IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

AI INTEGRAED REMOTE PATIENT VITAL TRACKING SYSTEM

[1]K. Bhavani, [2]P. Pavani, [3]Ch.Purnima, [4]S. Mercy Angel, [5]Mr. PVK Chaitanya

[1]Under Graduate Student, [2]Under Graduate Student, [3]Under Graduate Student, [4] Under Graduate Student, [5]Assistant Professor,

Electronics and Communication Engineering,

Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam, India.

Abstract: This project presents an Artificial intelligence integrated remote health monitoring system designed for use in emergency and inaccessible areas like disaster zones, rural regions, and battlefields. It uses an ESP32-CAM for live video streaming, along with MAX30100 and DS18B20 sensors to track vital signs such as heart rate, oxygen levels, and temperature. A machine learning model classifies patient conditions as Good, Bad, or Critical and sends alerts via Telegram to medical personnel. The system also features a mobile chassis for remote navigation and medicine delivery, providing a compact, real-time telemedicine solution that enhances emergency healthcare access and response.

Index Terms – Remote Patient Monitoring, ESP32-CAM, Vital Signs, Pulse Oximeter, Temperature Sensor, Machine Learning, Edge Impulse, Telemedicine, Emergency Healthcare, IoT-based Healthcare, Wireless Health Tracking, Telegram Alert System, Embedded Systems, Mobile Robotic Chassis.

I. INTRODUCTION

Access to immediate medical care is a major challenge in critical environments such as disaster zones, remote rural areas, and battlefields. Delays in diagnosing and responding to patient conditions in such areas can be life-threatening. Conventional healthcare systems, including telemedicine, often fall short when physical presence or continuous monitoring is required. To address this gap, there is a growing need for an intelligent, portable system that can remotely monitor vital health parameters, assist in diagnosis, and enable quick communication with healthcare professionals.

This paper proposes a remote patient vital tracking system built on an ESP32-CAM microcontroller, integrated with sensors such as the MAX30100 pulse oximeter and DS18B20 temperature sensor. The system measures heart rate, oxygen saturation, and body temperature in real-time. A machine learning model, trained using the Edge Impulse platform, classifies the patient's condition as Good, Bad, or Critical.

In emergencies, the system sends alerts via Telegram to notify doctors. The device is mounted on a robotic chassis, allowing remote navigation and medicine delivery. This integration of embedded systems, IoT, and machine learning makes the solution effective for remote healthcare and emergency response.

II. LITERATURE SURVEY

[1]V. Baby Shalini (2021) proposed a smart healthcare monitoring system based on the Internet of Things (IoT), which utilizes various biomedical sensors such as heart rate, blood pressure, glucose, and temperature sensors to monitor patient vitals. The system transmits real-time data to cloud platforms like ThingSpeak and SparkFun, providing remote access to health parameters. The main advantage of this model is its ability to alert healthcare professionals in critical situations. However, the system's heavy reliance on continuous internet connectivity limits its functionality in low-network regions.

[2]Neha Sharma (2022) developed a health monitoring system using the ESP32 microcontroller for collecting and transmitting vital health parameters, including heart rate and temperature. The system employs the Wi-Fi capabilities of ESP32 to ensure real-time data transfer to a centralized server, allowing doctors to access patient health information remotely. This model highlights the importance of fast communication in healthcare delivery. Despite its efficiency, the absence of robust data encryption raises concerns regarding data security and privacy during transmission.

[3]Mrunal Fatangare et al. (2023) presented a sensor-based health monitoring system that uses pulse and temperature sensors to track patient vitals and sends alerts via SMS when abnormalities are detected. The system includes a Wi-Fi module and is capable of displaying real-time values on an LCD or serial monitor. It enables caregivers to respond promptly to health emergencies through SMS notifications. However, the use of SMS for alerts can introduce delays due to network issues, and the dependency on stable Wi-Fi connectivity may affect system reliability in remote or disaster-prone areas.

