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 10, October 2026

DOI: 10.17148/IJARCCE.2026.1510137

RESQMESH: AN OFFLINE EMERGENCY COMMUNICATION AND RESCUE COORDINATION SYSTEM

Pushpalatha M[1], Pavitra N[2], Ramyashree S S[3], Nithya S[4], Sirisha D[5], Varshini R[6]

Assistant Professor, Department of computer science, S.J.M. Institute of Technology, Chitradurga, Karnataka, India[1]

Assistant Professor, Department of computer science, S.J.M. Institute of Technology, Chitradurga, Karnataka, India[2]

B.E. Students, Department of Computer Science and Engineering, S.J.M. Institute of Technology, Chitradurga, Karnataka, India[3-6]

Abstract: During disasters and other emergencies, cellular networks and Internet services may become unavailable because of infrastructure damage, congestion, power failure, or limited coverage. This paper presents ResQMesh, an offline-first emergency communication and rescue coordination system that uses nearby Android devices to exchange emergency information through Bluetooth-based communication. Emergency requests contain details such as emergency type, priority, location, battery level, and contact information. Records are stored locally using the Room persistence library and can be forwarded between participating devices. When Internet connectivity becomes available, unsynchronized records are transferred to Firebase Firestore. A web-based administration panel enables authorized administrators to monitor recent, active, and completed cases, inspect case details, and update rescue status. Functional and integration tests documented in the project report covered emergency creation, local storage, device discovery, communication, forwarding, duplicate prevention, synchronization, authentication, and case management; the reported test cases passed. ResQMesh is intended to support emergency information transfer and coordination, not to replace professional emergency services or guarantee communication in every environment.

Keywords: ResQMesh, emergency communication, offline-first system, Bluetooth, mesh forwarding, Room Database, Firebase Firestore, rescue coordination.

I. INTRODUCTION

Timely communication is essential during natural disasters, accidents, and medical emergencies. However, conventional communication services may become unavailable when network infrastructure is damaged, mobile networks are congested, or users are in areas with poor coverage. If an emergency request cannot reach a centralized service, response and coordination may be delayed.

ResQMesh is an emergency communication and rescue coordination system designed to support local handling and transfer of emergency information without requiring every participating device to have an active Internet connection. The Android application creates emergency requests and exchanges information with nearby devices through Bluetooth. Emergency data is retained locally and synchronized with Firebase Firestore when Internet connectivity returns. A separate web-based Admin Panel provides centralized case monitoring and management.

II. RELATED WORK

Research on disaster communication has explored wireless mesh networks, mobile ad hoc networks, Bluetooth-based communication, delay-tolerant networking, intelligent routing, and emergency dashboards. Abidde et al. [1] discuss optimization of wireless mesh networks for disaster-response communication, while Chaitrashree et al. [2] examine a disaster-resilient mesh network with AI-based load balancing. Rondon et al. [3] study reliability, delay, and scalability considerations in Bluetooth Mesh. WiMesh and BLUEMERGENCY describe approaches to communication in disaster scenarios using mesh networking [4], [5].

Delay-tolerant networking provides a store-and-forward model for environments with intermittent connectivity [6]. Other work explores reinforcement-learning-based routing [7], sensor-based disaster management [8], emergency dashboards [9], and broader emergency communication architectures [10]. ResQMesh brings together offline emergency creation,

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 10, October 2026

DOI: 10.17148/IJARCCE.2026.1510137

nearby-device transfer, local persistence, cloud synchronization, and centralized case management in one project implementation.

III. PROBLEM STATEMENT AND OBJECTIVES

Traditional emergency applications often rely on cellular networks, Internet access, or a centralized server. During network outages, emergency information created on an offline device may not immediately reach the server. The project therefore aims to provide an offline-first mechanism for creating and storing emergency requests, transferring them between nearby devices, and synchronizing them when connectivity becomes available.

The main objectives are to develop an Android emergency application; support Bluetooth-based device-to-device communication and multi-hop forwarding; preserve records locally using Room; synchronize records with Firebase Firestore; prevent duplicate emergency records; and provide an authenticated web panel for monitoring cases and updating rescue status.

IV. PROPOSED SYSTEM ARCHITECTURE

ResQMesh consists of two main components: an Android application for users and a web-based Admin Panel for authorized administrators. The mobile application collects emergency information, creates a unique emergency record, stores it locally, discovers nearby participating devices, and sends or receives emergency information through Bluetooth. Received records are checked using their emergency identifier to reduce duplicate storage and forwarding.

