# Hospital Management System with IoT and MERN Stack

## Introduction

In modern healthcare, efficient patient monitoring is crucial for timely medical interventions. This project aims to develop a **real-time hospital management system** that integrates **IoT (NodeMCU) with the MERN stack** to monitor patient health parameters. The system ensures **instant alerts** if any critical condition is detected, thereby enhancing patient safety and improving hospital efficiency.

## System Overview

The system consists of three major components: 1. **IoT-Based Data Collection** 2. **MERN Stack Backend and Database** 3. **Real-Time Dashboard with Alerts**

## Technology Stack

- **Hardware:** NodeMCU, Sensors (Temperature, Heart Rate, SpO2, BP, etc.), Buzzer, LED, LCD Screen
- **Software:** MERN Stack (MongoDB, Express.js, React.js, Node.js), WebSockets (Socket.IO)
- **Communication Protocol:** Wi-Fi (NodeMCU to Server)

## Workflow

1. **Data Collection:**

   - Sensors attached to the patient collect vital parameters.
   - The NodeMCU microcontroller reads the sensor data and sends it to the backend server using **HTTP requests** or **WebSockets**.

2. **Data Processing & Storage:**

   - The server (Node.js + Express.js) receives the data and stores it in **MongoDB**.
   - The backend also analyzes the data to detect abnormal readings.

3. **Dashboard & Alerts:**

   - A **ejs** web dashboard displays real-time patient data.
   - If a parameter exceeds a set threshold, the system triggers:
   - **Buzzer & LED Alert** on the NodeMCU device
   - **LCD Color Change** for visual alert
   - **Dashboard Notifications** for hospital staff

## Implementation Details

### 1. IoT Hardware Setup

- **Sensors Used:**
- **DHT11/DHT22** - Temperature & Humidity
- **MAX30100/MAX30102** - Heart Rate & SpO2
- **BMP180** - Blood Pressure
- **Microcontroller:** NodeMCU (ESP8266)
- **Actuators:** Buzzer, LED, LCD Screen
- **Data Transmission:** WebSockets via Wi-Fi

**2. Backend & Database**

- **Server:** Node.js with Express.js
- **Database:** MongoDB (stores patient data with timestamps)
- **Real-Time Updates:** Implemented using **Socket.IO** for bidirectional communication between the server and client.
- **Alert Logic:** Threshold-based system detects critical values and triggers alerts.

**3. Frontend & Dashboard**

- **Built Using**: ejs + WebSockets
- **Key Features**:
- Live Patient Data Display
- Alert Notifications for Critical Values
- Historical Data View for Trends & Reports

## Key Features & Benefits

- **Real-Time Monitoring**: Ensures quick response to critical conditions.
- **Automated Alerts**: Notifies medical staff instantly when needed.
- **Scalability**: Can support multiple patients & sensors.
- **Remote Access**: Hospital staff can monitor patients from anywhere.
- **Data Storage & Analysis**: Helps in tracking patient history and trends.

## Future Enhancements

- **Integration with AI**: Predictive analytics for early disease detection.
- **Mobile App Support**: Accessibility for doctors on smartphones.
- **Cloud Storage**: Secure access to patient data from any location.

## Conclusion

This **IoT-powered hospital management system** provides a smart, real-time solution for patient monitoring. By leveraging **MERN stack** and **WebSockets**, it ensures seamless communication between sensors, the server, and hospital staff, leading to improved patient care and efficiency.