ParamediX - Intelligent Emergency Response:
A Smart Ambulance System for Seamless Hospital Connectivity

BECE204L

Microprocessors and Microcontrollers

Bachelor of Technology

in

Electronics and Communication

By

S Nazeer Ahammed (23BEC0205)

B Raghava Reddy (23BEC0254)

B V Amogh Satwik (23BEC0346)

VIT

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

April,2025

Table of Contents

Abstract

In emergency medical services, the "**golden hour**" is crucial for patient survival, emphasizing the need for rapid communication and informed decision-making. **ParamediX** is an IoT-based real-time ambulance-to-hospital communication system designed to enhance pre-hospital care by enabling seamless monitoring and transmission of patient vitals and location data during transit. The system utilizes the **ESP32** microcontroller integrated with biomedical and geolocation sensors, including the **MAX30100** (pulse rate and SpO_2), **DS18B20** (body temperature), and **NEO-6M** GPS module. This hardware framework allows continuous data collection and wireless transmission to remote hospital dashboards.

ParamediX is implemented using two distinct yet complementary approaches. The first employs the **Blynk IoT platform** to provide a lightweight, mobile-accessible dashboard for rapid deployment and scalable field use. The second approach involves a **custom-developed web application** built using HTML, CSS, JavaScript, Node.js, Express.js, EJS, and MySQL. This platform supports secure login for medical personnel, patient biodata management, real-time GPS tracking via OpenStreet Maps integration, and live video conferencing between ambulance paramedics and hospital staff.

By facilitating early preparation of medical resources and pre-arrival diagnosis, ParamediX reduces delays in treatment initiation and improves the efficiency of emergency response. The system is scalable, modular, and adaptable, forming a foundation for future enhancements such as AI-based triage, machine learning-driven route optimization, and integration with hospital information systems. ParamediX represents a significant step toward intelligent, responsive, and connected emergency healthcare infrastructure.

Introduction

Emergency medical services (**EMS**) operate in a high-stakes environment where every second counts. One of the most critical concepts in this domain is the "golden hour"—the narrow window of time following a traumatic event in which prompt medical attention can dramatically increase a patient's chance of survival. While hospitals have made significant strides in diagnostics and treatment, a persistent bottleneck remains: the absence of real-time communication and patient monitoring capabilities during ambulance transit.

The typical ambulance-to-hospital communication model relies heavily on voice calls or incomplete pre-arrival notes, offering minimal insights into the patient's evolving condition. This lack of continuity often results in delayed medical intervention, inefficient resource mobilization, and a reactive rather than proactive approach to emergency care. In a time when almost every industry is being transformed by smart technologies and real-time data, pre-hospital care continues to rely on outdated, manual systems.

Meanwhile, the proliferation of IoT devices and cloud-based platforms has reshaped modern healthcare, from wearable fitness monitors to telehealth consultations. However, the integration of such technologies into EMS is still at a nascent stage. There exists a pressing need to bridge this gap by designing solutions that are not only technologically capable but also scalable, user-friendly, and cost-effective.

ParamediX is conceived within this context—not just as another IoT project, but as a response to a system-wide challenge in emergency healthcare. Rather than focusing solely on post-arrival care, ParamediX brings intelligence and connectivity into the critical "**en route**" phase. What sets this system apart is not just its ability to collect data, but its architectural flexibility—catering to both quick-deploy solutions via mobile platforms, and comprehensive, enterprise-level deployments with full-stack dashboards.

In conceptualizing ParamediX, we were driven by real-world emergency scenarios: rural ambulances with limited connectivity, urban hospitals struggling to triage during peak loads, and medical staff left guessing until the ambulance arrives. Through a modular and scalable design, ParamediX aims to empower paramedics, inform hospital teams, and ultimately shift emergency response from being reactive to data-informed and anticipatory.

This paper explores the system's architecture, hardware-software integration, implementation strategies, and broader implications for smart EMS. It further discusses how such systems can be extended in the future to incorporate artificial intelligence, predictive analytics, and smart city infrastructure—

signalling a **paradigm shift** in how we think about and manage pre-hospital medical emergencies.

Motivation

India's emergency healthcare infrastructure is challenged by a critical disconnect between pre-hospital care providers and hospitals. While the nation has seen progress in digital health, especially under initiatives like **Ayushman Bharat Digital Mission (ABDM)**, the **integration of ambulance-based patient monitoring systems with hospital networks remains underdeveloped and largely unregulated**.

In most parts of the country, ambulances serve merely as transport vehicles with little or no ability to transmit **real-time patient vitals** or **GPS data**. Communication between paramedics and hospital staff is typically manual and unstructured, often relying on phone calls, which are subject to human error, network availability, and delays. In emergencies, where every second counts—the so-called **"golden hour"**—this lag in communication can be fatal. Unfortunately, **India lacks any regulatory mandate** requiring ambulances to be equipped with real-time monitoring, telemedicine, or data-sharing capabilities.

Several regional and institutional efforts have been made to address this issue:

- **Gujarat's 108 Emergency Services**, managed by GVK EMRI, have piloted initiatives to install GPS tracking and basic patient data capture in ambulances. However, these are still limited in scope and lack end-to-end hospital integration.
- The **All India Institute of Medical Sciences (AIIMS), New Delhi**, initiated a **pilot IoT-based ambulance connectivity project**, using real-time telemetry and dashboards for critical patient data. While promising, this was limited to select ambulances and has yet to be adopted on a national scale.
- **Apollo Hospitals** has been a pioneer in **telemedicine and smart ambulance deployment**, offering video consultations during transit and data transmission to emergency rooms. However, such systems remain expensive and limited to urban, private healthcare setups.

- Other states like **Tamil Nadu** and **Kerala** have experimented with ambulance tracking and hospital alert systems through centralized control rooms, but lack robust integration of **biomedical sensor data** with hospital EHRs or decision-support systems.

Despite these initiatives, there is no **centralized protocol** or **national health standard** that mandates a uniform digital infrastructure for ambulances across government and private sectors. Many ambulances, especially in **rural and tier-2 cities**, still operate without essential equipment, let alone digital connectivity. Furthermore, interoperability between hospitals, EMS providers, and cloud-based systems is virtually non-existent.

The **ParamediX** project was motivated by this fragmented ecosystem. Recognizing the need for a **cost-effective, scalable, and IoT-driven solution**, we set out to develop a system that can operate across diverse infrastructure levels—from rural health centres to super-specialty hospitals. By using readily available hardware like the **ESP32**, and integrating **sensors such as the MAX30100 (SpO2 & pulse), DS18B20 (temperature), and NEO-6M GPS**, the system enables ambulances to transmit patient vitals and location data in real time to hospital dashboards.

What sets ParamediX apart is its **dual-deployment architecture**—supporting both the **Blynk platform for quick visualization and mobile use**, and a **custom-built web dashboard** for hospital systems, complete with secure login, biodata integration, Google Maps, and live video support. This makes it uniquely adaptable for varying levels of healthcare readiness.

Through this project, we aim not only to provide a technological solution but to contribute toward establishing a **blueprint for standardized, intelligent EMS communication systems** in India. ParamediX serves as a prototype that could inspire future integrations with **AI-based triage**, **route optimization**, and **real-time EHR systems**, moving India closer to a truly connected emergency care infrastructure.

Literature Survey

India's healthcare system has undergone significant transformation in recent years, yet emergency medical services (EMS) continue to face numerous challenges, especially in terms of coordination, accessibility, and technological integration. In a country with vast geographical diversity, uneven infrastructure, and varying levels of digital adoption, the ambulance-to-hospital communication ecosystem remains largely fragmented.

A. Current State of EMS in India

According to the **National Health Profile 2021** and reports from the **Ministry of Health and Family Welfare**, there is a growing recognition of the importance of EMS in reducing pre-hospital mortality. However, studies [1], [2] have highlighted that **only 30–40% of ambulances in India are equipped with Advanced Life Support (ALS)** capabilities. In rural and semi-urban regions, ambulances often serve as mere transport vehicles, lacking real-time monitoring systems or trained personnel to manage critical cases.

Moreover, communication between paramedics and hospital staff is typically limited to verbal updates via mobile phones, without the transmission of vital physiological data [3]. This leads to uncoordinated handovers, diagnostic delays, and compromised patient outcomes—issues that are especially pronounced during mass casualty incidents or in high-traffic urban areas.

B. Technological Interventions in Indian Healthcare

India has seen a rise in **eHealth** and **mHealth** solutions, particularly since the rollout of the **Ayushman Bharat Digital Mission (ABDM)**, which aims to digitize health records and promote telemedicine [4]. However, while such programs have improved access to healthcare services, there has been limited application in the domain of real-time emergency response.

Studies such as [5] and [6] have explored the use of **IoT-enabled devices for remote health monitoring**, primarily focusing on chronic disease management and elderly care. Projects involving the **MAX30100** and **DS18B20** sensors for monitoring SpO_2, pulse rate, and body temperature have shown reliable performance in both lab and field trials [7], [8]. Nevertheless, few efforts have been made to integrate these technologies into moving ambulances or to design systems tailored to India's diverse ambulance infrastructure.

C. Need for Real-Time Ambulance-to-Hospital Communication

Research presented in [9] and [10] highlights the critical need for **real-time patient telemetry and location tracking** during emergencies. Hospitals with access to pre-arrival data are significantly better prepared to triage and mobilize resources, leading to faster intervention. Despite this, such systems are mostly found in developed countries and are often cost-prohibitive or technically demanding to deploy in the Indian context.

