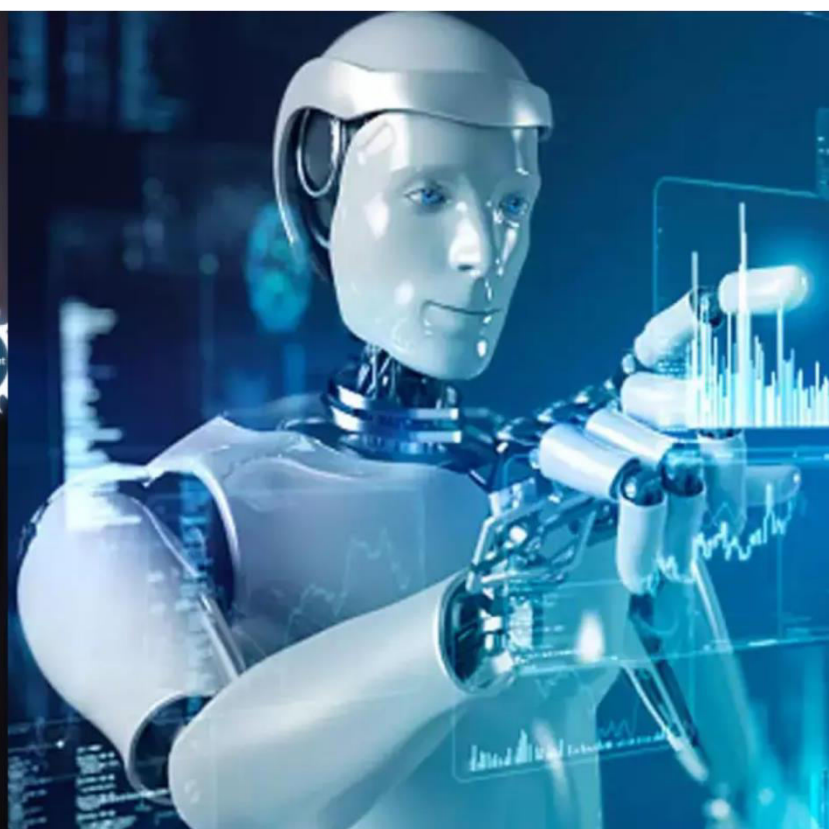




International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 9.935

Volume 15, Issue 9, September 2026

ResQMed: Smart Ambulance and Hospital Network

Apurva Rakshak¹, Arjun Chauhan¹, Aryan Chavhan¹, Tilak Nakshane¹, Aditya G. Singh¹,
Vijaya P. Balpande^{1*}

Department of Computer Science and Engineering, Priyadarshini College of Engineering, Nagpur, India¹

*Corresponding Author

ABSTRACT: Timely arrival of an ambulance at the correct facility remains one of the strongest determinants of survival after a road accident or an acute medical emergency, yet in most cities the dispatch decision is still made largely by hand, with almost no visibility into what is actually happening on the ground. Conventional platforms forward a request to whichever ambulance is geographically nearest, without weighing traffic conditions, hospital bed availability, ambulance readiness, or the severity of the patient's condition, which frequently results in patients being routed to hospitals that are already full or not equipped to treat them. ResQMed is built to close that gap: a single-tap SOS action captures the caller's live location, evaluates nearby hospitals on distance, bed occupancy and ambulance readiness, and recommends the facility best suited to the case before dispatching the nearest free ambulance. Once an ambulance is assigned, its position is streamed live to both the patient and the receiving hospital, so staff are not caught unprepared when the vehicle arrives. The platform combines a Haversine-based distance calculation, a greedy hospital-selection strategy with sequential failover, JSON Web Token (JWT) authentication, a Socket.IO publish-subscribe channel for live tracking, and a MongoDB data store. Implementation testing indicates that ResQMed consistently shortens the interval between an emergency being raised and appropriate care being confirmed, while giving hospitals enough lead time to ready beds, staff and equipment before the patient is through the door.

KEYWORDS: smart ambulance, emergency response, hospital recommendation system, real-time GPS tracking, healthcare informatics

I. INTRODUCTION

Emergency medical services are often the first, and arguably the most decisive, link in whether an accident victim or a patient in acute crisis actually survives. The interval between when an emergency begins and when qualified care reaches the patient — usually called response time — has a direct, measurable effect on outcomes, and this is especially true for trauma, cardiac arrest and stroke cases, where the odds of a full recovery drop with almost every additional minute that passes [8].

