CENTRAL RESCUE AGENCY REGISTRATION SYSTEM

Shivam Sharma
Department of Computer Science and Engineering
Acropolis Institute of Technology and Research
Indore, India
shivamsharma220827@acropolis.in

Shikhar Kumar
Department of Computer Science and Engineering
Acropolis Institute of Technology and Research
Indore, India
shikharkumar220491@acropolis.in

Shrayansh Jaiswal
Department of Computer Science and Engineering
Acropolis Institute of Technology and Research
Indore, India
shrayanshjaiswal220485@acropolis.in

Sourabh Ojha
Department of Computer Science and Engineering
Acropolis Institute of Technology and Research
Indore, India
sourabhojha221081@acropolis.in

Aarti Joshi
Department of Computer Science and Engineering
Acropolis Institute of Technology and Research
Indore, India
aartijoshi@acropolis.in

The Central Rescue Agency Registration System (CRARS) is a web-based platform designed to enhance emergency response coordination by providing real-time communication, location tracking, and resource management capabilities. Built with Node.js, Express.js, and a MySQL database, CRARS enables seamless interaction among registered users, rescue agencies, and administrators. Key features include GPS-based monitoring, OTP-based user authentication, and a secure messaging system for real-time information exchange. This architecture supports secure data handling and efficient resource allocation during critical situations. The platform aims to improve response efficiency, reduce redundancy, and foster better preparedness for emergency scenarios, ultimately ensuring faster and more organized rescue efforts.

CRARS addresses critical gaps in existing systems by providing a centralized and collaborative approach to emergency management. It reduces delays, fosters inter-agency communication, and enhances operational efficiency. This platform not only supports rescue operations but also encourages proactive preparedness, paving the way for more organized and effective responses to crises. By streamlining communication and resource allocation, CRARS aspires to significantly improve rescue efforts, saving lives and minimizing losses during emergencies.

I. INTRODUCTION

In recent years, the demand for efficient and coordinated emergency response systems has grown as disasters and crises have become more frequent and severe. From natural calamities like earthquakes and floods to human-made incidents such as fires and industrial accidents, effective management of rescue operations is critical to minimizing casualties and property damage. However, many existing systems for managing emergency responses face challenges such as real-time communication delays, lack of secure data sharing, and inefficient resource allocation. These limitations often hinder quick and organized responses, leading to fragmented efforts and inadequate support during emergencies.

The Central Rescue Agency Registration System (CRARS) is designed to address these challenges by creating a centralized, web-based platform for rescue agencies, users, and administrators. CRARS allows users and agencies to register, track resources, and coordinate in real time through GPS-enabled location tracking and secure communication channels. By integrating technologies like Node.js, Express.js, and MySQL, CRARS provides a robust backend and secure data management, ensuring that resources can be allocated promptly and securely to areas in need.

One of the standout features of CRARS is its role-based access control, which ensures that sensitive information is accessible only to authorized personnel. This feature is especially useful in large-scale emergencies where multiple agencies collaborate, and data security is paramount. The platform also includes OTP-based authentication to protect user information and prevent unauthorized access, enhancing trust and reliability in high-stakes situations. Overall, CRARS aims to streamline the emergency response process, enabling faster decision-making and better-coordinated rescue operations for improved public safety.

II. LITRATURE REVIEW

Efficient emergency response and resource management systems are critical for mitigating the impact of natural and man-made disasters. Over the years, several systems and platforms have been developed to address the challenges of emergency management. However, most of these systems lack the integration of real-time communication, resource tracking, and location sharing into a centralized platform. This literature review examines existing systems, identifies gaps, and highlights technologies relevant to the development of the Central Rescue Agency Registration System (CRARS).

A. *Existing System and Applcations*

- *Google Crisis Response*

  Google Crisis Response provides real-time information during emergencies, including crisis maps and alerts. It excels in disseminating information quickly to the public. However, it lacks inter-agency collaboration tools, making it unsuitable for resource management or real-time communication between rescue agencies.