In India, platforms like **108 Ambulance Services** (run by GVK EMRI) have laid the groundwork for centralized EMS coordination [11]. However, the integration of real-time vitals, GPS data, and digital dashboards remains limited. This technological gap presents an opportunity for low-cost, IoT-driven solutions tailored to Indian settings.

D. Emerging IoT-Based EMS Solutions

Recent research efforts have begun to explore low-power microcontrollers such as the **ESP32**, which support both sensor integration and wireless transmission. In [12], an ESP32-based system was successfully deployed for home health monitoring, while [13] demonstrated its use in telehealth diagnostics. These solutions point to the potential of microcontroller-based systems in building scalable EMS communication frameworks.

Additionally, cloud-integrated dashboards—developed using **Node.js**, **MySQL**, and **JavaScript**—have shown promise in healthcare analytics and patient management [14]. However, the full-stack implementation of such systems specifically designed for ambulance-hospital workflows in India is still underexplored, thereby motivating the development of comprehensive frameworks like **ParamediX**.

Proposed Work

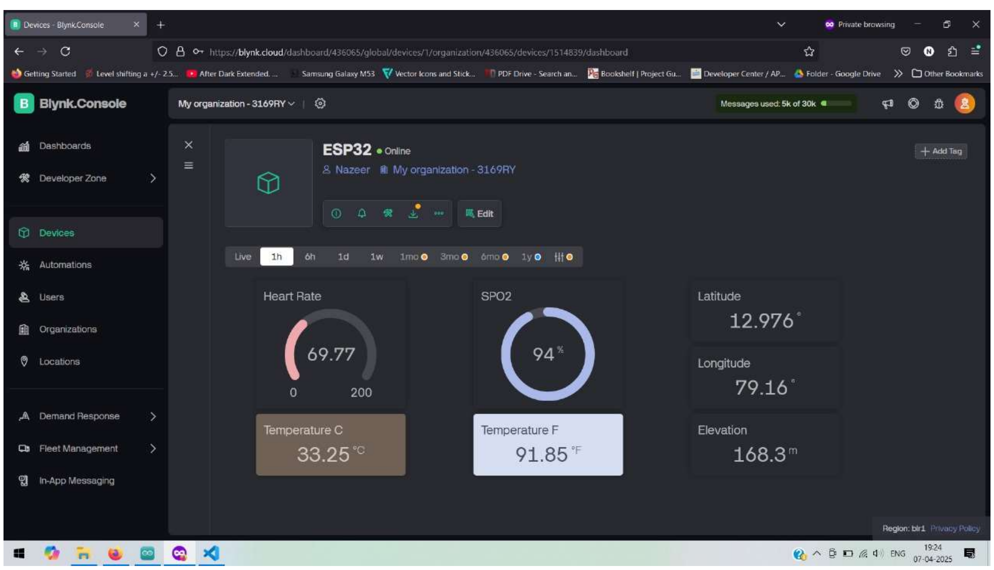
The proposed system, **ParamediX**, is designed to enable real-time communication between ambulances and hospitals to bridge the critical gap in pre-hospital emergency care in India. The architecture consists of two major components:

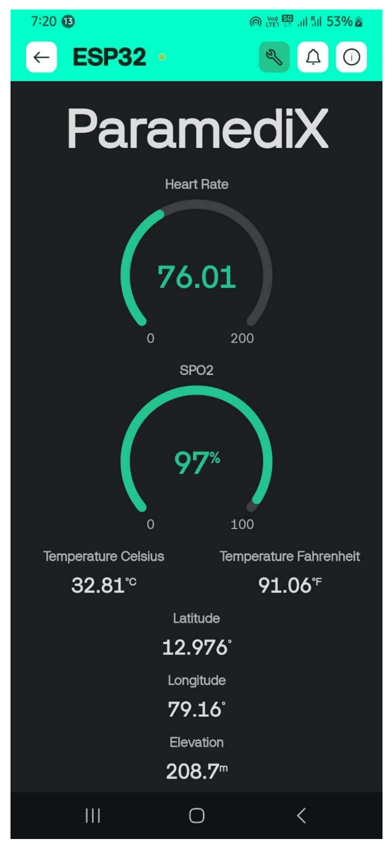
1. **Ambulance Node**
2. **Hospital Node**

Each node is developed to handle specific tasks, facilitate data acquisition and visualization, and maintain low-latency communication between first responders and hospital staff. The system is built using two parallel approaches:

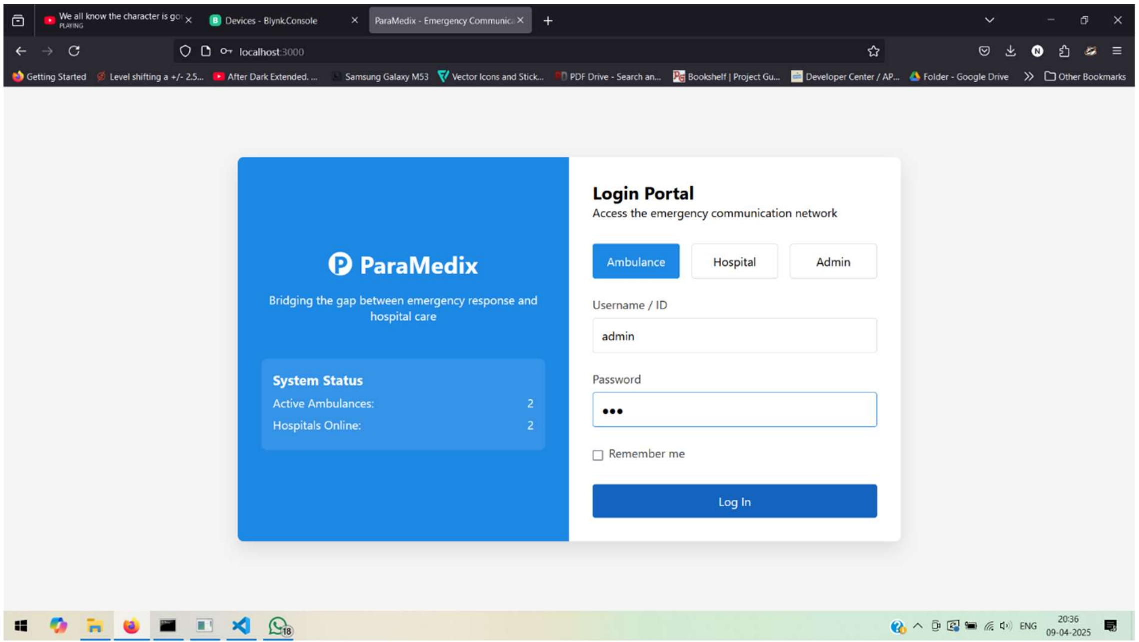
(1) A rapid-deployment solution using the **Blynk IoT platform**,

(2) A custom-built **web dashboard-based system** using modern web development technologies and database integration.





Blynk IoT App and Web Interface



Custom Web- Dashboard

A. Ambulance Node

The ambulance node is the frontline data acquisition unit deployed in emergency vehicles. This node is responsible for interfacing with biomedical sensors, packaging patient data, and transmitting it in real-time to the hospital node.

1) Hardware Architecture

The core controller of the ambulance node is the **ESP32 microcontroller**, chosen for its built-in Wi-Fi and Bluetooth capabilities, power efficiency, and dual-core processing. The ESP32 interfaces with:

- **MAX30100** sensor for pulse rate and SpO_2 monitoring
- **DS18B20** digital temperature sensor for core body temperature
- **NEO-6M GPS** module for live location tracking

These sensors send data serially to a host computer. In a production-grade version of this system, this host would be replaced by a compact **Single Board Computer (SBC)** such as a **Raspberry Pi 4**, capable of handling serial and Bluetooth data from old and new biomedical devices.

An **LED indicator** is also attached to the ESP32 to reflect network connectivity status through blinking patterns. Currently, the ESP32 connects via a **mobile hotspot**, but in real-world scenarios, this could be replaced with **GSM**, **LoRaWAN**, or other communication protocols, depending on network strength and availability.

Additionally, a display unit is attached to the SBC for local visualization of patient data, enabling real-time feedback to the ambulance crew.

2) Software Architecture – Approach 1: Blynk IoT Platform

To demonstrate proof of scalability and accessibility, the first implementation uses the **Blynk platform**, a cloud-based IoT dashboard that supports both web and mobile interfaces.

In this setup:

- The **ESP32** transmits sensor data using **virtual pins** to Blynk's cloud servers.
- The **Blynk mobile app** and **web dashboard** simultaneously display vitals, offering a synchronized view to both hospital and ambulance teams.
- Data visualization is enabled through real-time widgets (gauges, graphs, maps), providing a highly intuitive UI.

While Blynk offers rapid deployment and cross-platform compatibility, it lacks centralized control, user login security, and role-based access. It serves primarily as a **proof-of-concept** for real-time cloud synchronization between multiple endpoints.

B. Hospital Node

The hospital node consists of a centralized web portal developed using **HTML, CSS, JavaScript, Node.js, Express.js**, and **MySQL** for backend data management. The system utilizes **EJS** for server-side rendering, **PeerJS** for WebRTC-based video conferencing, and **OpenStreetMap APIs** for location tracking and ETA estimation.

