BedManager

Real-Time Hospital Bed and ICU Occupancy Management Dashboard

Team Number: 25

December 2, 2025

Team Number	25
Team Members	2025204042, 2025201050, 2025201031, 2025201025, 2025201057
Code Repository	`https://github.com/abhinavborah/bedmanager-team25`
Presentation	`https://docs.google.com/presentation/d/1PE5IcKpXsabehe0gSDZcuKYiT1-yDqzcjYFyImkzZbM/ed`
Demo	`https://youtu.be/PPvSJ_8TNek?si=T0oYBi_i2-URLEF6`

About Project

Anuradha is the ICU Manager at a large city hospital managing 40 ICU beds with 87% occupancy. She faces daily challenges with emergency admissions, bed allocation, and capacity planning due to fragmented information across departments. **BedManager** is our solution—a comprehensive real-time hospital bed and ICU occupancy management dashboard that transforms bed management through live monitoring, intelligent allocation, and ML-powered forecasting.

The system provides role-based dashboards for five stakeholders (Hospital Admin, ICU Manager, Ward Staff, ER Staff, Technical Team) with real-time WebSocket updates, emergency request workflows, cleaning management, and predictive analytics. When a critical patient arrives, the ER staff submits a request, Anuradha receives an instant alert, reviews available beds with equipment status, and allocates a bed—all within 2 minutes compared to 15-20 minutes with manual coordination.

Solution Diagram

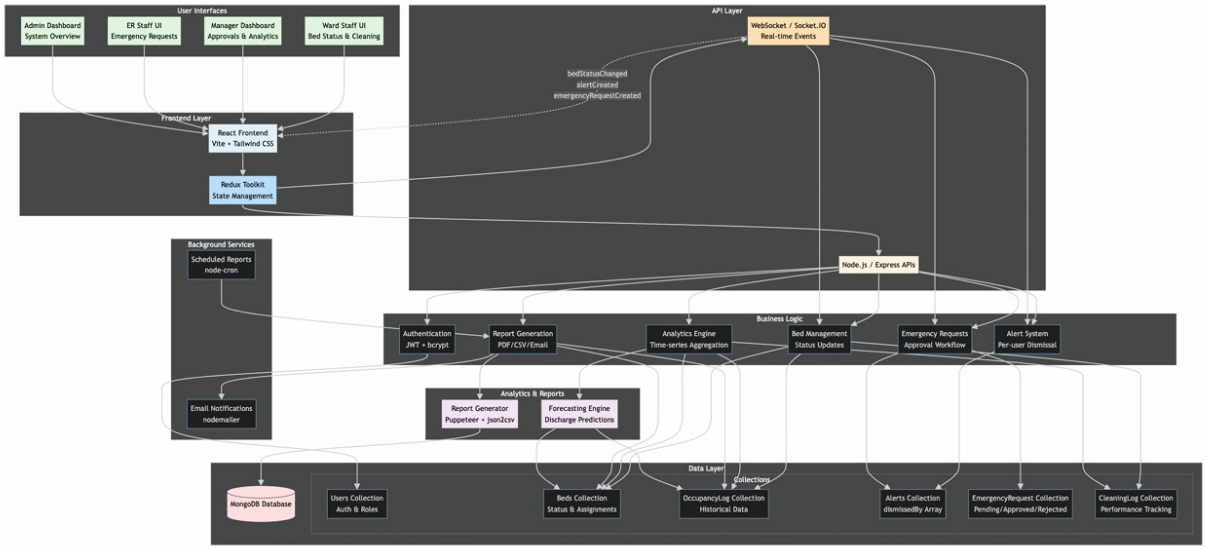

Figure 1: Complete BedManager System Architecture showing User Interfaces (Admin, ER Staff, Manager, Ward Staff Dashboards), Frontend Layer (React + Tailwind CSS with Redux Toolkit for state management), API Layer (WebSocket/Socket.IO for real-time events and Node.js/Express APIs), Background Services (Scheduled Reports via node-cron and Email Notifications via Nodemailer), Business Logic (Authentication with JWT+bcrypt, Report Generation in PDF/CSV/Email formats, Analytics Engine with time-series aggregation, Bed Management with status updates, Emergency Requests with approval workflow, and Alert System with per-user dismissal)