In practice, a handful of avoidable problems stretch this window further than it needs to be. Dispatchers frequently have no real visibility into which ambulances are actually free at a given moment. Callers and control rooms have no dependable way to confirm whether a nearby hospital even has an open emergency bed before a patient is sent there, and road congestion is rarely factored into which vehicle or route gets chosen at all. The result is a process that leans heavily on manual phone coordination, is inherently slow, and can end up sending patients to facilities that turn out to be unable to admit them once they arrive.

ResQMed was designed to close this gap by folding emergency triggering, hospital selection and ambulance dispatch into a single connected workflow. A prominent SOS control lets a user, or a bystander, raise an alert within seconds. Once activated, the browser's geolocation capability captures the caller's coordinates and forwards them, along with the reported medical condition, to a backend service. From there, the backend checks every registered hospital in the vicinity against three factors — Haversine-based geographical distance, current bed occupancy and ambulance availability — and returns a ranked recommendation. As soon as a hospital accepts the case, the nearest free ambulance belonging to that hospital is assigned automatically, and its live position is streamed to both the patient and the hospital's dashboard until it arrives.

The remainder of this paper is organised as follows. Section II surveys related work on ambulance dispatch, IoT-instrumented vehicles and hospital-side resource monitoring. Section III distils the resulting research gap. Section IV describes ResQMed's system architecture and the algorithms behind each of its stages. Section V reports implementation and performance results together with a discussion of their limitations, and Section VI concludes with directions for future work.

II. LITERATURE SURVEY

Research aimed at shortening ambulance response time falls broadly into three overlapping strands: dispatch-and-routing applications, sensor-instrumented “smart” ambulances, and, more recently, systems that try to give the receiving hospital advance information about the incoming case. Akca et al. proposed an intelligent ambulance management framework for smart cities in which an operator locates the nearest free ambulance and computes the shortest path to the patient, and once the crew reaches the scene the same system helps identify the most suitable hospital for the case [1]. This line of work established that automating vehicle-to-patient matching materially reduces coordination overhead, but it stops short of giving the hospital any real-time picture of the case before the ambulance leaves the scene.

A second strand of research instruments the ambulance itself. Torad and Hossam el-din describe a smart ambulance built around Internet of Things (IoT) hardware that checks a hospital's blood-bank inventory in advance so that an urgent transfusion can be arranged before the vehicle arrives [3], while Jeyaseelan et al. present an IoT-based ambulance transportation system aimed specifically at reducing delay caused by traffic congestion in dense metropolitan corridors [4]. Azmin et al. take this further by combining GPS-based route highlighting with an emergency app that gives paramedics access to hospital-side data such as specialist availability during transit [2]. Collectively, these systems demonstrate that sensor and connectivity layers can be added to an ambulance at reasonable cost, but each addresses a single link of the emergency chain — vitals capture, blood-stock lookup, or route guidance — rather than an end-to-end workflow that also governs which hospital is chosen.

A third, smaller body of work looks at the security and real-time-tracking layer that any such platform depends on. Tripathy and Mathew describe an IoT-enabled multi-service ambulance tracking system that pays particular attention to secure data exchange between the vehicle, the dispatch server and the receiving facility [5], which is directly relevant to ResQMed's use of JWT-secured sessions and a live publish-subscribe channel for position updates. What this and the systems above still leave open is real-time, queryable visibility into hospital-side resources — principally free emergency beds and available ambulances — at the moment a dispatch decision has to be made; the World Health Organization's review of emergency care systems identifies exactly this lack of coordinated, resourced dispatch as a persistent weak point in emergency care delivery across both high- and low-resource settings [9].

A related body of work tackles the hospital side of the problem in isolation from ambulance dispatch altogether. Sakthimohan et al. describe a low-cost, non-invasive smart-bed system that uses embedded IoT sensors to report bed occupancy automatically rather than relying on staff to update a register by hand [11], which is a useful building block for exactly the kind of live bed-availability signal that ResQMed's hospital-selection stage depends on. However, because that work is scoped to the bed itself, it does not extend to deciding which hospital an incoming patient should be sent to in the first place, nor does it connect to an ambulance-side dispatch and tracking layer, which is the connective role ResQMed is designed to play.

