Grenze International Journal of Engineering and Technology, January Issue

Application of a Lightweight Protocol to Visualize the Behaviour of Autonomous Vehicle in a Virtual Environment

Vaidehi M[1], Vaibhav Magadum[2], Sharadhi Hegde[3], Shravya R[4] and Sahaj Gupta[5]
[1-5]Dayananda Sagar College of Engineering, Department of Information Science and Engineering, Bengaluru, India
Email: vaidehim-ise@dayanandasagar.edu, vaibhavmagadum859@gmail.com, sharadhih00@gmail.com, shravyar024@gmail.com, sahajguptarocks@gmail.com

***Abstract*— This paper introduces the design of a system that implements the MQTT protocol in order to model and study the operation of autonomous vehicles in a simulated IoT-based environment. The main goal is to facilitate the constant monitoring and communication of autonomous vehicles in a simulated environment, where vehicles interact with their environment and with each other. With the incorporation of IoT technologies in transportation, conventionally operated vehicles will eventually be replaced with autonomous systems. To achieve this, a design of a communication framework that is aware of its environment to support dynamic changes is required. This framework differs from the traditional approach that uses onboard sensors and perception systems, which is constrained to the knowledge of the immediate local environment rather than relying on collaborative decision-making. Vehicles will be able to monitor the actions of surrounding vehicles, beyond their immediate local surroundings and navigate accordingly based on events such as an accident or a geofenced area. The system continuously collects parameters: velocity, battery state, and GPS measurement, among others, from distributed sensors and broadcasts them to a centralized server using a lightweight communication protocol (MQTT). Results show the feasibility of IoT-enabled communication to enhance coordination between autonomous vehicles. Additionally, the data collected can be used to manage a fleet of vehicles.**

***Keywords*— Internet of Things (IoT), MQTT protocol, autonomous driving, vehicle-to-environment interaction, fleet management.**

I. INTRODUCTION

The technology of autonomous vehicles is growing swiftly as a revolutionary technology and is likely to have a considerable impact on transportation and society as a whole in the near future [12], [15]. For these systems to operate effectively, efficient vehicle-to-environment (V2E) and vehicle-to-vehicle (V2V) communication is essential [17]. However, although much advancement has been made in self-driving technologies, real-time vehicle communication with centralized monitoring systems exists as a gap [3], [10]. Therefore, the current research presents a virtualized ecosystem where autonomous cars substitute traditional vehicles, and each vehicle shares operational information, i.e., speed, GPS coordinates, and battery life, to a central server [7]. This central node serves as the operating hub for organizations and establishes fleet management and operational decision-making based on the compilation of vehicle information [5],[14].

Grenze ID: 01.GIJET.12.1.144

In the suggested solution, a mobile device is set up to serve as a virtual vehicle by means of the Termux application. Metrics such as GPS latitude, longitude, and speed are shared to a central server that can simulate a vehicle's action in a real-life context [11]. The backend server using FastAPI processes the incoming information and refreshes the live interface through WebSockets. At the same time, it logs the information to an SQLite database, allowing the data to be available for historical purposes as well. While eliminating the need for actual physical hardware, it provides real-time updates on vehicle positions continuously.

To promote effective communication, the use of the MQTT protocol [1], [8], [16] is a lightweight solution for communicating small messages, and is designed to send a message in real-time [9]. The user interface (UI), which is built using React and Vite, shows the state of the network and physical locations of vehicles, key operation parameters, and alerts from the system. In fact, the user interface is aligned to meet the needs of fleet management [2], [13], to help organizations manage an entire fleet, assess each vehicle, and recommend improved operations based on better use of their data [5].

Using autonomous vehicles in the real-world environment typically comes with high costs, safety concerns and challenges with scaling [6]. The proposed system provides a flexible and scalable virtual environment to study vehicle behaviour across a range of conditions [11]. Through GPS tracking, accident alerts and vehicle dashboards, the platform allows developers, researchers and fleet managers to have a safe and effective resource for assessing and improving autonomous vehicle performance. Employing IoT capabilities, lightweight protocols like MQTT, and stream-based applications, this paper shows how valuable a virtual environment can be for a vehicle telemetry system for monitoring and testing autonomous vehicles [4].