Persona and User Journey

Stakeholders

1. **Hospital Administrator**: Strategic decision-maker responsible for analyzing hospital-wide bed utilization metrics across all wards (Emergency, General, ICU), generating comprehensive reports for regulatory compliance and stakeholder communication, monitoring occupancy trends over configurable time periods (7/30/90 days), and making data-driven capacity planning decisions for resource allocation and expansion. Uses the Admin Dashboard with executive summary KPIs, ward utilization reports, trend analytics, and automated report scheduling features.

2. **ICU/Ward Manager (Anuradha)**: Primary operational user managing daily bed allocation within assigned ward. Responsibilities include monitoring real-time bed occupancy status, approving or rejecting emergency bed requests from ER staff based on availability and patient needs, coordinating with ward staff on cleaning schedules and maintenance, managing patient discharge timelines (both manual and ML-predicted), handling high-occupancy crisis situations, and using predictive analytics for capacity planning and staffing decisions. Uses Manager Dashboard with bed status grid, emergency request queue, forecasting panel, and nearby hospital coordination.

3. **Ward Staff (Nurses & Support Staff)**: Frontline healthcare workers responsible for hands-on bed management activities including updating bed status in real-time (available/cleaning/occupied), admitting patients to assigned beds with medical details entry, marking beds for cleaning after patient discharge, confirming cleaning completion through mobile-optimized workflows, reporting maintenance issues, and preparing beds for incoming emergency patients. Uses simplified Ward Staff Dashboard optimized for mobile devices with one-tap status updates and ward-filtered bed views.

4. **ER Staff (Emergency Department)**: Emergency response personnel who handle critical patient arrivals and bed allocation requests. Responsibilities include submitting emergency bed requests with detailed patient information (name, condition, priority level, equipment requirements), communicating urgency levels (Critical/High/Medium/Low) to managers, monitoring real-time request status (Pending/Approved/Rejected), checking bed availability across all wards before requesting, and coordinating patient transfer logistics from ER to allocated beds. Uses ER Staff Dashboard with quick request submission form, availability summary, and auto-refresh status tracker.

User Journey: ICU Manager Morning Shift

8:00 AM - Anuradha logs in, sees dashboard: 35/40 beds occupied (87%), 3 cleaning, 2 available
8:15 AM - Critical patient arrives in ER, staff submits emergency request via dashboard
8:15 AM - Anuradha receives instant WebSocket alert notification
8:17 AM - Reviews request details, sees Bed A5 available with ventilator equipment
8:18 AM - Approves request, allocates Bed A5 (system broadcasts update to all users)
8:25 AM - Ward staff admits patient to Bed A5 via mobile dashboard
10:30 AM - Patient in Bed B3 discharged, marked for cleaning (30-min timer starts)
11:00 AM - Cleaning completed, Bed B3 turns green on all dashboards
2:00 PM - Reviews ML forecasting: 8 beds expected available by 6:00 PM

