

An Intelligent Predictive Framework for Flood and Fire Risk Assessment and Disaster Decision Support

AASHA J , CHITHARTHANI K



[https://doi.org/ 10.55041/ijst.v2i9.008](https://doi.org/10.55041/ijst.v2i9.008)

Cite this Article: J, A. & K, C. (2026). An Intelligent Predictive Framework for Flood and Fire Risk Assessment and Disaster Decision Support. International Journal of Science, Strategic Management and Technology, 02(9), 1-9. <https://doi.org/10.55041/ijst.v2i9.008>

License:  This article is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting use, distribution, and reproduction in any medium, provided the original author(s) and source are properly credited.

ABSTRACT

Floods and forest fires are major natural hazards that can cause loss of life, damage to infrastructure, and environmental destruction. Effective disaster management requires timely prediction, early warning, evacuation planning, and efficient resource management. This project proposes a **Multi-Hazard Prediction and Decision Support System for Flood and Fire Disaster Management** that integrates real-time weather information, machine learning, geographic services, and emergency management functions into a unified web-based platform. The system collects real-time environmental parameters such as temperature, rainfall, humidity, and wind speed through the OpenWeather API. Machine learning models analyze these parameters to predict the risk of floods and forest fires and generate appropriate risk levels and warnings. The system also provides location-based distress reporting and evacuation assistance by identifying suitable routes to nearby emergency shelters using Google Maps.

To support coordinated disaster response, the system provides separate interfaces for citizens, NGOs, and government authorities. Citizens can report emergencies and access evacuation routes, while NGOs can monitor distress requests, update rescue status, and manage shelter occupancy. Government authorities can monitor emergency activities, issue alerts, and manage essential resources based on the requirements of affected shelters. The proposed system integrates disaster prediction, early warning, evacuation planning, and resource management into a single platform. It helps improve the speed and coordination of emergency response while supporting informed decision-making during flood and fire disasters. The reported evaluation achieved **92.4% prediction accuracy**, with an average system response time of approximately **2.4 seconds**.

Keywords: Flood Prediction, Forest Fire Prediction, Machine Learning, Multi-Hazard Disaster Management, Decision Support System, OpenWeather API, Evacuation Routing, Resource Allocation, Emergency Management.

CHAPTER 1 INTRODUCTION

1.1. DISASTER MANAGEMENT SYSTEMS

Disaster management is a systematic approach aimed at reducing the impact of natural and human-induced disasters through effective planning, preparedness, response, and recovery strategies. Natural disasters such as floods and forest fires have severe consequences on human life, infrastructure, agriculture, and the environment. These disasters often occur with little warning and can rapidly escalate into large-scale emergencies if proper monitoring and response systems are not in place. Therefore, efficient disaster management systems are essential to minimize damage, protect communities, and ensure quick recovery after a disaster event.

Traditionally, disaster management relied heavily on manual monitoring methods, historical records, and periodic weather reports. While these approaches provided basic awareness about potential hazards, they often lacked real-time responsiveness and predictive capabilities. As a result, authorities were sometimes unable to issue timely warnings or deploy resources efficiently. In many cases, delayed response and inadequate planning led to increased casualties, property damage, and economic losses. With the advancement of information technology and data analytics, disaster management systems have evolved significantly. Modern disaster management frameworks incorporate technologies such as Geographic Information Systems (GIS), remote sensing, cloud computing, and data-driven analytics to monitor environmental conditions continuously. These systems allow authorities to collect and process large volumes of environmental data, enabling more accurate risk assessment and disaster prediction. Furthermore, the integration of Artificial Intelligence (AI) and Machine Learning (ML) has enhanced the capability of disaster management systems to analyze complex environmental patterns. By utilizing predictive models, these systems can identify potential disaster risks in advance and provide early warnings to vulnerable communities. Early warning systems play a crucial role in disaster preparedness by giving authorities and citizens enough time to take preventive actions such as evacuation, shelter preparation, and resource mobilization.

An effective disaster management system also includes components such as risk assessment, emergency communication, evacuation planning, and resource allocation. These components work together to ensure coordinated actions among different stakeholders including government agencies, non-governmental organizations (NGOs), and local communities. Through proper integration of these elements, disaster management systems can significantly reduce the negative impact of disasters and enhance community resilience.

1.2. FLOOD AND FIRE HAZARDS

Floods and forest fires are among the most destructive natural disasters affecting various regions across the world. Both hazards have the potential to cause widespread environmental damage, disrupt economic activities, and threaten human safety. The increasing frequency and severity of these disasters in recent years highlight the urgent need for effective monitoring and prediction systems. Floods generally occur due to excessive rainfall, overflowing rivers, poor drainage infrastructure, or sudden release of water from reservoirs and dams. When large volumes of water accumulate in low-lying areas, they can submerge residential zones, agricultural fields, transportation networks, and public infrastructure. Floods often lead to displacement of communities, contamination of water resources, and outbreaks of water-borne diseases. In coastal regions and urban environments, flooding can become even more severe due to inadequate drainage systems and rapid urbanization.

Forest fires, on the other hand, occur when vegetation ignites and spreads rapidly across forest areas. These fires are typically triggered by high temperatures, prolonged drought conditions, lightning strikes, or accidental human activities such as unattended campfires or agricultural burning. Once initiated, forest fires can spread quickly due to wind speed and dry vegetation, making them extremely difficult to control. The consequences include destruction of forests, loss of wildlife habitats, reduction in air quality, and increased greenhouse gas emissions. By utilizing environmental parameters such as rainfall intensity, temperature, humidity, wind speed, and atmospheric pressure, modern systems can identify potential high-risk zones for both floods and forest fires. When combined with geospatial analysis and predictive modeling, these parameters enable authorities to generate accurate hazard maps and implement preventive measures before disasters occur.

1.3. ROLE OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING IN DISASTER MANAGEMENT

Artificial Intelligence (AI) and Machine Learning (ML) have emerged as powerful tools for improving disaster prediction, monitoring, and response planning. These technologies enable systems to process large volumes of environmental data, identify complex patterns, and generate accurate predictions regarding potential disaster events before they strike.

Machine learning algorithms analyze various dynamic environmental parameters—such as rainfall levels, temperature variations, humidity, and wind speed—to determine the likelihood of a disaster. By training predictive models using historical disaster datasets, the system learns the exact thresholds that trigger flood and fire incidents. In this proposed

system, these predictive models are empowered by the integration of the **Open Weather API**, which fetches live, real-time meteorological data for Chennai and the Nilgiris. By feeding this live API data directly into the trained ML models, the system can autonomously evaluate real-time environmental conditions and calculate a highly accurate, localized probability of disaster occurrence.

DECISION SUPPORT SYSTEM (DSS) FOR MULTI-HAZARD MANAGEMENT

A Decision Support System (DSS) is a computer-based platform designed to assist decision-makers by analyzing large volumes of data and presenting meaningful insights for effective planning and management. In multi-hazard environments, where rapid coordination is a matter of life and death, a DSS becomes particularly valuable.

The proposed **AI-Driven Multi-Hazard Prediction and Decision Support System** is deployed as a dynamic web application using the Flask framework. It integrates predictive analytics, geographic data, and operational strategies into a unified platform built on a three-tier Role-Based Access Control (RBAC) architecture:

1. **Citizen Portal:** Allows the general public to monitor local flood and fire risks using live API data, send distress signals with their exact location, and use integrated Google Maps routing to navigate safely to the nearest available emergency shelters.
2. **NGO / Relief Worker Portal:** Empowers ground-level responders to monitor live distress signals from citizens, update rescue statuses to "In Process" or "Rescued", and manually update the real-time capacity and occupancy of emergency shelters.
3. **Government / Unified Operations Center:** Provides disaster managers with a centralized dashboard featuring a live activity log of all NGO and Citizen actions. Authorities can broadcast global emergency alerts and utilize an automated **Resource Allocation Engine**, which dynamically calculates the exact requirements for food packets, water, and medical kits based on the live shelter capacity updated by NGOs.

