submission.

Insurance Claim Tracking: Submit insurance claims directly through portal with policy details, track claim status.

Profile Management: Update contact information, emergency contacts, and insurance policy details.

2.3.8 Organization Admin Requirements

Branch Management: Create new branches, assign branch administrators, view branch-level operational metrics.

Staff Directory: Comprehensive view of all staff across all branches with role filtering, activity status, and assignment details.

Billing Analytics Dashboard: Interactive visualizations showing monthly revenue trends, total bills generated, average bill amounts, and outstanding payments.

Revenue Insights: Dual-axis charts displaying revenue vs. bill count trends over configurable time periods.

KPI Monitoring: Real-time calculation of key performance indicators including total revenue, bill count, average transaction value, and outstanding receivables.

2.3.9 Branch Admin Requirements

Staff Onboarding: Create user accounts for doctors, nurses, receptionists, and pharmacy staff with role assignment and credential generation.

Resource Management: Configure consultation rooms, ward beds, ICU capacity, and equipment inventory for the branch.

Operational Reports: Branch-level analytics showing daily appointments, patient admissions, and staff utilization.

Staff Activity Monitoring: Track login history, patient access logs, and action audit trails for compliance.

Chapter 3

ER DIAGRAM

This chapter presents the comprehensive Entity-Relationship model representing the IHORMS database structure, encompassing all entities, their attributes, relationships, and cardinalities required to support multi-tenant hospital operations.

The IHORMS ER diagram models a sophisticated multi-tenant hospital management system supporting multiple organizations, each with multiple branches, diverse user roles, clinical workflows, and financial operations. The diagram employs standard ER notation with entities represented as rectangles, relationships as diamonds, and attributes as ovals, with cardinality constraints clearly marked.

At the architectural foundation lies the **Organization** entity, representing hospital chains like Apollo or Fortis. Organizations contain attributes including name, contact details, operational status, and timestamps. Each organization owns multiple branches and maintains complete data isolation from other organizations.

The **Branch** entity represents physical hospital locations within an organization, storing location-specific details including address, city, state, pincode, and contact information. The one-to-many relationship between Organization and Branch enables a single hospital chain to manage numerous locations while maintaining centralized oversight.

The **User** entity implements a polymorphic design supporting eight distinct roles through a single table. Core attributes include role enumeration (super_admin, org_admin, branch_admin, doctor, nurse, receptionist, pharmacist, patient), email, password hash, personal information, and organizational relationships.

Role-specific entities extend the base User model. The **Doctor** entity adds specialization, qualification, experience years, license number, and consultation fee. The **Nurse** entity contains qualification and license number. The **Patient** entity stores patient-specific data including unique patient UID, blood group, emergency contacts, and insurance details.

The **Room** entity tracks physical spaces within branches, categorized by type (consultation, ICU, general ward, emergency, operation theater) with attributes for room number, floor, capacity, and availability status.

Equipment manages medical devices and instruments with serial numbers, equipment type, purchase dates, maintenance schedules, and operational status.

The **Appointment** entity serves as the central clinical workflow hub, linking patients and doc-

tors with scheduled date and time, room assignment, appointment status, chief complaint, clinical notes, diagnosis, prescription, and verdict.

Admission manages inpatient care, recording admission date, discharge workflow with request and approval stages, discharge summaries, and linking to the originating appointment.

MedicalHistory provides longitudinal patient records with visit dates, diagnoses, symptoms, severity levels, treatments, medications, follow-up requirements, doctor notes, and lab results.

TelemetryData captures patient vital signs including heart rate, blood pressure, temperature, oxygen saturation, respiratory rate, ICU status, and automated alert flags with messages.

The **Billing** entity handles financial transactions with unique bill numbers, itemized charges, subtotal, tax, discount, total amount, payment tracking, and payment method.

InsuranceClaim processes insurance workflows with claim numbers, insurance provider, policy details, claimed amounts, approved amounts, and status tracking.

Inventory manages pharmacy stock with medicine names, generic equivalents, manufacturer details, batch numbers, quantities, pricing, expiry dates, and reorder thresholds.

PharmacyOrder tracks medicine dispensing with order numbers, patient references, itemized lists, total amounts, and fulfillment status.

Audit entities including **PatientAccessLog**, **AuditLog**, and **SystemEvent** provide comprehensive compliance tracking.

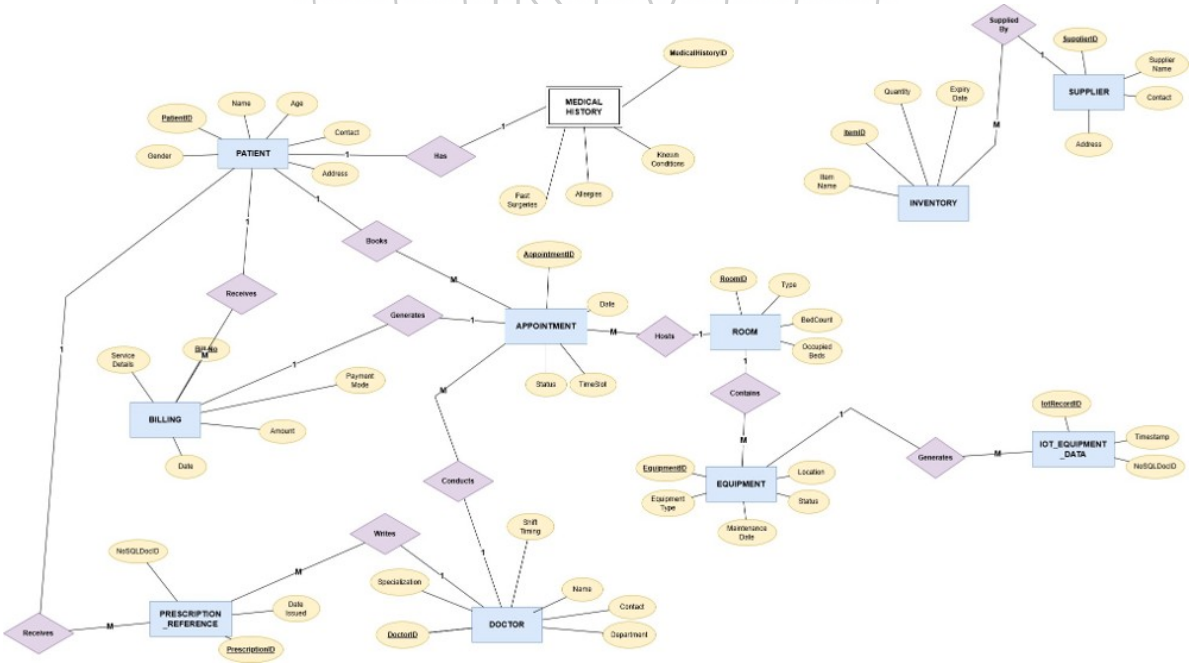

Figure 3.1: IHORMS Entity-Relationship Diagram depicting all entities, attributes, and relationships in the multi-tenant hospital management system

3.1 Key Relationships and Cardinalities

The ER diagram establishes the following critical relationships:

Organization to Branch (1:N): Each organization owns multiple branches, with Branch.organization_id as foreign key. This one-to-many relationship enables hospital chains to expand across multiple locations while maintaining centralized governance.

Branch to User (1:N): Each branch employs multiple staff members, with User.branch_id as foreign key.

Organization to User (1:N): Each organization has multiple users across all roles, with User.organization_id as foreign key. This relationship establishes the tenant boundary, ensuring data isolation between competing hospital organizations.

Doctor to Appointment (1:N): Each doctor manages multiple appointments over time, with Appointment.doctor_id as foreign key.

Patient to Appointment (1:N): Each patient can have multiple appointments throughout their lifetime, with Appointment.patient_id as foreign key.

Appointment to Admission (1:1): An appointment may result in exactly one admission, with Admission.appointment_id as unique foreign key.

Appointment to Billing (1:1): Each appointment generates exactly one bill, with Billing.appointment_id as unique foreign key.

Billing to InsuranceClaim (1:N): A bill may have multiple insurance claims processed sequentially, with InsuranceClaim.bill_id as foreign key.

Patient to MedicalHistory (1:N): Each patient accumulates multiple medical history records, with MedicalHistory.patient_id as foreign key.

Admission to TelemetryData (1:N): Each admission generates multiple telemetry readings, with TelemetryData.admission_id as foreign key.

Branch to Room (1:N): Each branch contains multiple rooms, with Room.branch_id as foreign key.

Branch to Inventory (1:N): Each branch maintains independent pharmacy inventory, with Inventory.branch_id as foreign key.

These relationships collectively form a robust data model supporting complex multi-tenant hospital operations while maintaining data integrity through foreign key constraints and cardinality enforcement.

3.2 Normalization Strategy

The ER diagram is designed to achieve Third Normal Form (3NF), eliminating data redundancy and ensuring update anomalies cannot occur:

No Redundancy: Patient names are stored only in the Users and Patients tables, not duplicated across Appointments or Bills. Doctor specializations reside solely in the Doctor entity, referenced through foreign keys rather than repeated.

Atomic Data: Attributes are decomposed into smallest meaningful units. Addresses are broken into separate fields enabling granular filtering and analytics. Blood pressure is split into systolic and diastolic readings for precise medical analysis.

Full Functional Dependency: Every non-key attribute depends entirely on the primary key. Doctor specialization depends on doctor_id, not on appointment_id. Room availability depends on room_id, not on the appointments scheduled within it.

No Transitive Dependencies: Non-key attributes do not depend on other non-key attributes. Organization address is stored in the Organization table, not duplicated in Branch records which only reference organization_id. Patient insurance details reside in the Patient entity, not propagated to every Billing record.

This normalization strategy ensures data consistency, simplifies updates, and eliminates insertion, deletion, and update anomalies.

Chapter 4

DETAILED DESIGN

This chapter presents the hierarchical decomposition of IHORMS functionality through Data Flow Diagrams (DFDs) at three levels of abstraction. DFDs provide a visual representation of information movement through the system, illustrating how data is processed, transformed, and stored across various functional modules.