II. Related Work

In last few years, Internet of Things (IoT) is has become a booming industry and also have shown great sign towards real-time data collection, transmission and processing to support autonomous and connected vehicle ecosystems [7],[15]. Hence also led to many research efforts towards the evolution of these autonomous vehicles communication and control systems.

To advance collective perception in vehicular networks, Wang et al. [10] introduced an edge-assisted cooperative sensing model, which exhibited decreased latency and enhanced detection accuracy in environments with multiple vehicles. Oliva et al. [3] implemented vehicle-to-infrastructure (V2I) to improve pedestrian safety and prioritize the escorting of emergency transport in urban environments. In another fleet management context, Zhang et al. [5] developed a cloud–edge framework to manage intelligent logistics fleets employing a digital twin approach that supports real-time decision making and synchronized vehicle state monitoring. Likewise, Reddy et al. [13] used MQTT and 5G to create a dynamic geofence for an autonomous delivery robot, illustrating the benefit of level optimization at the communication protocol layer enables operational control.

Node communication was further improved for safety-critical systems in the works of Dharani et al. [4], where a secure and low-latency MQTT-based transmission protocol was created for autonomous vehicle operation, while Al-Shareefi et al. [9] developed an adaptive Quality-of-Service (QoS) framework to improve V2I communication performance under dynamic network conditions. Cao et al. [11] introduced a virtual testing and validation platform for autonomous driving development as a scalable replacement for testing on the roadway; Argui et al. [6] provided an overview of recent progress inw testing experiences based on mixed reality to improve the reliability of autonomous vehicle evaluation. Altogether, these studies illustrate the increasing significance of IoT-enabled architectures, lightweight communication protocols and virtualised validation platforms for the realisation of robust and scalable autonomous mobility systems.

III. Methodology

A. System Architecture

As depicted in Figure 1, the proposed framework utilizes a client–server architecture in which the vehicles are clients, interacting with a centralized server developed using a FastAPI backend. The system also contains a frontend based on React that consists of two top-level modules: User Dashboard and Production Dashboard. User Dashboard focuses on real-time visualization of vehicles on an engaging map [4], but it also provides useful features such as SOS alerts, distance tables, geofencing, and weather updates. At the same time, Production Dashboard is used for sophisticated analytics where the collected vehicle data can be analyzed and visualized in several ways, including charts and graphical reports.

In this design, a mobile device is set up to simulate a vehicle and sends data (like GPS coordinates and velocity) to the server. In this way, the collaboration of multiple autonomous vehicle networks can be replicated for testing

purposes inside a controlled virtual environment without the necessity of physical vehicles. Data from the vehicle nodes to the server is transmitted using the MQTT protocol, which provides a lightweight and efficient method of data exchange. WebSockets are used for backend-to-front-end communications to update the user interface in real-time.