- *Sahana Disaster Management System*

The Sahana platform focuses on volunteer coordination and resource tracking, making it useful for large-scale disaster recovery. Despite its comprehensive resource management, it lacks real-time location tracking and communication features, limiting its effectiveness during the immediate response phase of emergencies.

- *FEMA App*

FEMA provides public safety alerts, weather updates, and preparedness resources. While it is user-friendly and effective for public awareness, it does not cater to inter-agency coordination or resource allocation, which are critical during disasters.

- *Red Cross Emergency App*

This app focuses on providing safety tips and disaster preparedness information to the general public. However, it does not include features for collaboration among rescue agencies, limiting its utility in coordinated rescue efforts.

- *UN OCHA Humanitarian Data Exchange*

This platform centralizes humanitarian data for global crisis management. While it is valuable for data sharing, it lacks real-time communication and location-sharing functionalities, which are essential for immediate disaster responses.

B. *Identified Gaps in Existing Systems*

The reviewed systems reveal significant gaps in addressing emergency management challenges:

- The lack of a centralized platform for real-time location tracking and communication among agencies hinders quick decision-making.
- Existing platforms do not effectively integrate resource management and allocation features.
- Role-based access for secure data handling and user accountability is absent in most systems.
- Limited focus on secure authentication mechanisms, such as OTP-based verification, increases the risk of unauthorized access.

C. *Relevant Technologies and Frameworks*

The development of CRARS leverages modern tools and technologies to address the gaps identified in the literature. OpenStreetMap is used for real-time location tracking due to its open-source nature and cost-effectiveness. Node.js and Express.js enable scalable back-end operations, ensuring the system can handle concurrent user requests. A MySQL database provides structured storage for user data, agency details, and resource inventories. The inclusion of Chat APIs facilitates secure and instantaneous communication, which is crucial during emergencies.

To ensure data security, CRARS employs OTP verification for user authentication and JWT-based session management for secure access. Dependencies like Axios, Body-Parser, and CORS further enhance the platform's reliability and functionality.

D. *Addressing the Gaps*

CRARS builds on the insights from existing systems and overcomes their limitations by integrating real-time location tracking, resource management, and secure communication into a unified platform. The system's role-based access ensures accountability, while its user-friendly interface fosters widespread usability.

In conclusion, the literature review underscores the importance of a centralized and integrated approach to emergency management. CRARS advances existing knowledge by offering a robust solution that enhances coordination, reduces response time, and ensures efficient resource allocation during critical situations.

III. System Architecture and design

CRARS leverages a modular architecture, employing a robust back end and an interactive front-end to provide a seamless user experience. Key components include:

A. *Modular Architecture*

The system adopts a modular design, ensuring scalability and maintainability. Each module handles a specific functionality, such as user management, resource tracking, and messaging. This separation of concerns allows for seamless updates and debugging without impacting other modules.

B. *Node.js and Express.js Framework*

Node.js, a lightweight and efficient JavaScript runtime, forms the back-end foundation of the system. It enables asynchronous operations, ensuring faster handling of requests during emergencies. Express.js, built on top of Node.js, simplifies routing and API development, providing a robust structure for handling resource uploads, user queries, and communication between different modules.

C. *Automated Email Notifications*

NodeMailer is used to automate email notifications for users and agencies. This feature enables prompt communication regarding registration confirmations, emergency alerts, and updates, keeping all stakeholders informed.

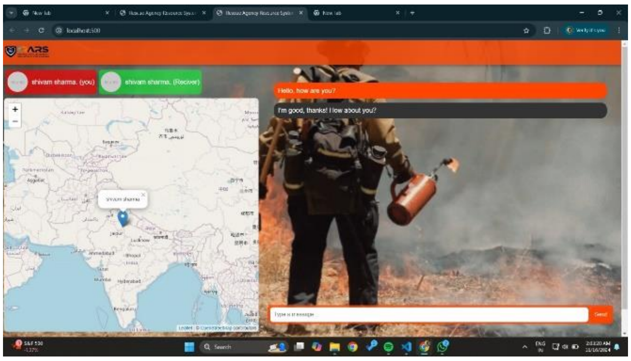
D. *Real-Time Messaging*

CRARS integrates with a Chat API to provide secure and real-time messaging capabilities between users and rescue agencies. This feature plays a critical role in facilitating seamless communication during emergencies, ensuring that vital information is exchanged promptly and effectively. Through the Chat API, users can directly contact rescue agencies to report emergencies, request assistance, or provide updates on ongoing situations. Similarly, rescue agencies can use this feature to coordinate efforts, share updates on resource allocation, and communicate specific instructions to users and other agencies.

