IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511146

Tracking and Route Deviation for Vehicle

Prof. Vishnudas Talware[1], Hardik M. Pawar[2], Sumit R. Pawar[2], Manish S. Sonawane[2], Darshan P. Mali[2]

[1]Assistant Professor; [2]UG Students, Department of Computer Science & Engineering (Data Science)

R. C. Patel Institute of Technology, Shirpur, Dist. Dhule, Maharashtra, India – 425405

Abstract: Watching vehicles in time is important for road safety, logistics, public transport and smart cities. Yet many existing tracking tools only show where a vehicle is at the moment and ask a supervisor to check if the vehicle is on the right route. This paper shows a web application that tracks a vehicle using GPS and digital maps and checks its movement against a route that was set earlier. If the vehicle steps outside the corridor that surrounds that route an alert is sent to the administrator or another authorised person. The work covers gathering GPS data creating routes with Google Maps API or OpenStreetMap measuring the distance between coordinates and the stored route sending alerts and keeping a record of each trip. The application is built with React.js, Node.js, Express.js and MongoDB. Is split into four parts: vehicle tracking, route management, route deviation detection and monitoring and reporting. Project Stage-I includes requirement analysis, technology selection, system design and a built prototype that has passenger, driver and admin screens. Numbers are shown when real experiment results exist and no unverified performance claim is made. The system is intended to reduce monitoring and to improve route compliance, in school bus tracking, logistics and public transport.

Keywords: Vehicle tracking, route deviation detection, GPS, real-time monitoring, fleet management, web application, MongoDB, OpenStreetMap.

I. INTRODUCTION

Vehicle tracking is now a part of modern transport, logistics and smart-city services and I see Vehicle tracking used everywhere. Most systems that people use depend on GPS to show the position of a vehicle but those systems rarely read the movement of a vehicle or check whether the vehicle follows its approved route. When monitoring is done by hand the administrator must watch the paths of vehicles. Decide for themselves if one vehicle has strayed. This process takes a lot of time. Becomes unreliable when many vehicles are involved.

Vehicle tracking shows that a vehicle may leave its route because of driving, traffic, misuse of fuel, private use or a deliberate decision. Those departures can increase fuel consumption delay deliveries and lower the efficiency of a fleet. For a school bus leaving the planned route is also a safety worry for children and for their parents and for emergency vehicles a route can slow the response. Existing systems also commonly lack alerts, detailed journey history route comparison records and deviation reports that could be studied later.

This project therefore looks at a system that continuously compares the live GPS position of a vehicle with a predefined route finds deviation on its own and informs administrators or users. Because GPS accuracy is limited the shape of the route and the chosen distance threshold all influence how well detection works,. The system should be tested on documented routes and judged on measured results.

The remainder of the paper is arranged as follows. Section II reviews related work Section III presents the proposed system and the reasons, behind its technology choices Section IV explains the workflow Section V describes data collection, the detection method and evaluation metrics Section VI gives the Stage-I status and Sections VII and VIII cover limitations and the conclusion.

A. Project Objectives

- Build a real-time vehicle tracking system based on GPS.
- Follow vehicle movement continuously on digital maps.
- Detect automatically when a vehicle departs from its predefined route.
- Keep travel history and route details available for later analysis.
- Give users accurate, live location updates.
- Provide a dependable and easy-to-use platform for monitoring transport.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511146

B. Potential Contributions

I think the integration of GPS tracking, predefined route management, automatic deviation detection and a monitoring dashboard is a game changer for anyone who needs a single web application to see where vehicles are and how they are moving. The project brings GPS tracking, predefined route management, automatic deviation detection and a monitoring dashboard together in one web application. The main contribution of the project is a workflow that starts with capturing the location compares that location to the route and then generates alerts while reporting trip history. The project is aimed at school bus monitoring, logistics and delivery public transport, emergency services and smart-city uses. Any statement about detection accuracy or alert delay must be backed by measured tests, on documented routes.

II. RELATED WORK

Earlier research on vehicle tracking mostly focuses on getting location through GPS sending it through a communication channel and showing it on a monitoring platform. Kumar and Kumar describe a real-time system where GPS provides the position GSM sends the data to a monitoring system. The movement is recorded and shown on a web platform [1]. Sharma and Jain use devices and GPS sensors along with cloud services to store and analyze tracking data and to send alerts when a vehicle moves in an unusual way [2].