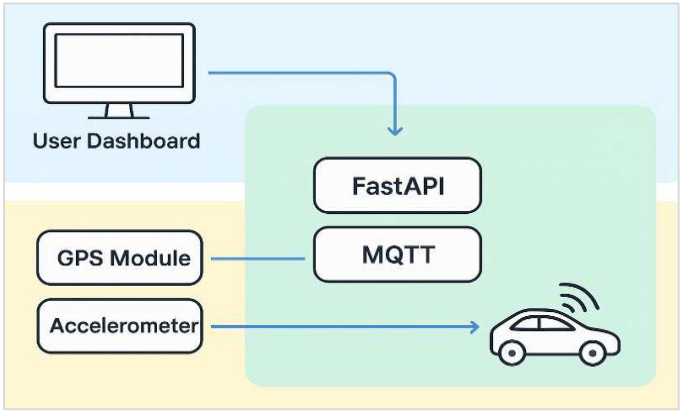

Figure 1. System Architecture

B. Component Modules

- Vehicle Module: The vehicle module is the one which uses MQTT protocol to send the essential vehicle related data to the backend server. The vehicle node uses the GPS sensor and accelerometer. Based on these sensors vehicle related data such as speed, exact coordinates, etc are shared to the backend.
- Backend Module: This is a central hub, the task of this module is to collect, store, organize and transmit data to the front end. It utilizes the MQTT for data collection, uses SQLite database for data storage and organization. IT uses WebSocket's to transmit data to the frontend. This backend processes the data coming from vehicle sensors and coordinates with front end to visualize the data.
- User Dashboard: The User Dashboard is the user end application which is visible to the end user. It helps to visualize the real time monitoring of the vehicle speed, location, accident alerts and notifications, etc. Also provide another option of fleet management system. This fetches the data from the Backend server using WebSocket's which refreshes dynamically and reduces the latency in real time.
- Production Dashboard: The Production Dashboard is designed for administrators or fleet managers who oversee multiple vehicles at the same time. It provides a clear, centralized view of all connected vehicles, displaying key details such as location, speed, average battery life, and maintainer information. With all data organized in one place, it simplifies fleet management, supports quick decision-making, and enhances overall safety. Additionally, the interactive charts and graphs make the dashboard more engaging and visually appealing for better data visualization.
- Frontend Module: This is the user interface consisting of the user dashboard as well as the production dashboard developed using React + Vite. The data stored in backend is fetched to the frontend using WebSocket for smooth and real time communication with backend in order to minimize the latency in the system. The interface enhances the user experience by providing a interactive, responsive and easy-to-use UI. Also allowing the user to access and interact with system from any internet-connected device.

C. Data Flow

The proposed architecture follows a linear flow which includes three main components: the vehicle node, the backend, and the frontend. The vehicle node sends data to the FastAPI backend using the MQTT protocol, and the backend, stores, process and transmits to the frontend through WebSocket connections.

In this setup, a mobile device running the Termux application acts as a replacement for actual vehicle hardware. A Python script inside Termux collects the device's GPS coordinates, effectively turning the phone into a virtual vehicle. This eliminates the need for separate hardware components like GPS sensors or accelerometers.

The gathered location data is treated as sensor input and sent to the backend using MQTT, a lightweight protocol designed for fast and efficient communication. The backend then stores and processes this data before transmitting it to the frontend. Both the User and Production Dashboards use this information for visualization, real-time tracking, and data analysis.

Tools and Technologies

- Backend: FastAPI, SQLite, WebSocket
- Frontend: React, Leaflet, Vite
- Data Transmission: MQTT via Termux (mobile GPS)

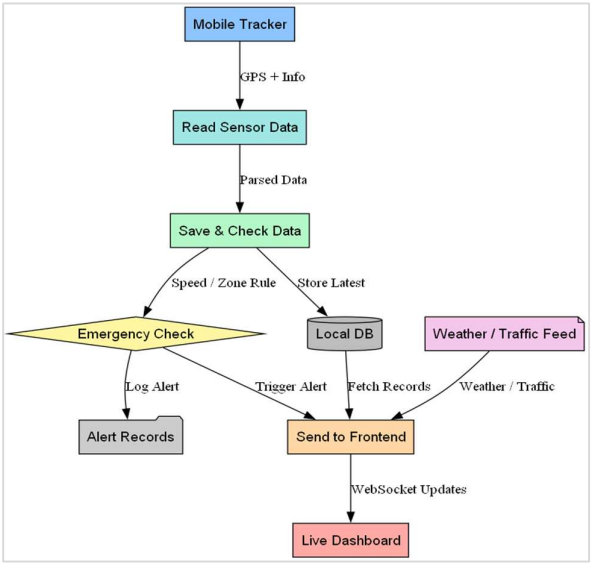

Figure 2. Dataflow diagram

Figure 2 shows the complete data flow of the intelligent vehicle monitoring system. It starts with the Mobile Tracker collecting GPS and sensor data, which is decoded, verified, and stored. The Emergency Check module detects rule violations or emergencies and sends real-time alerts to the frontend. Regular data is saved in the database, additional context (like weather or traffic) is fetched from APIs, and all updates are sent via WebSocket to the Live Dashboard for real-time monitoring and quick decisions.