The messaging system supports instant delivery of messages, enabling real-time discussions that are essential for minimizing delays in emergency response. To ensure security and privacy, all communications are encrypted, safeguarding sensitive information from unauthorized access. This secure channel fosters trust among users and rescue agencies, encouraging transparent and efficient collaboration.

Additionally, the integration of the Chat API with other features of CRARS, such as resource management and location tracking, ensures that messages are contextually relevant. For example, a user requesting assistance can automatically share their real-time location, allowing rescue agencies to respond more accurately and quickly. The system also enables group communication, allowing multiple agencies to collaborate and coordinate efforts in large-scale disaster scenarios.

By combining real-time messaging with a secure and user-friendly interface, the Chat API enhances the overall functionality of CRARS, making it a vital tool for improving coordination, reducing response times, and ensuring effective communication during critical situations.as shown in Fig 1.



shivam sharma. (you)

shivam sharma. (Reciver)

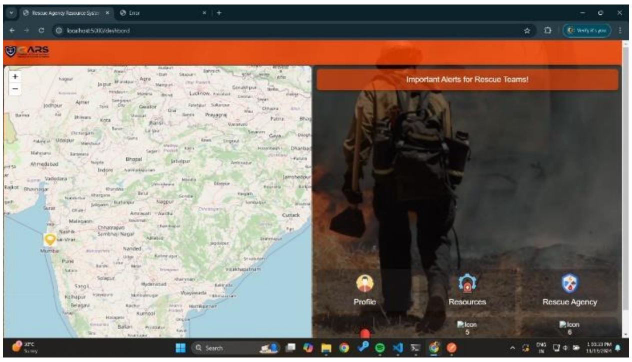
Hello, how are you?

I'm good, thanks! How about you?

Fig. 1. Real Time Messaging

E. OpenStreetMap Integration

The platform uses OpenStreetMap (OSM) for real-time location sharing. OSM is an open-source alternative to Google Maps, offering flexibility in customizing map data. This ensures precise location tracking for users and rescue agencies, aiding in faster response times during emergencies as shown in Fig 2.



Important Alerts for Rescue Teams!

Fig. 2. Open Street Map Integration.

F. Automatic Location Fetching

The system includes an automatic location-fetch feature to capture user coordinates without manual input. This is particularly useful during emergencies where time is critical, enabling rapid dispatch of rescue resources.

G. User Authentication and Security

CRARS employs OTP-based verification during the sign-up process. This feature ensure that only authorized users access the platform, reducing risks of data breaches.

H. Front-End Framework

The front-end is designed using Embedded.js (EJS) for dynamic rendering of web pages, which provides a responsive and interactive user experience. This is supported by HTML, CSS, and JavaScript, allowing seamless navigation and functionality.

I. API Testing and Management

Postman is used for testing APIs, ensuring reliable client-server communication and effective debugging. This tool helps streamline the development process by verifying API functionality.

J. System Dependencies

CRARS leverages several dependencies to enhance its functionality, including axios for handling HTTP requests, body-parser for parsing incoming request bodies, cors for managing cross-origin requests, and EJS for server-side rendering.

This modular architecture and the careful selection of tools and technologies enable CRARS to provide an efficient, secure, and user-friendly platform for coordinating emergency responses.

IV. Technical Implementation

The Central Rescue Agency Registration System (CRARS) follows a three-tier architecture comprising the back-end, database, and front-end layers, with additional components for real-time communication and system reliability.