Dashboard Screenshots

Hospital Administrator Dashboard

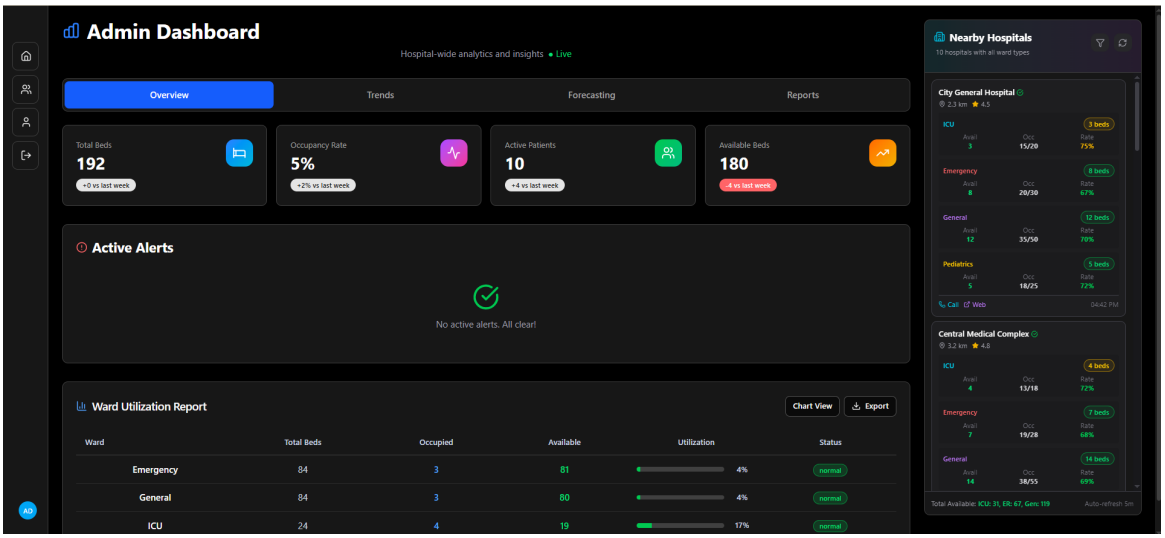

Figure 2: Admin Dashboard Overview showing KPI cards (192 total beds, 5% occupancy, 10 active patients, 180 available beds), Active Alerts monitoring panel for system notifications, and comprehensive Ward Utilization Report table displaying Emergency, General, and ICU wards with total beds, occupied counts, available beds, utilization percentages as progress bars, and real-time status indicators (normal/critical)

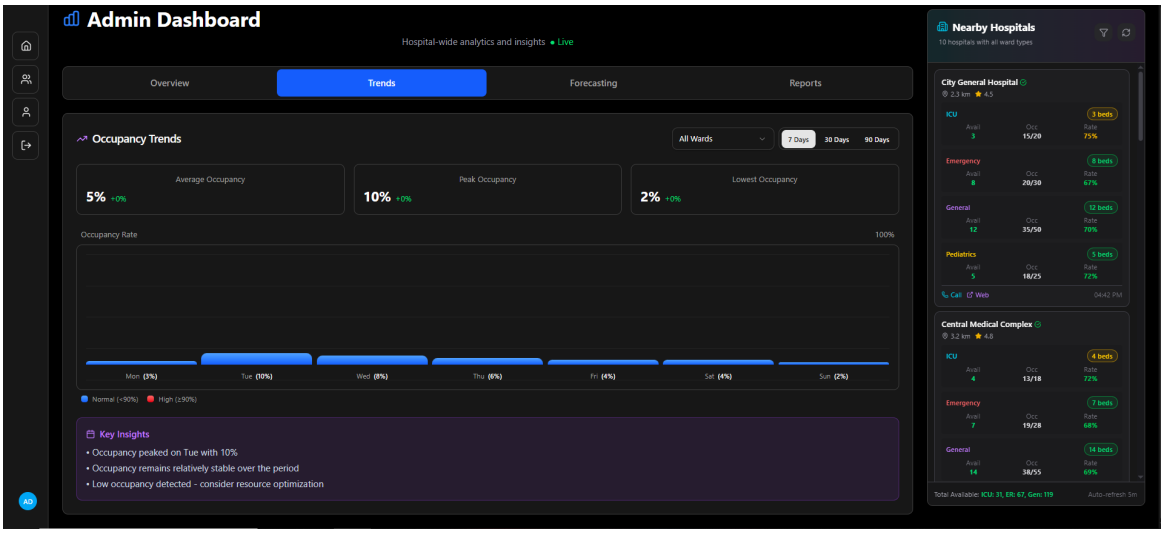

Figure 3: Occupancy Trends Analytics displaying time-series visualization with configurable periods (7, 30, or 90 days), interactive bar chart showing daily occupancy rates across all wards, summary cards revealing Average Occupancy (5%), Peak Occupancy (10%), and Lowest Occupancy (2%) with positive trend indicators, plus Key Insights panel highlighting peak timing, stability trends, and resource optimization opportunities