4.1 DFD Level 0: Context Diagram

The Context Diagram establishes the system boundary, presenting IHORMS as a single unified process interacting with six primary external entities. This highest level of abstraction illustrates the system's position within the broader healthcare ecosystem.

The diagram depicts the following external entities and their interactions:

Admin: Provides staff data, branch configurations, and resource allocation directives. Receives system reports including billing analytics, staff directories, appointment volumes, and operational metrics.

Patient: Submits personal details during registration, appointment requests with symptom descriptions, insurance policy information, and payment details. Receives booking confirmations, medical history reports, itemized bills, and insurance claim status updates.

Doctor: Inputs clinical data including diagnoses, treatment plans, prescriptions, and patient admission/discharge decisions. Receives comprehensive patient medical histories, real-time critical alerts, appointment schedules, and pending discharge approval requests.

Nurse: Records patient vital signs including heart rate, blood pressure, temperature, oxygen saturation, and respiratory rate. Submits discharge requests. Receives critical alerts when patient vitals exceed safe thresholds, ward assignments, and discharge approval notifications.

Pharmacist: Updates medicine stock levels, records new inventory receipts with batch numbers and expiry dates. Receives prescription orders, low-stock alerts, and expiry warnings.

Receptionist: Enters patient registration data, schedules appointments, creates itemized bills, and processes payments. Receives AI-powered doctor recommendations, appointment availability status, and billing reports.

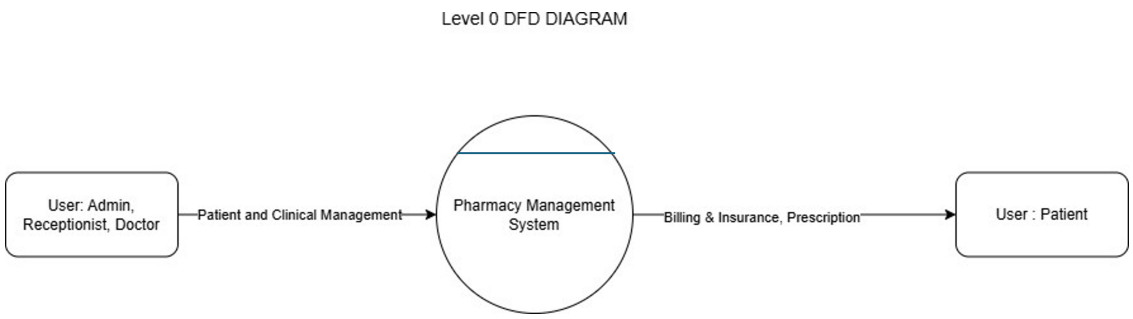

Figure 4.1: IHORMS Context Diagram showing external entities and system boundary

4.2 DFD Level 1: Major System Processes

The Level 1 diagram decomposes the IHORMS monolith into six core functional processes, revealing the major subsystems and their data exchange patterns.

Process 1.0 - User Authentication and Authorization: Validates user credentials against the User database, generates JWT tokens containing encrypted user identity and role claims, and routes users to role-appropriate dashboards.

Process 2.0 - Clinical Workflow Management: Orchestrates the complete patient care lifecycle from appointment scheduling through discharge. Sub-processes include ML-based doctor recommendation, appointment creation with room allocation, medical documentation, patient admission, and discharge approval workflows.

Process 3.0 - Patient Monitoring and Telemetry: Processes vital signs data entered by nurses, performs real-time threshold analysis to detect critical conditions, and generates automated alerts when patients deteriorate.

Process 4.0 - Billing and Insurance Management: Creates itemized bills aggregating charges from consultations, medications, room usage, and tests. Applies discounts and taxes, tracks payments, and manages insurance claim submission and status tracking.

Process 5.0 - Pharmacy Inventory Management: Tracks medicine stock levels, generates reorder alerts, monitors expiry dates, and processes prescription fulfillment orders.

Process 6.0 - Analytics and Reporting: Aggregates operational data across all branches, generates revenue trend visualizations, calculates appointment volumes and staff utilization metrics, and produces predictive insights.

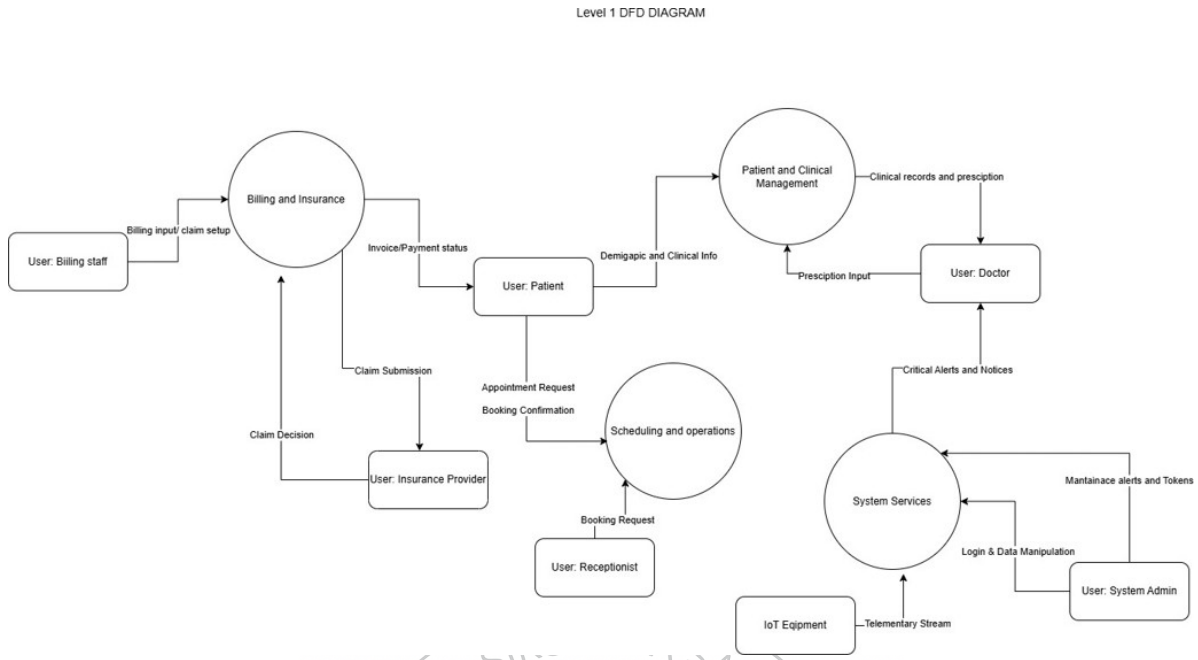

Figure 4.2: IHORMS Level 1 DFD decomposing the system into six core functional processes

4.3 DFD Level 2: Detailed Sub-Process Workflows

4.3.1 DFD 2.1 - Staff Management Sub-Process

This diagram details the administrative workflows for managing hospital staff across all roles:

Process 2.1.1 - Staff Registration: Branch administrators input staff details. The system validates email uniqueness, generates password hashes using bcrypt, assigns organization and branch IDs, and writes complete user records.

Process 2.1.2 - Role Assignment: After user creation, the system links users to role-specific tables. Doctors receive records with specialization, consultation fees, and license numbers.

Process 2.1.3 - Credential Verification: For medical professionals, the system performs verification of license numbers against external medical boards.

Process 2.1.4 - Staff Deactivation: Administrators mark users as inactive rather than deleting records, preserving audit trails and maintaining referential integrity.

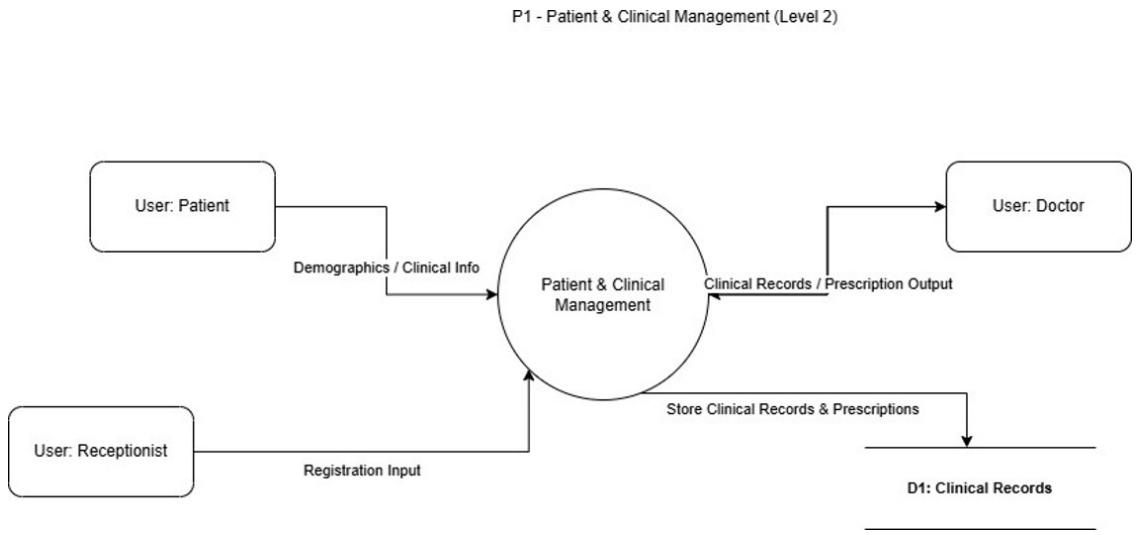

Figure 4.3: Detailed sub-processes for staff registration, role assignment, and credential verification

4.3.2 DFD 2.2 - Clinical Flow Sub-Process

This diagram illustrates the sequence of clinical activities from doctor perspective:

Process 2.2.1 - Patient History Retrieval: When a doctor begins a consultation, the system queries medical history filtered by patient ID, ordering by visit date descending.

Process 2.2.2 - Vital Signs Recording: During consultation, nurses input current vital signs through structured forms. The system validates ranges and writes to TelemetryData table.

Process 2.2.3 - Diagnosis Documentation: After examination, doctors enter diagnosis codes, clinical notes, and treatment plans. The system stores this in both Appointment and MedicalHistory tables.

