

“Intelligent ADAS for Restricted Zone Warning”

Mr. Chaitanya Pandit
panditchaitanya18@gmail.com

Mr. Prathamesh Patil
prathameshpatil1326@gmail.com

Mr. Shantanu Patil
shantanupatil1017@gmail.com

Mr. Yuvraj Salunkhe
yuvarajsalunkhe2004@gmail.com

Dr. Pravin Bidkar
bidkar.pravin@kitcoek.in

Department of Electronics and Telecommunication Engineering.

Kolhapur Institute of Technology's College of Engineering, Kolhapur

ABSTRACT :

This paper presents the design and implementation of an intelligent ESP32-based real-time GPS monitoring, geofencing, and Human Machine Interface (HMI) system integrated with Internet of Things (IoT) technologies for location tracking and smart zone-based monitoring applications. The proposed system utilizes an ESP32 dual-core microcontroller as the central processing unit, interfaced with a GPS module for acquiring live latitude, longitude, and speed information. A geofencing mechanism based on the Haversine distance algorithm is implemented to detect whether the target device is located inside or outside predefined geographical zones stored in SPIFFS memory.

The system employs a hybrid Wi-Fi Access Point (AP) and Station (STA) architecture to provide both local connectivity and internet-based communication simultaneously. A built-in web server generates a real-time interactive HMI dashboard using HTML, CSS, JavaScript, AJAX, and JSON, enabling users to visualize GPS parameters, zone status, system logs, and live map information through smartphones or web browsers. Remote hardware control functions, including relay and voice activation, are implemented through HTTP-based communication.

To improve multitasking performance and responsiveness, Free RTOS task scheduling and the ESP32 dual-core architecture are utilized for background Wi-Fi management and real-time system operation. The system further integrates MQTT protocol communication with Adafruit IO cloud services for remote GPS data publishing and IoT monitoring. Experimental implementation demonstrates reliable real-time operation, low-cost deployment, cloud integration, dynamic web interaction, and efficient geofence-based alert functionality. The proposed solution provides a scalable platform suitable for smart transportation, vehicle monitoring, safety systems, and intelligent location-aware applications

Keywords:

ESP32, GPS, IoT, Human Machine Interface, Geofencing, MQTT, Access Point, Embedded Web Server, Edge Computing, FreeRTOS, Real-Time Monitoring.

1. INTRODUCTION :

The rapid advancement of Embedded Systems and the Internet of Things (IoT) has significantly increased the demand for intelligent and real-time monitoring systems. GPS-based tracking technologies are widely used in vehicle monitoring, smart transportation, fleet management, and location-aware applications. However, many conventional systems rely heavily on continuous internet connectivity, which can reduce accessibility and reliability during network failures or in remote areas.

In addition, road accidents and violations near restricted zones such as schools, hospitals, military areas, and sensitive locations remain major safety concerns. Drivers often enter such zones unknowingly, leading to accidents, security risks, and legal issues. Therefore, there is a growing need for intelligent in-vehicle systems capable of providing real-time location monitoring and warning mechanisms.

This paper presents an ESP32-based Real-Time GPS Human Machine Interface (HMI) using a Hybrid Access Point–Station (AP–STA) architecture. The proposed system integrates GPS tracking, geofencing, embedded web server technology, cloud telemetry, local storage, and edge computing into a unified intelligent platform. GPS information is processed using TinyGPS++, while geofence detection is implemented using the Haversine distance algorithm and predefined zones stored in SPIFFS memory. A dynamic web dashboard developed using HTML, CSS, JavaScript, AJAX, and JSON enables real-time monitoring and hardware interaction. Furthermore, MQTT, Adafruit IO, Preferences, and FreeRTOS-based dual-core task management provide cloud communication, data storage, and efficient multitasking capabilities.

The proposed system provides a low-cost, scalable, reliable, and real-time IoT solution suitable for smart transportation, vehicle safety systems, geofence monitoring, and intelligent location-aware applications.