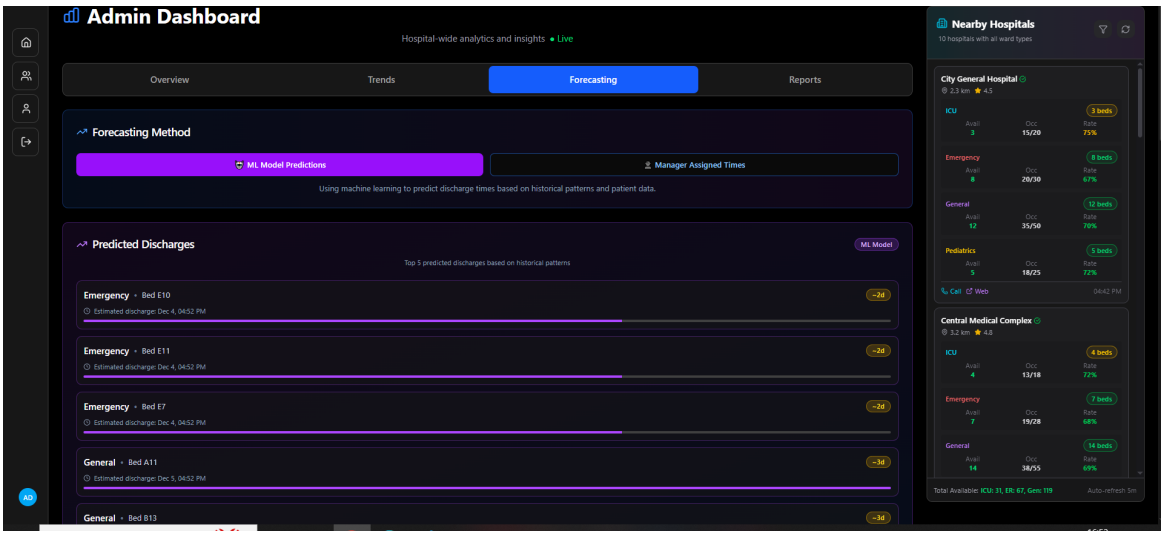

Figure 4: ML-Powered Discharge Predictions featuring top 5 predicted discharges generated by TensorFlow/Keras LSTM models, showing ward type (Emergency/General), specific bed numbers, estimated discharge dates and times, countdown timers in hours, and purple progress bars visualizing time remaining for proactive bed allocation planning