III. IMPLEMENTATION

A. Overview

The Smart Vehicle framework is designed to enable autonomous vehicle-to-vehicle (V2V) communication, aiming to enhance safety, improve operational efficiency, and allow real-time sharing of critical data across the network. The system features two dedicated dashboards: a User Dashboard for individual drivers and a Production Dashboard for fleet administrators. Together, these dashboards give stakeholders clear insights into vehicle activity and performance.

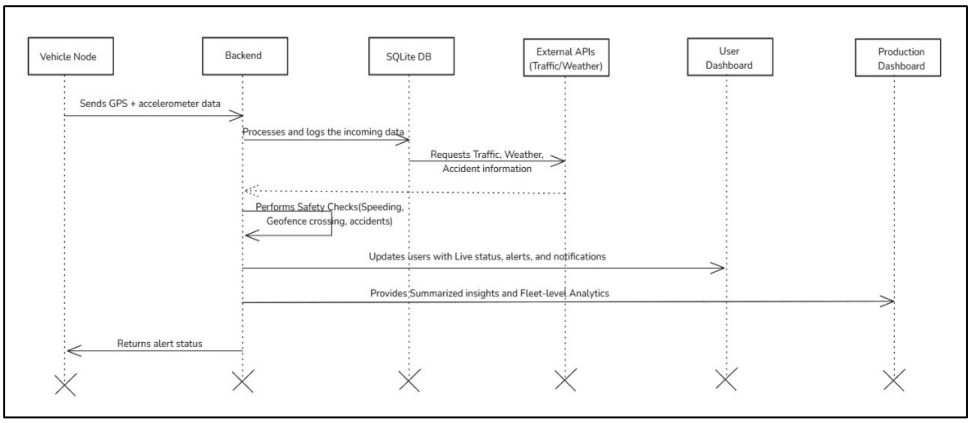

Figure 3. Sequence diagram indicating the communication flow within the System

Vehicles communicate using GPS and accelerometer data, which provide accurate information about their position and motion. This sensor data is transmitted to the backend using the lightweight MQTT protocol, ensuring reliable and timely delivery. The integration of these components demonstrates the feasibility of establishing a connected vehicular ecosystem capable of continuous monitoring and data-driven decision making.

B. Technologies used

The proposed fleet management system integrates a FastAPI backend, an MQTT-based data ingestion pipeline, and a real-time frontend dashboard. The backend server was hosted locally using Uvicorn, while the MQTT broker (Mosquitto) was configured on the same server to handle publish–subscribe communication. The SQLite database was designed with two tables: vehicle_data (containing id, vehicle_id, latitude, longitude, speed, and timestamp) and alerts (containing vehicle_id, alert_type, latitude, and longitude). Mobile devices, running Termux on Android, executed Python scripts that continuously collected GPS coordinates and transmitted them via MQTT to the backend. The frontend, served on localhost during development, connected to the backend through WebSocket for real-time updates. The dashboard consisted of a full-screen Leaflet map that displayed icons for all vehicles, with the user-entered vehicle ID highlighted in red and other vehicles in blue. To reduce visual clutter, only vehicles within a 2 km radius of the reference vehicle were shown. The dashboards also include interactive buttons that allow users to trigger SOS and accident alerts, which are displayed as contextual messages on the map. Beyond visualization, the frontend presents vehicle-specific details—such as speed, distance, and battery life—in tables, accompanied by charts and graphs that provide aggregated insights at the fleet level. This design ensures the coordinated, efficient, and safe operation of multiple vehicles in dynamic environments, while giving operators real-time

C. Challenges and Fixes

GPS Fluctuations: Mobile GPS data occasionally showed inaccuracies, which could lead to incorrect vehicle positioning. To address this, the built-in Termux accuracy parameter was used to filter out low-quality readings. Only reliable GPS data were processed, ensuring high positional accuracy.