By seamlessly connecting predictive AI, live weather data, and real-time communication between these three tiers, this Decision Support System enables authorities to respond proactively, allocate resources efficiently and ultimately save lives.

CHAPTER 2 LITERATURE SURVEY

REFERENCE-1

TITLE : Flood Hazard Mapping using Machine Learning

AUTHOR : R. Jain, S. Agarwal, Prof. K. Gupta

PUBLICATION: IEEE Xplore – International Conference on Machine Learning and Applications

YEAR : 2022

HIGHLIGHTS:

This paper presents the development of flood hazard prediction maps using the Random Forest algorithm with rainfall and elevation data as major inputs. The study utilized widely available geospatial datasets to classify regions based on their flood susceptibility and generated visual hazard maps for effective disaster preparedness. The model achieved high classification accuracy and proved to be a reliable tool for identifying flood-prone zones, supporting authorities in flood risk assessment and mitigation planning.

ADVANTAGES:

The use of widely accessible geospatial and meteorological data enhances the model's applicability in various regions. The Random Forest algorithm demonstrated strong predictive capability, achieving more than 85% accuracy in flood-prone area classification.

DRAWBACKS:

The research was limited to data from a single district, which restricts the generalization of the model across different geographical areas. Furthermore, the system does not incorporate real-time data streams or multi-disaster analysis, limiting its use in dynamic and large-scale disaster management scenarios.

REFERENCE-2

TITLE : Deep Learning for Forest Fire Detection from Satellite Images

AUTHOR : M. Hernandez, P. Silva

PUBLICATION : Elsevier – Remote Sensing of Environment

YEAR 2021

HIGHLIGHTS :

This paper explores the application of Convolutional Neural Network (CNN) models for early forest fire detection using MODIS and Landsat satellite imagery. The authors developed a deep learning framework capable of automatically identifying fire-prone regions and active fire spots based on spectral and thermal image features. The system demonstrated superior performance in detecting fire occurrences across multiple geographical regions and provided timely detection outputs suitable for early warning systems

ADVANTAGES:

The proposed approach achieved high detection accuracy due to the use of deep CNN architectures trained on large-scale remote sensing datasets. The model's ability to generalize across different regions was validated through extensive experiments on diverse landscapes, showing robustness under varying environmental conditions.

DRAWBACKS:

The deep learning model requires substantial computational power for training and inference, making real-time deployment challenging without advanced hardware infrastructure. Additionally, the study focuses solely on fire detection without incorporating modules for damage estimation, evacuation planning, or resource allocation, limiting its use in end-to-end disaster management applications.

REFERENCE-3

TITLE : A GIS-based Decision Support System for Disaster Management

AUTHOR : S. Patel, K. Deshmukh **PUBLICATION** : Springer – Natural Hazards Journal **YEAR** : 2020

HIGHLIGHTS:

This paper presents the development of a Geographic Information System (GIS)-based Decision Support System (DSS) aimed at enhancing disaster management, particularly for flood evacuation planning. The system integrates spatial data such as road networks, elevation, and population density to identify safe evacuation routes and shelter locations. Through spatial analysis and mapping tools, the DSS assists authorities in visualizing high-risk zones and making informed decisions during emergencies.

ADVANTAGES:

The proposed GIS-based DSS effectively combines mapping capabilities with decision-making processes, enabling authorities to visualize and plan evacuation strategies efficiently. The framework is user-friendly and practical for field use, as it provides spatial context to emergency operations.

DRAWBACKS:

The system does not incorporate Artificial Intelligence (AI) or Machine Learning (ML) algorithms, which limits its predictive capability and automation potential. Its visualization and analytical functions are relatively basic, relying primarily on manual interpretation of spatial maps rather than interactive, real-time dashboards. This restricts scalability and adaptability in complex, multi-hazard scenarios.

REFERENCE-4

TITLE : Multi-Hazard Early Warning System using IoT and Cloud Computing
AUTHOR : L. Wang, H. Zhou
PUBLICATION : IEEE Sensors Journal
YEAR : 2021
HIGHLIGHTS :

This paper introduces a multi-hazard early warning framework that integrates Internet of Things (IoT) sensors with cloud computing to detect and respond to disasters such as floods and landslides. The system collects real-time environmental data such as rainfall, soil moisture, and ground vibration through distributed sensor networks and transmits it to a cloud platform for analysis and alert generation.

ADVANTAGES:

The architecture supports real-time sensor data acquisition, ensuring immediate detection of environmental anomalies that may trigger disasters. Its cloud-based infrastructure enables scalability, allowing the system to be adapted to various disaster types beyond floods and landslides. The combination of IoT and cloud technologies ensures seamless data sharing and remote monitoring, making the framework suitable for integration into national-level disaster management systems.

DRAWBACKS:

The system's performance heavily depends on physical sensor hardware, making deployment and maintenance costly, particularly in remote or underdeveloped areas. Additionally, the approach emphasizes sensor-based detection rather than AI or Machine Learning-driven prediction, limiting its analytical depth and adaptability to data-driven forecasting models.

REFERENCE-5

TITLE : Optimizing Disaster Relief Resource Allocation using Linear Programming.
AUTHOR : A. Roy, T. Banerjee
PUBLICATION : Elsevier – Computers & Industrial Engineering
YEAR : 2019

HIGHLIGHTS

This paper presents an optimization-based framework for post-disaster resource allocation using Linear Programming (LP). The model focuses on efficiently distributing essential relief materials such as food, medical supplies, and shelter capacity to affected regions. By formulating the problem as a mathematical optimization model, the authors aim to minimize resource wastage and transportation costs while maximizing coverage and equity among disaster-affected zones.

ADVANTAGES

The use of linear programming ensures mathematically sound and optimal resource distribution, minimizing redundancy and ensuring that all affected areas receive adequate aid. The model effectively demonstrates cost-efficiency and improved logistics in disaster relief operations. It also offers a flexible framework that can be easily adapted to different resource types and disaster scenarios, making it valuable for emergency planning and management

authorities.

DRAWBACKS

The approach assumes static and pre-defined input parameters, such as demand and supply quantities, which may not reflect real-time disaster dynamics. Moreover, the model does not integrate predictive or AI-driven components, limiting its applicability in forecasting-based or dynamic decision support systems where conditions evolve rapidly.

REFERENCE-6

TITLE : Forest Fire Spread Simulation using Cellular Automata

AUTHOR : Y. Zhang, Q. Li

PUBLICATION: ACM Transactions on Environmental Modelling

YEAR : 2020

HIGHLIGHTS :

This paper presents a simulation-based approach for modeling forest fire spread using Cellular Automata (CA). The model divides the forest area into grid cells, where each cell's state changes dynamically based on environmental factors such as wind direction, vegetation density, and moisture content. The simulation effectively captures the spatial and temporal behavior of fire propagation, allowing researchers to study how various environmental and climatic conditions influence fire spread.

ADVANTAGES:

The Cellular Automata approach offers a realistic and visually interpretable model for simulating forest fire dynamics. It allows disaster planners to test multiple environmental and management scenarios, aiding in strategic decision-making and emergency preparedness. The model is also computationally efficient and adaptable, making it suitable for simulating fire behavior across different terrains and ecological conditions.

DRAWBACKS:

The system functions purely as a simulation tool without incorporating machine learning or predictive modeling, limiting its capacity to forecast future fire events. Additionally, the accuracy of simulation outcomes heavily depends on the precision and availability of environmental input data, such as wind speed, vegetation maps, and soil moisture levels.

REFERENCE-7

TITLE : AI for Climate Risk Prediction: A Survey

AUTHOR : K. Sharma, V. Singh

PUBLICATION : IEEE Access

YEAR : 2022

HIGHLIGHTS

This paper provides a comprehensive survey of Artificial Intelligence (AI) and Machine Learning (ML) techniques used for predicting climate-related hazards such as floods, droughts, and wildfires. The authors systematically analyze various algorithms, including neural networks, decision trees, and ensemble models, highlighting their effectiveness in different environmental contexts.