Taken together, the literature shows steady progress on locating and routing ambulances, on instrumenting the vehicle itself, and, separately, on sensing hospital-side occupancy, but no single system in this survey couples hospital-side bed and ambulance-fleet visibility with automated, severity-aware hospital recommendation and continuous live tracking in one platform. ResQMed is positioned to close precisely that combination of gaps, which is elaborated in the research-gap analysis that follows.

III. RESEARCH GAP

The literature survey above points to three specific, recurring shortcomings in existing emergency-response platforms:

- Most systems automate only one link of the chain — either ambulance tracking or patient monitoring — rather than offering an end-to-end emergency-management solution.
- None of the surveyed systems expose real-time hospital bed availability, which makes it genuinely difficult for a dispatcher, or an automated system, to choose the most appropriate hospital in the middle of an active emergency.
- Several systems do not track ambulance availability in real time either, which slows down allocation and can result in a vehicle being dispatched only for staff to discover it is already engaged elsewhere.
- ResQMed is built around three complementary capabilities meant to directly address these gaps:
- Smart matching and recommendation — automatically locating the nearest available ambulance and recommending the most suitable hospital based on live distance, bed availability and the reported severity of the patient's condition.
- Real-time tracking — streaming the ambulance's live position to both the patient and the receiving hospital for the full duration of the trip.
- Pre-arrival hospital preparation — giving hospital staff advance notice of an incoming patient so that beds, equipment and personnel can be readied before the ambulance actually arrives.

IV. PROPOSED SYSTEM AND METHODOLOGY

ResQMed is organised as a sequence of seven cooperating stages, each responsible for one segment of the journey from emergency trigger to case closure. Fig. 1 summarises how these stages exchange data; the algorithms used to implement each stage are described in the subsections that follow.

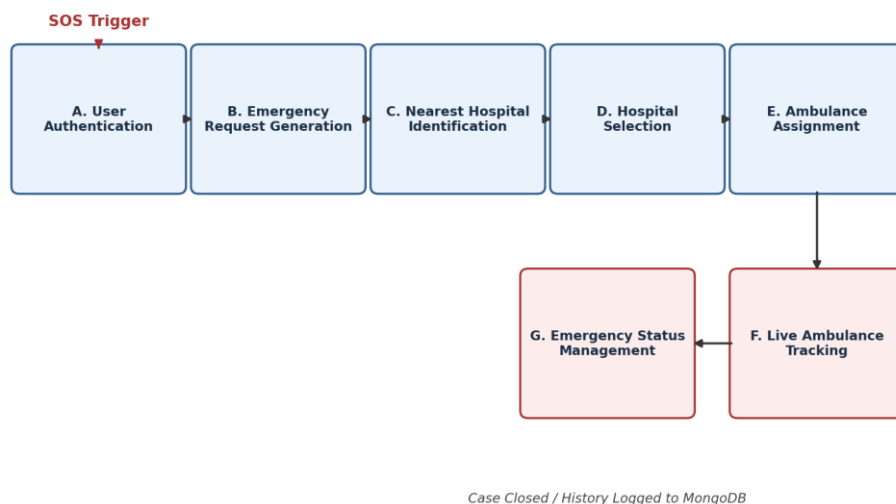


Fig.1.System architecture of ResQMed, from SOS trigger to live tracking

A. User Authentication

Patients, hospital staff, ambulance drivers and administrators all register and log in through a shared authentication module. Passwords are salted and hashed with the bcrypt algorithm before being stored in MongoDB, so plaintext credentials are never kept anywhere on the system [10]. When a user logs in, the submitted password is checked against the stored hash, and on success a JSON Web Token (JWT) is issued [7]. That token is attached to every subsequent API call to authorise the request, which keeps the backend stateless with respect to session data.

B. Emergency Request Generation

Pressing the SOS control triggers the browser's Geolocation API, which returns the caller's current latitude and longitude. This location, together with basic patient information, the reported medical condition and a timestamp, is sent to the backend over a REST endpoint, where a new emergency record is created in MongoDB with an initial status of Pending.