Process 2.2.4 - Prescription Generation: Doctors specify medications, dosages, and frequencies. The system checks for drug interactions, allergies, and stock availability.

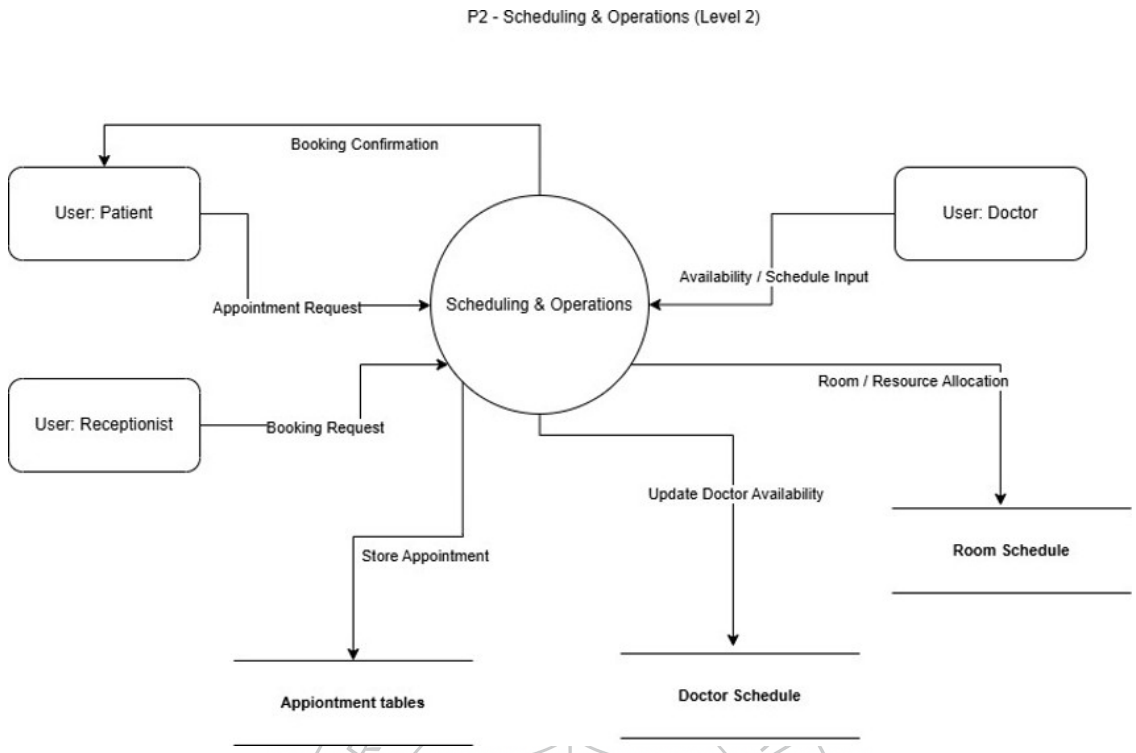

Figure 4.4: Clinical workflow from patient history retrieval through prescription generation

4.3.3 DFD 2.3 - Appointment Logic Sub-Process

This diagram details the appointment scheduling algorithm preventing double-booking:

Process 2.3.1 - Doctor Availability Check: The system queries all appointments for selected doctor and date, calculating occupied time slots.

Process 2.3.2 - Room Allocation: For available time slots, the system queries Room table and assigns using round-robin algorithm.

Process 2.3.3 - Appointment Creation: Upon confirming available doctor and room, the system generates unique appointment record.

Process 2.3.4 - Conflict Resolution: If double-booking is attempted, PostgreSQL's serializable isolation ensures one transaction succeeds.

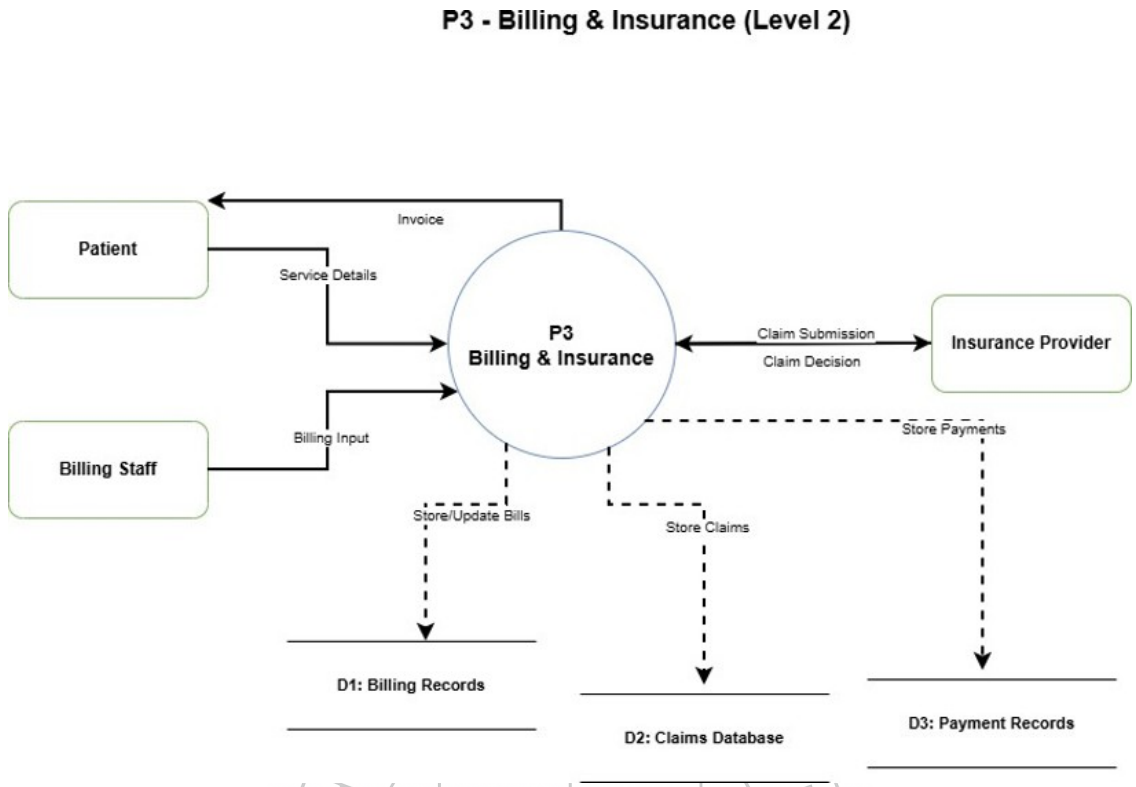

Figure 4.5: Appointment scheduling algorithm with availability checking and room allocation

4.3.4 DFD 2.4 - Billing and Pharmacy Integration

This diagram shows the financial workflow connecting clinical services to billing:

Process 2.4.1 - Charge Aggregation: When receptionist initiates billing, the system retrieves consultation fees, room charges, and pharmacy orders.

Process 2.4.2 - Tax and Discount Calculation: The system applies configurable tax rate and discount percentage.

Process 2.4.3 - Payment Processing: Receptionist selects payment method. For insurance payments, an InsuranceClaim record is created.

Process 2.4.4 - Invoice Generation: The system generates unique bill number and stores complete billing record.

Process 2.4.5 - Inventory Deduction: When PharmacyOrder is fulfilled, the system atomically decrements inventory quantities.

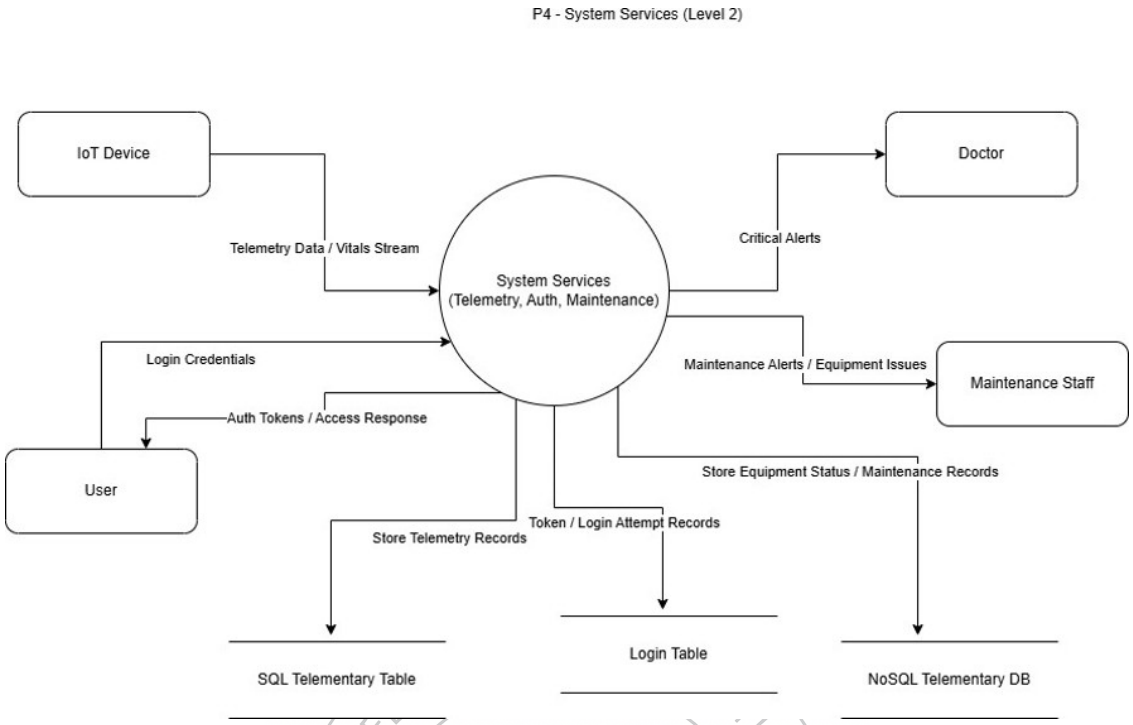

Figure 4.6: Financial workflows linking clinical services to billing and inventory management

Chapter 5

RELATIONAL SCHEMA AND NORMALIZATION

This chapter presents the finalized PostgreSQL relational schema for IHORMS, designed to ensure strong data integrity, scalability, and multi-tenant isolation. The database is normalized up to Third Normal Form (3NF) to reduce redundancy, eliminate update anomalies, and maintain consistent clinical, operational, and financial workflows across organizations and branches.