ADVANTAGES:

The paper offers **broad algorithmic coverage** and a **comparative performance analysis**, enabling readers to

understand the suitability of different AI/ML models for various disaster types. It also identifies key research gaps, emphasizing the need for hybrid and multi-hazard models. The comprehensive nature of the review makes it an excellent foundation for developing integrated, AI-driven disaster management systems.

DRAWBACKS:

The study remains **theoretical** in nature, with no real-world implementation or empirical testing of models. It focuses primarily on reviewing existing literature rather than designing or validating an operational system. As a result, while informative, it lacks practical demonstrations or system-level applications for real-time disaster management.

REFERENCE-8

TITLE : Hybrid CNN-RNN Model for Flood Forecasting

AUTHOR : D. Chen, F. Li

PUBLICATION : Elsevier – Expert Systems with Applications

YEAR : 2021

HIGHLIGHTS

This paper introduces a hybrid deep learning framework that combines Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for accurate rainfall-runoff modeling and flood forecasting. The CNN component extracts spatial features from rainfall and topographic data, while the RNN captures temporal dependencies in hydrological time series. The integration of both models allows the system to effectively learn complex spatial-temporal relationships, improving prediction accuracy for flood events.

ADVANTAGES:

The hybrid CNN-RNN architecture efficiently processes both spatial and temporal data, making it well-suited for complex flood forecasting tasks. It provides high predictive accuracy and adaptability to varying hydrological and meteorological inputs. Additionally, the framework enhances early warning capabilities, supporting data-driven disaster preparedness and mitigation strategies.

DRAWBACKS:

The model demands large amounts of training data and significant computational resources, making it challenging for deployment in data-scarce or resource-limited regions. Furthermore, the training process is complex and time-consuming, requiring careful parameter tuning and high-performance hardware to achieve optimal results.

REFERENCE-9

TITLE : Spatial Decision Support for Emergency Evacuation Routes

AUTHOR : H. Kim, J. Park

PUBLICATION : International Journal of Disaster Risk Reduction

YEAR : 2019

HIGHLIGHTS

This paper presents a spatial decision support system designed for emergency evacuation route planning using Dijkstra's shortest path algorithm integrated with Geographic Information System road maps. The system identifies optimal evacuation routes from disaster-affected areas to designated shelters, considering spatial constraints such as road connectivity, elevation, and hazard zones. The study focuses primarily on flood scenarios, demonstrating how GIS-based graph models can enhance decision-making efficiency during emergencies.

ADVANTAGES :

The approach provides a clear and practical application of graph theory for real-world evacuation planning. It is computationally efficient, low-cost, and easy to implement using readily available GIS data. The model's visual mapping capability allows authorities to quickly interpret and deploy safe evacuation routes, making it suitable for on-ground disaster management operations.

DRAWBACKS :

The study assumes static road conditions, not accounting for dynamic factors such as traffic congestion, blocked roads, or changing hazard intensity during evacuation. Additionally, the system is limited to flood scenarios, lacking generalization to other disaster types like fires or earthquakes.

REFERENCE-10

TITLE : Multi-Hazard Disaster Management using AI and Big Data
AUTHOR : P. Mehta, R. Kumar **PUBLICATION** : Springer – AI and Society Journal **YEAR** : 2023

HIGHLIGHTS

This paper proposes an AI-driven multi-hazard disaster management framework that leverages big data analytics to handle multiple disaster types—specifically floods, forest fires, and cyclones—within a unified system. The framework integrates predictive modeling, risk assessment, and response coordination using large-scale heterogeneous data sources such as satellite imagery, meteorological data, and social media feeds

ADVANTAGES :

The approach provides a comprehensive multi-disaster management framework, addressing the need for integrated solutions that can handle various hazard scenarios simultaneously. By merging AI algorithms with big data technologies, it enables improved prediction accuracy, faster situational awareness, and more effective resource allocation.

DRAWBACKS :

The proposed model remains conceptual, lacking real-world implementation or validation using operational data. Furthermore, it does not address practical deployment challenges, such as data standardization, computational infrastructure requirements, and coordination among disaster management agencies, which may hinder large-scale adoption

LITERATURE SURVEY SUMMARY

S.NO	TITLE	KEY FINDINGS	STRENGTHS	LIMITATIONS
1	Flood Hazard Mapping using Machine Learning	ML(Random Forest) used to create flood susceptibility maps using rainfall & elevation data.	High prediction accuracy; uses widely available geospatial data.	Limited to one district; no real-time or multi-disaster support.
2	Deep Learning for Forest Fire Detection from Satellite Images	CNN-based system detects fire-prone regions and active fires using MODIS & Landsat images.	High detection accuracy; generalizes across regions.	Requires high computational power; no end-to-end disaster management integration.
3	AGIS-Based Decision Support System for Disaster Management	GIS used for evacuation planning by integrating elevation, road networks, and population density.	User-friendly; effective spatial visualization for planning.	No AI/ML integration; manual, static analysis.

4	Multi-Hazard Early Warning System using IoT and Cloud Computing	IoT sensors + cloud platform used for real-time flood and landslide monitoring.	Real-time detection; scalable through cloud.	High deployment cost; limited predictive intelligence.
5	Optimizing Disaster Relief Resource Allocation using Linear Programming	LP model optimizes post-disaster resource distribution.	Cost-efficient, mathematically optimal resource allocation.	Uses static parameters no predictive or dynamic adaptation.
6	Forest Fire Spread Simulation using Cellular Automation	CA used to simulate fire spread based on wind, vegetation, and moisture.	Visually interpretable; fast and adaptable simulations.	No predictive ML; accuracy depends on data precision.
7	AI for Climate Risk Prediction: A Survey	Survey of AI/ML techniques for hazards (floods, droughts, fires).	Broad coverage; identifies research gaps; helpful for integrated systems development.	Theoretical only; no real-world testing or system implementation.
8	Hybrid CNN-RNN Model for Flood Forecasting	Combines CNN (spatial) and RNN (temporal) for rainfall-runoff modeling and flood forecasting.	High predictive accuracy; captures spatial-temporal features effectively.	Data-hungry; computationally expensive; complex training
9	Spatial Decision Support for Emergency Evacuation Routes	Uses GIS + Dijkstra's algorithm to compute optimal evacuation routes.	Low-cost; efficient; easy to implement on GIS platforms.	Assumes static road conditions; limited to flood scenarios.
10	Multi-Hazard Disaster Management using AI and Big Data	Proposes integrated AI- and big-data-driven multi-disaster framework.	Comprehensive multi-hazard approach; improved situational awareness.	Conceptual; lacks real-world deployment and validation.

CHAPTER 3 SYSTEM ANALYSIS

EXISTING SYSTEM

In current disaster management frameworks, hazard prediction and emergency response are typically handled as disjointed processes that rely heavily on fragmented communication channels. Traditional systems predominantly depend on periodic, static meteorological updates rather than real-time data streaming, which severely limits the ability to predict rapidly escalating disasters—such as urban floods in Chennai or sudden forest fires in the Nilgiris—with high accuracy. Furthermore, existing platforms often lack a unified, multi-tier architecture, resulting in a significant communication gap between affected citizens, ground-level relief workers (NGOs), and government authorities. When a disaster strikes, citizens usually have to rely on generic public broadcasts and lack a centralized portal to transmit location-specific distress signals or access dynamic, map-based routing to emergency shelters. Similarly, on the administrative side, resource allocation—such as the distribution of food, water, and medical supplies—is largely based on rough historical estimations rather than real-time, ground-truth shelter occupancy. This lack of a synchronized, real-time decision support system leads to delayed rescue operations, the misallocation of critical relief materials, and an overall reactive, rather than proactive, approach to multi-hazard disaster management.