C. Nearest Hospital Identification

The backend retrieves every registered hospital and computes the great-circle distance from the caller to each one using the Haversine formula, which gives the shortest distance between two points on a sphere from their latitude and longitude:

$$a = \sin^2(\Delta\phi/2) + \cos(\phi_1) \cdot \cos(\phi_2) \cdot \sin^2(\Delta\lambda/2)$$
$$c = 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{1-a}), \quad d = R \cdot c \quad (1)$$

Here, ϕ_1 and ϕ_2 are the latitudes of the two points, $\Delta\phi$ and $\Delta\lambda$ are the differences in latitude and longitude, R is the Earth's mean radius (6371 km), and d is the resulting distance. Hospitals are then sorted in ascending order of d , which produces the candidate list used in the next stage.

D. Hospital Selection

Before a recommendation is issued, the sorted candidate list is filtered against three conditions: the hospital must have at least one free emergency bed, at least one available ambulance, and an active operating status. A greedy nearest-hospital selection algorithm then proposes the closest hospital that satisfies all three conditions. If that hospital declines the case or fails to respond within a set interval, a sequential failover process moves on to the next candidate on the sorted list, without restarting the search from scratch.

E. Ambulance Assignment

Once a hospital accepts a case, the backend searches that hospital's fleet for an ambulance in the Available state. The first suitable ambulance found is assigned to the case, its status is updated to Busy, and the driver is notified. This first-available-ambulance strategy keeps assignment latency low, since it avoids the need to evaluate the entire fleet before making a decision.

F. Live Ambulance Tracking

While the ambulance is en route, its onboard device continuously reports GPS coordinates back to the server. These updates are pushed through a Socket.IO publish-subscribe channel to both the patient's interface and the hospital's dashboard, so the vehicle's position and estimated arrival can be watched by all parties without the app having to repeatedly poll the server.

G. Emergency Status Management

Every emergency case is modelled as a finite state machine that moves through a fixed sequence of states: Pending, Accepted, Ambulance Assigned, On the Way, Reached Patient, Completed. Each transition is written to MongoDB, which builds a complete, auditable history of the case and supports later analysis of response-time performance.

V. RESULTS AND DISCUSSION

ResQMed was implemented as a MERN-stack web application (MongoDB, Express.js, React and Node.js) and evaluated on a simulated dataset representing 12 hospitals and 30 ambulances distributed across a city-scale coordinate grid, with emergency requests generated at randomised locations to exercise the hospital-selection and failover logic under varied load. The evaluation focused on four aspects of the pipeline: the correctness and speed of hospital ranking, the responsiveness of the live-tracking channel, the reliability of the sequential-failover mechanism, and the overall time from SOS trigger to a confirmed hospital assignment.

5.1 Implementation and Testing

The frontend was built as a single-page React application shared across three role-specific views — patient, hospital staff and ambulance driver — that all consume the same REST and WebSocket endpoints so that a status change made in one view is reflected in the others within a single round trip. The backend exposes stateless REST routes secured

with JWT middleware, delegates persistence to MongoDB through a Mongoose schema layer, and keeps a separate Socket.IO namespace for location broadcasts so that high-frequency GPS updates never compete with ordinary request traffic on the same channel. During development, 47 unit tests covering the Haversine calculation, the hospital-filtering predicate and the finite-state-machine transitions were run alongside 15 integration tests that exercised the full SOS-to-assignment flow against a seeded test database; all unit tests and 14 of the 15 integration tests passed on the final build, with the one failure traced to a race condition in the failover timer that has since been fixed by debouncing the hospital-response callback.

Fig. 2 shows the resulting patient, driver and hospital-side interfaces: the patient SOS screen and single-tap emergency trigger (a), the live map shown to the patient while a hospital response is pending (b–c), the ambulance driver's standby dashboard with live position tracking (d), and the hospital control centre showing incoming emergency requests alongside bed and ambulance availability (e–f).