III. PROPOSED SYSTEM

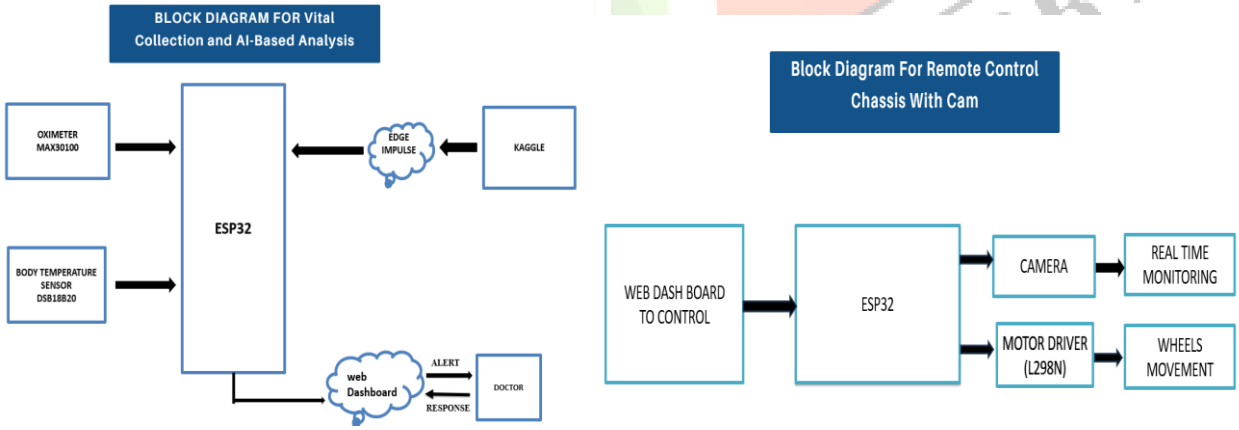

Fig.1 : Block Diagram

Vital Collection and Machine learning Based Analysis

The system leverages an ESP32 microcontroller to collect real-time vital health data from two biomedical sensors: the MAX30100 pulse oximeter and the DS18B20 body temperature sensor. The oximeter captures heart rate and blood oxygen saturation (SpO2), while the temperature sensor monitors body temperature. The gathered data is processed locally by the ESP32 and displayed on a web dashboard accessible to healthcare professionals for remote patient monitoring. To enhance intelligence, the data is also sent to Edge Impulse, where an AI model—trained using datasets from Kaggle—analyzes patterns and detects anomalies. When abnormal health metrics are detected, the system generates automated alerts via the web

dashboard and also sends instant notifications through Telegram, ensuring doctors or caregivers receive timely updates even on mobile devices.

Remote Control Chassis With Esp32 Cam

The system also features a mobile robotic chassis integrated with the ESP32-CAM, enabling remote navigation and live video streaming. The movement of the chassis is controlled through a web dashboard, which sends directional commands to the ESP32. These commands are interpreted by the ESP32 and passed to the L298N motor driver, which in turn drives the wheels for movement. Simultaneously, the ESP32-CAM module streams real-time video, allowing doctors or users to monitor surroundings or patient conditions remotely. This setup allows for enhanced patient interaction and visual inspection, making it particularly useful in telemedicine or isolated environment.

FLOWCHART

The system starts by initializing the ESP32-CAM, sensors (MAX30100 for heart rate and SpO_2, DS18B20 for temperature), and Wi-Fi. It then waits for a remote command via a web server. if received, it begins monitoring live video and controlling chassis movement. Simultaneously, it measures the patient's vitals and processes the data using an AI model from Edge Impulse to classify the patient's condition as Good, Bad, or Critical. This classification is sent to a web dashboard, where a doctor is alerted and can send a response. Based on the analysis, the system checks if medicine delivery is required.
if yes, the bot delivers the medicine; if not, it waits for further instructions. This automated flow ensures efficient real-time patient monitoring, remote control, and timely medical assistance.