DISADVANTAGES OF EXISTING SYSTEM

- Focuses mainly on **single-hazard monitoring** such as flood or fire individually.
- Relies on **manual data monitoring and traditional weather reports**
- Limited use of **Artificial Intelligence or Machine Learning for prediction**.
- **Delayed warning systems**, reducing the time available for evacuation.
- **Manual evacuation planning**, which may not provide optimal routes.
- Inefficient **resource allocation during emergency situations**.
- Lack of a **centralized monitoring and decision-support platform**.
- Poor coordination between **citizens, NGOs, and government authorities**.

PROPOSED SYSTEM

To overcome the limitations of existing frameworks, this project proposes a proactive, centralized, and real-time AI-Driven Multi-Hazard Decision Support System designed specifically for urban floods in Chennai and forest fires in the Nilgiris. The proposed system eliminates fragmented communication by introducing a unified, three-tier Role-Based Access Control (RBAC) web application built on the Flask framework, catering specifically to Citizens, NGOs, and Government Officials. Instead of relying on static weather reports, the system dynamically fetches real-time meteorological data via the Open Weather API and feeds it directly into trained Machine Learning models to generate highly accurate, localized threat predictions. For the general public, the system offers an interactive assistance portal to send location-based distress signals, track rescue statuses, and utilize dynamic Google Maps integration for optimal evacuation routing to the nearest emergency shelters. At the ground level, the NGO module empowers relief workers to efficiently manage these rescue operations and manually update the real-time capacity of active shelters. Concurrently, the Government portal functions as a Unified Operations Center, providing disaster management authorities with a centralized live activity log, a global emergency alert broadcasting system, and an automated Resource Allocation Engine. This engine replaces manual estimation by dynamically calculating the exact quantities of relief materials—such as food packets, water, and medical kits—required based on the live shelter occupancy data updated by NGOs, thereby ensuring a highly coordinated and data-driven disaster response.

ADVANTAGES OF PROPOSED SYSTEM

- Supports **multi-hazard prediction** for both floods and forest fires.
- Utilizes **Machine Learning algorithms** for accurate disaster prediction.
- Processes **real-time environmental data** for continuous monitoring.
- Generates **automated early warning alerts** based on risk levels.
- Provides **optimal evacuation routes** using shortest-path algorithms.
- Implements **efficient resource allocation** using optimization techniques.
- Includes a **centralized web-based dashboard** for monitoring disasters.
- Supports **role-based access for citizens, NGOs, and government authorities**.
- Improves **coordination, response time, and decision-making efficiency**.

INTRODUCTION

CHAPTER 4 SYSTEM REQUIREMENTS

The System Requirements chapter specifies the hardware and software resources necessary for the development and seamless operation of the AI-Driven Multi-Hazard Prediction and Decision Support System for Flood and Fire Disaster Management. These requirements ensure that the system can efficiently perform complex tasks such as dynamic environmental data collection, machine learning-based disaster prediction, real-time evacuation routing, and automated resource allocation. Since the proposed system integrates predictive machine learning algorithms, modern web technologies, and real-time weather data APIs (Open Weather), it requires a stable computing environment with adequate processing power, memory, and internet connectivity. Adhering to these system requirements ensures reliable performance, smooth data processing, and highly efficient disaster management operations.

HARDWARE REQUIREMENTS

The hardware components required for implementing the proposed system are listed below:

- **Processor:** Intel Core i5 (or equivalent)
- **RAM:** Minimum 8 GB
- **Hard Disk:** 256 GB
- **Display:** 1366 × 768 resolution monitor
- **Network:** Stable broadband Internet connection (required for live API fetching and Google Maps routing)

SOFTWARE REQUIREMENTS

The following software tools and technologies are required for developing and running the proposed system:

- **Operating System:** Windows 11
- **Programming Language:** Python 3.10
- **Development Environment:** Command Prompt
- **Frontend Technologies:** HTML5, CSS3, JavaScript
- **Backend Framework:** Flask
- **Web Browser:** Google Chrome
- **Python Libraries:** Pandas, NumPy, Scikit-learn, Requests, Pickle

SOFTWARE DESCRIPTION

Operating System Windows 11

Windows 11 is used as the operating system for developing and executing the project. It provides a stable environment for running Python applications, web servers, and machine learning libraries. The operating system also supports the installation of necessary tools, packages, and frameworks required for building the disaster prediction system.

Programming Language – Python 3.10

Python is the primary programming language used in this project due to its simplicity, flexibility, and extensive support for machine learning and data analysis. Python is used to implement the disaster prediction algorithms, process environmental data, connect with APIs, and integrate different modules of the system.

Frontend Technologies – HTML, CSS, and JavaScript

HTML, CSS, and JavaScript are used to design and develop the user interface of the web dashboard. HTML structures the web pages, CSS is used for styling and layout design, and JavaScript enables dynamic interactions such as displaying prediction results, alerts, and user requests. These technologies allow users to interact with the system through a web-based platform.

Backend Framework – Flask

Flask is a lightweight Python web framework used to develop the backend of the system. It handles server-side operations such as processing user requests, executing machine learning models, managing user authentication, and sending prediction results to the frontend interface. Flask enables smooth communication between the web interface and the prediction modules.

Web Browser – Google Chrome

Google Chrome is used to access and test the web-based dashboard of the system. It allows users to view prediction results, alerts, evacuation routes, and resource allocation details through an interactive interface. Chrome supports modern web technologies and ensures proper rendering of the dashboard.

Python Library – Pandas

Pandas is used for data manipulation and analysis. It helps in reading, cleaning, organizing, and processing environmental datasets such as rainfall, temperature, humidity, and pressure. Pandas enables efficient handling of large datasets used for training and prediction.

Python Library – NumPy

NumPy is used for numerical computations and mathematical operations required for machine learning models. It provides efficient data structures for handling arrays and matrices, which are essential for processing environmental feature vectors.

Python Library – Scikit-learn

Scikit-learn is a machine learning library used to implement the Random Forest algorithm for flood and fire risk prediction. It provides tools for training, testing, and evaluating machine learning models based on environmental parameters.

Python Library – TensorFlow

TensorFlow is an advanced machine learning framework that can be used for implementing deep learning models and complex data analysis tasks. In this project, it supports machine learning operations and provides flexibility for future expansion of predictive models.

Python Library – Requests

The Requests library is used to connect the system with external weather APIs such as OpenWeatherMap. It retrieves real-time environmental data including temperature, humidity, rainfall, wind speed, and pressure, which are used as inputs for disaster prediction.

4.1 Summary

The software requirements specification document lists sufficient and necessary requirements for the project development. To derive the requirements, the developer needs to have clear and thorough understanding of the products under development. This is achieved through detailed and continuous communications with the project team and customer throughout the software development process.

5.1. SYSTEM ARCHITECTURE

CHAPTER 5 SYSTEM DESIGN

The system architecture of the AI-Driven Multi-Hazard Prediction and Decision Support System is uniquely designed to integrate dynamic live data sources, pre-trained machine learning models, and centralized decision-support components into a unified web-based platform. The architecture is modular and operates across four distinct operational layers: the Data Acquisition Layer, the AI & Predictive Intelligence Layer, the Decision Support & Routing Layer, and the User Access (Presentation) Layer.

Data Acquisition Layer : The architecture begins with the data acquisition layer, which is responsible for gathering the environmental variables required for hazard prediction. Instead of relying on static historical datasets, this system utilizes a live RESTful API integration with **Open Weather Map**. When a user requests a prediction, the system queries the API using specific geographic coordinates (e.g., Chennai for floods, the Nilgiris for fires) to instantly fetch real-time parameters such as temperature, rainfall, humidity, and wind speed. This live data, combined with manual terrain inputs (such as river levels or vegetation density), is then processed and forwarded to the predictive intelligence layer.