5.1 Relational Schema Diagram

The IHORMS relational schema defines the complete set of database entities and their relationships required to support hospital operations in a multi-tenant environment. It models organizational structure, user roles, clinical workflows, admissions, billing, telemetry monitoring, and pharmacy inventory using primary and foreign key constraints.

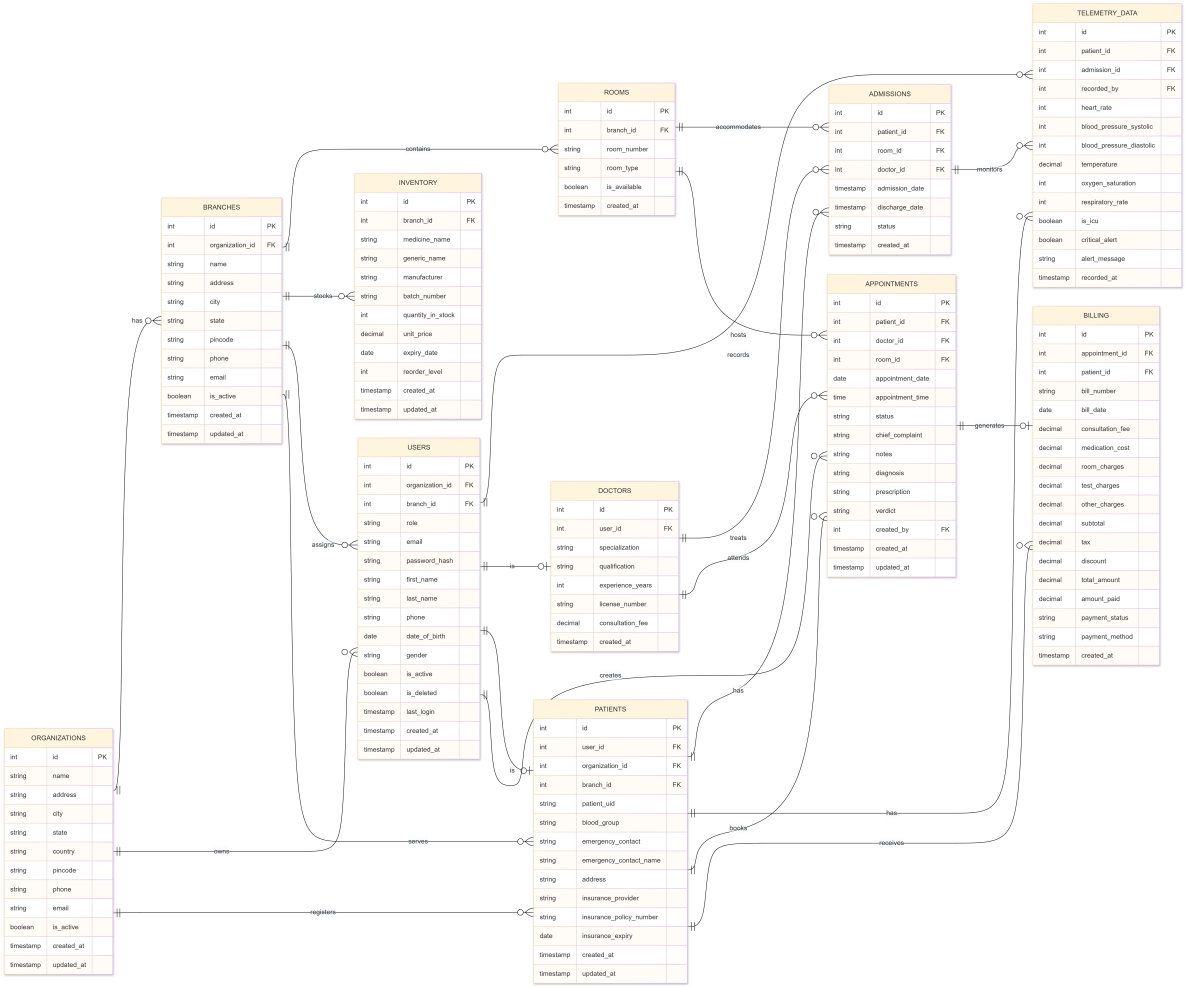

Figure 5.1: IHORMS Relational Schema (ER Diagram)

5.2 Normalization Summary (1NF to 3NF)

5.2.1 First Normal Form (1NF) — Atomicity

Rule	**Implementation in IHORMS**
Atomic values	Every attribute stores a single indivisible value (no arrays, lists, or repeating groups).
No repeating groups	Multi-value clinical readings, admissions, appointments, and billing transactions are stored as separate rows in dedicated tables.
Consistent row meaning	Each row represents one entity instance (one user, one appointment, one admission, one bill, one inventory batch).

Table 5.1: 1NF Compliance Summary

5.2.2 Second Normal Form (2NF) — Full Dependency on Primary Key

Rule	**Implementation in IHORMS**
No partial dependency	All non-key attributes depend fully on their table primary key.
Separated professional details	Doctor-specific fields are stored in `doctors` and referenced through foreign keys from appointments and admissions.
Independent resource metadata	Room details are stored in `rooms` and referenced only where required by workflow tables.
Branch metadata isolation	Branch contact and location details remain in `branches` and are not duplicated across user or patient records.

Table 5.2: 2NF Compliance Summary

5.2.3 Third Normal Form (3NF) — No Transitive Dependencies

Rule	Implementation in IHORMS
No transitive dependency	No non-key attribute depends on another non-key attribute within the same relation.
Organization data separation	Organization-level details are stored in `organizations`; branches reference organizations using `organization_id`.
Role-based entity separation	Common identity attributes remain in `users`, while role-specific details exist in `doctors` and `patients`.
Explicit billing state	Billing totals and payment status are stored explicitly to ensure consistent reporting and payment tracking.

Table 5.3: 3NF Compliance Summary

5.3 Normalized Relational Schema (3NF)

5.3.1 Core Multi-Tenant Tables

Table	PK	Foreign Keys (FKs)	Major Attributes
organizations	id	–	name (Unique), address, city, state, country, pincode, phone, email (Unique), is_active, created_at, updated_at
branches	id	organization_id → organizations(id)	name (Unique per org), address, city, state, pincode, phone, email, is_active, created_at, updated_at

Table 5.4: Core Multi-Tenant Tables

5.3.2 User and Role Tables

Table	PK	Foreign Keys (FKs)	Major Attributes
users	id	organization_id → organizations(id), branch_id → branches(id)	role (Enum), email (Unique), password_hash, first_name, last_name, phone, date_of_birth, gender (Enum), is_active, is_deleted, last_login, created_at, updated_at
doctors	id	user_id → users(id) (Unique)	specialization, qualification, experience_years, license_number (Unique), consultation_fee, created_at
patients	id	user_id → users(id) (Unique), organization_id → organizations(id), branch_id → branches(id)	patient_uid (Unique), blood_group (Enum), emergency_contact, emergency_contact_name, address, insurance_provider, insurance_policy_number, insurance_expiry, created_at, updated_at

Table 5.5: User and Role Tables

5.3.3 Facility and Admission Tables

Table	PK	Foreign Keys (FKs)	Major Attributes
rooms	id	branch_id → branches(id)	room_number, room_type, is_available, created_at
admissions	id	patient_id → patients(id), room_id → rooms(id), doctor_id → doctors(id)	admission_date, discharge_date, status, created_at

Table 5.6: Facility and Admission Tables

5.3.4 Clinical Workflow Tables

Table	PK	Foreign Keys (FKs)	Major Attributes
appointments	id	patient_id → patients(id), doctor_id → doctors(id), room_id → rooms(id), created_by → users(id)	appointment_date, appointment_time, status (Enum), chief_complaint, notes, diagnosis, prescription, verdict, created_at, updated_at
telemetry_data	id	patient_id → patients(id), admission_id → admissions(id), recorded_by → users(id)	heart_rate, blood_pressure_systolic, blood_pressure_diastolic, temperature, oxygen_saturation, respiratory_rate, is_icu, critical_alert, alert_message, recorded_at

Table 5.7: Clinical Workflow Tables

5.3.5 Financial Tables

Table	PK	Foreign Keys (FKs)	Major Attributes
billing	id	appointment_id → appointments(id), patient_id → patients(id)	bill_number (Unique), bill_date, consultation_fee, medication_cost, room_charges, test_charges, other_charges, subtotal, tax, discount, total_amount, amount_paid, payment_status (Enum), payment_method (Enum), created_at

Table 5.8: Financial Tables

5.3.6 Pharmacy Tables

Table	PK	Foreign Keys (FKs)	Major Attributes
inventory	id	branch_id → branches(id)	medicine_name, generic_name, manufacturer, batch_number, quantity_in_stock, unit_price, expiry_date, reorder_level, created_at, updated_at

Table 5.9: Pharmacy Tables

5.4 Key Constraints and Indexes

5.4.1 Referential Integrity Rules

Constraint Type	Purpose in IHORMS
Foreign Keys (FK)	Enforces valid relationships between tenants, branches, users, and workflow records.
Unique Constraints	Prevents duplicate critical identifiers such as email, patient_uid, license_number, and bill_number.
Check Constraints	Restricts invalid values for enumerations and numeric ranges (roles, statuses, vitals, payments).
Nullable FKs where required	Supports operational flexibility such as optional room allocation for appointments.

Table 5.10: Referential Integrity Rules

5.4.2 Performance Index Strategy

Index Name	Target Columns	Optimization Benefit
idx_user_email	users(email)	Accelerates login authentication and user lookup by email.
idx_appointment_date_doctor	appointments(appointment_date, doctor_id)	Optimizes doctor schedule retrieval by date and doctor.
idx_telemetry_critical	telemetry_data(critical_alert) WHERE critical_alert=TRUE	Enables fast filtering of critical patient telemetry alerts.
idx_billing_date	billing(bill_date)	Speeds up billing analytics and revenue reporting by date range.

