INTRODUCTION

Hospital Data Management System is a unified platform that streamlines key hospital operations, including blood bank tracking, emergency alert handling, drug issuance, and patient queue management. It features a centralized dashboard with real-time updates and supports hospital-wide display integration for broadcasting token queues and critical alerts, improving coordination and patient care.

KEY MODULES

🩸 Blood Bank Module

This module tracks the real-time inventory of all blood types, ensures visibility across departments, and provides instant alerts when critical thresholds are breached—enabling better planning, quicker responses, and reduced wastage.

🚨 Emergency Request Handler

Captures emergency situations (e.g., Code Blue/Red) raised by hospital departments and prioritizes fulfillment based on urgency. It ensures faster intra-hospital communication and improves critical care response times.

💊 Drug Inventory Management

Maintains a log of drug distributions per department, supports inventory checks, and prevents misuse or shortages by enabling real-time updates and stock level validations.

🏥 Hospital Display System

Connects 72+ display screens to a centralized system, broadcasting real-time token queues, emergency alerts, and OT/consultation schedules. Enhances operational visibility for patients, staff, and administrators across the hospital.

🛌 Operation Theatre (OT) Management System  
Allows departments to book and manage OT slots for surgeries, avoiding scheduling conflicts and ensuring optimal utilization of surgical infrastructure.

**How to Set Up the Project**

Follow these steps to set up and run the hospital dashboard system on your local machine.

**• Clone the Repository**

– Run these commands to clone the project:

* https://github.com/YENCS-T8/YENCS-HDMS.git

**• Create a Virtual Environment**

– Set up a virtual environment for dependencies:

* python -m venv venv

– Activate it:

◦ Windows: venv\Scripts\activate

◦ macOS/Linux: source venv/bin/activate

**• Install Dependencies**

– Install required packages:

* pip install -r requirements.txt

**• Set Up the Database**

– Ensure MySQL is installed and running.

– Create a database : hospital\_db

**–– Configure Environment Variables**

– Create a .env file or edit config.py with your database details:

DB\_HOST=localhost

DB\_USER=your\_username

DB\_PASSWORD=your\_password

DB\_NAME=hospital\_db

**– Run the Application**

–Start the Flask app:

* python main.py

– Access it at: <http://localhost:5000>

– Use Application Modules

– Once running, you can:

◦ Track blood inventory

◦ Manage emergency requests

◦ Issue and monitor drugs

◦ Handle patient queues and OT schedules

◦ Display hospital updates on screens

– **Set Up Display System**

– For hospital TV screens:

1. Run the Display Server

On the admin system (the main control system connected to all screens):

* python screen.py

1. Make sure you have installed the required packages:

* pip install flask flask-socketio pillow mss

1. Connect Display Screens

Connect each hospital TV screen to the same local network (Wi-Fi or LAN) as the admin system.

On each display screen, open a browser and navigate to:

For Monitor 1 : http://<admin-ip>:5000/viewer/monitor1

For Monitor 2 : http://<admin-ip>:5000/viewer/monitor2

For Monitor 3 : http://<admin-ip>:5000/viewer/monitor3

Same for all 70 Devices

1. Manage Display Content

* The admin can start/stop casting to each monitor via the control panel (<http://localhost:5000>).
* Displays can show desktop streams or custom visuals (e.g., token queues, emergency alerts, OT schedules).

Notes

• The backend uses Flask.

• Database queries use SQLAlchemy or raw SQL.

• Display screens fetch real-time token queues and OT updates.