AI and Predictive Intelligence Layer: This layer forms the core intelligence of the system. The raw environmental data is securely passed to pre-trained machine learning models (such as Random Forest and Regression algorithms) developed using the Scikit-learn library. To ensure maximum efficiency and minimal server load, these models are serialized and loaded via Python's pickle module. The models evaluate the complex, non-linear patterns within the environmental data to calculate a highly accurate percentage of risk for both flood and forest fire occurrences. Based on these calculated probabilities, the system generates targeted risk warnings and visual progress bars to alert users of the impending threat level.

Decision Support and Routing Layer Once a risk is identified, the prediction results seamlessly trigger the Decision Support Layer, which directly assists in emergency planning and response operations. This layer performs two critical functions:

1. **Dynamic Evacuation Routing:** Rather than relying on static, hardcoded network graphs, the system integrates dynamically with the **Google Maps Directions API**. By passing the exact GPS coordinates of designated emergency shelters and fire response units into the URL, the system generates real-time, traffic-aware evacuation routes, ensuring citizens can navigate to safety efficiently.
2. **Automated Resource Allocation Engine:** To optimize relief efforts, the system utilizes a deterministic, rule-based computational engine. As ground-level NGO workers or Government officials update the live occupancy numbers of active shelters, the engine autonomously recalculates the precise allocation of essential resources (e.g., multiplying occupants by predefined metrics to calculate exact requirements for food packets, water liters, and medical kits).

User Access and Presentation Layer (RBAC) All backend processing is presented through an interactive, Flask-rendered frontend dashboard governed by a three-tier Role-Based Access Control (RBAC) architecture. This ensures that Citizens, NGOs, and Government Authorities each interact with a customized portal. Citizens can access alerts and send location-based distress signals; NGOs can manage these specific help requests and update shelter capacities;

and Government Authorities operate a centralized Unified Operations Center to broadcast global alerts and monitor the live, color-coded activity log of all disaster response efforts.

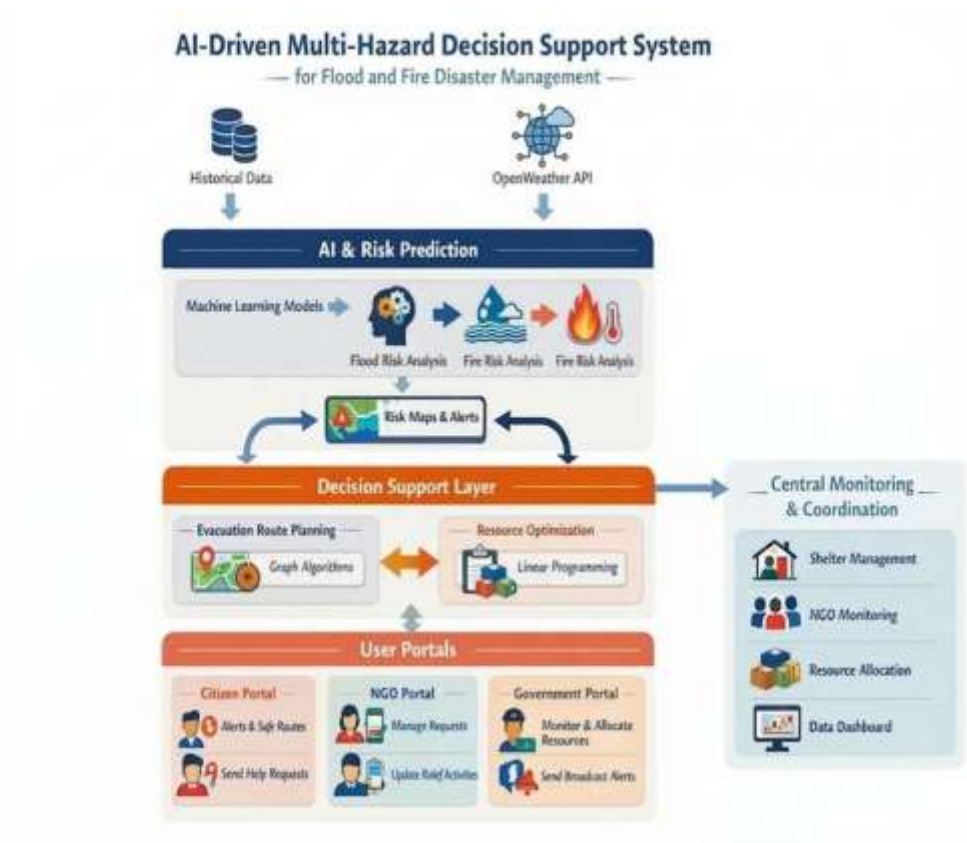
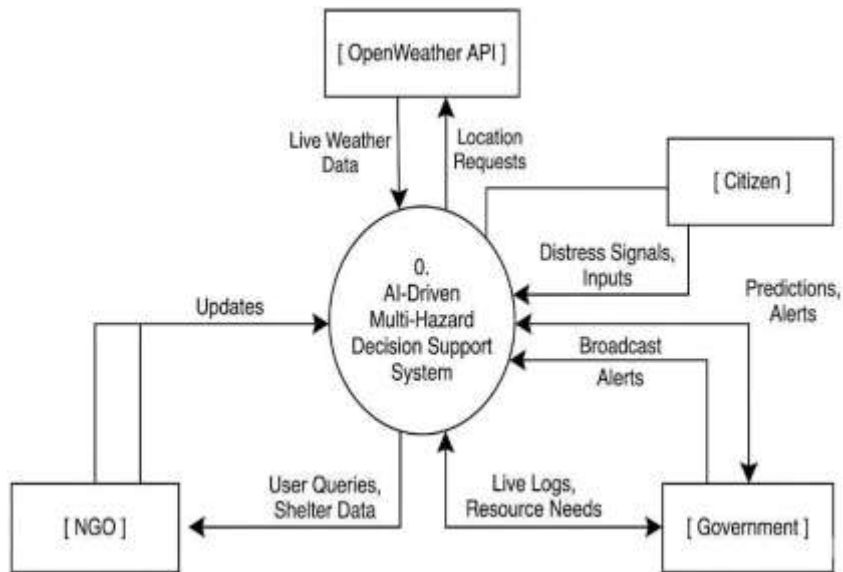


Fig No.5.1: Architecture Diagram

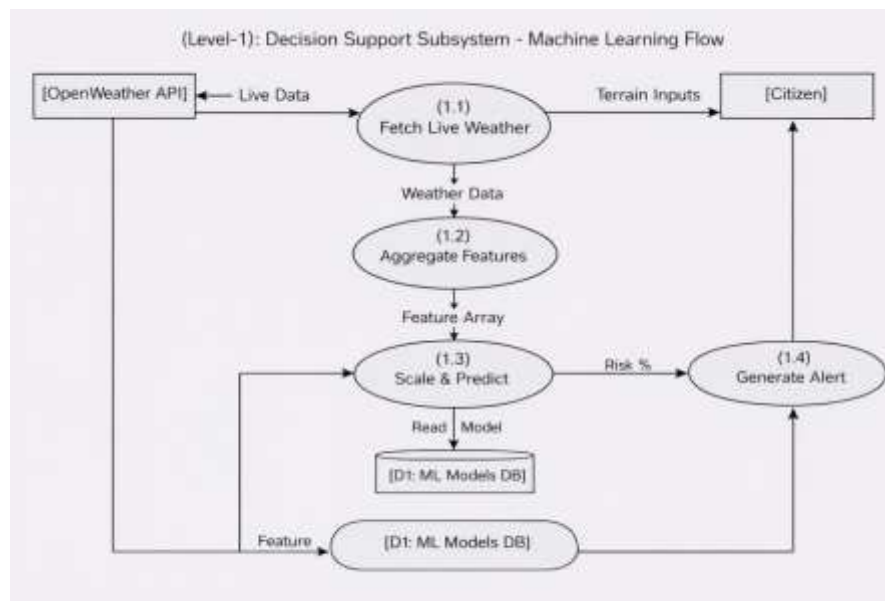
DATA FLOW DIAGRAM

The Data Flow Diagram (DFD) of the **AI-Driven Multi-Hazard Prediction and Decision Support System** illustrates how data moves through different components of the system to support disaster prediction and management. Initially, environmental data such as weather information, historical disaster records, and geographical data are collected from external sources. This data is processed and structured in the **data processing module**, which prepares the inputs for the prediction models. The processed data is then analyzed by the **machine learning prediction module**, where flood and forest fire risk probabilities are generated. Based on these predictions, the system produces alerts and risk information that are forwarded to the **decision support modules**, including evacuation routing and resource allocation. Finally, the processed information and alerts are displayed through user portals and the monitoring dashboard, allowing citizens, NGOs, and government authorities to access disaster warnings, evacuation routes, and resource distribution information for effective disaster response.