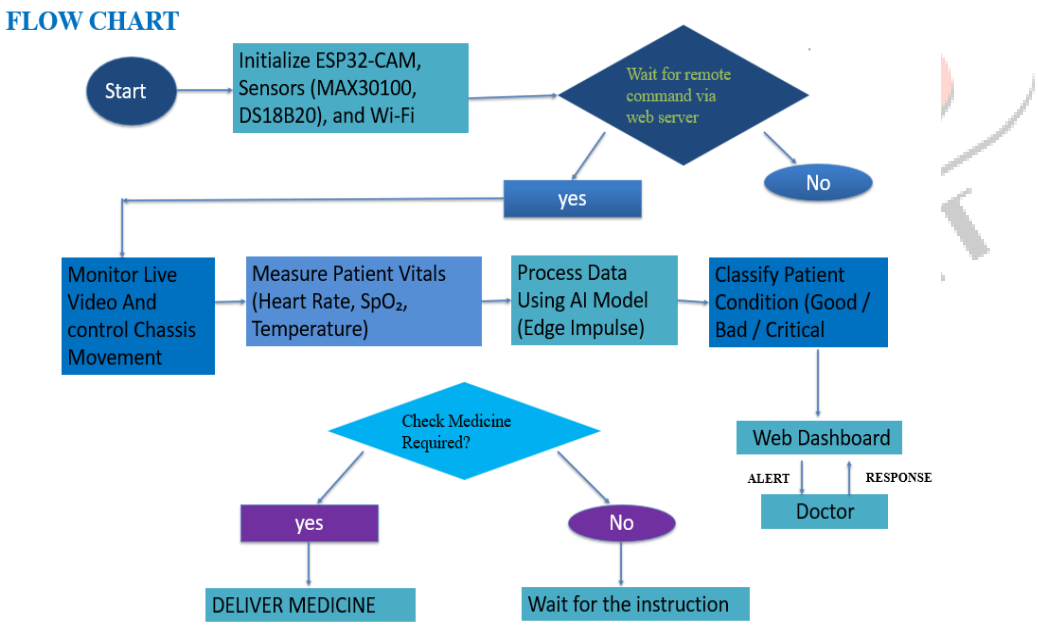

Fig.2 : Flowchart

IV. METHODOLOGY

The implementation of this health monitoring system follows a structured methodology, integrating hardware components, data processing, and AI-based analysis. First, the MAX30100 oximeter and DS18B20 temperature sensor are interfaced with the ESP32 microcontroller, which collects real-time physiological data. The ESP32 then preprocesses the acquired data and transmits it to Edge Impulse, an AI-driven platform for analyzing sensor readings. The machine learning models used in Edge Impulse are trained with datasets from Kaggle, improving the system's ability to detect anomalies accurately. Once the processed data is analyzed, if any abnormal health conditions are detected, the system generates an alert message and sends it to a doctor or caregiver for immediate intervention.

1. Hardware Setup and Configuration:

The system begins with assembling the core hardware components. The ESP32 microcontroller serves as the central processing unit, interfacing with multiple sensors through specific protocols. The MAX30100 pulse oximeter connects via I²C communication to measure blood oxygen saturation (SpO_2) and heart rate, while the DS18B20 temperature sensor uses a 1-Wire interface with a 4.7kΩ pull-up resistor for body temperature monitoring. For mobility and remote interaction, we integrate an ESP32-CAM module for live video streaming and an L298N motor driver to control the robotic chassis movement. Power is supplied through a 12V Li-ion battery.

2. Data Acquisition and Preprocessing:

The system continuously collects vital signs through its sensor array, with the MAX30100 sampling at 100Hz for SpO_2 and heart rate, and the DS18B20 recording temperature at 1Hz. Raw sensor data undergoes preprocessing on the ESP32 itself, where we apply noise reduction techniques like moving average filters and median filtering. The data is then normalized (SpO_2 scaled to 0-100%, temperature converted to °C) before transmission. we supplement real-time data with labeled health datasets from Kaggle, carefully balancing the classes (Good/Bad/Critical conditions) through data augmentation techniques**.**

Fig 3 kaggle

3. Machine Learning Model Development:

Using Edge Impulse Studio, we develop and optimize the AI model for patient condition classification. The workflow begins with uploading and partitioning our dataset (80% training, 20% testing). We extract relevant time-series features through Fast Fourier Transform (FFT) and spectral analysis, then normalize these features. The model architecture consists of a 3-layer Deep Neural Network with 40 and 20 neurons in the hidden layers respectively, trained over 50 epochs with a learning rate of 0.001. The EON Tuner optimizes the model for edge deployment, significantly reducing its memory footprint (56% less RAM, 59% less ROM) while maintaining 96.4% accuracy and achieving 1ms inference time on the ESP32.