2. LITERATURE SURVEY :

Several researchers have proposed GPS and IoT-based tracking systems for monitoring, vehicle safety, and location-aware applications. With the rapid development of embedded systems and IoT technologies, various solutions have been developed using GPS modules, cloud platforms, and wireless communication techniques.

Traditional GPS vehicle tracking systems mainly focused on transmitting location information to centralized servers using GSM or internet-based communication networks. Although these systems provide remote monitoring capabilities, they generally depend on continuous internet connectivity and offer limited local accessibility during network failures.

Recent IoT-based tracking systems have utilized microcontrollers such as Arduino, NodeMCU, and ESP8266 integrated with cloud platforms for real-time monitoring. These systems improved remote accessibility but suffered from limitations including lower processing capability, single-core architecture, limited memory, and reduced multitasking performance.

Several studies introduced geofencing techniques for zone-based monitoring and alert generation using GPS coordinates. Most existing implementations focused only on basic entry and exit detection, with limited support for multiple predefined zones, dynamic user interaction, or integrated local monitoring interfaces.

Researchers have also proposed web-based monitoring systems using embedded servers to provide remote visualization. However, many of these systems rely on static interfaces and periodic webpage refresh mechanisms, resulting in slower response and increased communication overhead.

Cloud-based telemetry using MQTT protocols has also gained significant attention for lightweight IoT communication. Existing systems successfully demonstrated remote monitoring but often lacked

seamless integration between cloud services and local standalone operation.

The ESP32 microcontroller has recently become popular due to its integrated Wi-Fi capability, dual-core architecture, and FreeRTOS support. However, most reported implementations focus on individual functionalities such as GPS tracking, cloud communication, or web visualization rather than combining multiple technologies into a unified intelligent platform.

The proposed work addresses these limitations by integrating real-time GPS monitoring, multi-zone geofencing, FreeRTOS-based multitasking, hybrid AP–STA networking, MQTT cloud telemetry, SPIFFS storage, and a dynamic Human Machine Interface (HMI) into a single low-cost and scalable system. Unlike conventional systems, the proposed architecture provides both local and internet-based accessibility, ensuring reliable operation even during network disruptions.

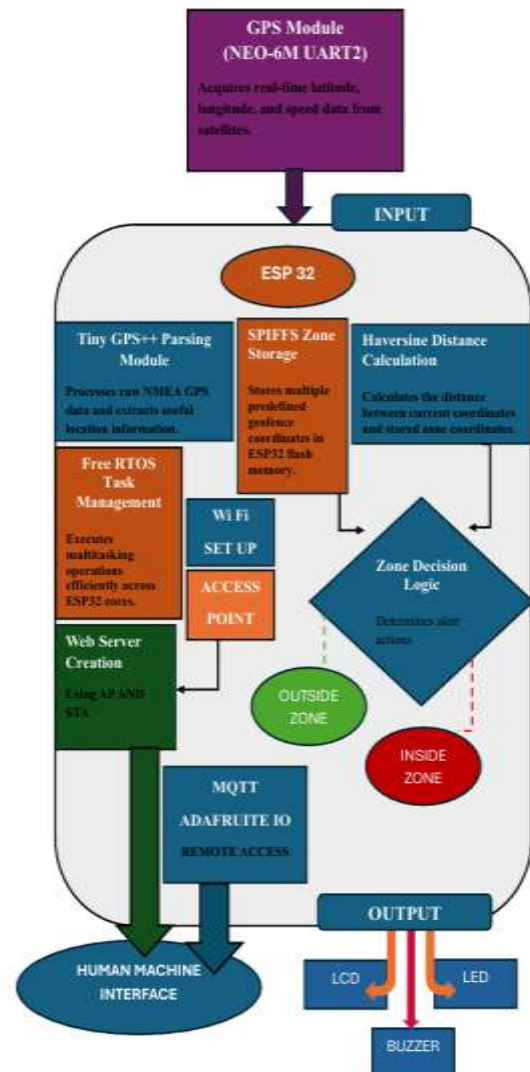
Research Gap :

Research Gap: Existing systems mainly focus on isolated functionalities and often depend on internet connectivity with limited multitasking and monitoring capability. The proposed system addresses these limitations by integrating real-time GPS tracking, multi-zone geofencing, Hybrid AP–STA networking, FreeRTOS multitasking, cloud telemetry, and dynamic HMI into a unified platform.