Schiller and Voisard talk about location-based services such as route monitoring, location analysis and map-based visualization [3]. Thrun et al. Present probabilistic models for estimating movement in changing environments, which are connected to navigation accuracy and route estimation [4]. Misra and Enge give the reference on GPS signals, measurements and performance [5]. Anderson and Smith use GPS tracking and route optimization for fleets using real-time location data to make route planning and efficiency [6].

Table I shows how each of these studies connects to this work. The studies reviewed support real-time monitoring, map display and alert generation. Most of them do not provide efficient automatic detection of deviation from a predefined route. This gap is the reason for the project. Results, from the literature are not results of this project and a fair comparison would need to check the setup, route definition and evaluation method of each study.

TABLE I SUMMARY OF RELATED WORK

Ref.	Focus of the study	Relevance to this project
[1]	GPS + GSM real-time tracking with web-based position display	Location capture and logging; no predefined-route comparison
[2]	IoT and GPS sensors with cloud storage; alerts on abnormal movement	Alerting and data storage; abnormality is not tied to an assigned route
[3]	Location-based services: route monitoring, map visualization	Conceptual basis for map-based monitoring
[4]	Probabilistic movement estimation in dynamic environments	Background on navigation accuracy and route estimation
[5]	GPS signals, measurements, and performance	Background on GPS error that influences the threshold
[6]	GPS tracking and route optimization for fleets	Fleet efficiency; planning rather than deviation detection

III. PROPOSED SYSTEM

The proposed system contains GPS-enabled vehicles or smartphones a Node.js and Express.js backend a MongoDB database, a mapping service such, as Google Maps API or OpenStreetMap and a React.js dashboard. The system is divided into four modules, which are described below and Fig. 1 displays the architecture of the system.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511146