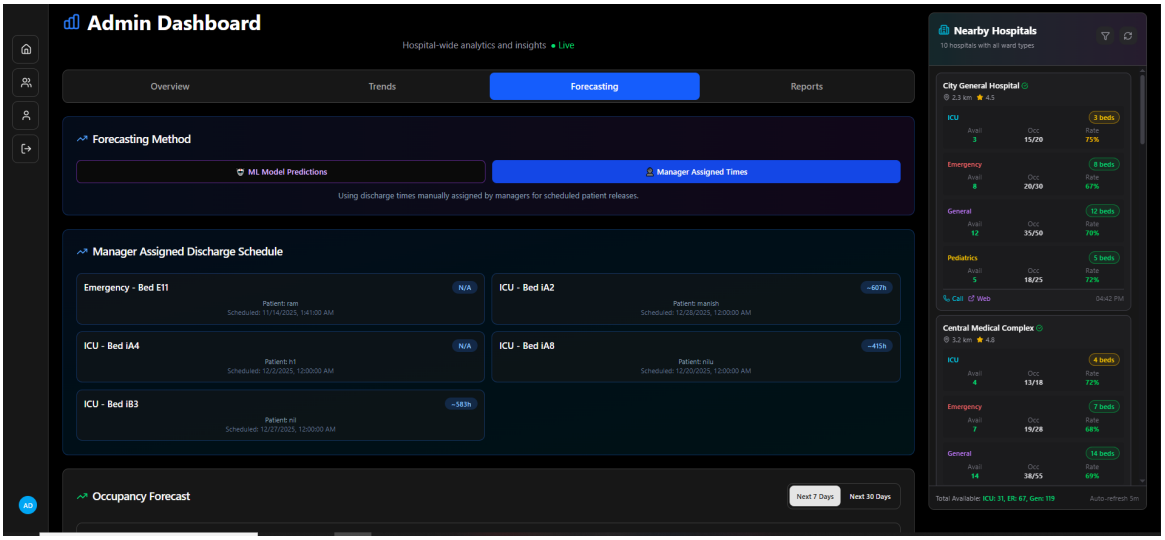

Figure 5: Manager-Assigned Discharge Schedule displaying manually set discharge times that override ML predictions, showing bed assignments (Emergency-E11, ICU-IA2, ICU-IA4, ICU-IA8, ICU-IB3), patient names (ram, marion, h1, nilu, nil), scheduled dates/times, countdown timers, toggle between ML Model Predictions and Manager Assigned Times modes, and Occupancy Forecast section for 7/30-day predictions with confidence percentages

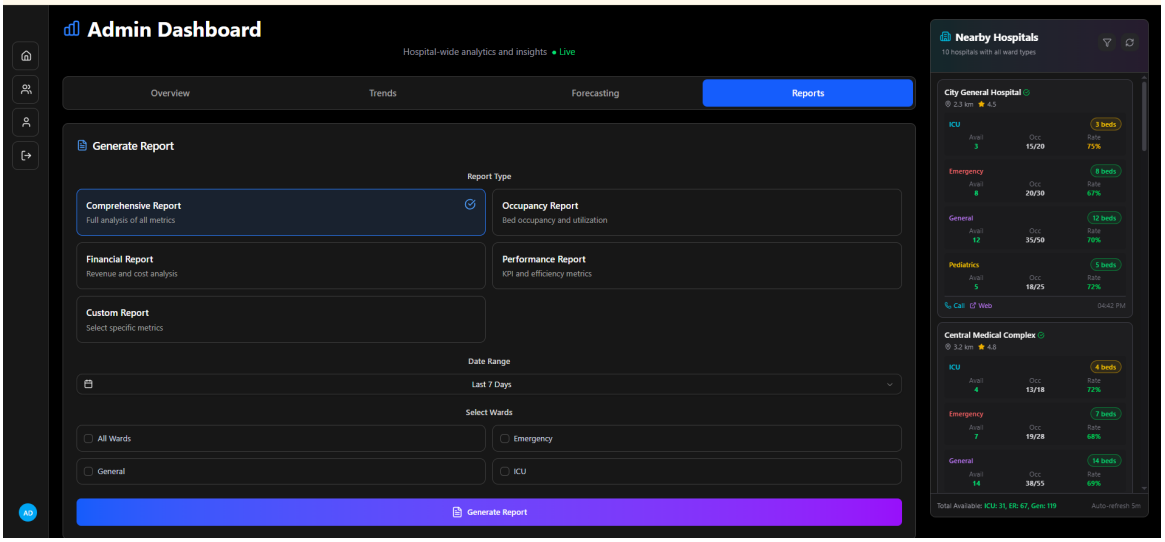

Figure 6: Comprehensive Report Generation Interface enabling administrators to create customizable analytics reports with five report type options (Comprehensive, Occupancy, Financial, Performance, Custom), date range selection (Last 7 Days dropdown), ward filtering (All Wards, Emergency, General, ICU checkboxes), and Generate Report button for downloadable PDF/CSV formats for stakeholder review and regulatory compliance documentation

Manager Dashboard (ICU/Ward Manager)