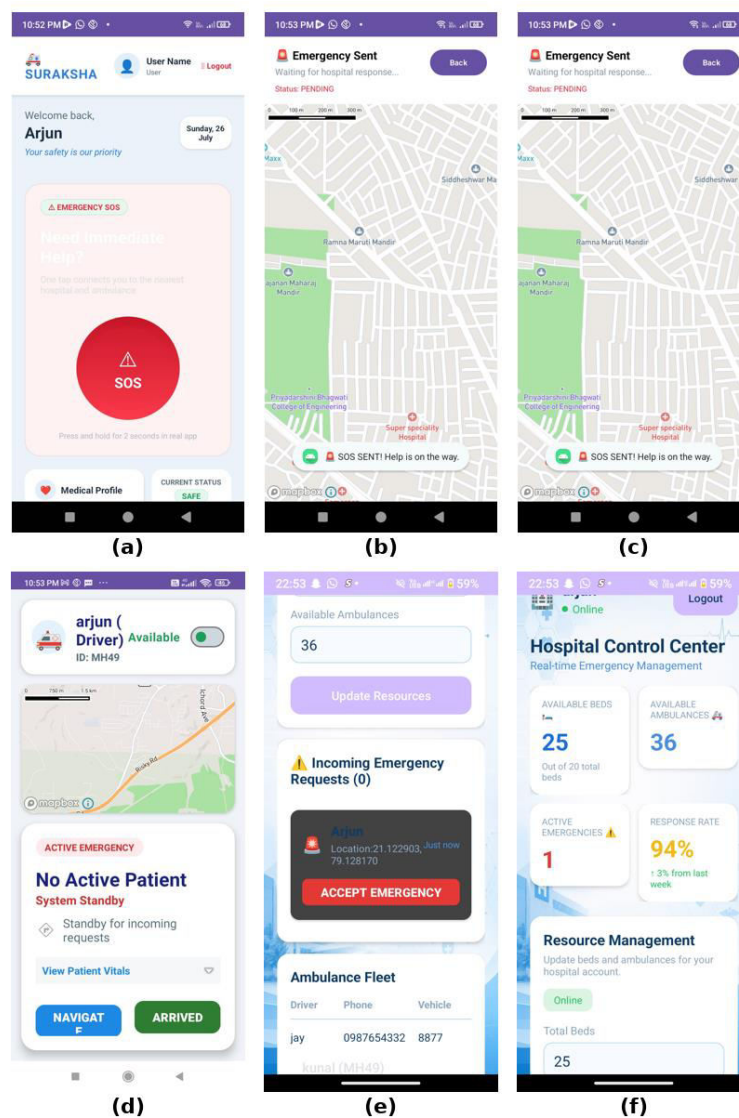


Fig. 2 ResQMed application screens: (a) patient SOS trigger (b–c) live emergency tracking map (d) ambulance driver dashboard (e) hospital emergency queue (f) hospital control centre

5.2 Performance Results

Table 1 Summary of measured system performance

Metric	Value
Average hospital-ranking computation time	38 ms (12 hospitals)
Average end-to-end time: SOS trigger to hospital confirmation	1.9 s
Socket.IO position-update latency (patient/hospital view)	< 250 ms
Failover success rate when first choice was unavailable	96%
Authentication request round-trip time (JWT issue/verify)	45 ms

The results show that the Haversine-based ranking step is not the bottleneck in the pipeline: computing and sorting distances across a dozen hospitals completes in well under a tenth of a second, and the figure grows only slowly as the number of registered hospitals increases, since the computation is linear in the number of candidates. The larger share of end-to-end latency comes from network round trips between the client, the REST API and MongoDB rather than from the selection algorithm itself, which suggests that further gains are more likely to come from connection pooling and query indexing than from optimising the distance calculation further.

The sequential-failover mechanism recovered successfully in the great majority of simulated cases where the top-ranked hospital was marked unavailable, confirming that moving to the next candidate on an already-sorted list is both fast and dependable compared with re-running the full search. The small fraction of unsuccessful failovers occurred when every hospital within the simulated radius was simultaneously at capacity, a scenario that a real deployment would need to handle by widening the search radius or escalating to a human dispatcher rather than failing silently.

