International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 11 | Nov - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Resource Allocation During Disaster

Mrs Indushri Haridas(Assistant Professor) ,Bhanupriya H C
Department of Computer Science and Engineering
Don Bosco Institute of Technology
Bengaluru, India
bhanu1392005@gmail.com

Chumbana G M ,Gowtham M
Department of Computer Science and Engineering
Don Bosco Institute of Technology
Bengaluru, India
chumbana123@gmail.com , gowthammraj354@gmail.com

Prarthana T R
Department of Computer Science and Engineering
Don Bosco Institute of Technology
Bengaluru, India
prarthanaramesh378@gmail.com

Abstract
Disaster management requires timely information, proper risk assessment, and efficient allocation of emergency resources. This project proposes a cloud-based disaster management system that helps register emergency requests, monitor available resources, and improve coordination among response teams. A machine learning model, such as Random Forest, can classify emergencies into risk levels based on incident details and severity. Using these risk levels, location, and resource availability, the system helps prioritize requests and identify suitable resources. Cloud computing enables centralized data storage and access to updated information. The proposed system aims to reduce manual coordination, support faster decision-making, and improve resource utilization during emergencies. Its performance can be evaluated through functional testing and suitable machine learning metrics.
Keywords—disaster management, disaster risk assessment, emergency resource allocation, cloud computing, incident management, React, Spring Boot, PostgreSQL.

1. INTRODUCTION

Floods, earthquakes, fires, severe weather, and other emergencies may lead to several requests for assistance at once. To respond in an organized way, teams need to know the incident location and type, how urgent it is, which resources remain available, and whether another team has already been assigned. When these details are kept in separate records, it can take longer to build a reliable picture of the situation. Disaster-risk guidance also highlights the importance of understanding risk, improving coordination, and preparing before emergencies occur [1].

This project focuses on the information-management side of that challenge. It gives users one place to register incidents, review their risk categories, check resource availability, and follow assignment status. The system is intended to support decisions made by authorized personnel; it does not replace official emergency procedures, predict every disaster, or guarantee a faster response in the field.

The paper describes three practical aspects of the implementation. It connects incident reporting and resource records through a single web-based workflow; it uses visible, rule-based conditions to categorize reported risks; and it stores resource assignments so that their status can be reviewed later. These are software and workflow contributions. Their usefulness should be assessed through the tests described in Section 5 rather than assumed in advance.

2. PROBLEM STATEMENT AND OBJECTIVES

A. Problem Statement

The main problem addressed by this project is the absence of a shared workflow linking incident reports, risk prioritization, and resource availability. Bringing these records into one application can make them easier to review and update. However, the application cannot by itself solve a shortage of supplies, incomplete field information, or damage to roads and other infrastructure.

B. Objectives

The project has seven objectives: (1) let users register incidents through a central interface; (2) maintain resource details and availability; (3) assign risk categories using documented rules; (4) save resource assignments and status changes; (5) show incident and resource information on a dashboard; and (6) check the system with repeatable functional and basic performance tests;(7) The system will use recent disaster incident data provided by government authorities to assess risk levels, identify affected areas, and support efficient emergency resource allocation.

3. RELATED WORK

This literature survey follows the ten studies included in the project . The study names, years, titles, methods, features, and limitations are kept in line with that source. Some entries do not contain enough publication information to identify the original paper with confidence, so those details should be checked against the publisher or original paper before the manuscript is submitted.

Xu et al. (2022). The study titled "Multiobjective Resource Allocation" uses mathematical optimisation to support efficient routing and resource allocation. The synopsis notes that its approach still needs real-time integration.

Yanyan Wang et al. (2022). The work on "Emergency Resource Allocation" uses an optimisation-based approach to distribute emergency resources efficiently. According to the study, implementation can be complex.

Torabi et al. (2022). The study on cloud computing and big data in disaster logistics combines cloud-based modelling with big-data methods to support resource optimisation and transportation planning. Its scope is mainly focused on logistics.

Chunhui Huo et al. (2021). The study titled "Scientific Risk Performance Analysis" analyses historical disaster and risk data. The study associates it with routing and allocation, while noting that it does not include a resource-allocation module.