1) Software Architecture – Approach 2: Custom Web Dashboard

This second implementation addresses the limitations of the Blynk-based setup by offering a fully **customizable**, **secure**, and **multi-user web application**. The web app is divided into three functional views:

- **Admin View**: Controlled by a central authority (e.g., health department), this portal manages authentication, assigns ambulance and hospital permissions, and oversees system operations.
- **Ambulance View**: After patient pickup, the ambulance crew logs in and enters basic patient data, selects the nearest hospital based on treatment availability, and initiates data transmission. The interface provides:
    - Real-time dashboard displaying vitals from connected sensors
    - Live video conferencing with hospital staff
    - Dynamic ETA tracking using GPS coordinates
    - Option to switch between hospitals depending on specialty or availability
- **Hospital View**: Hospital staff can monitor a live feed of incoming ambulances. Upon selecting an ambulance label:
    - Vital statistics and GPS-based ETA are displayed
    - Preliminary prognosis and symptoms are visible
    - A live video link allows remote consultation and early intervention
    - Doctors can be pre-assigned based on condition

This approach ensures a **closed-loop emergency response system** where hospitals are pre-alerted, patients are remotely monitored, and triage decisions can begin before the ambulance reaches the facility.

2) Network and Security Considerations

The platform is **locally hosted**, ensuring control and data integrity within the institution. Future upgrades may involve:

- Encrypted data transfer via HTTPS
- Secure logins with OTP or token-based authentication

- Integration with national health IDs and hospital EHRs

System Architecture

The architecture of **ParamediX** is designed as a modular and scalable IoT-based framework that facilitates seamless communication between ambulances and hospitals. It is divided into two primary nodes: the **Ambulance Node** and the **Hospital Node**, interconnected via cloud or local web infrastructure.

1. Ambulance Node

The ambulance node is responsible for collecting, processing, and transmitting patient health data and location in real-time.

- **Sensor Layer**: Includes the MAX30100 (Pulse & SpO_2), DS18B20 (Body Temperature), and NEO-6M GPS. These sensors interface with the ESP32 microcontroller via I2C and 1-Wire protocols.
- **Microcontroller Layer**: The ESP32 reads sensor data, formats it, and transmits it via Wi-Fi. It also handles connection status and visual feedback through a status LED.
- **Connectivity Layer**:
    - *Approach A*: Sends data to **Blynk Cloud**, enabling visualization on mobile and web dashboards using virtual pins.
    - *Approach B*: Sends data to a **custom server** via HTTP or WebSocket, supporting role-based access and data logging.
- **Display Interface (optional)**: A small onboard display (OLED) can visualize vitals directly for ambulance personnel.
- **Local Controller (SBC)**: In real deployments, a Single Board Computer (e.g., Raspberry Pi 4) acts as the edge device, collecting serial/Bluetooth data from both legacy and modern medical equipment.

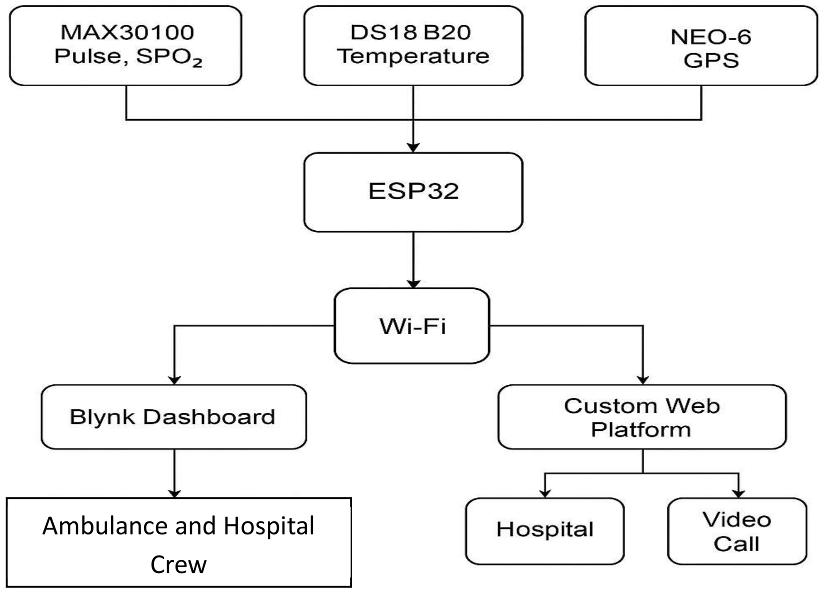
2. Hospital Node

The hospital node provides healthcare providers with a live overview of incoming patients and ambulances.

- **Data Reception & Visualization**:
  - In the **Blynk System**, staff access vitals via mobile/web dashboards and track location through Blynk Map widgets.
  - In the **Custom Web App**, a secure login system enables access to live dashboards, patient data, and video conferencing.
- **Functional Views**:
  - *Ambulance Dashboard*: Allows entry of patient biodata, hospital selection based on treatment availability, and viewing of vitals, ETA, and video chat.
  - *Hospital Dashboard*: Lists all active ambulances, with options to view vitals, track ETA, and initiate early treatment preparation.
  - *Admin Dashboard*: Central authority can register ambulances/hospitals, approve access, and oversee deployment.

3. Communication Layer

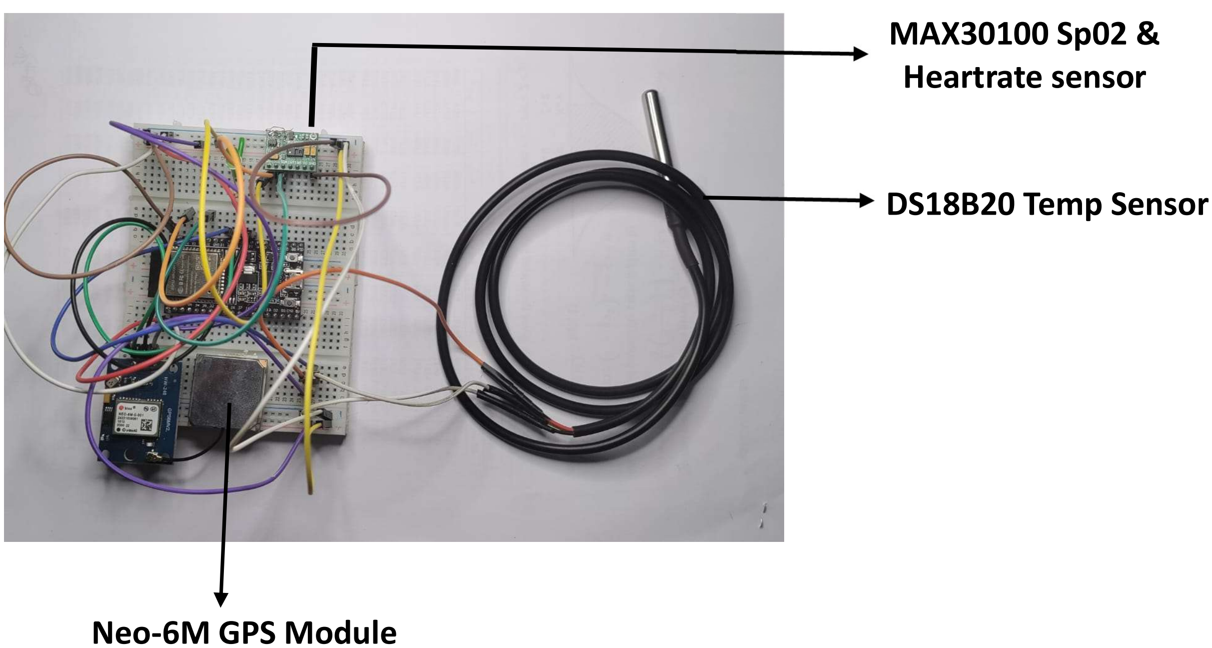
- **Blynk Architecture**: ESP32 uses the Blynk library to send sensor values as virtual pin data to the Blynk Cloud, which is then displayed on a custom-designed mobile and web interface.
- **Custom Web Architecture**: Built using Node.js, Express.js, EJS, and MySQL. It supports REST APIs, real-time WebSocket updates, PeerJS for video calling, and OpenStreetMap for location tracking.
- **Security & Access Control**: Role-based authentication ensures only authorized personnel access sensitive data.