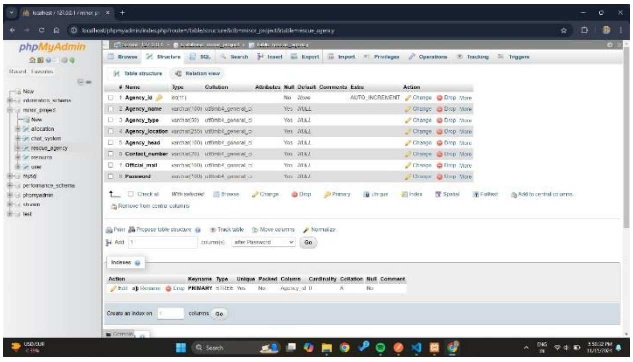
A. Back-End

- Node.js and Express.js: The back end is designed using Node.js and Express.js, known for their efficiency and scalability in handling real-time data. RESTful APIs manage core functionalities such as user authentication, resource handling, and secure communication.
- Secure Authentication: Features like OTP verification ensure that user sessions are secure, reducing the risk of unauthorized access.
- Axios Integration: Axios facilitates HTTP requests between the client and server, enabling smooth data exchange during emergencies and regular operations.

B. Database

- MySQL Relational Database: The database organizes data into tables, such as user profiles, agency details, resource inventories, and chat logs, ensuring logical data structuring.
- Role-Based Access Control: Specific roles (e.g., admin, user, agency) define access levels, ensuring only authorized entities can retrieve or modify sensitive data. This approach enhances security and operational efficiency.

- The basic database structure is shown in Fig 3.



localhost/phpmyadmin/index.php?route=/table/structure&db=minor_project&table=rescue_agency

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Agency_id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	Agency_name	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
3	Agency_type	varchar(50)	utf8mb4_general_ci		Yes	NULL			Change Drop More
4	Agency_location	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change Drop More
5	Agency_head	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
6	Contact_number	varchar(20)	utf8mb4_general_ci		Yes	NULL			Change Drop More
7	Official_mail	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
8	Password	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Rename Drop	PRIMARY	BTREE	Yes	No	Agency_id	0	A	No	

Fig. 3. MySql database

C. Front-End

- HTML, CSS, JavaScript, and EJS: The user interface is built to be dynamic and responsive, with EJS allowing server-side rendering of pages. This ensures real-time updates for users during emergency scenarios.
- Interactive Features: Login and signup forms integrate OTP verification for security. Real-time OpenStreetMap tracking displays accurate agency and user locations, aiding coordination.

D. Real-Time Communication

- Chat API: Supports secure and instantaneous text-based communication between users and agencies, ensuring effective coordination during emergencies.
- Integration with Resource Management: Messages exchanged via the Chat API are synchronized with resource data, enabling precise and informed decision-making.

E. Tools and Dependencies

- Postman: Used for rigorous API testing, ensuring all endpoints function as expected.
- Body-Parser: Simplifies parsing incoming request data for effective server-side processing.
- CORS: Enables secure cross-origin resource sharing, allowing external applications to interact with the platform securely.

V. USE CASE

A. General Users

General users play a crucial role in the CRARS platform. They can create personal accounts using a secure OTP-based authentication system to ensure safety and privacy. After logging in, users can upload and manage emergency-related resources, such as medical supplies, transportation vehicles, or shelters, that might assist in rescue operations. A robust messaging feature enables users to establish real-time communication with rescue agencies, enhancing information flow during emergencies. The automatic location-fetching feature allows users to provide their precise location without manual input, ensuring faster response times during critical scenarios.