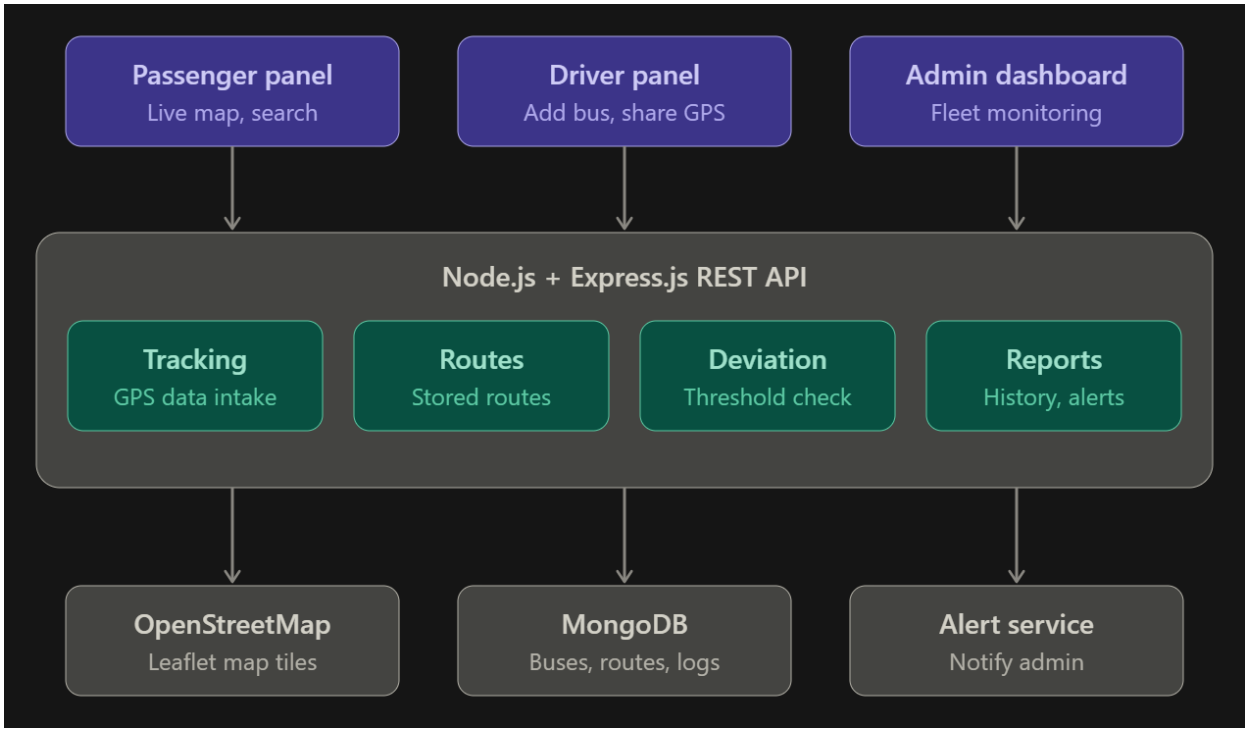


Fig. 1 Architecture of the proposed vehicle tracking and route deviation detection system.

A. Vehicle Tracking Module

This module collects the location of the car from a GPS device installed in the car or from the GPS on a mobile phone. The latitude and longitude are recorded all the time. Sent to the server using APIs through the internet. The user interface displays the locations on a digital map and the times information, about the car and the journey details are stored in MongoDB.

B. Route Management Module

The administrator creates a route by picking a source and a destination. The route is then made using Google Maps API or OpenStreetMap. Route coordinates, checkpoints, intermediate locations and route limits are stored in the database. These stored data serve as the reference path for deviation analysis. The module also lets users see a route and review routes, from the past.

C. Route Deviation Detection Module

This is the module of the system. The central module compares GPS coordinates with the saved route coordinates. The central module uses distance calculations. A threshold to set the acceptable movement limit.If the vehicle goes beyond the allowed boundary the central module logs the event as a route deviation. The central module then sends an alert away, to the administrator or authorised user.Deviation records and alert details are stored in MongoDB. The calculation is explained in Section V-B.

D. Monitoring and Reporting Module

The dashboard shows real-time car locations the paths they take when they go off track and past trips on maps that you can interact with. It allows an administrator to watch cars at the same time. Data, about tracking the routes taken alerts that happened and travel details are stored for checking. Reports can be made about how cars moved, when they went off course how long trips lasted and the history of the routes taken.

E. Technology Selection and Justification

Each part of the stack was picked to suit three needs: continuous location updates, flexible storage of routes, and a dashboard that runs in the browser (Table II). Using one language, JavaScript, for both front end and back end also keeps the work manageable for a student project.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511146

TABLE II TECHNOLOGY SELECTION AND JUSTIFICATION

Component	Choice	Justification
Front end	React.js	Component-based interface that can refresh map markers and alert lists when new positions arrive
Back end	Node.js, Express.js	Event-driven, non-blocking I/O suits many small and frequent location requests; Express offers REST-style API routing [9]
Database	MongoDB	Document model holds nested coordinate arrays, checkpoints and trip records without a rigid schema; supports GeoJSON and geospatial indexes
Maps	Google Maps API or OSM	Google provides routing and visualisation services under its terms of use; OpenStreetMap offers open, community-maintained map data [8]
Distance	Haversine formula	Great-circle distance between coordinates that stays well-behaved over short distances [7]
Testing	Postman, GitHub	API testing and version control during development

F. User Roles and Interfaces

The system has three kinds of users each, with an interface (Table III). By keeping the roles separate the passenger view is limited to trip information only while the tasks of route creation, monitoring and handling of alerts stay with the administrator.

TABLE III USER ROLES AND INTERFACES

Role	Functions
Passenger	Route planning and live bus tracking; view of bus number, driver, source, destination, running status and route-deviation status
Driver	Registers a bus with its number, driver name, and source and destination cities
Administrator	Creates routes; lists registered buses with status; sees counts of total buses, active routes and reached buses; receives deviation alerts; views trip history and reports

IV. SYSTEM WORKFLOW

As shown in Fig. 2 The workflow starts with turning on the GPS tracking service. The GPS provides the vehicle location as latitude and longitude. The GPS data is sent to the server. The server keeps the GPS data in the database. Next the route is loaded from the database. The GPS coordinates are then compared continuously with the route coordinates. The distance, between the GPS position and the route path is calculated. If the vehicle goes outside the allowed boundary a deviation is. An alert is sent to the administrator or authorised user.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511146