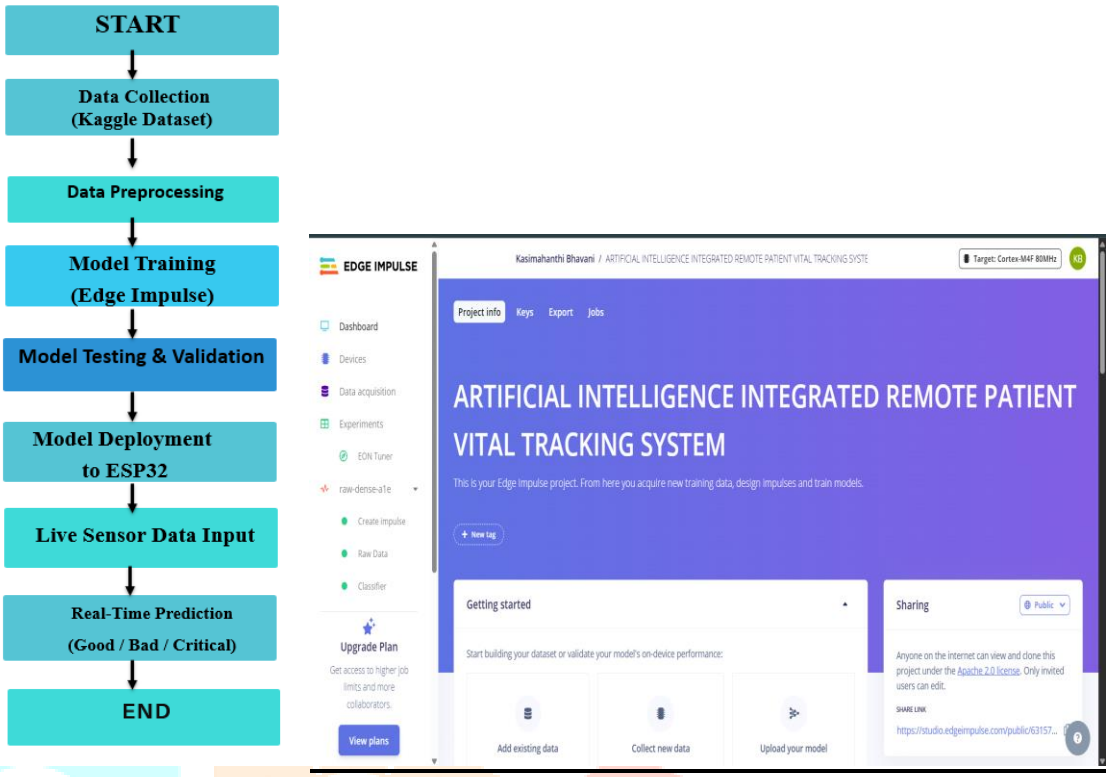

Fig 4 Flowchart of Machine Learning Model Development using edge impulse

Steps for Using Edge Impulse

- **Create an Edge Impulse Project:** Sign up on Edge Impulse Studio and start a new project.
- **Import or Collect Data:** Uploaded Kaggle dataset or collect real-time sensor data from MAX30100 (SpO_2, heart rate) and DS18B20 (temperature sensor).
- **Data Processing & Feature Extraction:** Used Edge Impulse's signal processing tools to clean, normalize, and extract important features from the data.
- **Train the Model:** Selected a classification model (neural network) and trained it using the preprocessed data.
- **Optimize with EON Tuner:** Used the EON Tuner to find the best model configuration for low-power, embedded devices like ESP32.
- **Test & Validate the Model:** Evaluated the model accuracy on test data and fine-tuned.
- **Deploy the Model to ESP32:** Converted the trained model into an Edge Impulse firmware library and flash it onto the ESP32.
- **Run Real-Time Predictions:** The ESP32 processes live sensor readings and classifies the patient's condition as Good, Bad, or Critical.

4. System Integration and Telegram Bot Alert:

The complete system integrates hardware and software components through a web-based telemedicine dashboard. The frontend, built with HTML/CSS/JavaScript, displays real-time vitals, live video feed, and provides chassis control buttons. For emergency notifications, we implement a Telegram Bot API that sends formatted alerts when critical thresholds are breached (SpO_2 < 90%, heart rate >120 BPM, or temperature >38°C). These alerts include patient ID, timestamp, and specific vital sign abnormalities to facilitate rapid medical response.