Table 5.11: Index Strategy Summary

Chapter 6

CONCLUSION

The Integrated Hospital Resource Management System (IHORMS) demonstrates the practical application of strong database design principles to address real-world healthcare workflow challenges. By normalizing the schema up to Third Normal Form (3NF) and implementing a structured ER model and DFD decomposition, the system delivers a scalable multi-tenant hospital management solution.

Key outcomes include ML-based doctor recommendation, real-time telemetry monitoring with automated critical alerts, integrated billing, and role-based access control to support secure and compliant operations. The database design ensures minimal redundancy, strong referential integrity through foreign key constraints, data validation via check constraints, and optimized performance using strategic indexing.

Overall, IHORMS bridges academic database concepts with enterprise implementation requirements and provides a solid foundation for future upgrades such as IoT integration, telemedicine, insurance API connectivity, and predictive analytics.

Chapter 7

REFERENCES

1. M. A. Hannan et al., "A Review of Internet of Things (IoT) Embedded Sustainable Supply Chain for Industry 4.0 Requirements," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3954-3971, Oct. 2018.

2. S. M. R. Islam et al., "The Internet of Things for Health Care: A Comprehensive Survey," *IEEE Access*, vol. 3, pp. 678-708, 2015.

3. R. H. Weber et al., "Internet of Things: Privacy issues revisited," *Computer Law & Security Review*, vol. 31, no. 5, pp. 618-627, 2015.

4. A. Dwivedi et al., "Security Analysis of Lightweight Authentication Schemes for IoT Healthcare Systems," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12132-12144, Aug. 2021.

5. J. Zhang et al., "Data Security and Privacy Protection in Cloud Computing," *IEEE Network*, vol. 30, no. 4, pp. 86-94, July-Aug. 2016.

6. K. Patel et al., "Machine Learning Techniques for Healthcare: A Survey," *IEEE Access*, vol. 8, pp. 197369-197397, 2020.

7. M. Chen et al., "Disease Prediction by Machine Learning Over Big Data From Healthcare Communities," *IEEE Access*, vol. 5, pp. 8869-8879, 2017.

8. C. J. Date, *An Introduction to Database Systems*, 8th Edition, Addison-Wesley, 2004.

9. R. Elmasri and S. Navathe, *Fundamentals of Database Systems*, 7th Edition, Pearson, 2015.

Chapter 8

APPENDIX: SNAPSHOTS

This appendix presents comprehensive system screenshots documenting all user interfaces, demonstrating IHORMS functionality across eight distinct user roles.

8.1 Authentication Interface

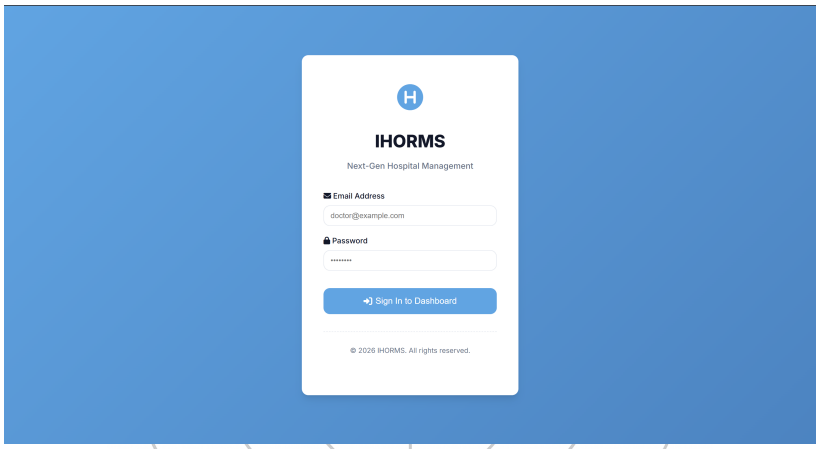

Figure 8.1: Unified Login Portal - Single sign-on interface where all users authenticate using email and password. The backend validates credentials, determines user role, and routes to role-appropriate dashboard.

8.2 Doctor Portal Interfaces

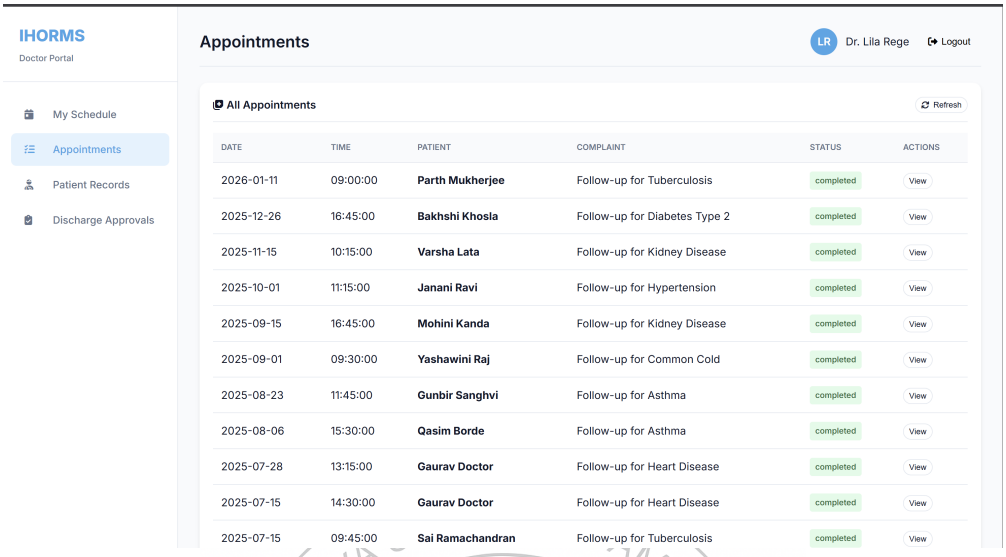

Figure 8.2: Doctor Dashboard Interface - Daily appointment schedule showing patient names, appointment times, chief complaints, and consultation room assignments. Doctors can search patients and access complete medical histories.

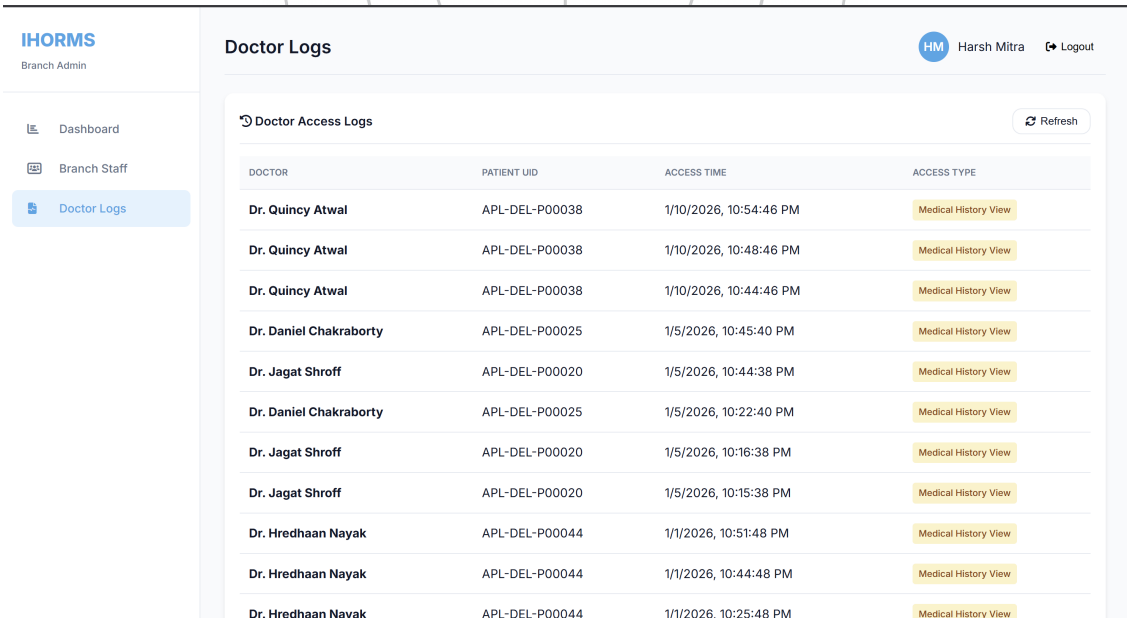

Figure 8.3: Patient Access Audit Logs - HIPAA-compliant tracking of every medical record access showing accessor name, timestamp, access reason, and IP address for compliance investigations and privacy oversight.

8.3 Nurse Portal Interfaces

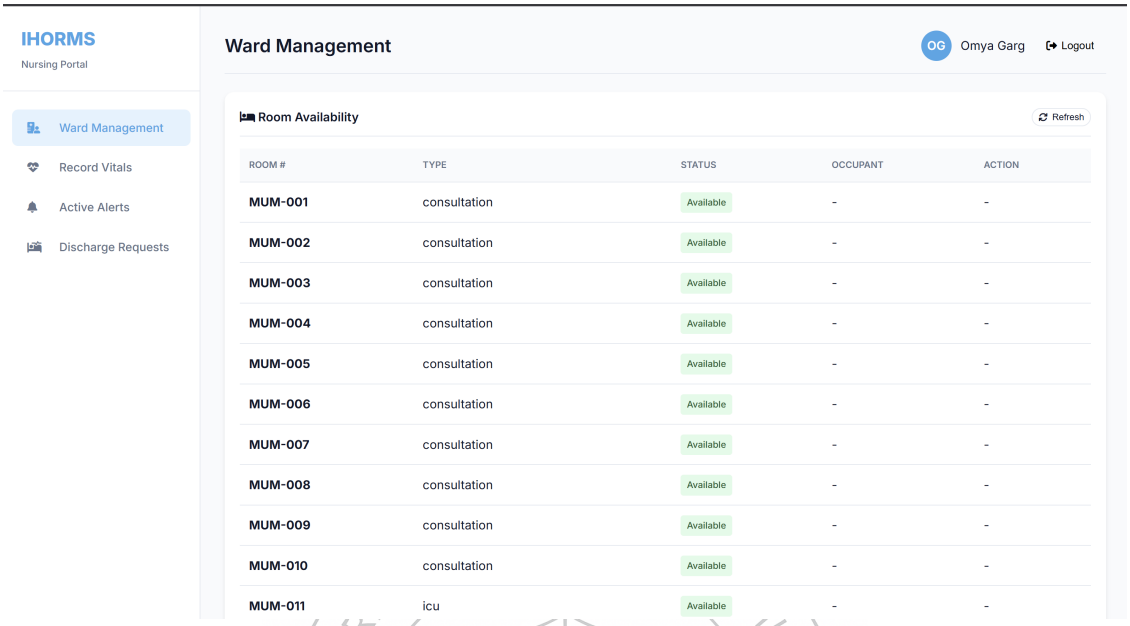

Figure 8.4: Nurse Ward Management Interface - Real-time view of all admitted patients showing room numbers, admission dates, attending physicians, and current status. Nurses can record vital signs and submit discharge requests.