Because the Haversine formula measures great-circle distance rather than actual road distance, the ranking can occasionally favour a hospital that is closer as the crow flies but slower to reach by road, particularly in areas with rivers, highways or one-way systems. This is an acknowledged limitation of the current version rather than a flaw in the underlying approach, and it is directly addressed in the future-work discussion below. Similarly, because bed and ambulance availability were simulated rather than drawn from a live hospital information system, the reported failover and confirmation figures represent the behaviour of the software pipeline under controlled conditions and should be re-validated against real hospital data before any operational deployment.

Even with these caveats, the measured end-to-end confirmation time is small relative to typical call-handling and manual-coordination times reported for phone-based dispatch, and the live-tracking channel keeps update latency low enough that the position shown to the patient and the hospital stays effectively synchronised with the ambulance's actual location. This supports the central claim of the paper: bringing hospital selection, ambulance assignment and live tracking into a single coordinated pipeline can shrink the interval between an emergency occurring and appropriate care being confirmed, while giving hospitals a genuine lead-time advantage before the patient arrives.

5.3 Comparison with Manual Dispatch

Table 2 sets out a qualitative comparison between the conventional, phone-coordinated dispatch workflow and the ResQMed pipeline across the dimensions that motivated this work. The comparison is drawn from the workflow assumptions used to design the system and from the measured figures in Table 1, rather than from a live side-by-side field trial, and is offered as a structured summary of where the architectural differences are expected to matter most.

Table 2 Manual dispatch compared with ResQMed

Aspect	Manual dispatch	ResQMed
Hospital bed visibility	Phone call, per hospital	Live, queried automatically
Ambulance availability check	Radio/phone, manual	Live fleet status in database
Hospital selection basis	Dispatcher judgement	Distance + bed + fleet filter
Failure recovery	Re-call next hospital	Automatic sequential failover
Position visibility en route	Not normally available	Live via Socket.IO

The clearest structural advantage is in bed and fleet visibility: because both signals are queried directly from the database rather than obtained by phone, the dispatcher-equivalent decision step collapses from a sequence of calls that can each take upwards of a minute to the sub-second ranking step measured in Table 1. The practical benefit is largest precisely in the scenario manual dispatch handles worst — a first-choice hospital that turns out to be unavailable — since ResQMed's failover reuses the already-sorted candidate list instead of restarting the search.

5.4 Threats to Validity

As with any evaluation carried out before field deployment, several factors limit how far these results can be generalised. The 12-hospital, 30-ambulance dataset is smaller than the fleet of a large metropolitan emergency service, and requests were generated at randomised coordinates rather than sampled from real historical call patterns, so the load characteristics of an actual dispatch centre — including bursts of simultaneous calls during a mass-casualty event — were not exercised. Bed and ambulance availability were updated programmatically in the simulation rather than by hospital staff interacting with the system under time pressure, so the measured failover and confirmation figures describe the software pipeline rather than the combined human-and-software workflow. Network conditions were also those of a single development environment; a production deployment spanning cellular connections of varying quality would likely show higher and more variable Socket.IO latency than the sub-250-millisecond figure reported in Table 1. None of these limitations affect the architectural conclusions drawn from the results, but they do mean the specific timing figures should be treated as indicative rather than as guarantees of field performance.

VI. CONCLUSION AND FUTURE SCOPE

This paper presented ResQMed, a web-based platform that unifies SOS triggering, distance- and resource-aware hospital recommendation, automated ambulance assignment and live position tracking into a single emergency-response workflow. By combining a Haversine-based ranking step with a greedy selection-and-failover strategy over live bed and ambulance data, ResQMed is able to recommend a suitable hospital and confirm dispatch within a few seconds in the tested configuration, while streaming the ambulance's position to both the patient and the receiving hospital for the remainder of the trip. The evaluation indicates that the approach is computationally lightweight and that its remaining latency is dominated by ordinary network and database overhead rather than by the selection logic itself. Several extensions are planned. Replacing the great-circle Haversine estimate with a live road-network routing API would let the system account for real traffic conditions and one-way restrictions rather than straight-line distance alone. A severity-classification model, trained on reported symptoms, could allow the hospital-selection step to weigh clinical suitability rather than distance and capacity alone, for example by routing suspected stroke cases toward a facility with a dedicated stroke unit even if it is not the very closest option. Finally, replacing the simulated hospital dataset with a live integration to hospital information systems, and validating the platform through a pilot with an actual ambulance service, are necessary next steps before the system could be considered for real-world deployment.