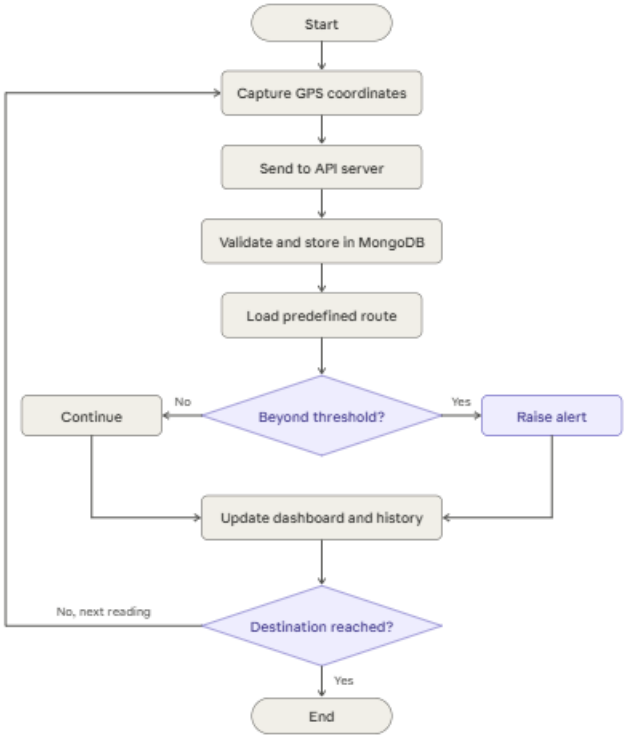


Fig. 2 Workflow of tracking, route comparison, and alert generation.

The dashboard shows where the vehicle is now the path it is taking what trips it has done before and any reports. Tracking continues until the vehicle gets to where it's going or until someone stops the tracking. All the information, about the tracking and the route is saved so that any unexpected events can be looked at after the trip is done.

V. DATA COLLECTION AND METHODOLOGY

GPS data is collected from a module that is mounted on the vehicle or, from a device. This data includes the vehicle location, the route path, speed, travel time and movement status. The data is sent to the backend using APIs and internet communication services. Predefined route data, which includes the source, destination, intermediate points and route boundaries is obtained through the Google Maps API or OpenStreetMap. This route data is stored in MongoDB. The project does not rely on a -labelled public dataset. The exact GPS sampling interval, the test routes and the number of trips must be recorded based on the implementation. The main collections are listed in Table IV.

TABLE IV MAIN DATA COLLECTIONS IN MONGODB

Collection	Main fields
Vehicles	Vehicle number, driver name, source, destination, running status
Routes	Source, destination, route coordinates, checkpoints, intermediate locations, route limits
Tracking log	Vehicle reference, latitude, longitude, timestamp, speed, movement status
Deviations	Vehicle and route reference, time of event, distance from route, alert information

I think the development environment should run on Windows 10 or Linux, The development environment relies on React.js, HTML, CSS and JavaScript for the end and on Node.js and Express.js for the back end, The development environment uses MongoDB as the database, Visual Studio Code as the editor Postman for testing APIs and GitHub for version control., Visual Studio Code as the editor, Postman for testing APIs, and GitHub for version control. I recommend hardware that includes an Intel Core i3 or i5 processor or better least 4 GB of RAM and 100 GB of storage.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511146

A. Deviation Detection Criterion

For every GPS fix the distance d between the vehicle and the stored route path is calculated using the coordinates for example with the haversine great-circle distance [7]. For two points with latitudes φ1, φ2 and longitudes λ1, λ2 (, in radians) the distance is given by

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi 1 \cdot \cos\varphi 2 \cdot \sin^2(\Delta\lambda/2) \qquad (1)$$

$$d_h = 2R \cdot \arcsin(\sqrt{a}) \qquad (2)$$

where R is the mean radius of the Earth. A deviation is flagged when d is greater than a threshold T, that is,

$$d > T \qquad (3)$$

T needs to be picked by thinking about GPS mistakes and how wide the road's. A low value causes wrong warnings while a very high one makes detection slower. As a rule to follow when designing T must be bigger than the GPS position mistake that is expected plus half the road width. The exact number, for T should come from tests instead of guesses. Figure 3 shows the area allowed around the saved path.