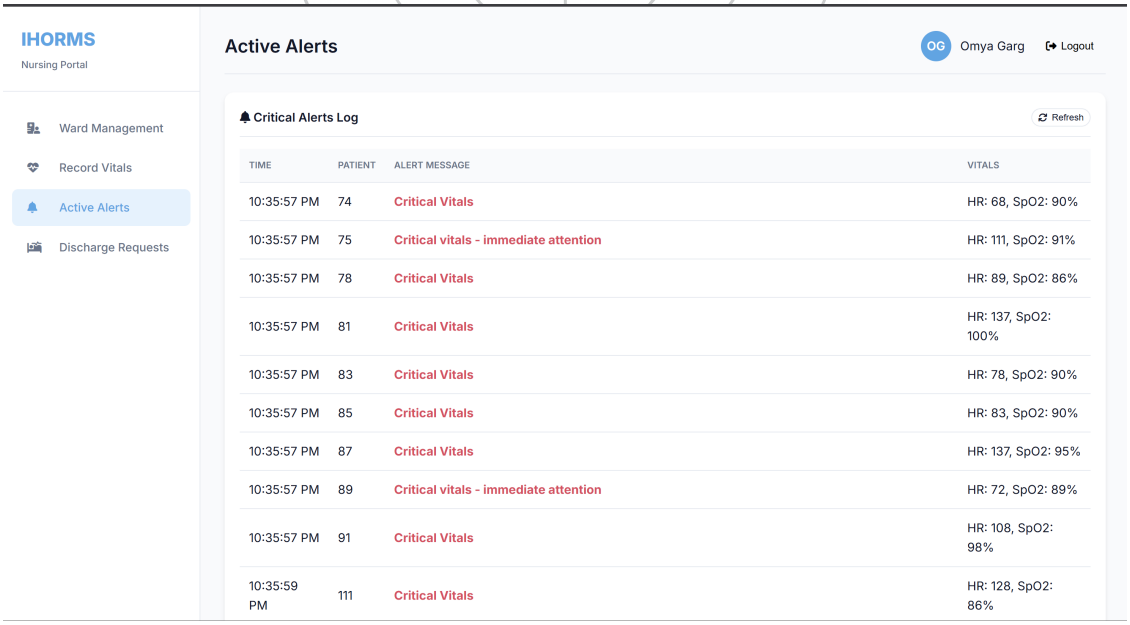

Figure 8.5: Real-Time Critical Alert System - Automated threshold monitoring displaying urgent warnings when patient vitals exceed safe ranges, enabling immediate medical intervention.

8.4 Receptionist Portal Interfaces

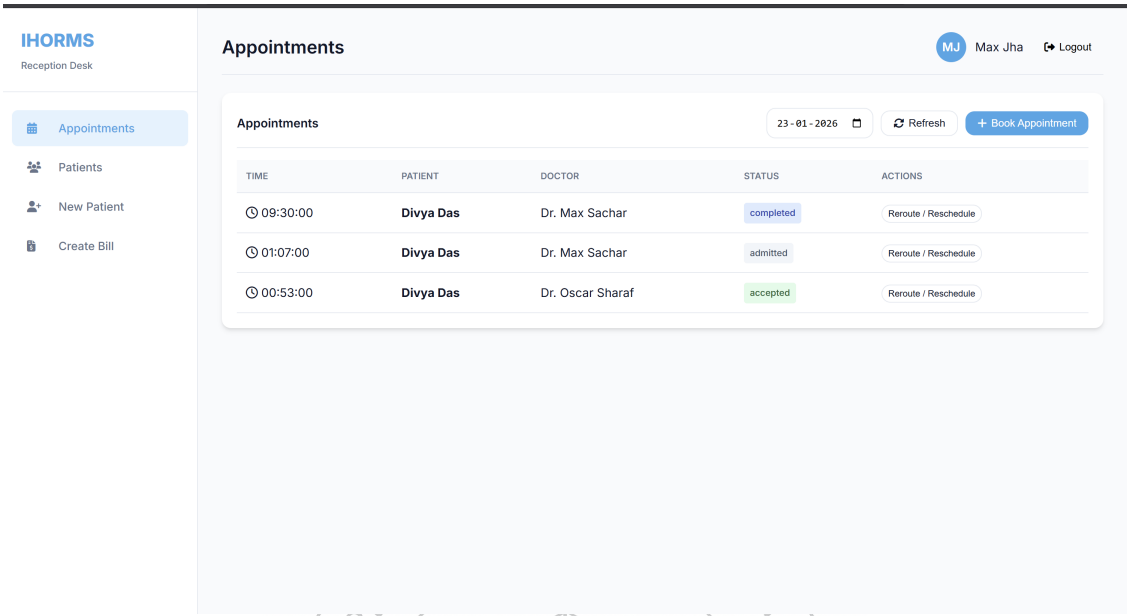

Figure 8.6: Appointment Scheduling with AI Recommendations - ML-powered appointment booking interface analyzing patient symptoms to recommend appropriate specialists with availability checking to prevent double-booking.

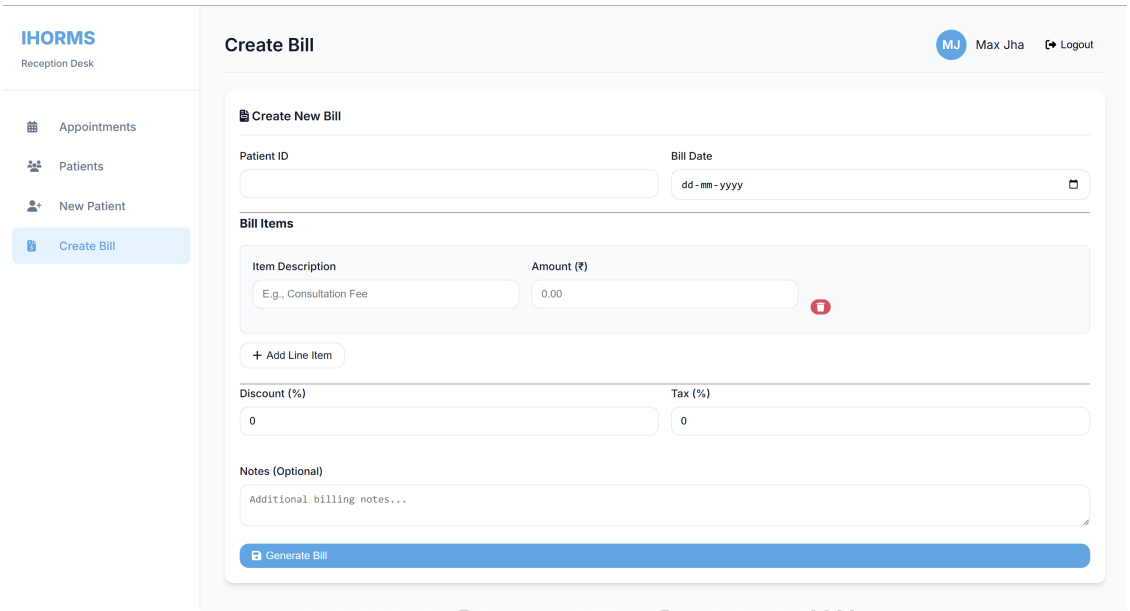

Figure 8.7: Itemized Billing Interface - Comprehensive invoice generation supporting multiple charge categories with automatic tax and discount calculations. Multiple payment methods supported with partial payment tracking.

8.5 Patient Portal Interfaces

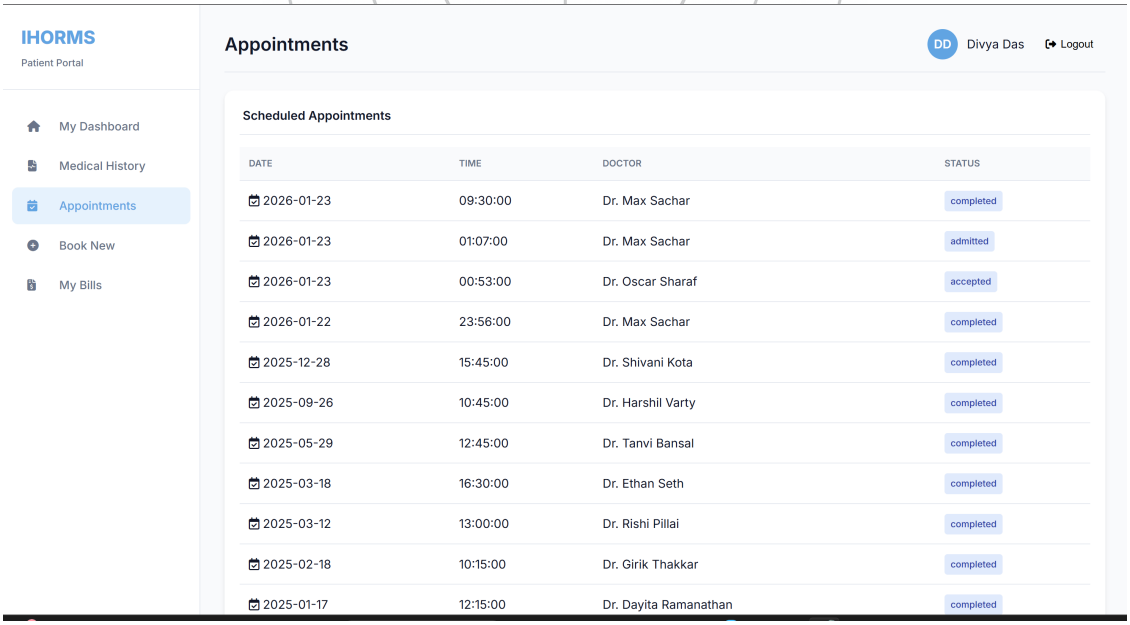

Figure 8.8: Patient Self-Service Dashboard - Patient interface for viewing medical history, booking appointments, and managing bills without staff intervention, empowering patients with transparency and control.