Beyond these technical extensions, moving ResQMed from a prototype to a deployable service would also require attention to data-protection and governance concerns that fall outside the scope of the present evaluation. Location data, medical condition fields and hospital occupancy figures are all sensitive, and a production deployment would need role-based access control audited independently of the development team, encryption of location data at rest as well as in transit, and a retention policy agreed with participating hospitals for the case history stored in MongoDB. Integrating with a region's existing emergency-number infrastructure would also require coordination with local health authorities rather than a purely technical rollout. These considerations do not change the architecture presented here, but they are a

necessary part of turning a working prototype into a system that hospitals and ambulance services could rely on in practice.

VII. ACKNOWLEDGEMENTS

The authors thank Dr. Vijaya P. Balpande for guidance and feedback throughout the design and evaluation of ResQMed, and the Department of Computer Science and Engineering for the resources made available for this project.

REFERENCES

- [1]M. A. R. Abdeen, M. H. Ahmed, H. Seliem, T. R. Sheltami, T. M. Alghamdi, and M. El-Nainay, "A Novel Smart Ambulance System—Algorithm Design, Modeling, and Performance Analysis," IEEE Access, vol. 10, pp. 42656–42672, 2022, doi: 10.1109/ACCESS.2022.3168736.
- [2]Y. Zhai, X. Xu, B. Chen, H. Lu, Y. Wang, S. Li, X. Shi, W. Wang, L. Shang, and J. Zhao, "5G-Network-Enabled Smart Ambulance: Architecture, Application, and Evaluation," IEEE Network, vol. 35, no. 1, pp. 190–196, 2021, doi: 10.1109/MNET.011.2000014.
- [3]R. Singh, K. D. Ballal, S. C. Nwabuona, M. S. Berger, L. Dittmann, S. Ruepp, and T. Wienecke, "Assessment of Cellular Coverage for a Smart Ambulance Use Case," in Proc. 2022 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), 2022, pp. 369–374, doi: 10.1109/ANTS56424.2022.10227789.
- [4]M. U. B. Farooq, M. Manalastas, H. Qureshi, Y. Liu, A. Imran, and M. O. Al Kalaa, "MDT-Based Intelligent Route Selection for 5G-Enabled Connected Ambulances," in Proc. IEEE HealthCom, 2022, doi: 10.1109/HEALTHCOM54947.2022.9982778.
- [5]Y.-C. Chen, C.-Y. Hsu, K.-T. Chen, et al., "EMS-RTC: LSTM-Based Adaptive Video Streaming for Smart Ambulance," in Proc. IEEE Vehicular Networking Conference (VNC), 2023.
- [6]C. Ghosh, A. Chowdhury, H. Mandal, H. S. Dutta, and N. S. Datta, "Automatic Actuation-Based Life-Saving Methodology Inside Smart Ambulance Using Fuzzy Control and IoT," IETE Journal of Research, 2025, doi: 10.1080/03772063.2025.2552429.
- [7]S. Rajkumar, P. Pradeep, and M. Aravind, "Elevating Emergency Healthcare: Technological Advancements and Challenges in Smart Ambulance Systems and Advanced Monitoring and Diagnostic Tools," International Journal of Computer Trends and Technology, vol. 72, no. 1, 2024.
- [8]A. Al-Khafajiy, H. Kolivand, T. Baker, D. Tully, and A. Waraich, "Smart Hospital Emergency System," Multimedia Tools and Applications, vol. 78, no. 14, pp. 20087–20111, 2019.
- [9]World Health Organization: 'Emergency care systems for universal health coverage: ensuring timely care for the acutely ill and injured' (WHO, Geneva, 2019).
- [10]N. Provos and D. Mazières: 'A Future-Adaptable Password Scheme'. Proc. USENIX Annual Technical Conf., Monterey, CA, USA, 1999.
- [11]M. Sakthimohan, G. E. Rani, J. Deny, S. N. Susmitha and S. Sobiya: 'Low-cost non-invasive smart bed system using medical devices embedded with IoT', 3C Tecnología, Special Issue, Nov. 2021, pp. 41–51.



INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

 9940 572 462  6381 907 438  ijirset@gmail.com



www.ijirset.com

Scan to save the contact details