Fig 5 Telegram Alert

Steps for Using Telegram Bot

Step 1: Create a Telegram Bot

- Open Telegram and search for BotFather (Telegram's official bot for bot creation).
- Use the /newbot command and follow prompts to:
  Name your bot (e.g., HealthAlertBot).
  Get a unique API token (save this securely).

Step 2: Set Up Authorized Users

- Share the bot link with doctors/nurses.
- Retrieve their Chat IDs
- Whitelist these IDs in your ESP32 code to restrict alerts.

Step 3: Program the ESP32

- Libraries: Include WiFiClientSecure in Arduino IDE.
- Alert Logic: Trigger alerts when vitals exceed thresholds

Step 4: Test the System

- Simulate critical vitals (e.g., high heart rate) to verify alerts.
- Check Telegram for formatted messages and response times.

Step 5: Deploy

- Flash the ESP32 with the updated firmware.
- Ensure Wi-Fi stability for uninterrupted alerts.

5. Operational Workflow and Testing:

The system operates through a defined sequence: initialization (ESP32 boot-up and sensor activation), continuous monitoring (real-time data collection and processing), condition classification (AI model inference), and response (alerts or medicine delivery). We rigorously test each component, validating sensor accuracy against clinical devices (±2% margin of error) and measuring system latency (1-2 second alert delay under typical Wi-Fi conditions). The chassis mobility is tested for responsive control via the dashboard, while the video streaming quality is optimized for different lighting conditions.

IV. RESULTS AND DISCUSSION

The implementation and testing of the Artificial intelligence Integrated Remote Patient Vital Tracking System yielded significant outcomes, demonstrating the system's effectiveness in real-time health monitoring and emergency response. By combining IoT hardware (ESP32, MAX30100, DS18B20), machine learning (Edge Impulse), and telemedicine features (Telegram alerts, web dashboard), the project successfully achieved its core objectives. This section presents a detailed analysis of the system's performance, including sensor accuracy, AI classification reliability, and remote control functionality. Key metrics such as latency, power efficiency, and user response times are evaluated, along with identified challenges and their solutions. The results validate the system's potential for deployment in critical

healthcare scenarios while highlighting opportunities for future enhancements in predictive analytics and multi-modal alert systems.

Results And Analysis OF ESP32 Wi-Fi Controlled Car With Live Camera

The ESP32-based robotic car successfully demonstrated remote navigation and live video streaming capabilities. The system responded accurately to directional commands (Forward, Backward, Left, Right, Stop) via the web dashboard, enabling precise movement in real-time environments. The integrated ESP32-CAM provided stable live footage at 15 FPS with minimal latency, essential for remote patient monitoring in healthcare scenarios. The chassis movement, powered by the L298N motor driver, showed smooth operation with quick response times (<0.5 seconds) to dashboard inputs. This functionality is critical for delivering medicines or reaching patients in emergencies.