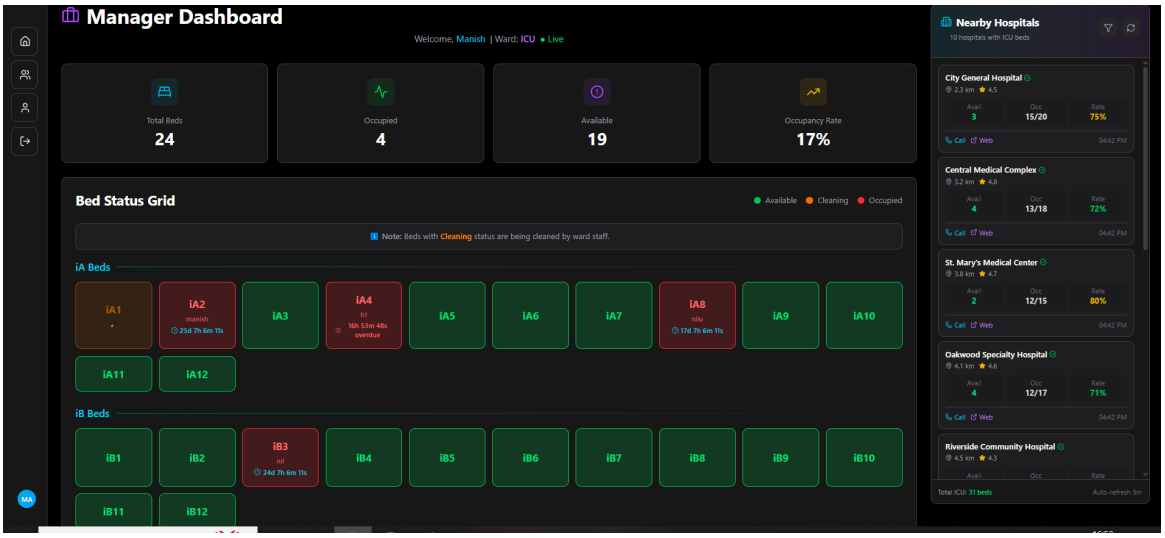

Figure 7: Manager Dashboard for ICU ward displaying four KPI summary cards (24 total beds, 4 occupied, 19 available, 17% occupancy rate), comprehensive Bed Status Grid with color-coded bed tiles (green=available, orange=cleaning, red=occupied with countdown timers, blue=occupied), real-time cleaning status notifications with estimated completion times, and Nearby Hospitals panel showing 10 regional hospitals with available ICU beds, distances, occupancy rates, and contact options for emergency overflow coordination