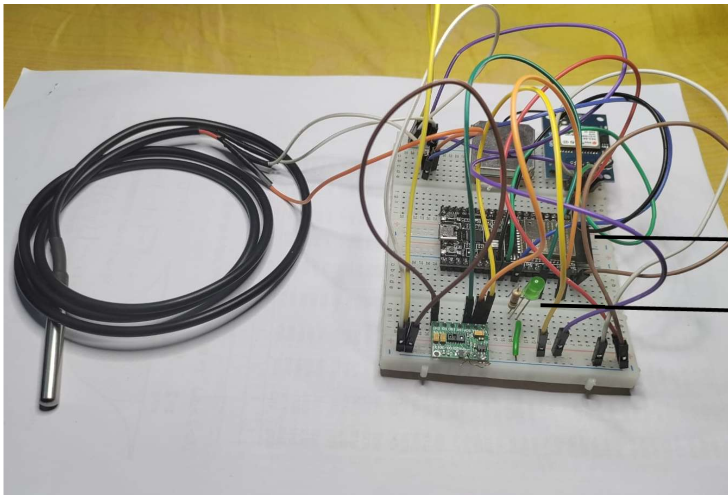


Block Diagram

Results

Hardware Overview

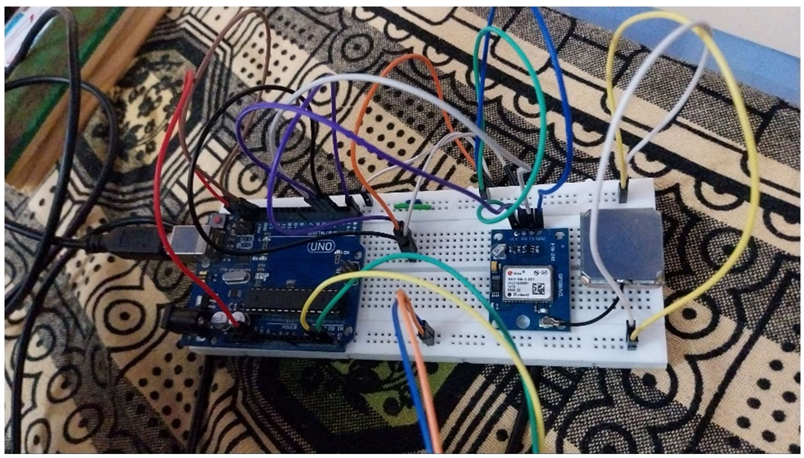




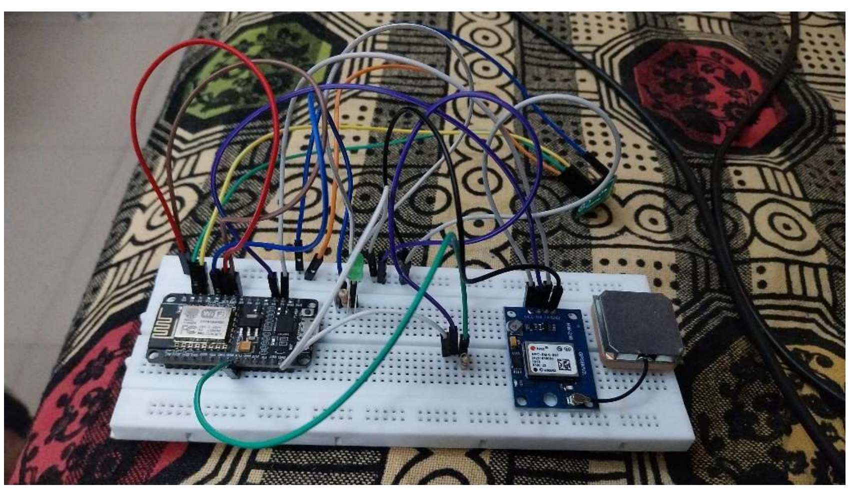
ESP32 WROOM MCU

WiFi Status LED

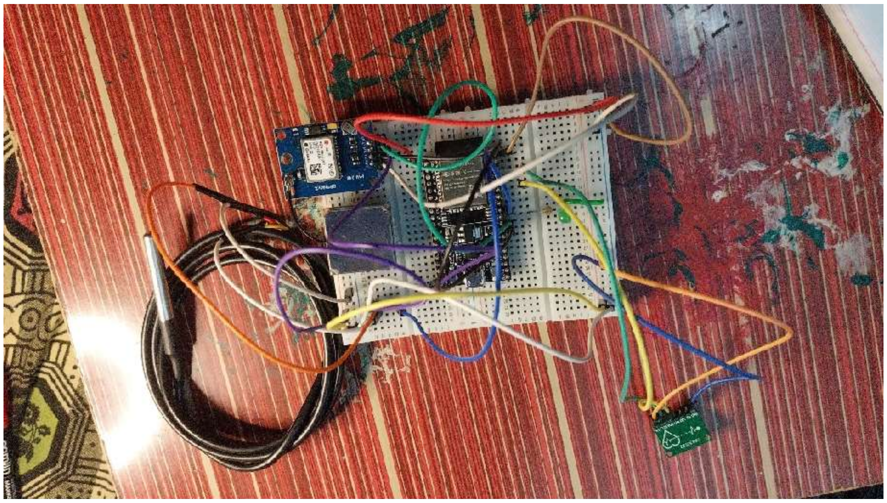
<u>Prototyping Phase</u>



Initial Interfacing with Arduino UNO but rejected this approach because its memory space was too small and external WiFi Module was required for IoT interfacing

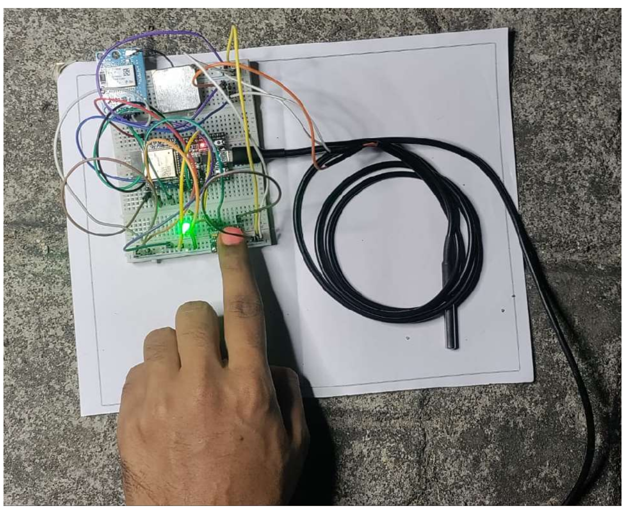


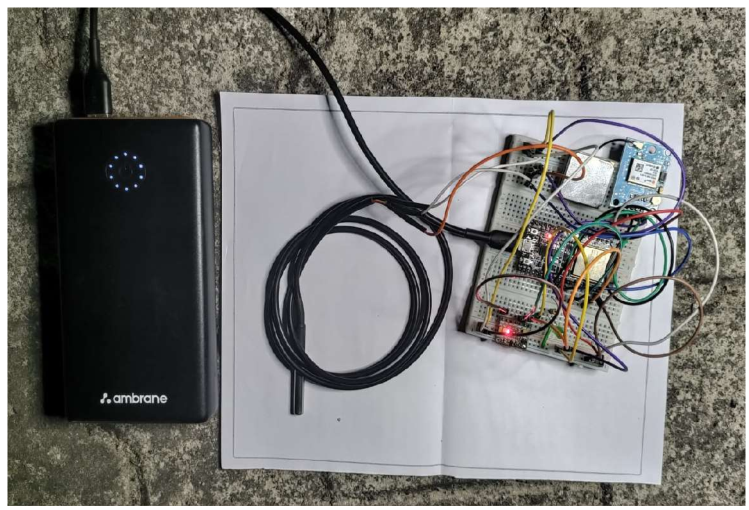
Second iteration with Esp8266 but rejected this approach because its memory space was also small for IoT interfacing and had some internal conflicts.