WebSocket Connectivity: Frontend WebSocket connections were sometimes prone to timeouts or disconnections, particularly when multiple browser tabs were open or the connection remained idle. Stable, real-time data streams were maintained by introducing periodic ping messages from the frontend and server-side heartbeat checks.

Data Synchronization: Keeping the backend and multiple frontend dashboards consistently updated was challenging under high load. This was solved by implementing WebSocket-based broadcast mechanisms with atomic updates, ensuring that all dashboards reflected the same real-time state.

Visualization Overhead: Displaying a large number of vehicle markers and alert icons on the frontend map could slow down the interface. To improve responsiveness, marker clustering and dynamic loading strategies were implemented, showing only relevant vehicles within a specific radius or zoom level.

IV. RESULTS AND CONCLUSION

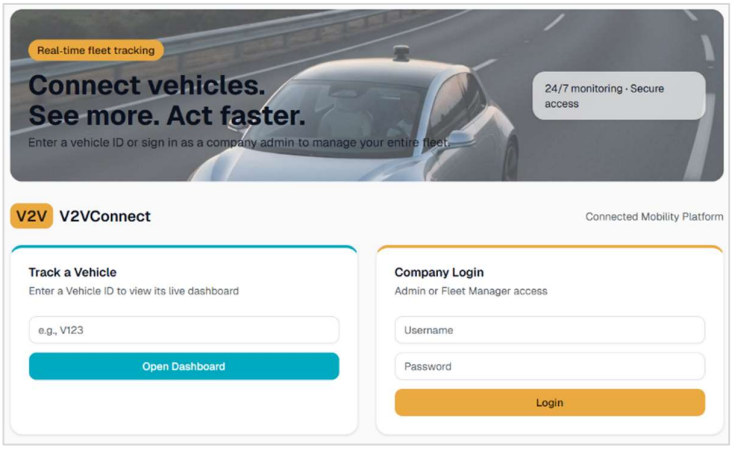

Figure 4. The Landing Page

Figure 4 shows the first page that appear as we reach the website the landing page. It displays 2 mode available: User mode and a Production mode.To enter user mode enter the vehicle Id in the input field which leads to user page. To access production page, Enter username and password credentials.

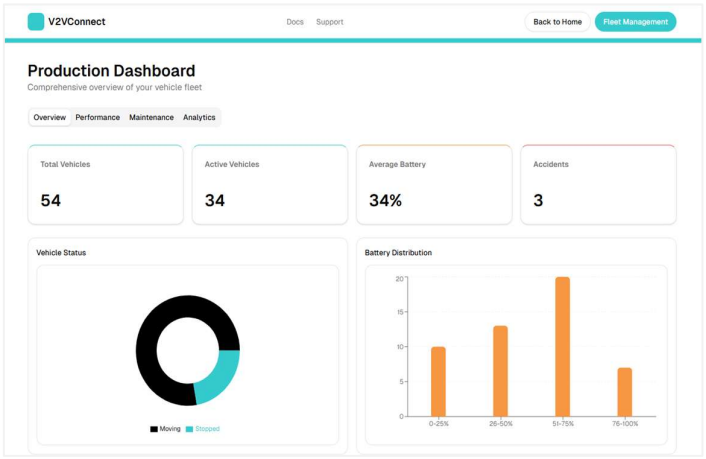

Figure 5 The Production Mode Dashboard

Figure 5 shows the Production Mode Dashboard, which shows different graphs to visulaise the overall performance of the fleet. The dashboard shows different graphs and charts plotted for different metrics like vehicle speed, battery life, average distance coverage every day,etc. Hence, usefull for analytics and visualization of overall fleet efficiency and operations.