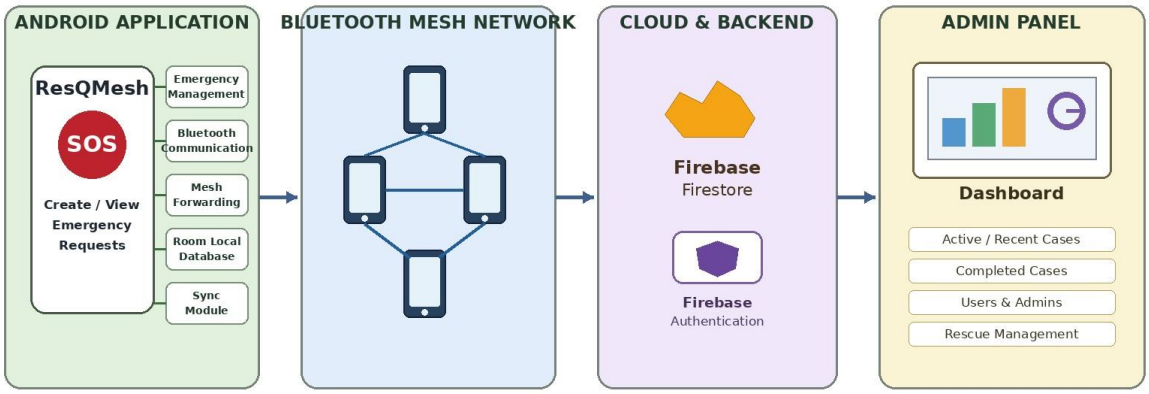


Fig. 1. ResQMesh system architecture showing the Android application, Bluetooth mesh forwarding, Firebase backend, and Admin Panel.

A. Android Emergency Application

The Android application supports user information and emergency-contact details, emergency type and priority selection, location and battery information, emergency history, and local emergency storage. The mobile application is developed using Kotlin and Jetpack Compose. Room provides local persistence, allowing records to remain available during temporary Internet unavailability.

B. Bluetooth and Mesh Forwarding

Nearby participating devices can discover one another and exchange emergency information through Bluetooth communication. A receiving device stores the record locally and may forward it to another suitable device. Emergency identifiers and forwarding information help prevent repeated processing. If forwarding fails, the report describes the emergency as remaining stored locally rather than being discarded.

C. Cloud Synchronization and Admin Panel

When Internet connectivity becomes available, unsynchronized emergency records are transferred to Firebase Firestore and their synchronization state is updated. Firebase Authentication is used for administrator login. The React-based Admin Panel displays dashboard statistics and recent, active, and completed cases, supports viewing emergency details, and allows authorized users to update rescue status and manage users or administrator accounts according to their privileges.

 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 10, October 2026

DOI: 10.17148/IJARCCE.2026.1510137

V. METHODOLOGY

The system workflow begins when a user creates an emergency request. The application records an emergency identifier, user and contact details, emergency type, priority, available location, battery level, timestamp, and status. The record is saved in the Room database before network-dependent synchronization is attempted. The application then searches for nearby participating devices and transfers the emergency information when a suitable connection is available.

A receiving device checks whether the emergency identifier has already been recorded. New records are stored locally and can be forwarded as appropriate. Local records remain available while the device is offline. When Internet connectivity returns, the synchronization module identifies unsynchronized records and uploads them to Firebase Firestore. The Admin Panel retrieves cloud records for authorized monitoring and case management.

A. Technology Stack

The project report identifies Kotlin and Jetpack Compose for Android development, Bluetooth for nearby-device communication, Room for local storage, Firebase Firestore for cloud data, Firebase Authentication for administrator authentication, and React with Vite and JavaScript/JSX for the web Admin Panel. Git is used for source-code management.

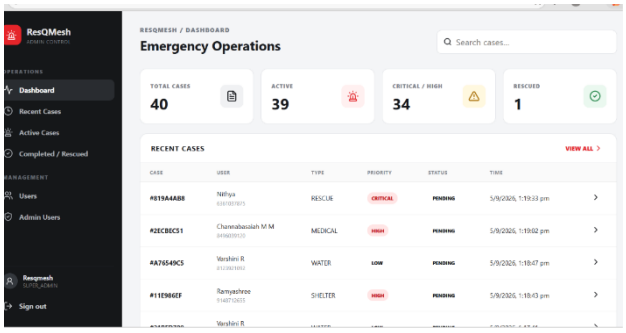
B. Testing Method

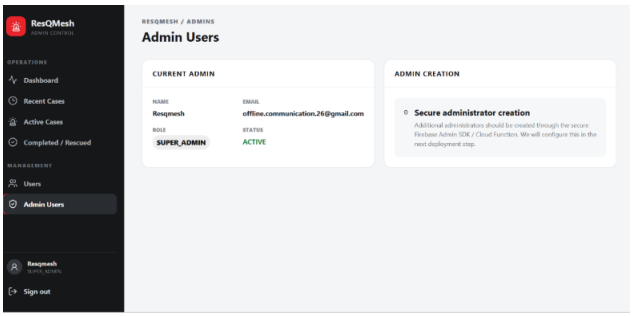
Functional and integration testing were reported for user registration, emergency creation, type and priority selection, location recording, local storage, Bluetooth device discovery, connection, emergency transmission and reception, multi-hop forwarding, duplicate detection, offline-to-online synchronization, Firebase storage, administrator login, dashboard display, case details, rescuestatus updates, and user/administrator management.