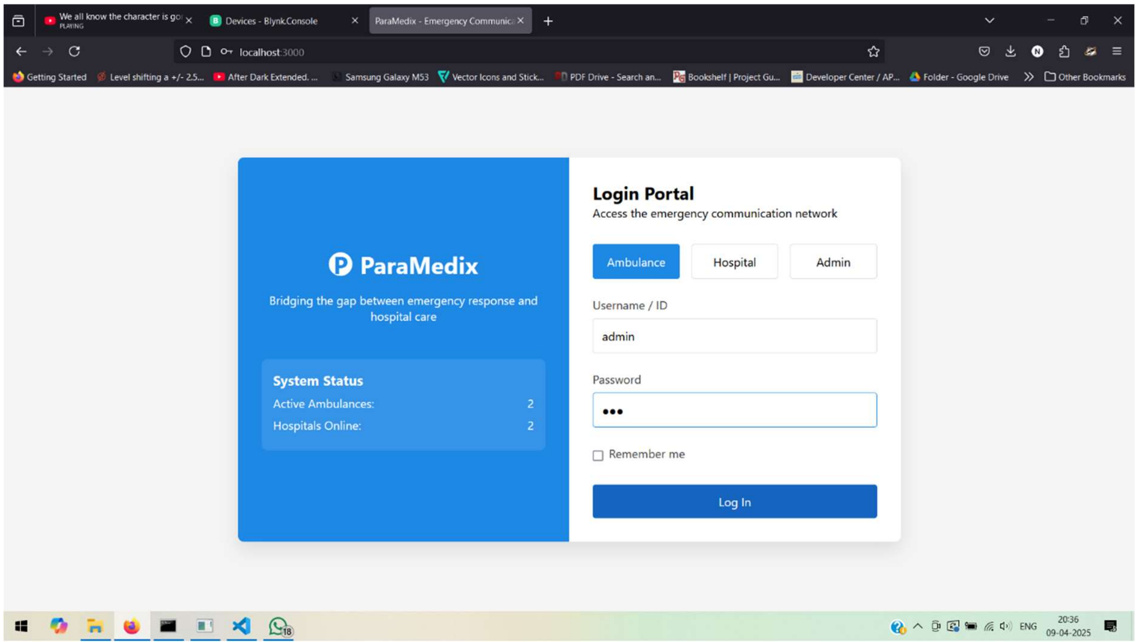
Third and final iteration with Esp32

Testing Phase

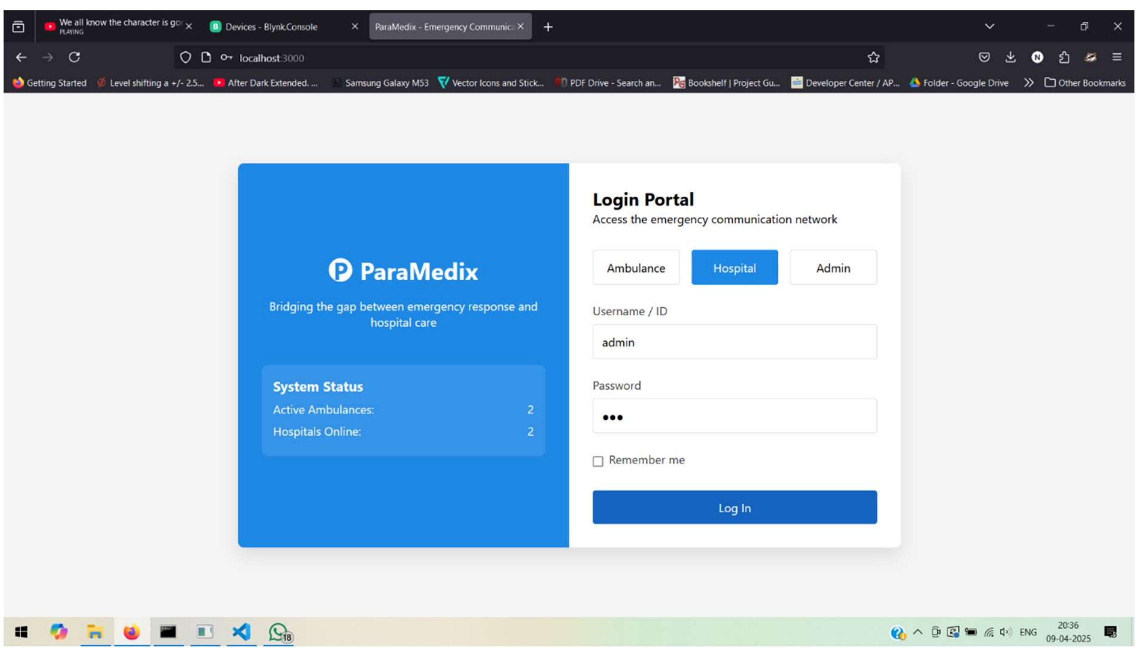




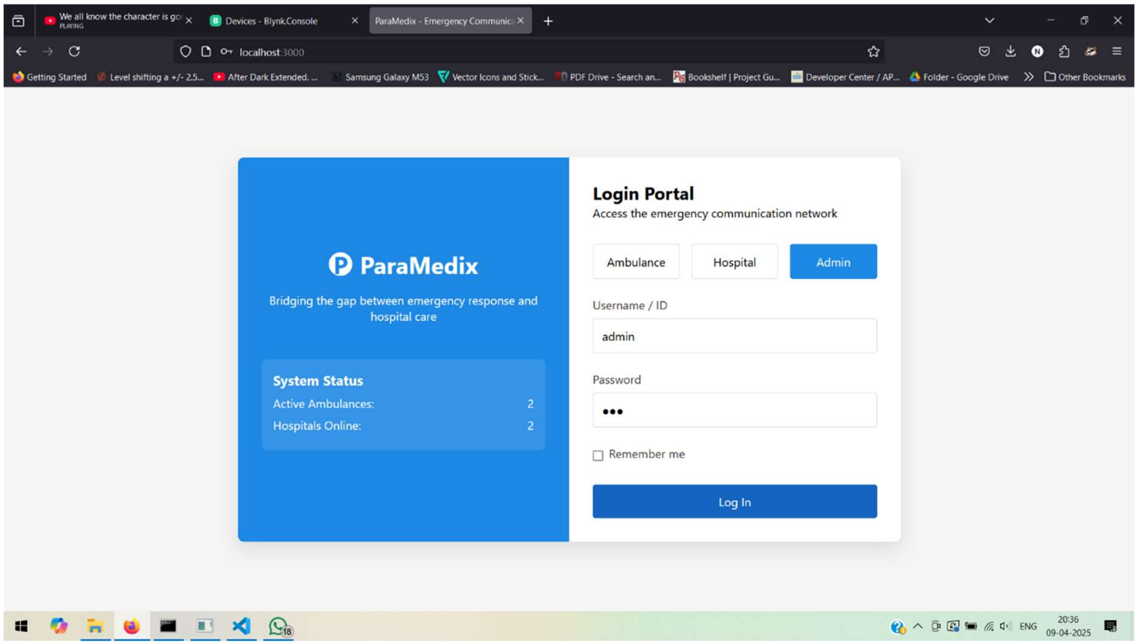
Custom Web Solution



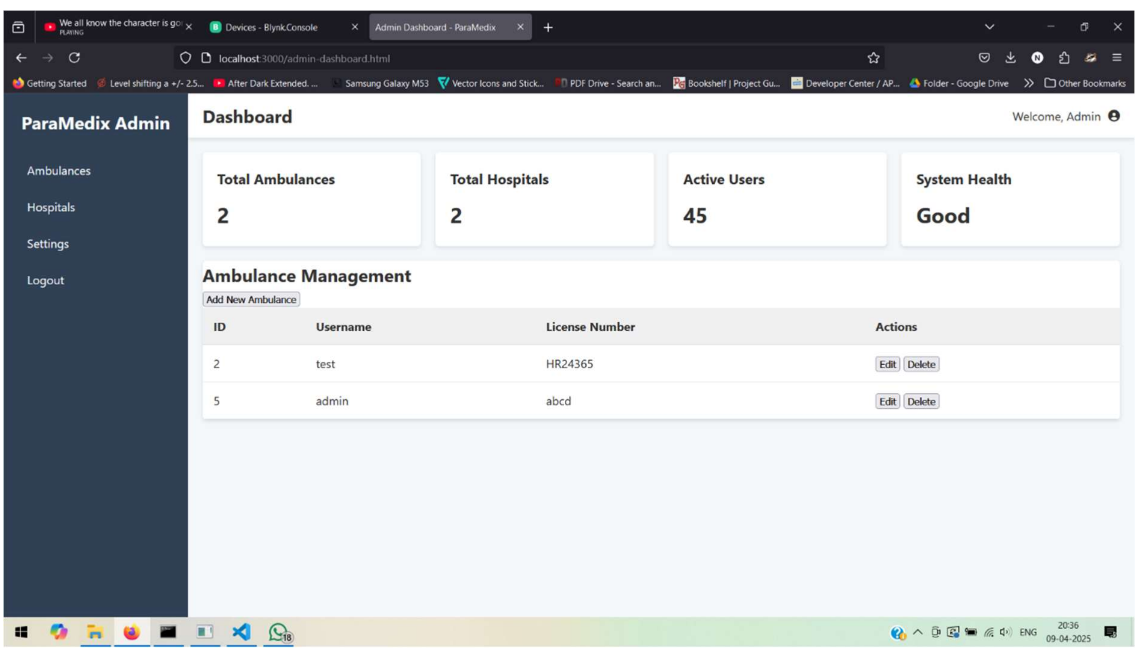
Ambulance Login Page



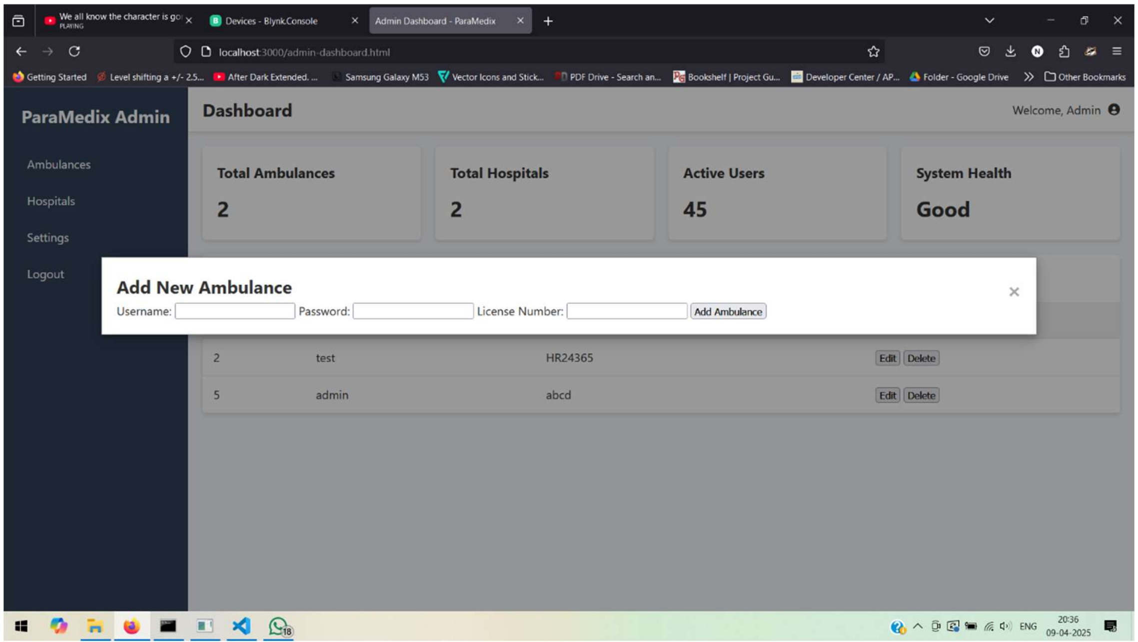
Hospital Login Page