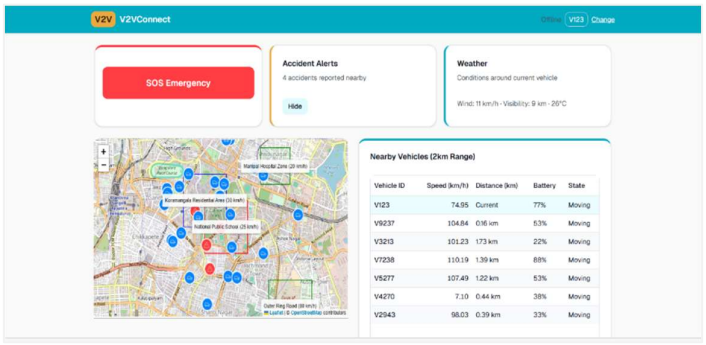

Figure 6 The User Mode dashboard

Figure 6 shows the User Mode Dashboard, which displays an interactive map displaying all vehicles in the system. The vehicle ID entered by the user is highlighted differently from other vehicle for easy identification, and the map also shows accident locations and geofences. User can view the weather conditions and also report accidents, which one reporting marks a new icon with accident alert on the map. Additionally, a table lists vehicles within a 2 km radius, including their IDs, speed, distance from the user's vehicle, battery status, and motion status, allowing users to monitor nearby vehicles in real time.

V. Conclusion and Future Scope

The shift from human-controlled to autonomous vehicle navigation involves a number of challenges, such as choosing optimal routes, tracking energy consumption, and keeping other vehicles informed about traffic, accidents, and environmental conditions. This work illustrates the practice of utilizing real-time GPS tracking, accident detection, and fleet management for autonomous and semi-autonomous vehicles in a virtual space.
Utilizing technologies like MQTT, Termux, FastAPI, and WebSockets, the system facilitates uninterrupted and high-reliability communication between vehicles and a server. The platform is deployed in two modes: User Mode for individual users and Production Mode for fleet managers, both delivering actionable insights into vehicle behavior and performance.

Simulating car interactions within a testbed environment provides a cost-efficient and scalable means of testing autonomous systems prior to deployment in the real world. By providing live data visualization and dashboard-enabled fleet management, the system improves safety and operational efficiency, a major milestone towards developing robust autonomous vehicle networks.

REFERENCES

[1] H. I. Ali, H. Kurunathan, M. H. Eldefrawy, F. Gruian, and M. Jonsson, "Navigating the Challenges and Opportunities of Securing Internet of Autonomous Vehicles With Lightweight Authentication," IEEE Access, vol. 13, pp. 24207-24222, 2025, doi: 10.1109/ACCESS.2025.3537800.

[2] J. Balcerek and P. Pawłowski, "Interaction-Based Vehicle Automation Model for Intelligent Vision Systems," Electronics, vol. 14, no. 17, p. 3406, 2025, doi: 10.3390/electronics14173406.

[3] F. Oliva, E. Landolfi, G. Salzillo, A. Massa, S. M. D'Onghia, and A. Troiano, "Implementation and Testing of V2I Communication Strategies for Emergency Vehicle Priority and Pedestrian Safety in Urban Environments," Sensors, vol. 25, no. 2, p. 485, 2025, doi: 10.3390/s25020485.

[4] S. G. Dharani, R. K. Megalingam, and A. K. K., "A Novel MQTT-Based Secure and Efficient Data Transmission Protocol for Autonomous Vehicles," in 2024 IEEE International Conference on Communications (ICC), Denver, CO, USA, 2024, pp. 1-6, doi: 10.1109/ICC.2024.10578890.

[5] L. Zhang, W. Li, and C. Wang, "Digital Twin-Driven Fleet Management for Smart Logistics: A Cloud-Edge Framework," IEEE Internet of Things Journal, vol. 11, no. 5, pp. 8624-8637, 1 March1, 2024, doi: 10.1109/JIOT.2023.3324322.

[6] I. Argui, M. Gueriau, and S. Ainouz, "Advancements in Mixed Reality for Autonomous Vehicle Testing and Advanced Driver Assistance Systems: A Survey," IEEE Trans. Intell. Transp. Syst., vol. 25, no. 12, pp. 19276-19294, Dec. 2024, doi: 10.1109/TITS.2024.3473740.