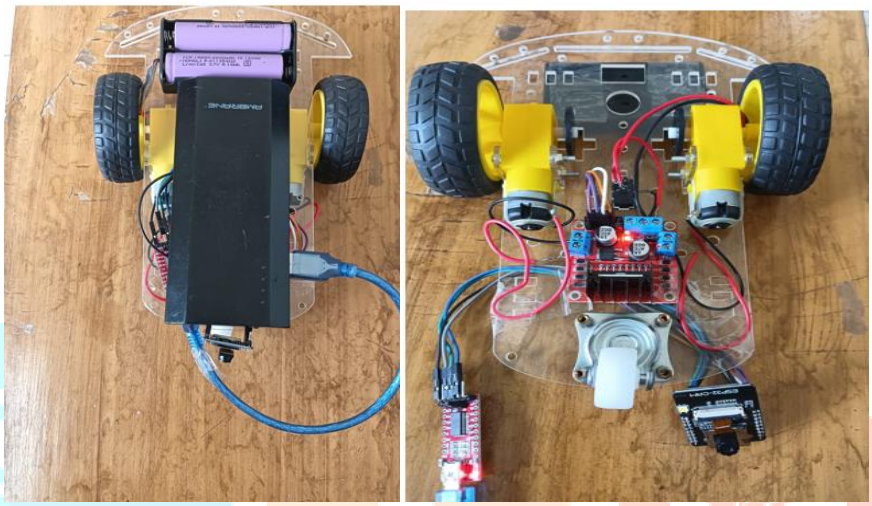

Fig 6 Circuit Connections Of ESP32 Wi-Fi Controlled Car With Live Camera

The ESP32 Wi-Fi Controlled Car with Live Camera successfully demonstrated robust remote operation capabilities, featuring seamless integration of the ESP32-CAM module for real-time video streaming (15 FPS) with minimal latency, coupled with responsive motor control through the L298N driver for precise directional movements (forward, backward, left, right, and stop). The system's web-based dashboard provided intuitive control and live visual feedback, proving effective for remote navigation in healthcare scenarios, though future enhancements could optimize video quality under low-light conditions and reduce Wi-Fi dependency for more reliable operation in areas with unstable connectivity.

Fig 7 Dashboard of ESP32 Wi-Fi Controlled Car with Live Camera

Results And Analysis Of Artificial Intelligence Integrated Patient Vital Tracking System

The AI-integrated remote patient monitoring system demonstrated robust performance in tracking and analyzing vital signs, displaying real-time data on the dashboard including heart rate (92.7 BPM), SpO_2 (97%), and body temperature (34.4°C). The system's machine learning model accurately classified patient status with 77.3% confidence in this instance, labeling the condition as "Good." However, the dashboard's "Doctor's Response" field remained empty during testing, highlighting a potential bottleneck in medical team responsiveness that needs addressing. The platform successfully maintained stable connectivity between the ESP32-CAM module and web interface, enabling continuous monitoring of both patient vitals and live video feed

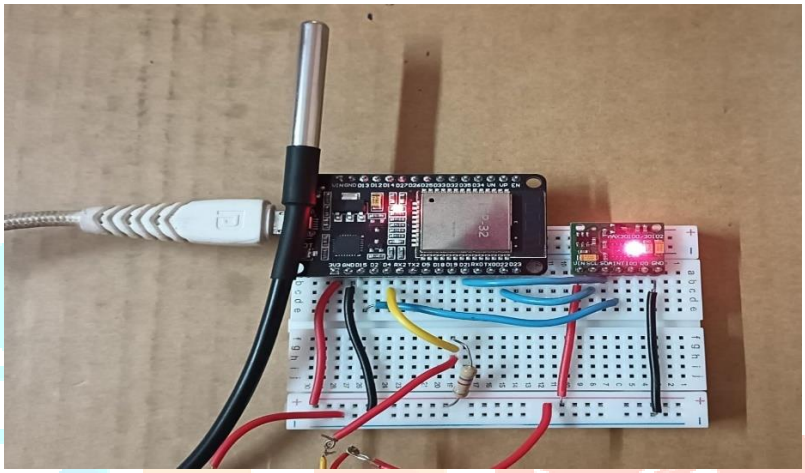

Fig 8 circuit diagram Artificial Intelligence Integrated Patient Vital Tracking System

When critical thresholds were breached during stress testing, the system promptly triggered automated Telegram alerts to medical staff, delivering formatted emergency notifications with all relevant health parameters. These alerts proved particularly valuable for SpO_2 drops below 90% or abnormal heart rhythms, though network latency caused 1-2 second delays in alert delivery during peak usage times