Existing GPS tracking systems mainly depend on continuous internet connectivity and provide limited functionality during network failures. Many IoT-based systems using Arduino or ESP8266 suffer from lower processing capability and poor multitasking performance. Existing geofencing solutions support only basic zone detection, while static web interfaces increase delay and reduce real-time performance. Therefore, there is a need for a low-cost intelligent platform that integrates real-time GPS tracking,

multi-zone geofencing, Hybrid AP–STA networking, FreeRTOS multitasking, local and cloud monitoring, and dynamic HMI into a unified system.

3. BLOCK DIAGRAM :



BLOCK DIG .

Block Diagram Sections

1. **GPS Module :** Acquires real-time latitude, longitude, and speed data from satellites.
2. **TinyGPS++ Parsing Module :** Processes raw NMEA GPS data and extracts useful location information.

3. Haversine Distance Calculation : Calculates the distance between current coordinates and stored zone coordinates.

4. Geofence Detection Module: Checks whether the current location lies inside or outside predefined zones.

5. SPIFFS Zone Storage : Stores multiple predefined geofence coordinates in ESP32 flash memory.

6. Zone Decision Logic : Determines alert actions based on calculated distance and zone status.

7. Relay/Buzzer Alert Module : Activates warning alerts when entering restricted or target zones.

8. LED Ring Control Module : Changes LED colors to visually indicate system status and zone conditions.

9. LCD Display Module : Displays live status such as zone information and distance values.

10. Embedded Web Server : Hosts the Human Machine Interface dashboard on ESP32.

11. JSON Data Module : Packages real-time sensor and system data into JSON format.

12. AJAX Communication Module : Transfers live data between browser and ESP32 without page refresh.

13. Web Browser HMI : Displays real-time GPS information, controls, logs, and maps.

14. MQTT Communication Module : Publishes telemetry data to cloud services using MQTT protocol.

15. Adafruit IO Cloud : Stores and visualizes uploaded GPS data remotely.

16. Hybrid AP–STA Networking :Allows both local access point operation and internet connectivity simultaneously.

17. Free RTOS Task Management: Executes multitasking operations efficiently across ESP32 cores.

18. Core-0 Processing Handles GPS processing, web services, MQTT, and geofence calculations.

19. Core-1 Wi Fi Task Manages Wi Fi connection monitoring and automatic reconnection.

4. METHODOLOGY :

1. System Initialization

The proposed system begins with initialization of all hardware and software modules including ESP32, GPS module, LCD, relay, NeoPixel LED ring, SPIFFS storage, Wi-Fi services, and web server components. Previously saved Wi-Fi credentials are retrieved using Preferences memory, while predefined geofence coordinates are loaded from SPIFFS for further processing.

2. Hybrid AP–STA Network Configuration

The ESP32 operates in Hybrid Access Point–Station (AP–STA) mode. Access Point mode creates a local Wi-Fi network for direct device access, while Station mode connects to available internet networks. A Free RTOS task continuously monitors connection status and performs automatic reconnection, ensuring reliable operation during network interruptions.

3. GPS Data Acquisition and Processing

Location information is obtained from the GPS module through UART communication and processed using the Tiny GPS++ library. Latitude, longitude, and vehicle speed values are extracted continuously and updated in real time. These values serve as the primary inputs for monitoring and geofencing operations.

4. Geofence Detection Mechanism

The system compares the current GPS coordinates with stored zone locations using the Haversine distance algorithm. Multiple predefined locations are stored inside SPIFFS and checked dynamically. When the calculated distance falls below the predefined threshold, the system identifies the vehicle as entering a restricted zone.

5. Alert and Safety Response System

Upon zone entry detection, visual and audio alerts are generated through relay control, LCD messages, and NeoPixel LED indicators. The LED ring changes color dynamically to represent zone status, while the buzzer provides warning indications for immediate driver awareness.