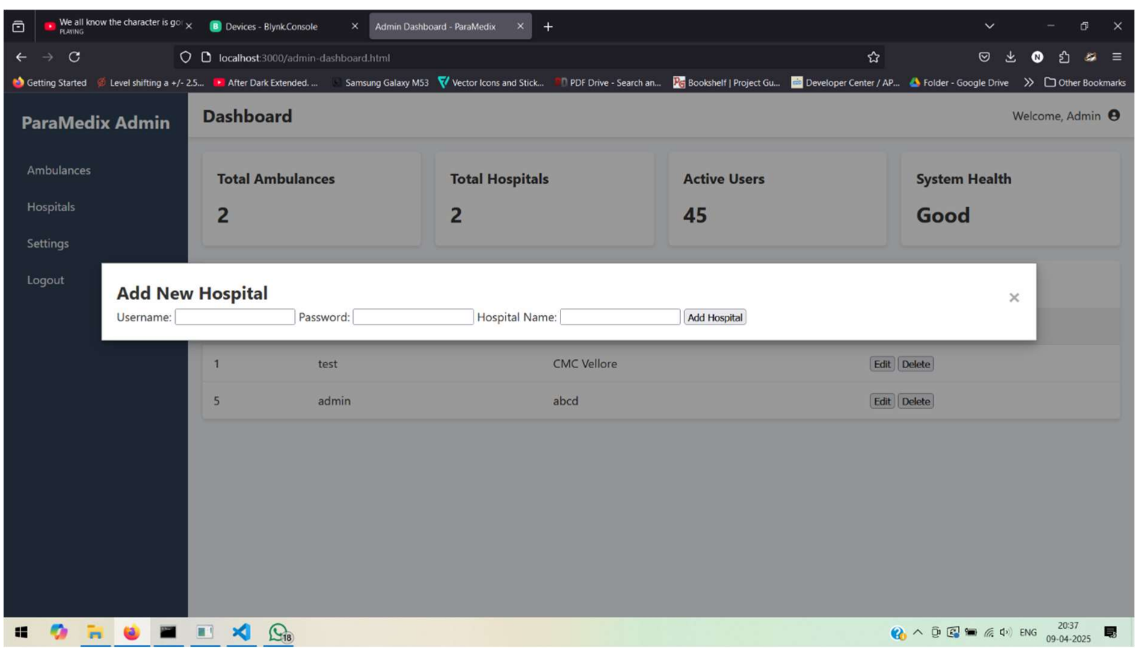
Admin Login Page



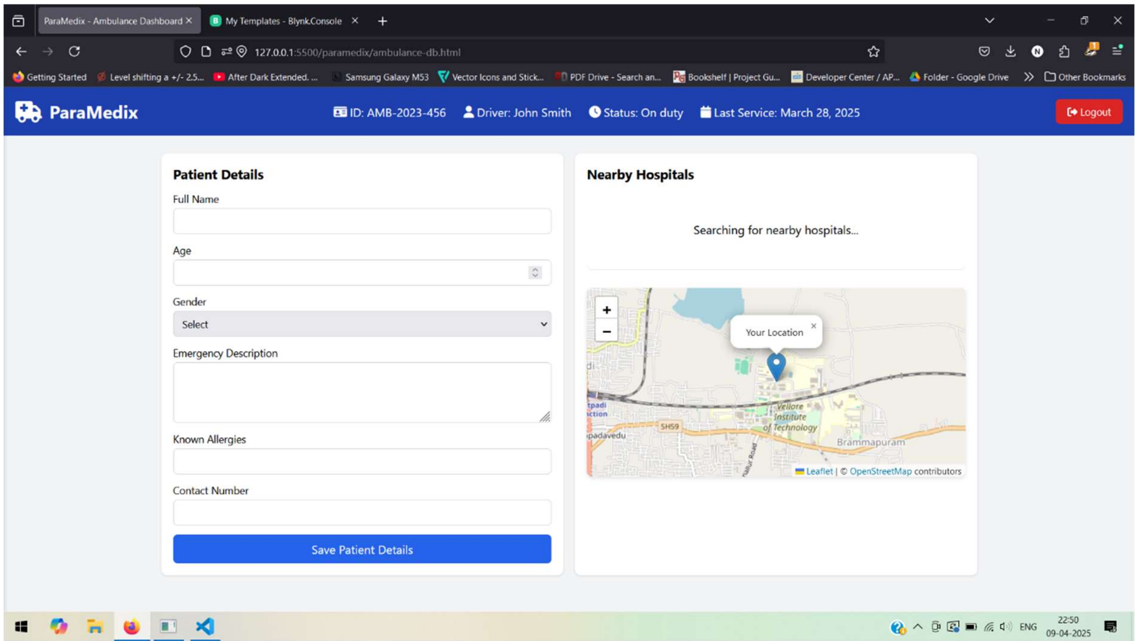
Admin Dashboard Page



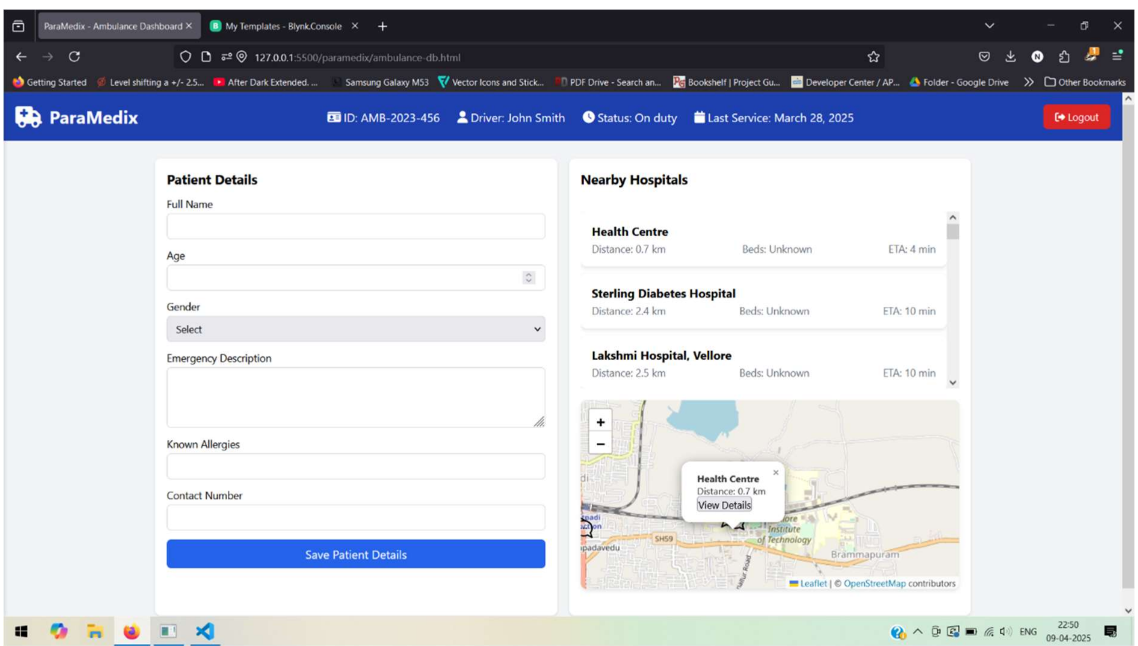
Entering/Editing Ambulance Login Credentials in Admin Dashboard



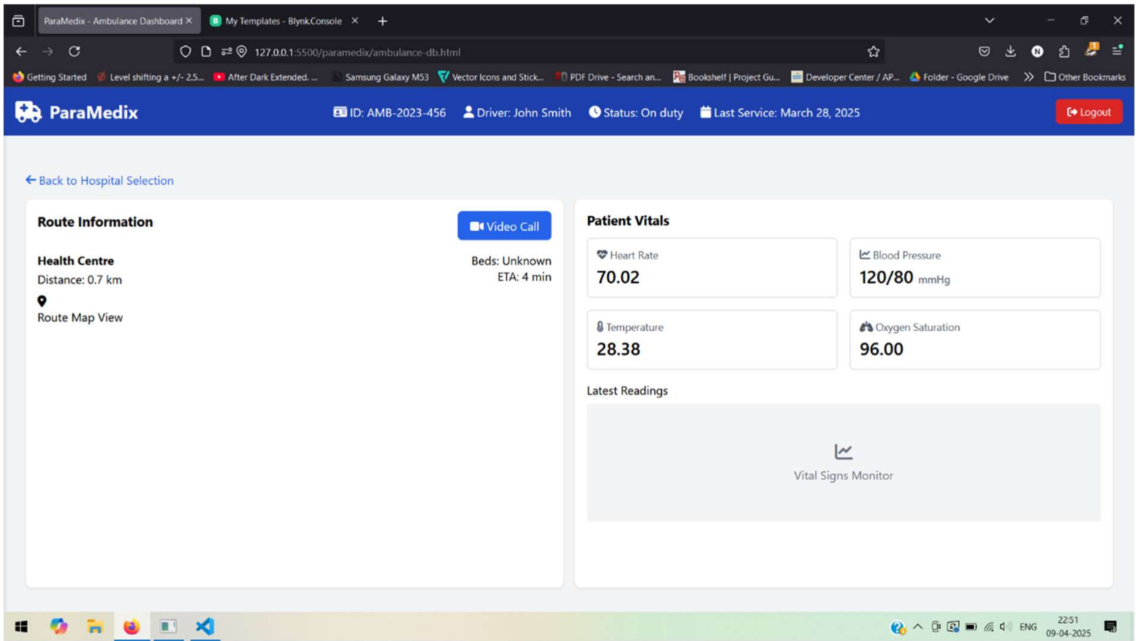
Entering/Editing Hospital Login Credentials in Admin Dashboard