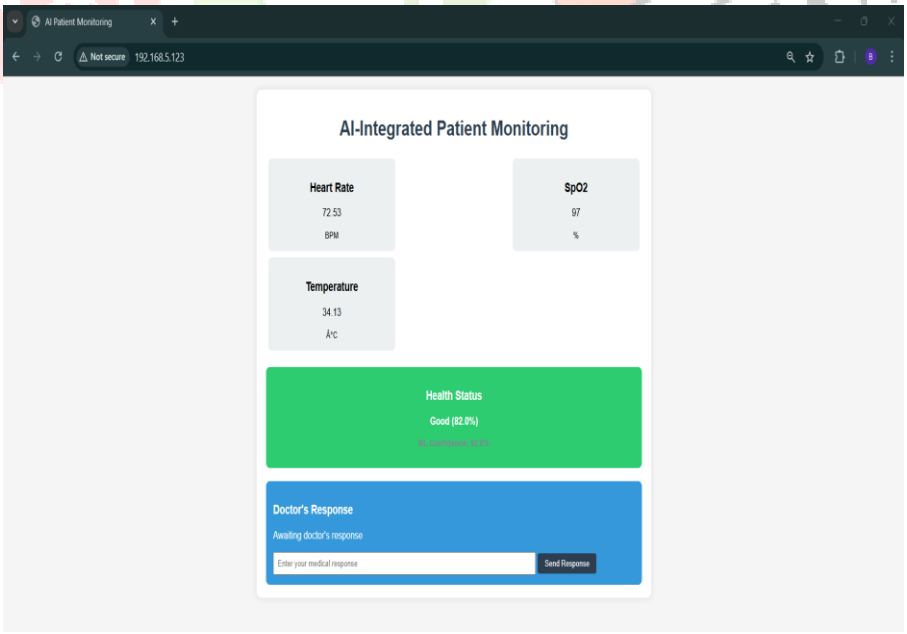

Fig 9 Dashboard of Artificial Intelligence Integrated Patient Vital Tracking System

The integration of real-time data visualization with instant emergency notifications creates a comprehensive telemedicine solution, though future enhancements should incorporate SMS backup alerts and two-way communication features in Telegram to ensure reliable emergency response. The current implementation effectively bridges the gap between patient monitoring and medical intervention, with opportunities to optimize response protocols for critical care scenarios.