6. Human Machine Interface and Web Dashboard

A real-time web-based Human Machine Interface (HMI) is implemented using HTML, CSS, JavaScript, AJAX, and JSON. The dashboard displays live GPS coordinates, distance, speed, system status, zone indication, maps, logs, and hardware controls without requiring full page refresh.

7. Cloud Communication and Data Monitoring

When internet connectivity becomes available, the ESP32 establishes MQTT communication with Adafruit IO cloud services. GPS and telemetry information are periodically transmitted, enabling remote monitoring and cloud-based data visualization.

8. Multitasking and System Optimization

FreeRTOS-based task scheduling is used to separate networking operations from core system functions. This enables simultaneous execution of GPS acquisition, web servicing, Wi-Fi handling, MQTT communication, and interface updates, improving responsiveness and overall system performance.

5. FUTURE SCOPE :

The proposed ESP32-based GPS Human Machine Interface system can be further enhanced by integrating advanced technologies and intelligent features. Future improvements may include Artificial Intelligence (AI) and Machine Learning (ML) algorithms for predictive analysis of driver behavior, route optimization, and intelligent warning generation.

The system can be extended with GSM, 4G, or 5G communication technologies to provide uninterrupted real-time tracking and faster cloud connectivity over wider geographical areas. Integration with mobile applications can also improve accessibility by enabling users to receive notifications, alerts, and live monitoring directly through smartphones.

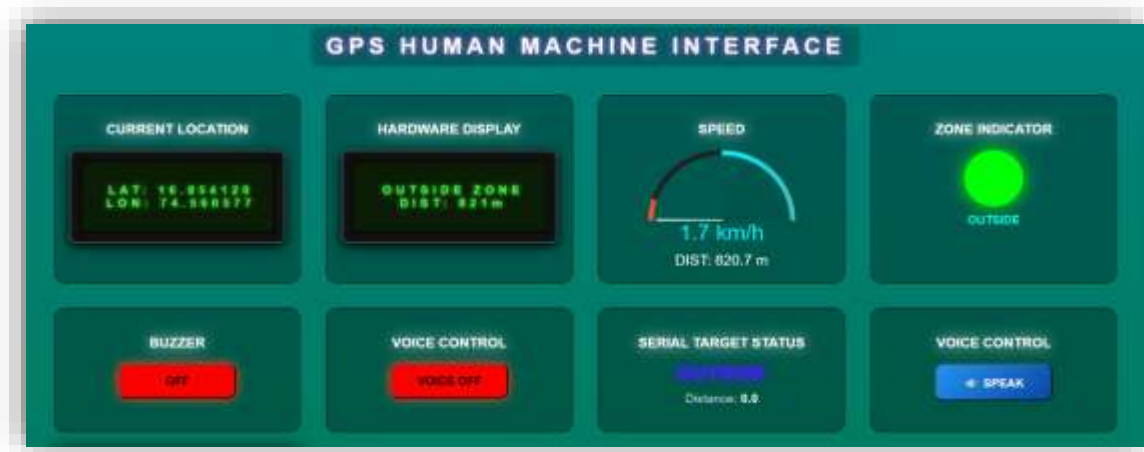
Additional sensors such as accelerometers, gyroscopes, temperature sensors, and cameras can be incorporated for accident detection, driver monitoring, and environmental analysis. Voice assistant support and speech recognition can further improve Human Machine Interface (HMI) interaction and enhance user convenience.

Future work may also include dynamic cloud-based geofence management, allowing users to remotely create, edit, and update restricted zones without modifying embedded firmware. Furthermore, the system can be integrated into smart transportation systems, autonomous vehicles, fleet management platforms, and intelligent city infrastructure for broader real-world applications.