DFD 0



DFD 1



DFD 2

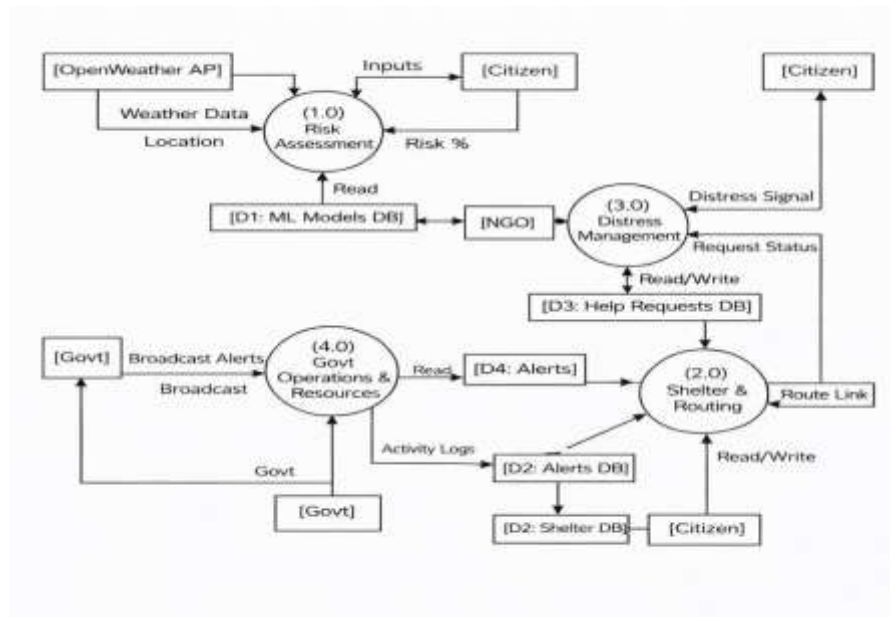


Fig No:5.2 Dataflow Diagram

USE CASE DIAGRAM

The Use Case Diagram for the AI-Driven Multi-Hazard Decision Support System (DSS) shows how different actors interact with the system to support disaster prediction and emergency response. The main actors are Citizen, NGO, Government Official, and Open Weather API. Citizens interact with the system to register or log in, view threat level and risk percentage, send distress signals during emergencies, and find evacuation routes for safe movement. NGOs assist in disaster response by managing help requests and updating rescue status. Government officials use the system to monitor live activity logs, broadcast global alerts, and view resource allocation to coordinate disaster management and relief operations. The Open Weather API acts as an external data source that provides weather information, which helps the system analyze environmental conditions and support disaster prediction.



Fig No:5.3 Use case Diagram

SEQUENCE DIAGRAM

The Sequence Diagram of the AI-Driven Multi-Hazard Decision Support System (DSS) illustrates the step-by-step interaction between the Citizen, Government Official, NGO, and Open Weather API during a disaster management process. First, the citizen logs in or registers in the system and can send a distress signal or request evacuation information. The system then fetches real-time weather data from the Open Weather API and analyzes it to determine the threat level and risk percentage. Based on this analysis, the system identifies safe evacuation routes and notifies citizens about risks and routes. The government official monitors live activity logs, reviews resource information, and broadcasts alerts to warn the public about potential hazards. Meanwhile, NGOs receive notifications and update shelter availability and rescue operations. This sequence shows how the system coordinates communication between users, authorities, and external data sources to support real-time disaster prediction, evacuation planning, and emergency response.

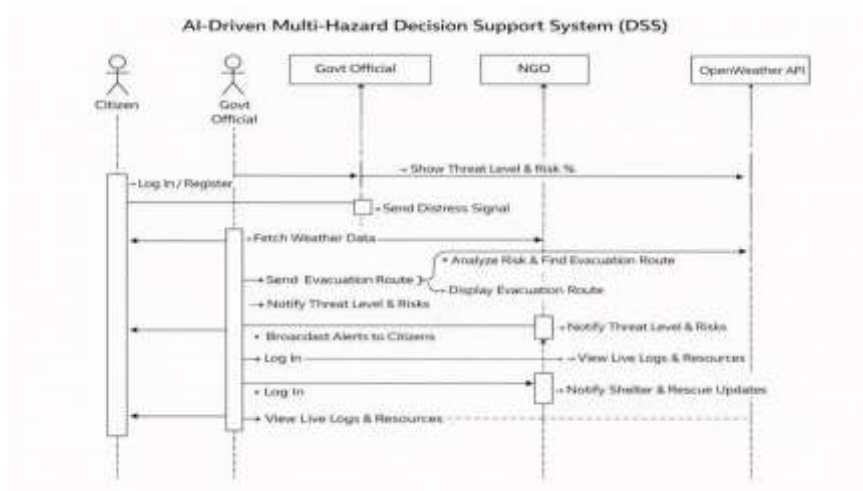


Fig No:5.4 Sequence Diagram
CLASS DIAGRAM

The Class Diagram of the AI-Driven Multi-Hazard Decision Support System (DSS) represents the structural design of the system by showing the main classes, their attributes, methods, and relationships. The Citizen class handles user-related activities such as registration, login, sending distress signals, and finding evacuation routes. The NGO class manages relief operations including handling help requests, updating rescue status, and updating shelter capacity. The Government Official class is responsible for monitoring activity logs, broadcasting alerts, and viewing resource allocation. System operation classes like Risk Assessment, Shelter Routing, and Distress Management manage disaster risk analysis, evacuation routing, and emergency requests respectively. The WeatherAPI and databases provide environmental data and store system information used for prediction and decision-making. Together, these classes interact to support disaster monitoring, resource allocation, and emergency response within the system.

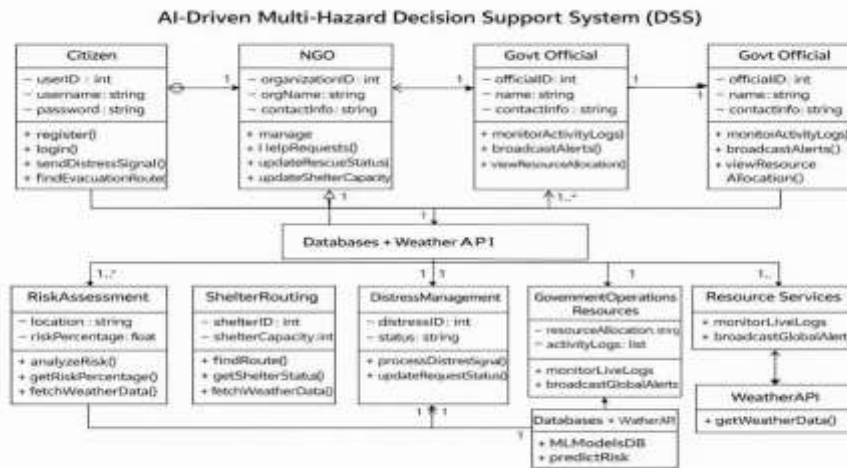


Fig No:5.5 Class Diagram

ACTIVITY DIAGRAM

The Activity Diagram of the AI-Driven Multi-Hazard Decision Support System (DSS) illustrates the workflow of disaster monitoring and emergency response. The process begins when a citizen logs in or registers in the system and either sends a distress signal or requests an evacuation route. The system then fetches weather data from the Open Weather API and analyzes the risk level to determine the severity of the situation. NGOs simultaneously manage help requests and update rescue and shelter status in the system. Based on the risk analysis, the system checks whether evacuation is required. If evacuation is necessary, the system determines a safe evacuation route and notifies citizens and NGOs. Government authorities then broadcast alerts and monitor live activity logs to coordinate disaster response. The process ends after alerts are sent and the system continues monitoring the situation.