ER Staff Dashboard

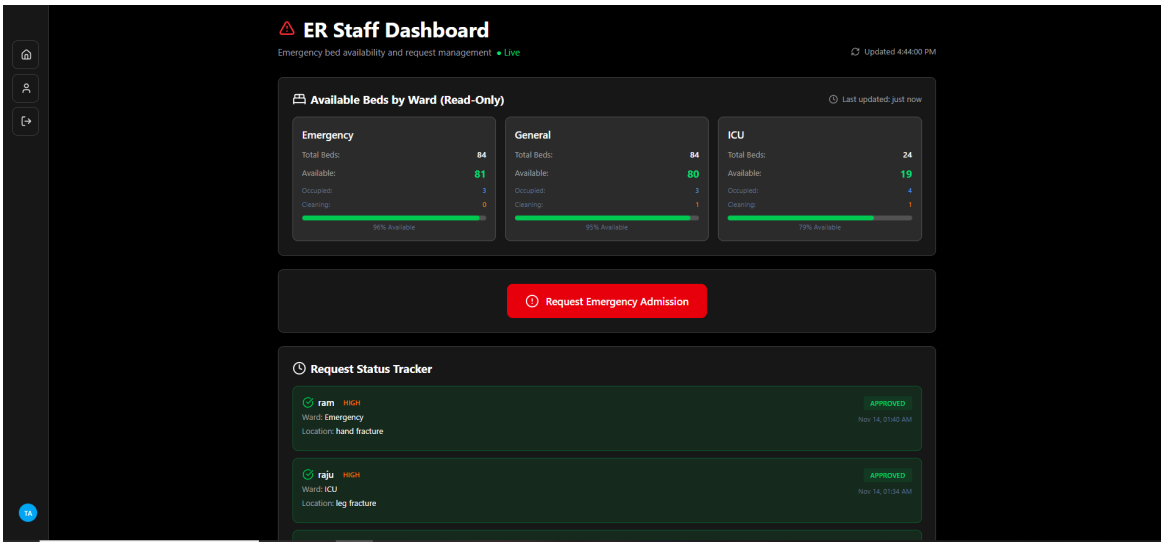

Figure 8: ER Staff Dashboard displaying Available Beds by Ward summary cards (Emergency: 81/84 available with 96% availability, General: 80/84 with 95%, ICU: 19/24 with 79%), prominent red Request Emergency Admission button for quick access during critical situations, and Request Status Tracker panel showing approved requests with patient names (ram, raju), HIGH priority badges, ward assignments (Emergency, ICU), admission reasons (hand fracture, leg fracture), and approval timestamps (Nov 14 timestamps) with auto-refresh every 10 seconds

Ward Staff Dashboard

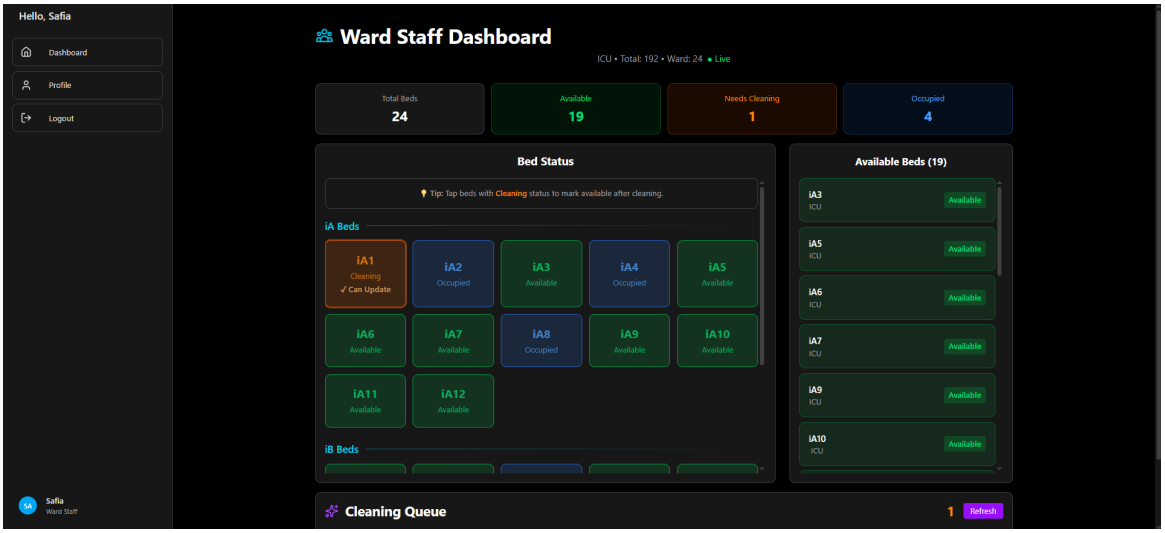

Figure 9: Ward Staff Dashboard (Safia - ICU Ward) with mobile-optimized interface showing four summary cards (24 total beds, 19 available in green, 1 needs cleaning in orange, 4 occupied in blue), interactive Bed Status grid with large touch-friendly tiles displaying bed IDs (iA1-iA12, iB1-iB12) and color-coded status (iA1=orange cleaning with "Can Update" prompt, iA2/iA4/iA8=blue occupied, rest=green available), Available Beds scrollable list showing 19 beds (iA3, iA5, iA6, iA7, iA9, iA10...) with quick-access Available buttons, and Cleaning Queue panel with 1 bed (iA1) requiring attention and Refresh button for status updates