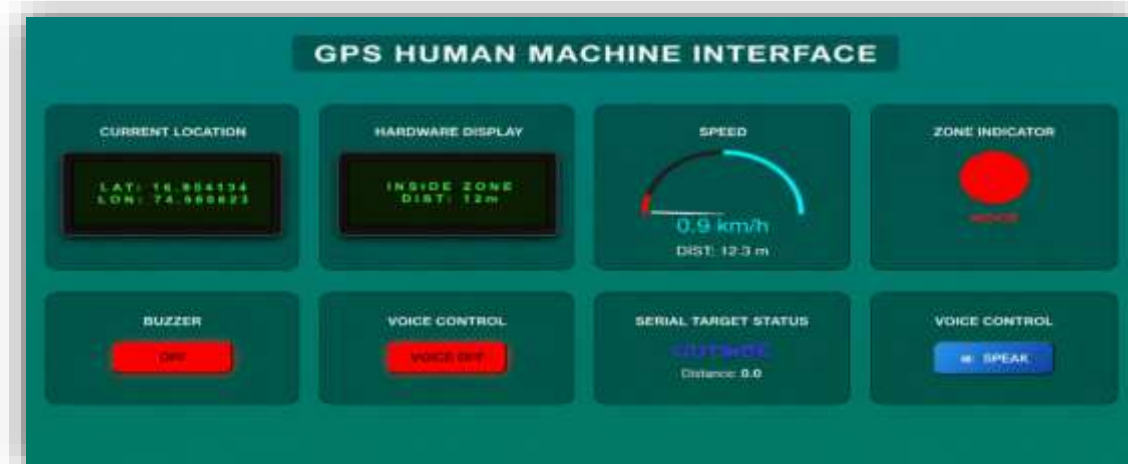
6. RESULTS :

A. OFFLINE MODE USING ESP32 HOTSPOT (ACCESS POINT) :

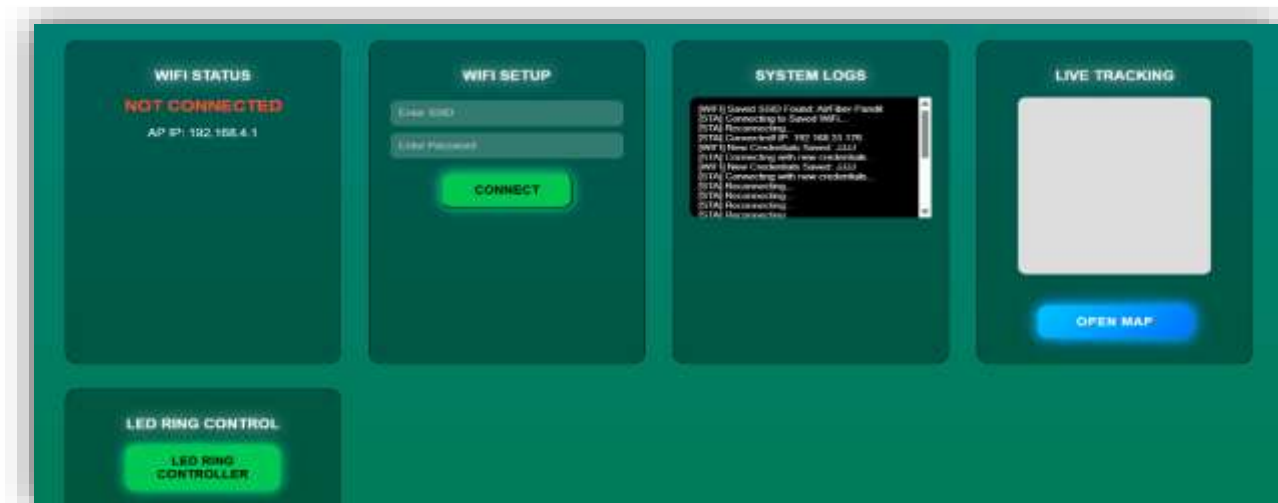
1. OUTSIDE ZONE :