[7] I. Ud Din, A. Almogren, and J. J. P. C. Rodrigues, "AIoT Integration in Autonomous Vehicles: Enhancing Road Cooperation and Traffic Management," IEEE Internet Things J., vol. 11, no. 22, pp. 35942-35949, 15 Nov. 2024, doi: 10.1109/JIOT.2024.3387927.

[8] S. Lakshminarayana, A. Praseed, and P. S. Thilagam, "Securing the IoT Application Layer From an MQTT Protocol Perspective: Challenges and Research Prospects," IEEE Commun. Surv. Tutor., vol. 26, no. 4, pp. 2510-2546, Fourthquarter 2024, doi: 10.1109/COMST.2024.3372630.

[9] M. A. Al-Shareefi, S. H. Al-Majidi, and H. A. Al-Raweshidy, "An Adaptive QoS Framework for MQTT Protocol in Vehicle-to-Infrastructure (V2I) Communication," IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 8, pp. 8878-8892, Aug. 2023, doi: 10.1109/TITS.2023.3268445.

[10] Y. Wang, J. Liu, and K. Li, "A Cooperative Perception System for Autonomous Vehicles Using IoT and Edge Computing," IEEE Sensors Journal, vol. 23, no. 15, pp. 17521-17534, 1 Aug.1, 2023, doi: 10.1109/JSEN.2023.3285994.

[11] X. Cao, H. Chen, S. Y. Gelbal, B. Aksun-Guvenc, and L. Guvenc, "Vehicle-in-Virtual-Environment (VVE) Method for Autonomous Driving System Development, Evaluation and Demonstration," Sensors, vol. 23, no. 11, p. 5088, 2023, doi: 10.3390/s23115088.

[12] L. Chen et al., "Milestones in Autonomous Driving and Intelligent Vehicles—Part I: Control, Computing System Design, Communication, HD Map, Testing, and Human Behaviors," IEEE Trans. Syst., Man, Cybern., Syst., vol. 53, no. 9, pp. 5831-5847, Sept. 2023, doi: 10.1109/TSMC.2023.3276218.

[13] K. P. K. Reddy, M. R. Asharif, and Y. Park, "Dynamic Geofencing for Fleet Management of Autonomous Delivery Robots Using 5G and MQTT," in 2023 International Conference on Electronics, Information, and Communication (ICEIC), Singapore, 2023, pp. 1-4, doi: 10.1109/ICEIC.2023.10078921.

[14] L. Chikazhe, S. Siziba, T. Bhebhe, O. Sifile, and B. Nyagadza, "Fleet management system, perceived service quality and the public health sector performance in Zimbabwe," Int. J. Public Sector Manag., vol. 36, no. 2, pp. 113-129, 2023, doi: 10.1108/IJPSM-04-2022-0103.

[15] M. Sadaf, Z. Iqbal, A. R. Javed, I. Saba, M. Krichen, S. Majeed, and A. Raza, "Connected and Automated Vehicles: Infrastructure, Applications, Security, Critical Challenges, and Future Aspects," Technologies, vol. 11, no. 5, p. 117, 2023, doi: 10.3390/technologies11050117.

[16] T. A. Z. R. Al-Azzawi, O. A. S. Al-Zubaid, and M. Y. I. Idris, "A Comprehensive Review on IoT Protocols for Autonomous and Connected Vehicles: Focus on MQTT and CoAP," IEEE Access, vol. 10, pp. 128201-128223, 2022, doi: 10.1109/ACCESS.2022.3226543.

[17] C. R. Thompson, N. R. B. Sridharlakshmi, R. Mohammed, N. R. Boinapalli, and A. R. Allam, "Vehicle-to-Everything (V2X) Communication: Enabling Technologies and Applications in Automotive Electronics," Asian J. Appl. Sci. Eng., vol. 11, no. 1, pp. 85-98, 2022.

[18] B. Agarwal, L. Ravichandra, M. Kumar, P. D. Thakkar and V. M, "Design and Implementation of Virtual Laboratory using Unity with a Web Interface," 2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS), Manama, Bahrain, 2024, pp. 728-732, doi: 10.1109/ICETSIS61505.2024.10459676.