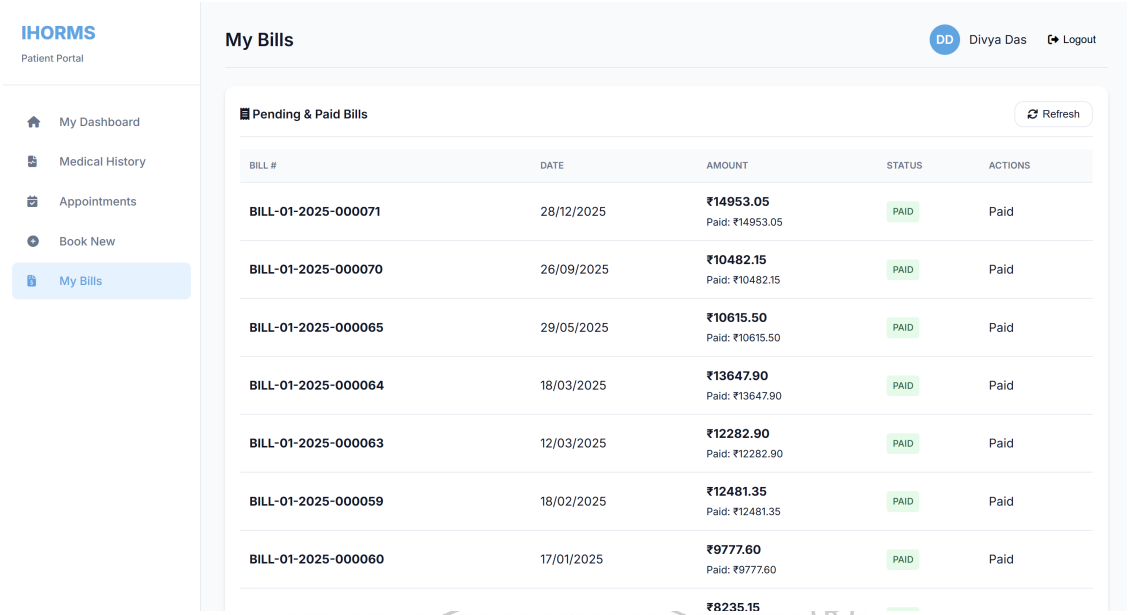

Figure 8.9: Patient Billing and Claims Management - Detailed bill viewing with itemized breakdowns, payment status tracking, and insurance claim submission capabilities simplifying financial management.

8.6 Pharmacy Portal Interfaces

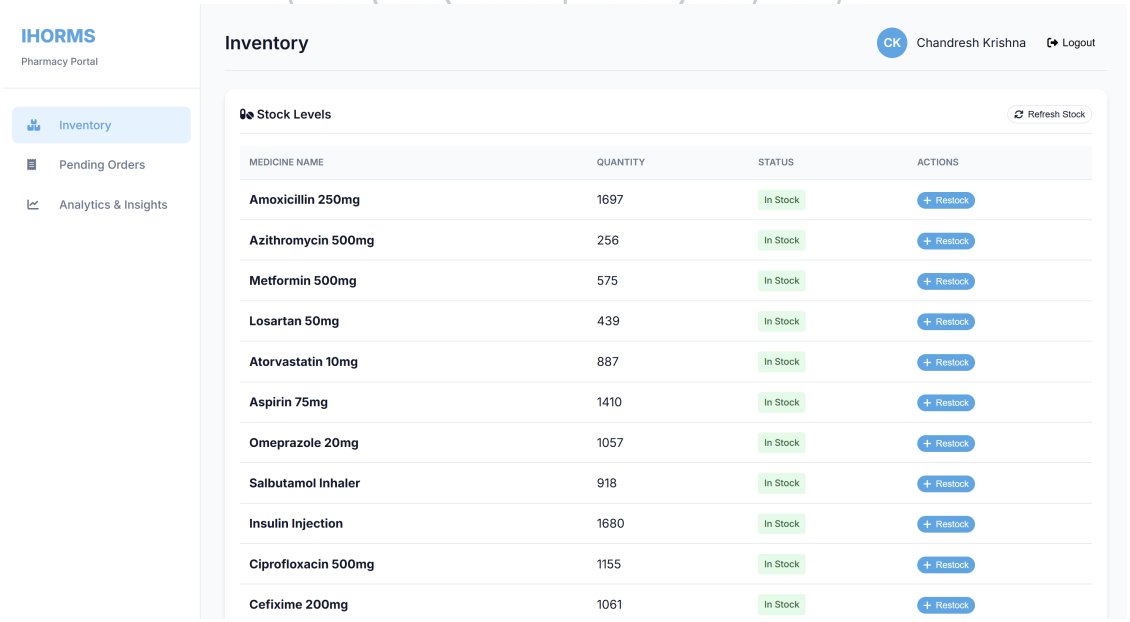

Figure 8.10: Pharmacy Inventory Management - Real-time stock monitoring displaying medicine names, batch numbers, quantities, expiry dates, and reorder thresholds with automated alerts for low stock and approaching expiry dates.

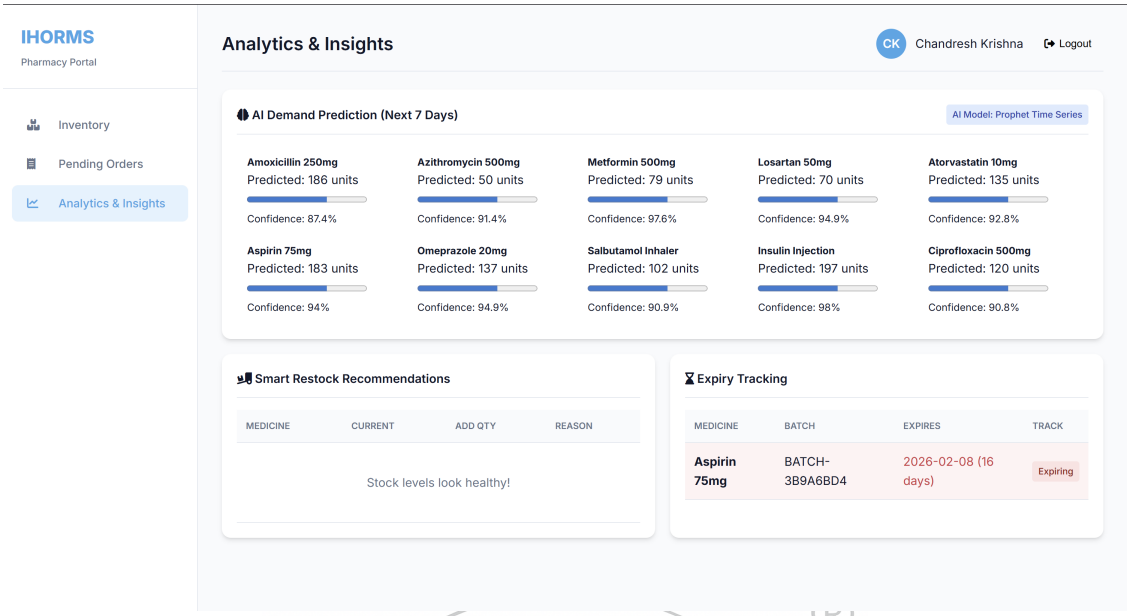

Figure 8.11: Predictive Demand Forecasting - Time-series analytics using historical consumption patterns to forecast future medicine demand, reducing stockouts and minimizing waste from over-ordering.

8.7 Organization Admin Portal Interfaces

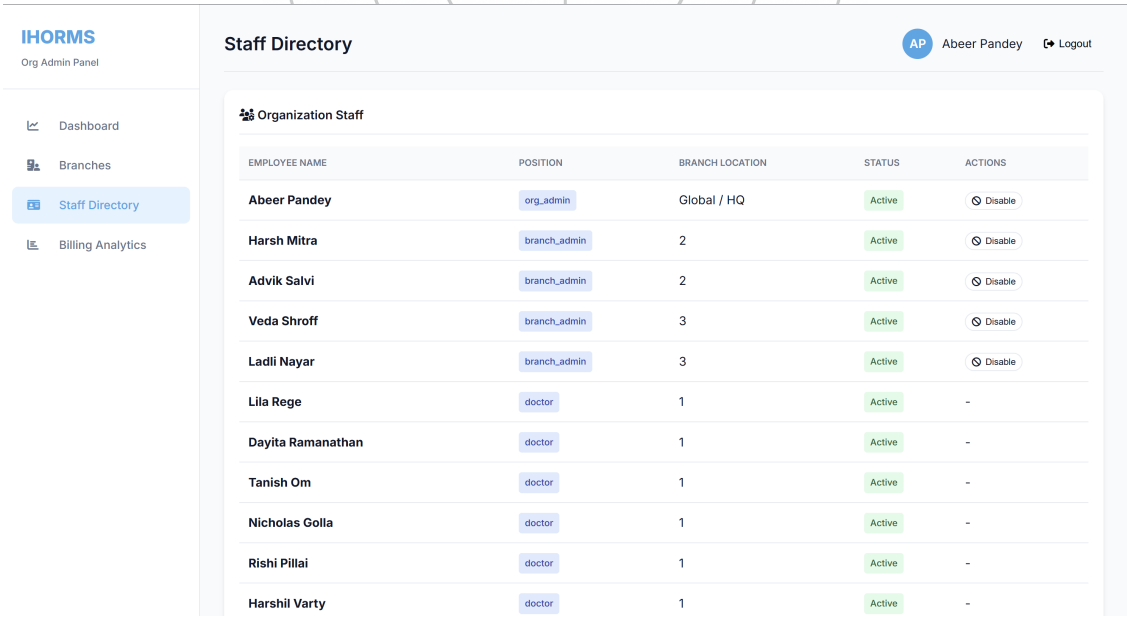

Figure 8.12: Organization-Wide Staff Directory - Comprehensive employee listing across all branches showing names, roles, specializations, contact details, and activity status for effective workforce management.