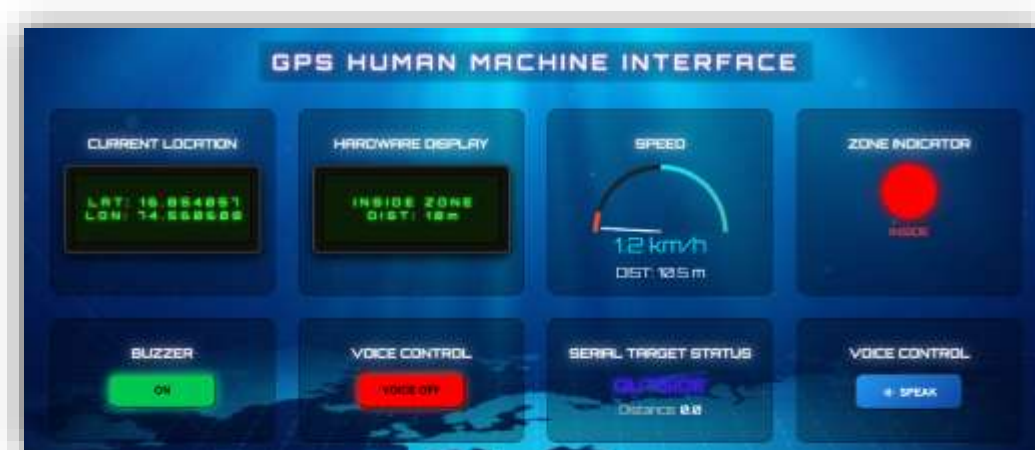
2. INSIDE ZONE :



3. LOGIN WINDOW TO CONNECT INTERNET :

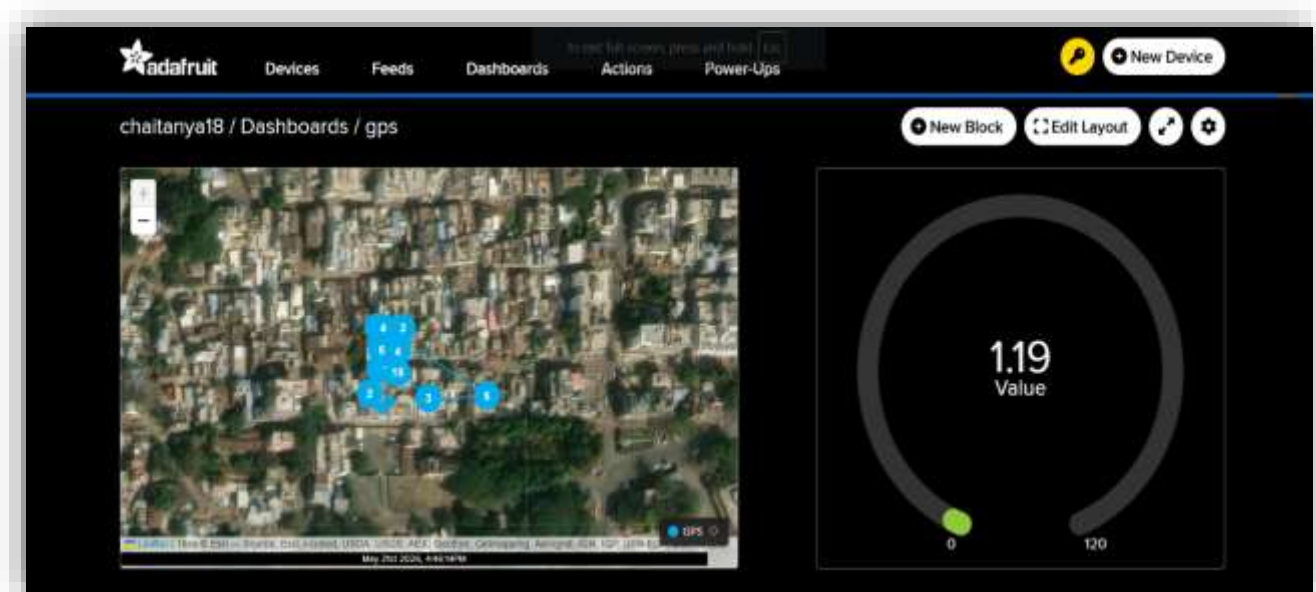


B . ONLINE MODE USING ESP32 Wi Fi (STA) :



C . ADAFRUIT IO : FOR REMOTE ACCESS

Feed Name	Key	Last value	Recorded
☐ GPS	gps	3.85	less than a minute ago



ESP32 GPS HUMAN MACHINE INTERFACE SYSTEM



7. SUMMARY :

The proposed work presents an ESP32-based Real-Time GPS Human Machine Interface (HMI) designed for intelligent monitoring and vehicle safety applications. The system aims to provide real-time location tracking and alert generation for restricted areas such as schools, hospitals, military zones, and sensitive locations.