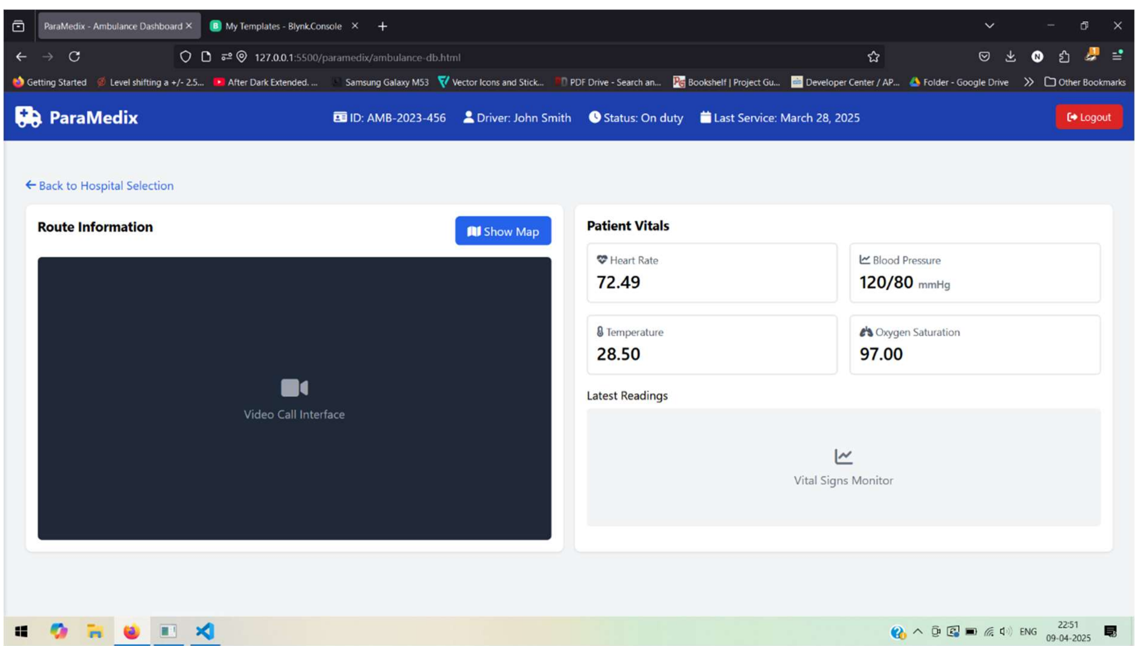
Ambulance Dashboard



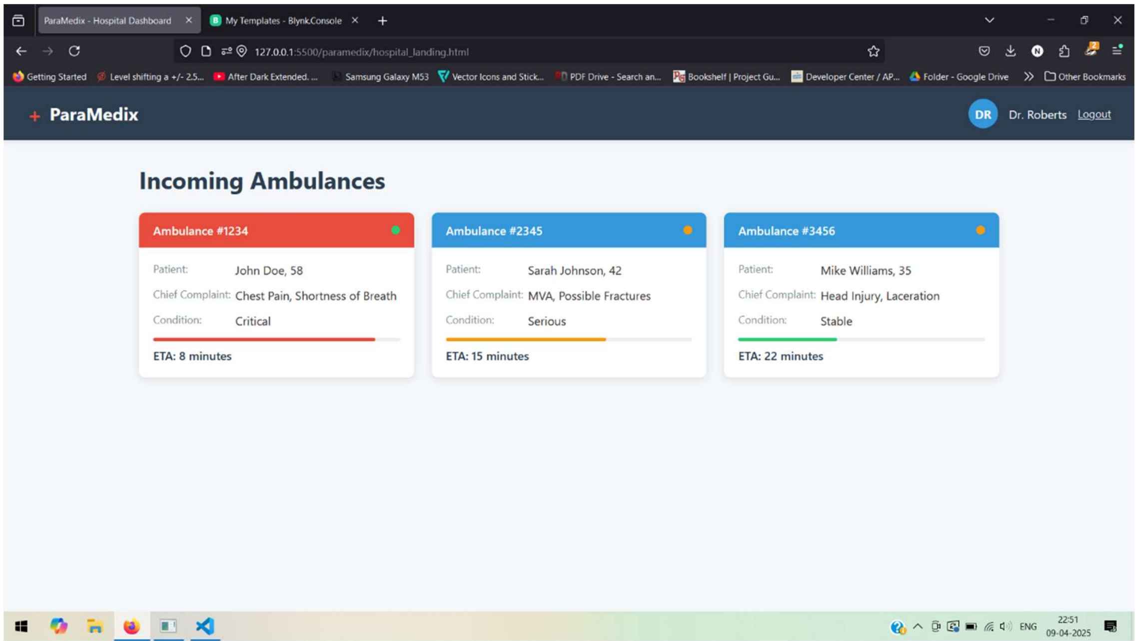
Ambulance Dashboard with Patient Form and Hospital Tracker



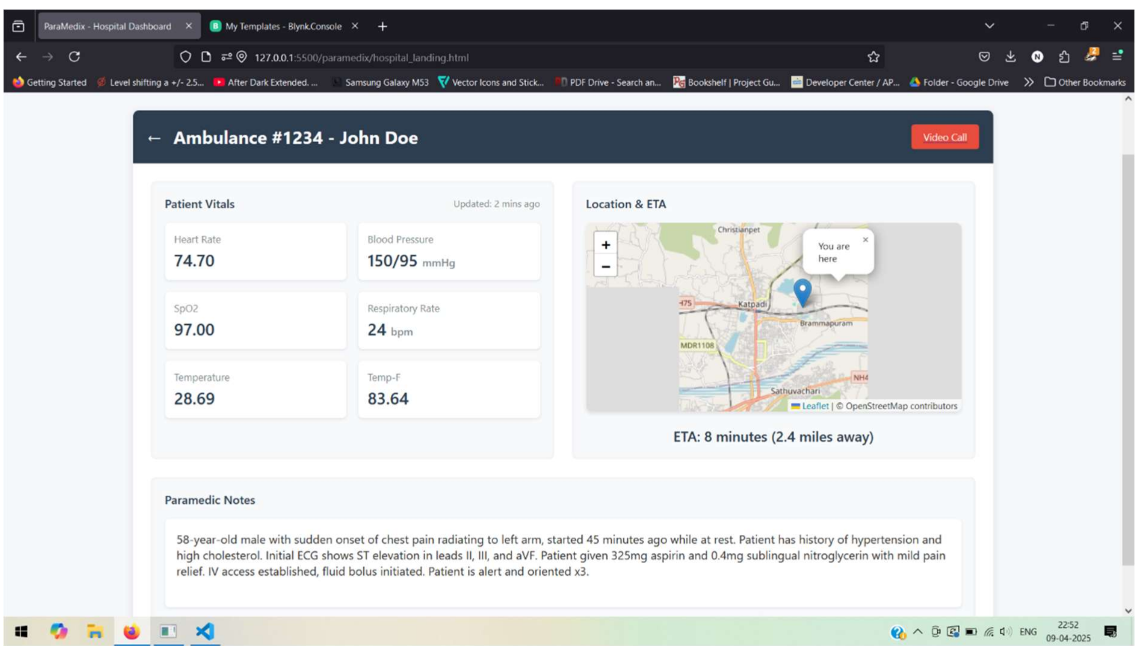
Ambulance Dashboard with Live Vitals Tracking



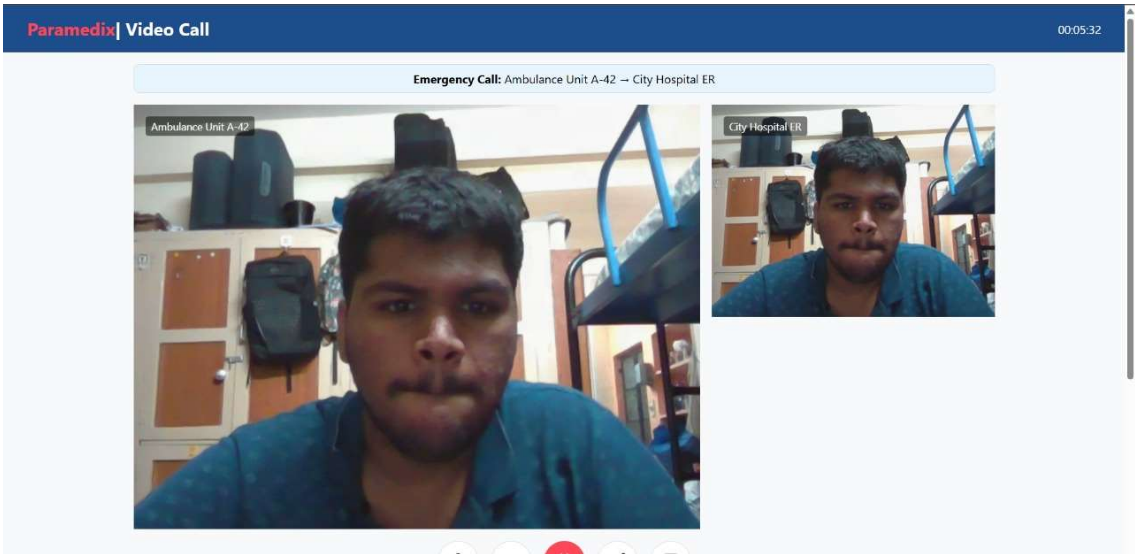
Ambulance Dashboard with Video Call Interface