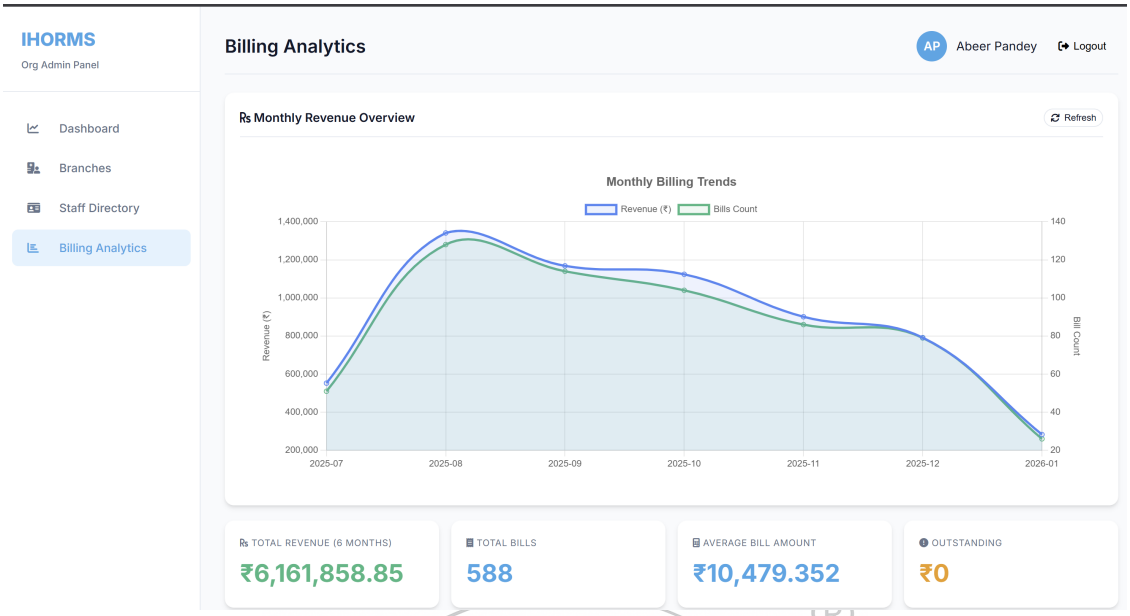

Figure 8.13: Organization Billing Analytics Dashboard - Interactive visualizations displaying monthly revenue trends, bill counts, and KPI metrics aggregated across all branches for strategic planning and performance monitoring.

8.8 Branch Admin Portal Interfaces

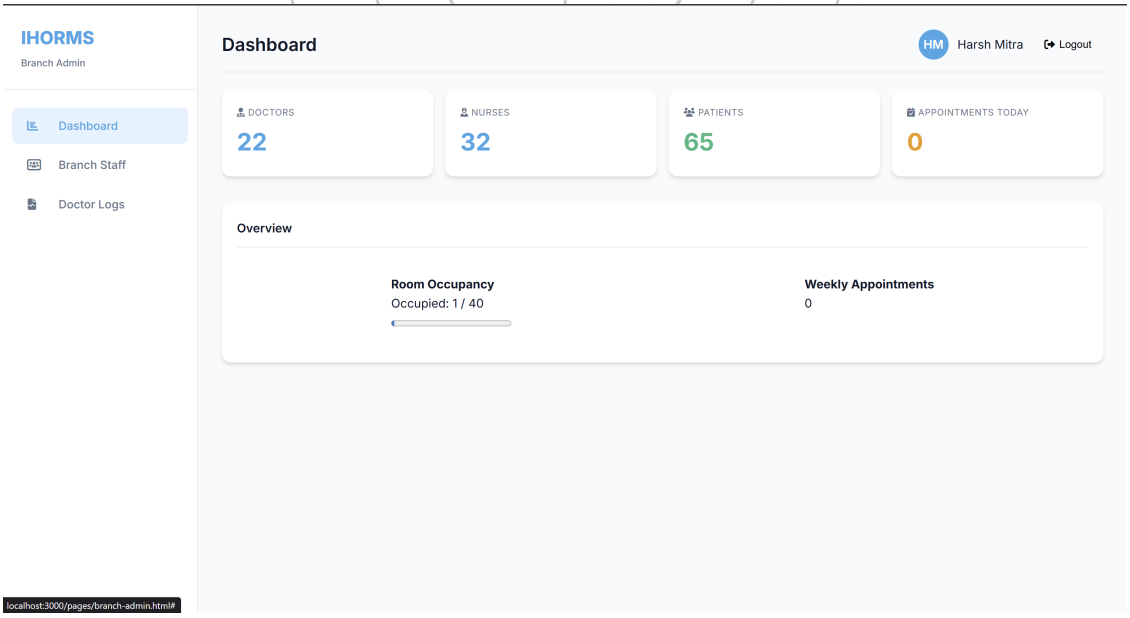

Figure 8.14: Branch Administrator Dashboard - Branch-level management interface for staff onboarding, resource allocation, and operational oversight with localized control while maintaining organization-wide data visibility.

8.9 Super Admin Portal Interfaces

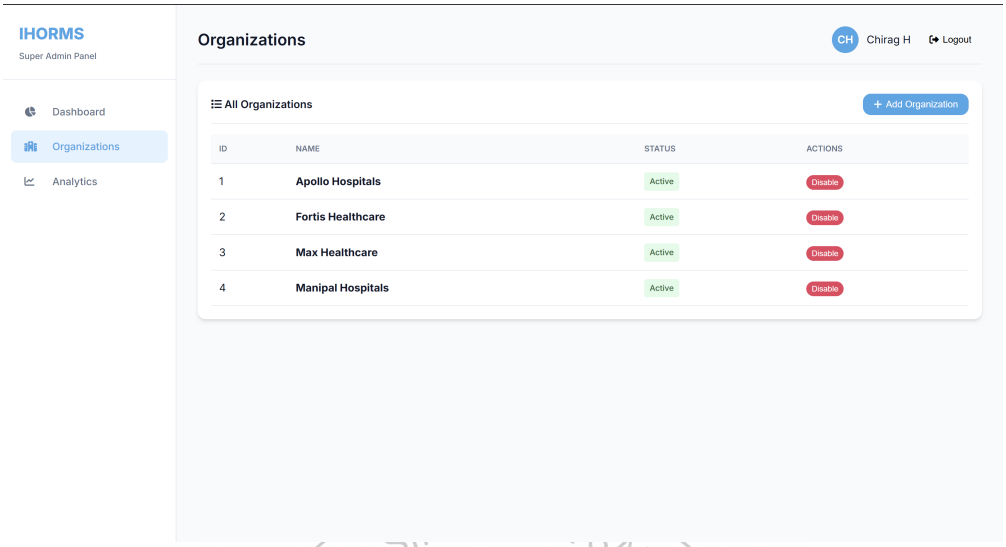

Figure 8.15: Super Admin Platform Management - Platform-level interface for managing multiple hospital organizations, viewing system-wide metrics, and configuring global settings for SaaS deployment scenarios.

8.10 Advanced Search Interfaces

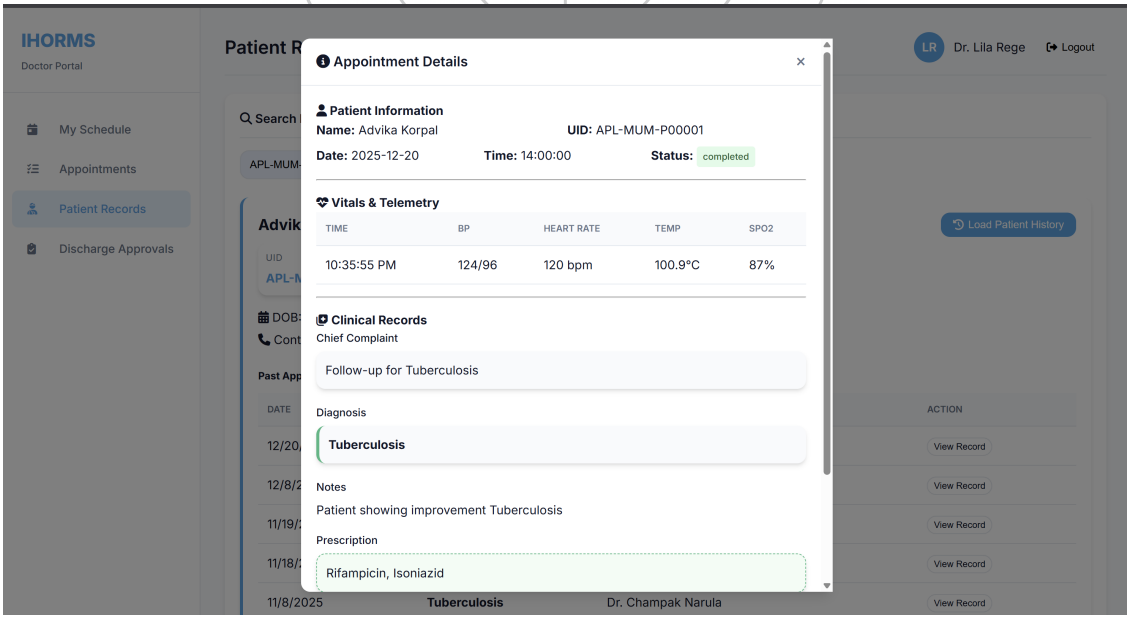

Figure 8.16: Patient UID Lookup – Simple and efficient search functionality allowing staff to retrieve patient records instantly using the unique Patient UID.