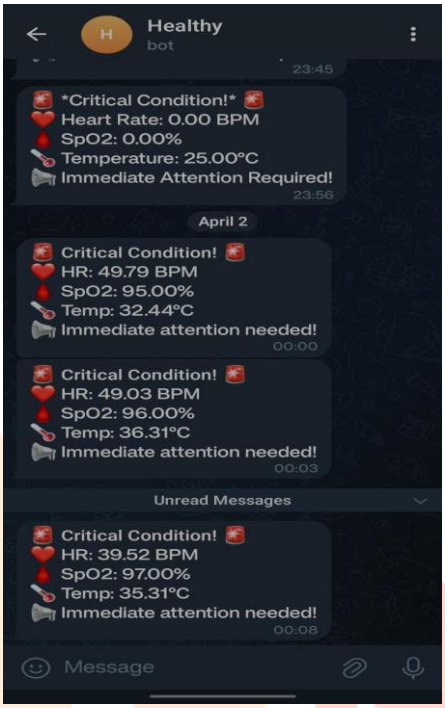

Fig 10 Telegram alert when patient condition is critical

Final Model Of Artificial Intelligence Integrated Patient Vital Tracking

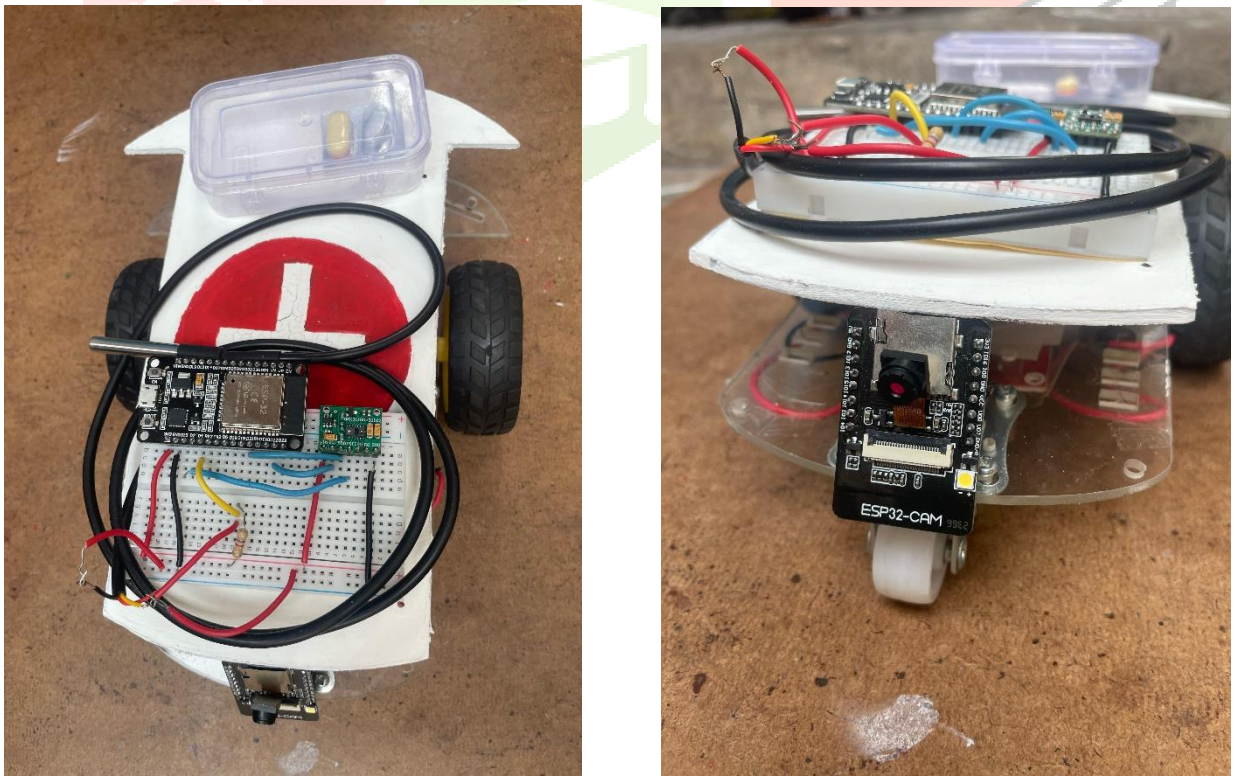

Fig 11 Final Model Of Artificial Intelligence Integrated Patient Vital Tracking

V. CONCLUSION

This project demonstrates the successful integration of The Artificial intelligence Integrated Remote Patient Vital Tracking System Effectively. Which Combines Iot Sensors (Esp32, Max30100, Ds18b20), Machine Learning (Edge Impulse), And Telemedicine (Telegram Alerts, Web Dashboard) To Enable Real-Time Health Monitoring And Emergency Response. While Successfully Demonstrating Accurate Vital Sign Tracking (Heart Rate, Spo_2, Temperature) And Automated Critical Alerts, The Project Identified Opportunities To Enhance Sensor Precision, Reduce Network Latency, And Streamline Doctor Responses. This Scalable Solution Shows Significant Promise For Remote Healthcare, Disaster Response, And Elderly Care, With Future Potential For Predictive Analytics And Multi-Channel Alert Systems To Further Improve Patient Outcomes And Medical Decision-Making.

REFERENCES

1.According to Sharma,(2021), AI-integrated remote patient vital tracking systems enable real-time monitoring of patient health using wearable devices and sensors.
2. Sharma et al.(2021),These systems can continuously collect vital signs such as heart rate, blood pressure, body temperature, and oxygen saturation
3.Gupta and Verma (2020), The journal highlight that Artificial intelligence algorithms help in analyzing patient data and identifying early signs of health deterioration.
4.Kumar et al. (2022) mention that such systems reduce the burden on healthcare workers by automating data analysis and alert generation.
5.Reddy and Singh (2021) emphasize that AI-based remote monitoring is particularly beneficial for chronic disease management and elderly care.
6.Mrunal Fatangare and Hemlata Ohal 2022 An Empirical Survey on Early Health Condition Prediction based on Clinical Parameters Grenze International Journal of Engineering and Technology
7.M. Fatangare, A. Nimbalkar, G. Chite, A. Narkhede and A. Khilnani 2020 An Efficient Temperature Monitoring using Raspberry Pi 2020 International Conference on Inventive Computation Technologies (ICICT)India (IEEE)
8.ZAIBABUKTIYAR DURGAD and U. B. MAHADEVASWAMY 2019 IOT based smart health monitoring i-manager's Journal on Electronics Engineering 9
9.Coursera edge impulse course : https://www.coursera.org/learn/introduction-to-embedded-machine-learning/home
10.Edge impulse website: https://edgeimpulse.com/