B. Distance to the Stored Route

The stored route is a sequence of vertices V0, … Vn. If the distance were measured to the nearest vertex that distance would be overstated whenever vertices are far apart and false alerts could appear between them. Therefore the distance is measured to each route segment VkVk+1. After projecting the coordinates onto a plane around the segment the closest point on the segment, to the fix P is found using

$$t_k = clamp[\,(P - V_k)\cdot(V_k{+}1 - V_k) \,/\, \|V_k{+}1 - V_k\|^2,\, 0,\, 1\,] \qquad (4)$$

$$d = min_k \,\|\, P - (V_k + t_k\,(V_k{+}1 - V_k))\,\| \qquad (5)$$

Across the distance, between neighbouring route vertices I find that local-plane approximation is sufficient. For distances I see that haversine formula (1)–(2) remains the reference. Algorithm 1 summarises the check carried out for each fix.

Algorithm 1. Route deviation check for one GPS fix

```
Input: fix P (latitude, longitude, time), route vertices V0…Vn, threshold T
1: d ← ∞
2: for k = 0 to n−1 do
3:     compute d_k, the distance from P to segment V_k V_k+1
4:     d ← min(d, d_k)
5: end for
6: store the fix in the trip history
7: if d > T then
8:     record deviation (vehicle, route, time, d)
9:     send alert to administrator or authorised user
10: end if
```

D. Evaluation Metrics

The evaluation needs to show how well the system detects deviations during test trips where deviations are already known. It should include the alert rate for trips that stay on the route. Also it should mention how long it takes between a deviation and the alert being sent. The GPS update frequency and how long it takes for the dashboard to update are also Table V). Precision is calculated as TP divided by (TP plus FP). Recall is TP divided by (TP plus FN). Alert delay is the time when the alert is sent minus the time when the first, off-route GPS point was recorded.

TABLE V EVALUATION METRICS

Metric	Definition
Precision	TP / (TP + FP): share of raised alerts that were true deviations
Recall	TP / (TP + FN): share of real deviations that raised an alert
False-alert rate	Alerts raised on trips that followed the route

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511146

Metric	Definition
Alert delay	Time of alert − time of first off-route fix
GPS update interval	Time between consecutive position fixes
Dashboard refresh delay	Time from a fix being stored to its display on the map

E. Evaluation Procedure

To convert the criterion in (3) into measured results, the tracking-to-alert workflow should be tested on documented routes in the following order:

1. Choose test routes. Write down each test routes vertices, length and road type so that the results can be reproduced.
2. Run trips that follow the test route to obtain the alert rate and run trips that depart deliberately from the test route and log those departures to obtain precision and recall.
3. Repeat the trips for values of T and pick the value of T that balances false alerts, against detection delay.
4. Record the GPS update interval and the timestamps of the off-route fix the alert and the dashboard update so that alert delay and refresh delay can be calculated.

VI. EXPERIMENTAL RESULTS

I am excited to share that Project Stage-I delivered system requirements, technology selection, architecture, workflow design and a built prototype. Basic GPS tracking and live location display were tested with mapping services and database connectivity and API integration were confirmed. The prototype contains passenger interface, driver interface and admin interface. The passenger interface supports route planning and live bus tracking. Shows bus number, driver, source, destination, running status and route-deviation status for example a bus on the Pune–Nashik route shown as running on the correct route. The driver interface registers a bus with bus number, driver name, source city and destination city and the admin dashboard lists registered buses, with status and counts of buses, active routes and reached buses.

TABLE VI IMPLEMENTATION STATUS

Component	Status	Quantitative result
GPS tracking and live map	Basic function tested	None claimed
Database and API integration	Connectivity confirmed	None claimed
Dashboard (passenger, driver, admin)	Partially implemented	None claimed
Deviation detection and alerts	Designed; testing planned	None claimed

The material that is available does not have numbers to show detection precision, recall, false-alert rate or alert delay so no performance numbers are being stated. These numbers, along with the pictures that show the matching results should come from experiments. The whole process from tracking to sending an alert needs to be tested on routes.

Problems caused by losing GPS signals being in cities with buildings choosing the right thresholds and delays in the network need to be talked about along, with these results. Table VII shows how each problem affects detection and what the design needs to handle; these are things to think about when designing and not results that have been measured.