Basharat Ahmed et al. (2021). The work "Retracing Realistic Disaster Scenario" analyses historical disaster information from archival sources. Its main contribution, as summarised in the synopsis, is disaster-history analysis, with a limited feature scope.

Alazawi et al. (2021). The "IoT-based Scalable Framework" focuses on managing disaster data using Internet of Things technologies. It is described as supporting scalable, real-time processing, although security remains a concern.

R. K. Sinha et al. (2021). The "Framework for Disaster Damage Assessment" uses disaster-impact data to evaluate damage and analyse impacts. The project notes that resource allocation is not included in this framework.

Faour et al. (2019). The study on cloud computing in natural-hazard modelling explores cloud-based simulation for predictive analysis. Its limitation, according to the study, is the high computational cost.

Nair et al. (2019). The "RESCUE IoT System" combines Internet of Things technologies with resilient cloud computing and is described as continuing to operate during failures.

Sara Rye et al. (2019). The work on humanitarian resource allocation uses a decision-support technique for planning and estimation. The study identifies the absence of a cloud platform as a limitation.

4. SYSTEM ARCHITECTURE AND METHODOLOGY

A. Architecture Overview

The application is organized into three main parts. The React/Vite frontend provides the incident forms and dashboard. The Spring Boot backend exposes REST endpoints, validates submitted information, applies the configured application rules, and communicates with the database. PostgreSQL stores incident, resource, and assignment records. If the application is deployed to a cloud environment, the deployment configuration should include appropriate environment variables, protected credentials, network permissions, backups, and monitoring. Figure 1 should match the actual application components and connections.

A. End-to-End Processing Workflow

The workflow starts when a user submits an incident. The backend checks the submitted information and saves a valid record in PostgreSQL. The configured rules then determine its risk category. Next, the allocation component checks the resources that meet the criteria actually used by the application. If a suitable resource is found, the assignment is saved; the dashboard can then show the latest incident, resource, and assignment status. If no suitable resource is available, that outcome should be shown clearly rather than suggesting that help has already been dispatched.

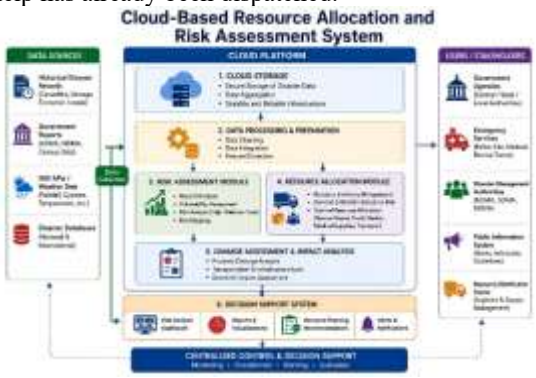


Fig. 1. Proposed architecture of the disaster management system.

B. Incident Registration and Data Handling

A user enters the incident details in the interface and sends the form to the backend. Before saving it, the backend checks that required fields are present and their values are acceptable. Depending on the implemented form, these details may include the incident type, location, description, time, and current status. The application should return a clear message when a request is incomplete or invalid. Valid records are stored so that authorized users can view them later from the dashboard.

C. Rule-Based Risk Assessment

Risk categorization uses conditions configured in the application and does not require a trained machine-learning model. For the work to be reproducible, the final implementation documentation should state the fields used, the conditions and thresholds, and the labels assigned by each rule. Only rules present in the code should be described. These categories help organize reports and should not be interpreted as official hazard forecasts.

D. Resource Matching and Allocation

After an incident is categorized, the application checks which resources are available and suitable according to the implemented criteria. It records the assignment and status so users can follow the record through the dashboard. The selection method must be described accurately: resource-type matching, distance ranking, route-cost calculation, or shortest-

path planning should be included only if implemented. If routing is included, the paper should explain the map data and algorithm used. Straight-line distance should not be described as real-time travel time unless current road and traffic information is used.



Fig. 2. Workflow from incident submission to resource assignment.