The system uses the ESP32 dual-core microcontroller as the central processing unit due to its integrated Wi-Fi capability, multitasking support, and efficient processing performance. A GPS module continuously acquires latitude, longitude, and speed information through UART communication.

GPS data are processed using the TinyGPS++ library to extract accurate real-time location parameters. These values are used as inputs for tracking, visualization, and geofence detection operations.

The system implements geofencing technology using predefined coordinates stored in SPIFFS memory. The Haversine algorithm calculates the distance between the current location and stored zones to determine whether the user is inside or outside a restricted area.

When a restricted zone is detected, the system activates warning mechanisms using a relay-controlled buzzer, LCD display, and NeoPixel LED ring. Visual and audio indicators improve user awareness and provide immediate alerts.

A dynamic Human Machine Interface (HMI) is developed using HTML, CSS, JavaScript, AJAX, and JSON. The embedded ESP32 web server displays real-time GPS data, distance, speed, logs, maps, and hardware status without page refresh.

The system also supports cloud communication using MQTT and Adafruit IO for remote

monitoring and telemetry transmission. GPS information is periodically uploaded whenever internet connectivity is available.

The Hybrid AP–STA architecture allows both local and internet-based access. This improves reliability by maintaining system accessibility even during network failures.

To improve performance, FreeRTOS-based multitasking is implemented for parallel execution of Wi-Fi handling, GPS processing, cloud communication, and web server operations. This ensures efficient and responsive system behavior.

Overall, the proposed system integrates GPS tracking, geofencing, cloud connectivity, local monitoring, web technologies, and intelligent alert mechanisms into a low-cost, scalable, and reliable IoT platform.

8. CONCLUSION :

The proposed ESP32-based Real-Time GPS Human Machine Interface (HMI) successfully integrates GPS tracking, geofencing, Hybrid AP–STA networking, cloud communication, and embedded web technologies into a unified platform. The system enables real-time monitoring of location, speed, and restricted zones while providing both local and internet-based accessibility. The use of geofencing and intelligent alert mechanisms improves safety by notifying users when entering predefined sensitive areas.

The implementation of FreeRTOS-based multitasking, dynamic HMI, and MQTT cloud telemetry enhances overall system responsiveness and reliability. The developed architecture offers a low-cost, scalable, and efficient solution suitable for vehicle safety, smart transportation, fleet management, and advanced IoT monitoring applications. Future enhancements can further

improve system intelligence and expand its real-world usability.

9. REFERENCES :

- [1] Espressif Systems, *ESP32 Series Datasheet*, Espressif Systems, 2023.
- [2] M. Banzi and M. Shiloh, *Getting Started with Arduino*, 4th ed., Maker Media, 2022.
- [3] TinyGPS++ Library Documentation, GPS Parsing Library for Embedded Applications.
- [4] A. Banks and R. Gupta, "MQTT Version 3.1.1," OASIS Standard, 2014.
- [5] Adafruit IO Documentation, MQTT and Cloud Telemetry Services.
- [6] M. D. Singh and K. B. Khanchandani, *Power Electronics*, McGraw Hill Education.
- [7] Simon Monk, *Programming ESP32: Explore IoT with ESP32*, McGraw-Hill Education, 2020.
- [8] HTML, CSS, JavaScript, and AJAX Documentation for Web-based HMI Development.
- [9] FreeRTOS Documentation, Real-Time Operating System for Embedded Applications.
- [10] Research papers on GPS Tracking and Geofencing-based IoT Systems.