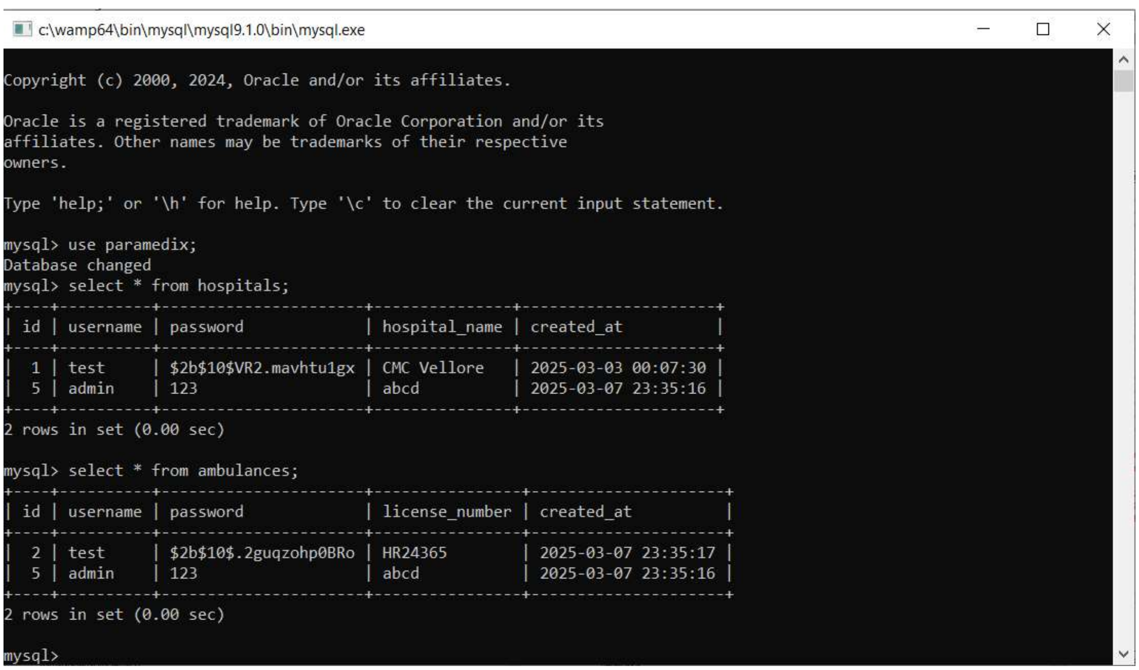
Hospital Dashboard



Hospital Dashboard with Live Vitals and Location Tracking



Hospital Dashboard with access to Live Video Feed



c:\wamp64\bin\mysql\mysql9.1.0\bin\mysql.exe

```
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use paramedix;
Database changed
mysql> select * from hospitals;
+----+----------+-----------------------+---------------+---------------------+
| id | username | password              | hospital_name | created_at          |
+----+----------+-----------------------+---------------+---------------------+
|  1 | test     | $2b$10$VR2.mavhtu1gx | CMC Vellore   | 2025-03-03 00:07:30 |
|  5 | admin    | 123                   | abcd          | 2025-03-07 23:35:16 |
+----+----------+-----------------------+---------------+---------------------+
2 rows in set (0.00 sec)

mysql> select * from ambulances;
+----+----------+-----------------------+----------------+---------------------+
| id | username | password              | license_number | created_at          |
+----+----------+-----------------------+----------------+---------------------+
|  2 | test     | $2b$10$.2guqzohp0BRo | HR24365        | 2025-03-07 23:35:17 |
|  5 | admin    | 123                   | abcd           | 2025-03-07 23:35:16 |
+----+----------+-----------------------+----------------+---------------------+
2 rows in set (0.00 sec)

mysql>
```

Backend – MySQL Database Entries

Software Overview: (Hardware Code)

```
#define BLYNK_TEMPLATE_ID "TMPL36AWMU8I8"
#define BLYNK_TEMPLATE_NAME "ParaMedix"
#define BLYNK_AUTH_TOKEN "qalMIkFVTWfE8Sqj2V9XV2gjdtjcp-yM"

#include <Arduino.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <TinyGPS++.h>
#include <HardwareSerial.h>
#include <Wire.h>
#include <WiFi.h>
#include <BlynkSimpleEsp32.h>
#include "MAX30100_PulseOximeter.h"

#define ONE_WIRE_BUS 18
#define LED_Status 5
#define RXD2 16  // GPS TX -> ESP32 RX
#define TXD2 17  // GPS RX -> ESP32 TX
#define REPORTING_PERIOD_MS 1000

const char* ssid = "Roomie's Galaxy A23 5G";
const char* password = "hvii09156";

PulseOximeter pox;
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
TinyGPSPlus gps;
HardwareSerial gpsSerial(2);

TaskHandle_t poxTaskHandle;
uint32_t tsLastReport = 0;

void poxTask(void *parameter) {
  while (1) {
    pox.update();
    vTaskDelay(1);
  }
}

void setup() {
  pinMode(LED_Status, OUTPUT);
  Serial.begin(115200);
  Wire.begin(21, 22);
  sensors.begin();
  gpsSerial.begin(9600, SERIAL_8N1, RXD2, TXD2);

  // Connect to WiFi
  WiFi.begin(ssid, password);
```

```
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi Connected");

  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, password);

  if (!pox.begin()) {
    Serial.println("FAILED");
    for (;;);
  } else {
    Serial.println("MAX30100 Initialized");
  }

  pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);

  xTaskCreatePinnedToCore(
    poxTask, "poxTask", 2048, NULL, 1, &poxTaskHandle, 1
  );
}

void loop() {
  Blynk.run();

  // Blink LED only if ESP32 is connected to Blynk
  if (Blynk.connected()) {
    digitalWrite(LED_Status, HIGH);
    delay(500);
    digitalWrite(LED_Status, LOW);
    delay(500);
  } else {
    digitalWrite(LED_Status, LOW);
  }

  sensors.requestTemperatures();
  float tempC = sensors.getTempCByIndex(0);
  float tempF = sensors.getTempFByIndex(0);

  while (gpsSerial.available()) {
    gps.encode(gpsSerial.read());
  }

  float latitude,longitude, altitude;
  int gpsValid;

  if (gps.location.isValid()) {
    latitude = gps.location.lat();
    longitude = gps.location.lng();
    altitude = gps.altitude.meters();
    gpsValid = 1;
```

```
  }

  if (millis() - tsLastReport >= REPORTING_PERIOD_MS) {
    float heartrate = pox.getHeartRate();
    float spo2 = pox.getSpO2();

    Serial.printf("{%.2f,%.2f,%.6f,%.6f,%.2f,%d,%.2f,%.2f}\n",tempC, tempF, latitude, longitude, altitude,
gpsValid, heartrate, spo2);

    // Send data to Blynk
    Blynk.virtualWrite(V0, heartrate);
    Blynk.virtualWrite(V1, spo2);
    Blynk.virtualWrite(V2, tempC);
    Blynk.virtualWrite(V3, tempF);
    Blynk.virtualWrite(V4, latitude);
    Blynk.virtualWrite(V5, longitude);
    Blynk.virtualWrite(V6, altitude);

    tsLastReport += REPORTING_PERIOD_MS;
  }

  vTaskDelay(10);
}
```

Conclusion:

This project, **ParamediX**, was developed to solve a real problem in India's emergency healthcare system—helping ambulances and hospitals communicate better and faster. Right now, there's often a delay in getting patients the treatment they need because hospitals don't know what to expect when an ambulance arrives. Our system helps fix that by sharing live updates like the patient's pulse, temperature, oxygen levels, and location while they are still in the ambulance.

We've tested two ways of doing this. The first uses the Blynk app, which makes it easy to see data quickly on mobile and web dashboards. The second is a more advanced custom-made website that lets ambulance staff choose hospitals, update patient info, see the live location, and even start a video call with the hospital doctor. The hospital, in turn, can see which ambulances are coming, check the patient's condition, and get ready before the patient even arrives.

We used low-cost components like the ESP32 and simple sensors to make it affordable and practical for use in real ambulances. The system can be connected through Wi-Fi, mobile data, or even LoRa depending on the area. In the future, this kind of system could be scaled up, improved with AI, and used across many ambulances and hospitals to save more lives.

Future Work:

While **ParamediX** already shows how real-time communication between ambulances and hospitals can help save lives, there's still a lot more we can do. In the future, we want to make this system even smarter and more useful.

One idea is to use **machine learning** to analyze the patient's vital signs and suggest possible conditions or risks even before reaching the hospital. This would help doctors get a better idea of what they're dealing with in advance. We also plan to build a proper **authentication system** so that only verified ambulance crews and hospitals can access the data, making everything more secure.

We'd like to explore better **connectivity options** like GSM modules, LoRaWAN, or 5G to make sure the system works smoothly even in rural or low-signal areas. On top of that, we can improve the **video call quality**, add **voice alerts**, and allow hospitals to send back suggestions or instructions to the ambulance in real time.

Eventually, we want to work with local governments and health departments to **integrate this with public emergency services** like 108 ambulances, so it becomes part of the official response system.

References:

[1] N. Bhalla et al., "An Overview of Emergency Medical Services in India," *Indian Journal of Emergency Medicine*, vol. 23, no. 1, 2021.

[2] WHO India Report, "Strengthening Emergency Care Systems," 2020.

[3] A. Singh and M. Ghosh, "Challenges in Pre-hospital Emergency Medical Services in India," *Journal of Health Management*, vol. 18, no. 4, pp. 549–556, 2020.

[4] Government of India, "Ayushman Bharat Digital Mission," National Health Authority, 2021.

[5] R. Rao and P. Kumar, "IoT Applications in Indian Healthcare: A Case Study of Vital Monitoring," *IEEE Conf. on Smart Technologies*, 2019.

[6] S. Mehta et al., "Mobile Health in India: Current Trends and Future Directions," *Journal of Medical Systems*, vol. 44, no. 7, 2020.

[7] D. Sharma and K. Jain, "Performance Analysis of MAX30100 Sensor for Remote Patient Monitoring," *IEEE Sensors Letters*, 2021.

[8] A. Banerjee, "Temperature Monitoring using DS18B20 in Wearable Health Devices," *Proc. of ICEICT*, 2020.

[9] J. Venkatesh and S. Mukherjee, "Real-Time Ambulance Tracking and Patient Monitoring," *Intl. Conf. on Healthcare Innovation*, 2018.

[10] M. Verma et al., "Smart Ambulance Systems: A Review," *International Journal of Advanced Computer Science and Applications*, 2019.

[11] GVK EMRI, "Annual EMS Report," 2022.

[12] K. Kiran and P. Raj, "IoT-based Health Monitoring using ESP32," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10215–10225, 2021.

[13] V. Latha and R. Reddy, "Telehealth Diagnostic Tools using ESP32 and MQTT," *IEEE Conf. on Embedded Systems*, 2020.