Software/Hardware

Technology Stack

Frontend: React 19.1.1, Redux Toolkit 2.9.1, Vite, Tailwind CSS 4.1.14, Socket.IO Client 4.8.1, Axios 1.12.2, Framer Motion, Recharts

Backend: Node.js, Express.js 5.1.0, MongoDB 8.19.1, Mongoose 8.19.1, Socket.IO 4.8.1, JWT 9.0.2, Bcrypt 3.0.2, Multer, Nodemailer, Puppeteer 24.29.1
ML Service: Python 3.8+, FastAPI, TensorFlow/Keras, Scikit-learn, NumPy, Pandas, Uvicorn
Development: ESLint, Nodemon, dotenv, Git, Postman

Hardware Requirements

Server: Modern server with Node.js and Python (AWS EC2, Azure VM, or local)
Database: MongoDB instance (local or MongoDB Atlas)
Client: Modern browser (Chrome 90+, Firefox 88+, Safari 14+, Edge 90+)
Memory: 4GB RAM minimum (8GB recommended)
Storage: 20GB for application, database, logs
CPU: 2+ cores recommended for ML inference

Step-by-Step Hosting Instructions

Prerequisites: Node.js (v14+), MongoDB, Python (v3.8+), Git

Step 1: Clone Repository

```
git clone https://github.com/abhinavborah/bedmanager-team25
cd bedmanager-team25
```

Step 2: Backend Setup

```
cd backend
npm install
```

Create `.env` file:

```
MONGO_URI=mongodb://localhost:27017/bedmanager
JWT_SECRET=your-secret-key-here
PORT=5001
NODE_ENV=development
FRONTEND_URL=http://localhost:5173
```

Step 3: Frontend Setup

```
cd ../frontend
npm install
```

Create `.env` file:

```
VITE_API_BASE_URL=http://localhost:5001/api
```

Step 4: ML Service Setup

```
cd ../ml-service
pip install -r requirements.txt
```

Step 5: Start Services (3 terminals)

```
# Terminal 1 - Backend
cd backend
npm start

# Terminal 2 - Frontend
cd frontend
npm run dev

# Terminal 3 - ML Service
cd ml-service
python main.py
```

Step 6: Access Application
Open browser: `http://localhost:5173`

Contribution

Abhinav (2025204042): UI/UX specialist implementing comprehensive styling across all dashboards, fixing mobile responsiveness and layout issues, establishing consistent theme and navigation standards.

Shubham Kumar Sunny (2025201025): Backend architect setting up Node.js/Express server, MongoDB models (User, Bed, Logs), middleware (error handling, validation, role-based security), and Manager Dashboard with forecasting.

Surjit (2025201057): State management specialist implementing Redux Toolkit stores, Alert/Emergency Request models with Socket.io integration, ML prediction features, and role-based security guards.

Nilkanta Karak (2025201031): Real-time communication specialist implementing WebSocket (Socket.io) system with JWT authentication, broadcasting notifications, emergency request flow, and historical data endpoints.

Diganta Sen (2025201050): Analytics lead developing Analytics Controller with ML integration, occupancy trends (7/30/90-day), KPI logic, report generation (Puppeteer PDF, CSV), scheduled reports (node-cron), and MongoDB aggregation queries.

References

1. React.js Official Documentation - `https://react.dev`
2. Node.js Best Practices - `https://nodejs.org/en/docs`
3. MongoDB Schema Design - `https://www.mongodb.com/docs`
4. Socket.IO Real-Time Events - `https://socket.io/docs`
5. TensorFlow/Keras Documentation - `https://www.tensorflow.org`
6. JWT Authentication Guide - `https://jwt.io/introduction`
7. Hospital Bed Management Research Papers (IEEE Xplore)
8. Healthcare IT Security Standards (HIPAA Compliance)