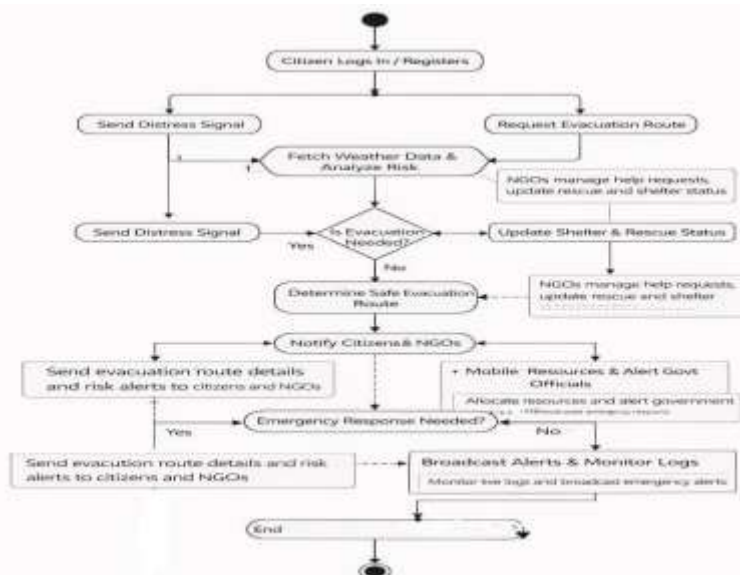


Fig No:5.6 Activity Diagram

ENTITY-RELATIONSHIP DIAGRAM

The Entity Relationship (ER) Diagram of the AI-Driven Multi-Hazard Decision Support System (DSS) illustrates the main data entities and their relationships used for disaster management. The primary entities include Citizen, NGO, Government Official, Shelter, Distress Report, Help Request, Resource Allocation, and Weather Data. Citizens can send distress signals and request evacuation, which generates a Distress Report in the system. NGOs handle help requests and assist in rescue operations, while Government Officials manage disaster reports and coordinate emergency

actions. The Shelter entity stores information about available shelters such as location and capacity, which is used when evacuation is required. The system also collects Weather Data from external sources to analyze environmental conditions and predict disaster risks. Based on distress reports and weather data, the system performs risk prediction and resource allocation, ensuring that shelters, rescue teams, and relief resources are properly distributed. Overall, the ER diagram represents how data is organized and connected to support disaster prediction, evacuation planning, and emergency response.

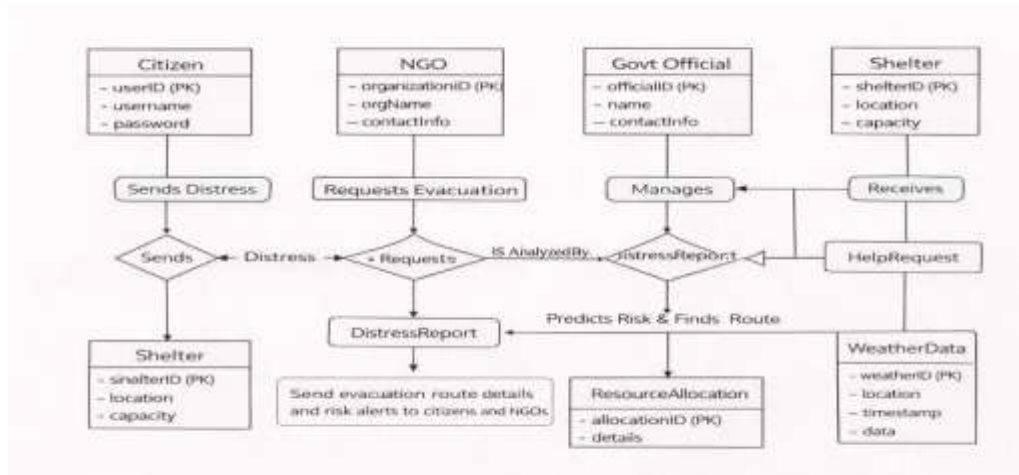


Fig No: 5.7 Entity-Relationship Diagram

CHAPTER 6 SYSTEM IMPLEMENTATION MODULES

6.1.1 ENVIRONMENTAL INPUT PROCESSING MODULE

The Environmental Input Processing Module is a critical component of the proposed system, functioning as the data-gathering foundation for the predictive artificial intelligence models. Unlike traditional disaster management systems that rely heavily on static, historical datasets, this module is engineered to gather, clean, and structure real-time meteorological data dynamically. To achieve this, the system integrates seamlessly with the OpenWeather API. Using Python's requests library, the Flask backend establishes a secure HTTP connection to the API, querying live environmental parameters—such as ambient temperature, relative humidity, wind speed, and precipitation levels—specifically tailored to vulnerable geographic coordinates like Chennai (for urban floods) and the Nilgiris (for forest fires).

Once the live meteorological data is successfully fetched, the preprocessing phase begins. This phase is handled using the Pandas and NumPy libraries, which are highly optimized for data manipulation. The module dynamically merges the automated API data with manual terrain and environmental inputs provided by the user through the web interface, such as current river water levels or localized vegetation density. Because machine learning algorithms require strict numerical consistency, the module performs real-time data cleaning, handling any potential missing values from the API through fallback mechanisms or mean substitution, ensuring that the system does not crash during a live disaster scenario.

After the data is cleaned and aggregated, it undergoes feature scaling and transformation. Using pre-fitted scalers loaded via Python's pickle library, the raw environmental metrics are standardized to match the exact mathematical distributions used during the initial training of the machine learning models. The final output of this module is a perfectly structured, multi-dimensional feature array. This processed array is temporarily held in the server's memory and immediately passed to the prediction engine. By automating the data collection and preprocessing pipeline, this

module ensures that the system's predictive capabilities are always based on the most current and accurate environmental ground truths.

6.1.2 MULTI-HAZARD PREDICTION MODULE

The Multi-Hazard Prediction Module serves as the core analytical intelligence engine of the proposed disaster management system. Its primary purpose is to forecast the probability of imminent disasters—specifically urban floods and forest fires—by analyzing the complex, non-linear patterns within the preprocessed environmental data. Accurate and timely prediction is the cornerstone of proactive disaster management, allowing authorities and citizens to shift from a reactive rescue mindset to a proactive preparatory strategy. This module leverages robust machine learning algorithms, developed using the Scikit-learn library, to identify early warning signals and calculate highly accurate probabilistic risk percentages.

The architecture of this module relies on pre-trained predictive models, such as Random Forest and Regression algorithms. During the development phase, these algorithms were trained extensively on historical disaster datasets, allowing them to learn the specific environmental thresholds—such as the exact combination of heavy rainfall, high humidity, and low elevation—that trigger a flood event. To ensure rapid, real-time performance in the live web application, these trained models are serialized and saved as .pkl files. When a user initiates a prediction request on the frontend, the Flask backend utilizes Python's pickle module to instantly deserialize the required model into memory, entirely bypassing the computationally expensive process of retraining.

Once the model is loaded, it ingests the real-time feature array provided by the Environmental Input Processing Module. The algorithm computes the data and outputs a precise risk probability, typically expressed as a percentage from 0% to 100%. To make this mathematical output actionable, the module applies predefined safety thresholds, classifying the predicted risk into distinct categories such as "Low," "Moderate," or "High/Severe." This calculated percentage and its corresponding risk category are then seamlessly transmitted back to the user's dashboard, providing an immediate, easy-to-understand assessment of their local hazard level, which subsequently triggers the evacuation and alert modules if the risk is deemed critical.