VI. RESULTS AND DISCUSSION

The project report records that all listed functional test cases (T01–T32) passed. The tests evaluated the Android application, Bluetooth communication, mesh forwarding, offline-to-online synchronization, Firebase services, and Admin Panel functions.

The offline-to-online test followed the sequence: Phone A creates an emergency without Internet, transfers it through Bluetooth to Phone B, Phone B stores it locally, and the record is synchronized after Internet access becomes available. The report states that the emergency remained available during the offline period and was subsequently synchronized with Firebase. Admin Panel tests covered authorized login, dashboard statistics, recent/active/completed cases, case details, rescue-status updates, and user and administrator management.



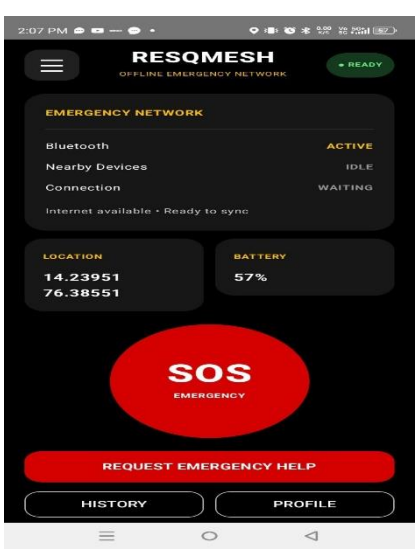


IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

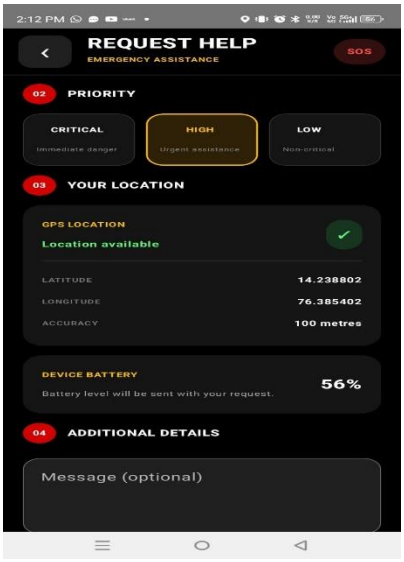
International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 10, October 2026

DOI: 10.17148/IJARCCE.2026.1510137







VII. CONCLUSION AND FUTURE WORK

This paper presented ResQMesh, an offline-first emergency communication and rescue coordination system combining an Android application, Bluetooth-based nearby-device communication, Room local storage, Firebase synchronization, and a web-based Admin Panel. The project report records successful functional tests for emergency generation, local storage, device communication, forwarding, duplicate prevention, cloud synchronization, authentication, and case management. By retaining emergency information locally until connectivity becomes available, the system provides a practical approach to handling emergency records during temporary network outages.

Future enhancements identified in the report include Wi-Fi Direct integration, long-range communication technologies such as LoRa, improved routing and relay selection, end-to-end encryption, map-based rescue tracking, rescue-team assignment, push notifications for critical cases, and iOS support. The system is intended as a supporting communication and coordination tool and does not replace professional emergency services.

ACKNOWLEDGMENT

The authors express their gratitude to the management, Principal, Department of Computer Science and Engineering, project coordinator, and faculty members of S.J.M. Institute of Technology, Chitradurga, for their guidance and support during the project. (Please add the project guide's name here if required by your college.)

REFERENCES

[1] W. N. Abidde, N. Eyidia, and C. I. Iyoloma, "Optimization of Wireless Mesh Networks for Disaster Response Communication," International Journal of Current Science Research and Review, vol. 8, no. 3, pp. 1312–1319, 2025, doi: 10.47191/ijcsrr/V8-i3-35.

[2] S. Chaitrashree, B. L. Kinagi, A. D. Gowda, G. R. Naidu, and D. Naresh, "Disaster-Resilient Mesh Network with AI Load Balancing," International Journal of Advanced Research in Computer and Communication Engineering, vol. 14, no. 12, 2025, doi: 10.17148/IJARCCE.2025.141297.

[3] R. Rondon, A. Mahmood, S. Grimaldi, and M. Gidlund, "Understanding the Performance of Bluetooth Mesh: Reliability, Delay and Scalability Analysis," arXiv:1910.03345, 2019.

[4] U. Ashraf, A. A. Khwaja, J. Qadir, S. Avallone, and C. Yuen, "WiMesh: Practical Wireless Mesh Networking for Disaster Scenarios in Low-Resource Regions," 2021.

[5] F. Alvarez, A. Rodriguez, A. Gonzalez, E. Calle, and M. Barba, "BLUEMERGENCY: Mediating Post-Disaster Communication Using Bluetooth Mesh Standards," arXiv:1905.02465, 2019.

[6] S. Jain, K. Fall, and R. Patra, "Delay-Tolerant Networking Architecture," Internet Research Task Force (IRTF), 2004.

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