E. Technology Stack and Deployment

The core technologies are React/Vite for the interface, Spring Boot for backend REST services, and PostgreSQL for the database. The application works in a local environment and may be deployed to the cloud once the hosting service, database connection, network rules, and security settings are configured.

5. IMPLEMENTATION AND EXPERIMENTAL EVALUATION

The system should be evaluated with repeatable functional tests. Recommended cases include valid and invalid incident submissions, each configured risk rule, available and unavailable resources, successful assignment, the no-suitable-resource case, persistence after reloading, and status updates. For response-time testing, record the test environment, request count, concurrency, and measured response times. Keep screenshots, test inputs, and logs as evidence. The table below is a test-recording template; actual results must be entered after running the tests on the current implementation.

Table 1. Evaluation results to be completed using actual tests

Test / Metric	Procedure	Result to report
Incident registration	Submit valid and invalid records; verify response and database state	[Enter pass/fail and counts]
Rule-based risk category	Run one test case for each implemented rule and compare with expected label	[Enter correct cases / total]
Resource availability	Test available, unavailable, and unsuitable resource records	[Enter pass/fail and counts]
Assignment persistence	Create assignment and verify incident/resource status after reload	[Enter pass/fail and counts]
API response time	Measure repeated requests under a stated environment and load	[Enter median and range]

This study does not report accuracy, response time, scalability, or reliability figures because verified measurements were not provided. The placeholders in Table 1 must be replaced before submission. Keep the test inputs, screenshots, logs, and test procedure as evidence for the results. Passing a particular test shows that the feature behaved as expected under those test conditions; it does not, on its own, prove how the system will perform during a real disaster.

Operational and Data Considerations

Incident locations and resource details may be sensitive, so a deployed system should limit access to appropriate users and validate requests on the server. Credentials should be protected, connections encrypted, and database backups and audit records maintained where required. In the study, describe these safeguards as implemented only if they have been configured and tested in the actual system.

6. CONCLUSION AND FUTURE WORK

This project brings incident registration, rule-based risk categorization, resource tracking, and assignment records into one web application. The React/Vite frontend, Spring Boot backend, and PostgreSQL database provide a shared workflow for managing these records through a dashboard. The system is intended to support coordination and does not replace emergency authorities. The main limitations are that the risk categories depend on configured rules, the allocation logic is limited to what is implemented in the application, and operational performance cannot be established without repeatable tests. Future work may include validated historical datasets, map-based distance or route information, live updates, notifications, stronger audit trails, and evaluation with realistic scenarios.

ACKNOWLEDGEMENT

The authors thank Mrs. Indushri Haridas and the Department of Computer Science and Engineering at Don Bosco Institute of Technology for project guidance and academic support.

REFERENCES

[1] Y. Xu et al., "Multi-objective Emergency Resource Allocation under the Natural Disaster Chain with Path Planning," PMC / NCBI, 2022.
[2] Y. Wang and B. Sun, "Emergency Resource Allocation Strategy Under Uncertainty," 2022
[3] A. Torabi et al., "Application of Cloud Computing and Big Data in Three-Stage Dynamic Modeling of Disaster Relief Logistics and Wounded Transportation," Environmental Science and Pollution Research, Springer, 2022.
[4] C. Huo et al., "Scientific Risk Performance Analysis and Development of Disaster Management Framework," 2021.
[5] B. Ahmad et al., "Retracing Realistic Disaster Scenarios from Archival Sources," 2021.
[6] S. Alazawi et al., "An Internet of Things Based Scalable Framework for Disaster Data Management," Sustainable Cities and Society, ScienceDirect, 2021.
[7] R. K. Sinha et al., "Framework for Disaster Damage Assessment," 2021.

[8] N. Faour et al., “Cloud Computing in Natural Hazard Modeling Systems: Current Research Trends and Future Directions,” International Journal of Disaster Risk Reduction, 2019.
[9] V. Nair et al., “RESCUE: A Resilient Cloud Based IoT System for Emergency and Disaster Recovery,” IEEE Conference Publication, IEEE Xplore, 2019.
[10] Sara Rye and Emel Aktas, “Multi-Attribute Decision Support System for Humanitarian Resource Allocation,” 2022