TABLE VII ERROR SOURCES AND DESIGN CONSIDERATIONS

Error source	Effect on detection	Design consideration
GPS signal loss	Missing or stale fixes; late alerts	Keep the timestamp of each fix and show the last known position as stale
Urban canyons	Position error [5] can push a fix outside the band	Choose T above the expected position error
Threshold selection	Small T: false alerts; large T: delayed detection	Fix T from measured on-route and off-route trips



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511146

Error source	Effect on detection	Design consideration
Network delay	Alert and dashboard lag behind the vehicle	Record the time of the fix at the device and measure refresh delay
Legitimate detours	Traffic or road closures raise false alerts	Let the administrator review the event in the alert history

VII. DISCUSSION: NOVELTY AND LIMITATIONS

The main contribution at project level is to combine live GPS tracking with comparison to a predefined route alert generation and trip-history reporting all on one web platform.I see that this fills the gap mentioned in Section II, where the reviewed systems [1] [2] [6] track or optimise vehicles but do not check each position against an assigned route.I believe that any claim of novelty or superiority, over existing systems would need a verified literature comparison and reproducible evidence.

A. Application Scenarios

We see that the same detection logic can be used in the settings named in Section I. For school buses, an alert when a bus leaves its planned route addresses the safety concern of students and parents. In logistics and delivery route compliance limits fuel misuse, delays and private use of vehicles. In transport an administrator can supervise many buses together instead of following each path by hand. In emergency services a record of departures, from the route supports review of response time.

B. Limitations and Future Work

These limitations can involve GPS inaccuracy [5] loss of signal or connectivity sensitivity, to the chosen threshold distance detours caused by traffic or road closures that may trigger false alerts and the fact that the present work is only a Stage-I prototype. Future work may add intelligence and predictive analytics, mobile applications and cloud-based services to improve performance, scalability and decision-making. Future work will also consider ways to reduce these limitations.

VIII. CONCLUSION

I am excited to share a web-based system, for vehicle tracking and route deviation detection that I have built on GPS, digital maps and a stack that includes React.js, Node.js, Express.js and MongoDB. The web-based system compares vehicle coordinates with a predefined route automatically detects deviations sends alerts and stores trip history for analysis. Stage-I produced the requirements, the design and a partial prototype and the effectiveness of the system should be reported using verified experimental measurements and documented test conditions.

IX. DATA AND CODE AVAILABILITY

Data: GPS coordinates are gathered from a device that is mounted in a vehicle or from a phone and route information is obtained from Google Maps API or OpenStreetMap; there is no use of any public dataset. The map service, the rules, for using it and the test-trip data that was collected need to be checked before anything is published. Code repository: share a link. Say if it is available.

ACKNOWLEDGMENT

The authors would like to thank Prof. Vishnudas Talware for his guidance. They also extend their gratitude to Dr. P. S. Sanjekar, the Project Stage-I Coordinator Dr. U. M. Patil, the Head of Department and Dr. J. B. Patil, the Director. Special thanks go to the Department of Computer Science & Engineering (Data Science) at R. C. Patel Institute of Technology Shirpur for their support and the facilities provided.

REFERENCES

[1]. R.. M. Kumar, "Real Time Vehicle Tracking System using GPS and GSM Technology " International Journal of Computer Applications volume 179 number 3 pages 15 to 19, 2017.
[2]. A. Sharma and S. Jain, "Smart Vehicle Monitoring and Tracking System using IoT " International Journal of Engineering Research & Technology (IJERT) volume 8 number 5 pages 234, to 238 2019.
[3]. J.. A. Voisard, Location-Based Services. Morgan Kaufmann Publishers, 2004.



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511146

[4]. S. Thrun, W. Burgard and D. Fox, Probabilistic Robotics. MIT Press, 2005.
[5]. P. Misra and P. Enge, Global Positioning System: Signals, Measurements and Performance. Ganga-Jamuna Press, 2011.
[6]. T. Anderson and K. Smith "Intelligent Fleet Management Using GPS Tracking " International Journal of Transportation and Logistics Management volume 12 number 4 pages 145 to 152, 2021.
[7]. R. W. Sinnott, "Virtues of the Haversine " Sky and Telescope volume 68 number 2 page 159 1984.
[8]. M. Haklay and P. Weber "OpenStreetMap: User-Generated Street Maps " IEEE Pervasive Computing, volume 7 number 4 pages 12 to 18, 2008.
[9]. R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures " Ph.D. Dissertation, University of California, Irvine, 2000.

 This work is licensed under a Creative Commons Attribution 4.0 International License