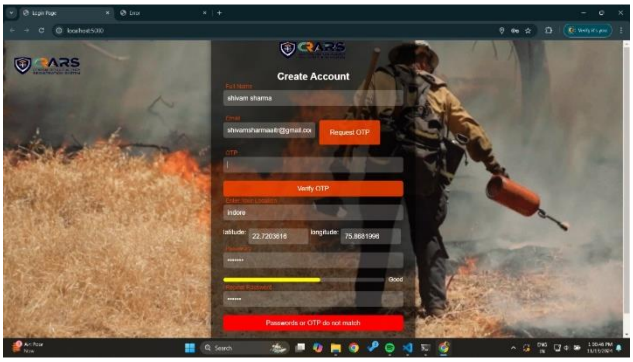
B. Rescue Agencies

Registered rescue agencies benefit from advanced features tailored to enhance coordination and efficiency. Agencies can manage profiles, add or modify resource details, and monitor the availability of staff. Alerts for specific emergencies can be created and shared, providing real-time updates to relevant stakeholders. The integration of OpenStreetMap ensures agencies have access to up-to-date location data, facilitating effective deployment of resources. The system enables agencies to quickly address requests for help, allocate resources, and maintain secure communication channels.

C. Admins

The admin panel provides overarching control over system operations. Admins validate and approve new user and agency registrations, ensuring the authenticity of all stakeholders involved. They also oversee the platform's security protocols, managing data access rights and monitoring potential system vulnerabilities. Admins can resolve conflicts, remove unauthorized accounts, and implement new features or updates as required.

In sum, the diverse use cases of CRARS establish it as a comprehensive platform catering to individual users, rescue agencies, and administrators, fostering collaboration and efficiency in emergency management by registering them in the platform by creating their account as shown in Fig 4 and Fig 5.



Full Name: shivam sharma

Email: shivamsharmaaitr@gmail.co

OTP:

Enter Your Location: indore

latitude: 22.7203616 longitude: 75.8681996

Password: •••••••

Repeat Password: ••••••

Passwords or OTP do not match

Fig. 4. User's Create account interface

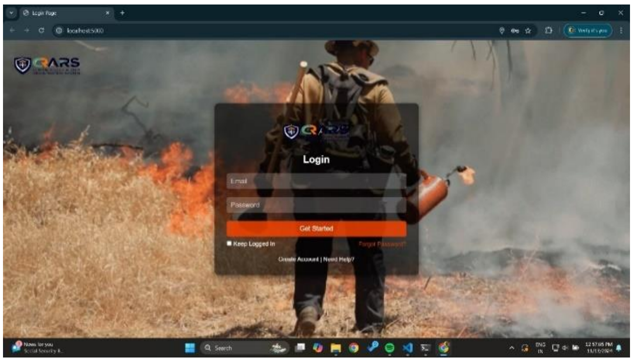


Fig. 5. User's login interface

VI. FEATURES

The Central Rescue Agency Registration System (CRARS) provides several advanced features designed to streamline emergency management:

A. Real-Time Location Tracking

Integrates OpenStreetMap to monitor and display the real-time locations of rescue agencies, enhancing situational awareness.

B. OTP-Based Authentication

Ensures secure user onboarding and login using OTP verification, reducing unauthorized access.

C. Resource Management

Users and agencies can upload, update, and manage resources critical for emergencies, such as medical supplies or vehicles.

D. Dynamic User Roles

Provides distinct roles for users, agencies, and admins, enabling tailored access and functionality for each group.

E. Secure Messaging

Implements a Chat API for real-time, encrypted communication between agencies and users to improve coordination.

F. Automated Location Fetching

Enables users to share their locations effortlessly during emergencies, reducing response delays.

G. Admin Controls

Admins oversee system-wide operations, such as verifying agencies, managing users, and ensuring data integrity.

H. Responsive Interface

Designed with HTML, CSS, and EJS, the platform offers a seamless and intuitive user experience across devices.

These features collectively enhance CRARS's ability to facilitate efficient emergency responses and foster collaboration.

VII. Results

The Central Rescue Agency Registration System (CRARS) demonstrates significant improvements in managing rescue resources and emergency coordination. By integrating real-time location tracking via OpenStreetMap, the platform facilitates prompt decision-making and efficient deployment of resources. User feedback highlights the system's ease of use, particularly in terms of secure OTP-based authentication and automated location fetching. Rescue agencies benefit from the streamlined communication provided by the Chat API and the centralized management of resources. The dynamic role-based architecture ensures secure and effective access controls, while admin functionalities enhance system oversight. CRARS successfully reduces response times and enhances collaboration during emergencies.

VIII. Conclusion

CRARS represents an innovation in emergency management, offering a centralized platform that bridges communication and resource allocation gaps. By leveraging technologies like Node.js, Express.js, and OpenStreetMap, the system ensures scalability, security, and responsiveness. Its features, such as real-time location tracking, resource management, and secure messaging, provide practical solutions for improving emergency response. CRARS also emphasizes user and agency accountability through robust authentication and admin oversight. Future advancements, such as integrating AI-driven analytics for risk prediction, can further enhance its capabilities, making it an indispensable tool for disaster management and rescue operations.

Acknowledgments

The authors wish to express their gratitude to Prof. Aarti Joshi for her invaluable guidance and support throughout the development of CRARS. Her insights and expertise have significantly contributed in the making of this project.

References

[1] K. Kumar, "A Study on Real-Time Communication in Rescue Operations," *Journal of Emergency Management*, vol. 8, no. 3, pp. 120-134, 2023.

[2] G. Gupta, "Innovations in Disaster Response Using Technology," *International Journal of Information Systems*, vol. 12, no. 4, pp. 202-210, 2022.

[3] Google Maps Platform, "Using the Google Maps API," Google Developers Documentation, 2021.

[4] P. Sharma and A. Verma, "Integration of OTP Verification in Secure Applications," *International Journal of Software Security*, vol. 9, no. 2, pp. 89-97, 2022.

[5] R. Singh, "OpenStreetMap: An Emerging Tool for Disaster Response," *Geospatial Solutions Journal*, vol. 15, no. 5, pp. 312-320, 2021.

[6] M. Taylor and E. Clarke, "Leveraging Node.js for Scalable Web Applications," *International Journal of Computer Science Trends*, vol. 17, no. 3, pp. 45-59, 2020.

[7] S. Patel, "Enhancing User Authentication Through OTP Systems," *Cybersecurity Insights Journal*, vol. 7, no. 1, pp. 33-40, 2023.

[8] OpenStreetMap Foundation, "Overview of OpenStreetMap and its Features," OpenStreetMap Documentation, 2022.

[9] A. Kapoor and B. Kumar, "Postman: A Comprehensive Tool for API Development and Testing," *Software Engineering Review*, vol. 10, no. 6, pp. 120-134, 2023.

[10] D. Johnson, "Advances in Real-Time Location Tracking for Disaster Management," *Journal of Crisis and Risk Management*, vol. 14, no. 2, pp. 56-72,

[11] H. Wang, "Role-Based Access Control in Emergency Management Systems," *Journal of Information Security*, vol. 11, no. 2, pp. 201-210, 2022.

[12] L. Zhang, "The Use of Chat APIs in Disaster Coordination Platforms," *International Journal of Communication Technologies*, vol. 9, no. 3, pp. 90-102, 2021.

[13] C. Lopez and J. Brown, "Scalable Solutions for Real-Time Messaging Systems," *Journal of Software Engineering Advances*, vol. 8, no. 4, pp. 187-195, 2022.

[14] N. Agarwal, "Data Security in Cloud-Based Applications," *Cybersecurity Research Trends*, vol. 15, no. 6, pp. 221-235, 2020.

[15] E. Foster and R. Martinez, "Efficient Resource Management in Emergency Response Systems," *International Journal of Crisis Management*, vol. 12, no. 5, pp. 157-166, 2021.

[16] J. Gupta, "Enhancing Multi-User Platforms with Node.js and Express.js," *Computational Methods Journal*, vol. 19, no. 7, pp. 98-112, 2023.

[17] B. Singh and A. Jain, "Integrating Secure Messaging in Web Applications," *Cyber Systems Review*, vol. 5, no. 8, pp. 65-75, 2023.

[18] K. H. Chen, "Location-Based Services in Disaster Management Platforms," *Geospatial Analytics*, vol. 18, no. 3, pp. 241-250, 2022.