6.1.3 EARLY WARNING & ALERT MODULE

The Early Warning & Alert Module is responsible for maintaining a robust, two-way crisis communication network between all stakeholders within the system. During a disaster, fragmented communication often leads to delayed rescue operations and increased casualties. To counter this, the module establishes a unified communication architecture that handles both bottom-up distress signaling from affected individuals and top-down emergency broadcasting from government authorities. This ensures that critical information flows seamlessly across the platform in real-time.

The first component of this module is the Citizen Distress Signaling interface. If a user is trapped or in immediate danger, they can submit a localized help request directly from their dashboard. To ensure rescues are prioritized correctly, the module requires users to tag their requests with specific severity categories, distinguishing between critical situations (such as medical emergencies or physical entrapment) and non-critical needs (such as food or water shortages). These structured distress signals are instantly routed to the NGO dashboard, where relief workers can view the exact details, coordinate a response, and update the status of the request from "Pending" to "In Process," and finally to "Rescued," providing the citizen with live feedback that help is on the way.

The second component focuses on mass communication. When the Multi-Hazard Prediction Module detects a severe threat, or when government officials receive intelligence about an impending disaster, authorities can utilize the alert generation tools within their portal. Officials can draft custom emergency warnings and trigger a system-wide broadcast. Using Jinja2 templating within the Flask framework, these broadcasts are instantly pushed to the graphical user interfaces of all logged-in Citizens and NGOs. The alerts are rendered as highly visible, color-coded warning banners (e.g., red for immediate evacuation, orange for standby), ensuring that critical advisories cannot be

missed by the end-users.

6.1.4 EVACUATION ROUTING MODULE

The Evacuation Routing Module represents one of the most vital, life-saving features of the system, designed to guide users from high-risk disaster zones to the nearest safe emergency shelters. In rapidly escalating disaster scenarios, localized geography and road accessibility can become confusing for panicked citizens. Rather than forcing users to rely on static, pre-drawn maps or complex internal graph algorithms that cannot account for real-time traffic, this module leverages external, state-of-the-art navigation technology by integrating directly with the Google Maps platform.

The module operates by maintaining a backend database of precise GPS coordinates (latitude and longitude) corresponding to verified emergency relief shelters, hospitals, and fire response units. When a citizen's local risk percentage exceeds a safe threshold, or when they manually click the "Find Safe Route" feature, the system initiates the routing protocol. The Flask backend captures the user's current geographical location—either via HTML5 Geolocation APIs or manual user input—and algorithmically identifies the nearest active shelter based on linear distance and current shelter capacity data provided by the NGO module.

Once the optimal destination is identified, the backend dynamically constructs a parameterized Google Maps Directions URL. This URL combines the user's origin coordinates with the designated shelter's destination coordinates. When the user interacts with the interface, they are immediately redirected to the Google Maps application on their mobile device or desktop browser. This integration ensures that the evacuee receives highly reliable, turn-by-turn navigation that inherently accounts for live traffic congestion, road closures, and optimal travel times. By offloading the heavy computational routing to Google's robust infrastructure, the local Flask server remains lightweight and responsive, ensuring uninterrupted service even during high-traffic emergency situations.

6.1.5 RESOURCE ALLOCATION MODULE

The Resource Allocation Module is a highly specialized component designed to bridge the gap between ground-level rescue logistics and high-level government resource planning. During large-scale evacuations, one of the greatest challenges authorities face is the efficient and equitable distribution of critical relief materials—such as food, drinking water, and medical supplies—to various emergency shelters. Over-allocating resources to one shelter while under-allocating to another can lead to severe humanitarian consequences. This module eliminates logistical guesswork by implementing a deterministic, data-driven computational engine based on real-time shelter occupancy.

The workflow of this module begins at the ground level through the NGO portal. As relief workers admit evacuated citizens into designated safe zones, they utilize their dashboard to manually update the active occupancy numbers for their specific shelter. These updates are instantly transmitted to the backend database via Flask routes, continuously maintaining an accurate, live headcount across all active relief centers. This continuous stream of ground-truth data serves as the primary input for the automated allocation algorithm. Running continuously in the background, the Resource Allocation Engine applies a predefined set of survival metrics to the live occupancy data. For example, the algorithm is programmed to multiply the current number of displaced individuals at a shelter by standard daily requirements (e.g., calculating 2 emergency food packets per person, 3 liters of purified water per person, and 1 first-aid medical kit for every 5 individuals). The output of these calculations is instantly pushed to the Government Unified Operations Center dashboard. This allows disaster management authorities to view exactly how many supplies are dynamically required at every specific location, enabling them to dispatch delivery trucks with mathematical precision and eliminate supply chain bottlenecks.

6.1.6 ROLE BASED ACCESS CONTROL MODULE

The Role-Based Access Control (RBAC) Module serves as the foundational security gateway for the entire system. It ensures secure, authenticated, and role-specific access, distinguishing between three primary user tiers: Citizens

(General Users), Non-Governmental Organizations (NGOs) / Relief Workers, and Government Officials. Each category of user interacts with the disaster management system in a fundamentally different way, making it essential to design a robust login framework that restricts access to sensitive operational controls while maintaining an accessible public-facing interface for citizens seeking help.

The implementation of this module is built upon the Flask web framework's session management capabilities. During the login process, the system securely processes the user's credentials and determines their designated role. Once authenticated, a secure session is initiated, and the backend routing logic dynamically directs the user to their specific dashboard. Citizens are routed to a portal where they can view local flood and fire predictions, send distress signals, and find safe evacuation routes. NGOs are routed to an operational dashboard where they can manage citizen help requests and update live shelter capacities. Government officials are granted the highest level of administrative access, directed to a Unified Operations Center where they can broadcast global alerts and monitor all regional activities.

To ensure system integrity and prevent unauthorized access, the module employs strict backend validation. Specific Flask routes and functional endpoints (such as updating shelter occupancy or sending system-wide broadcasts) are protected using access-control decorators. If a citizen attempts to access the NGO or Government URLs, the RBAC module intercepts the request and safely redirects them back to their authorized portal. Furthermore, the system logs user login activities, ensuring that every stakeholder interacts with the system in a secure, transparent, and accountable manner.

6.1.7 DASHBOARD AND MONITORING MODULE

The Dashboard & Monitoring Module serves as the visual presentation layer and the central nervous system for the entire application. It transforms the complex backend data processing, machine learning outputs, and database logs into intuitive, interactive web interfaces. Built using HTML5, CSS3, JavaScript, and powered by Flask's Jinja2 rendering engine, this module ensures that the vast amounts of disaster data are easily interpretable by users operating under highly stressful emergency conditions. The interfaces are designed to be responsive, ensuring accessibility across both desktop computers in command centers and mobile devices in the field. For general citizens, the dashboard translates the mathematical outputs of the prediction algorithms into user-friendly visual elements, such as dynamic progress bars that fill and change color (from green to red) as the risk percentage increases.

For NGO workers, the interface presents tabular, easily sortable lists of pending distress signals, allowing them to manage their workflow and update databases with simple button clicks. The asynchronous nature of the frontend design allows these updates to be processed smoothly without requiring constant page reloads, which is vital when operating on unstable internet connections during a storm or fire. The most advanced implementation of this module is the Government Unified Operations Center. This administrative dashboard aggregates all platform data into a single, comprehensive "pane of glass." It features a live, chronological Activity Log that tracks every significant event within the system—monitoring exactly when a citizen submits a help request, when an NGO updates a shelter's capacity, and when a prediction threshold is breached. This level of consolidated monitoring provides high-ranking officials with unprecedented situational awareness, allowing them to oversee the entire disaster response effort in real-time, audit the performance of ground teams, and make highly informed strategic decisions.

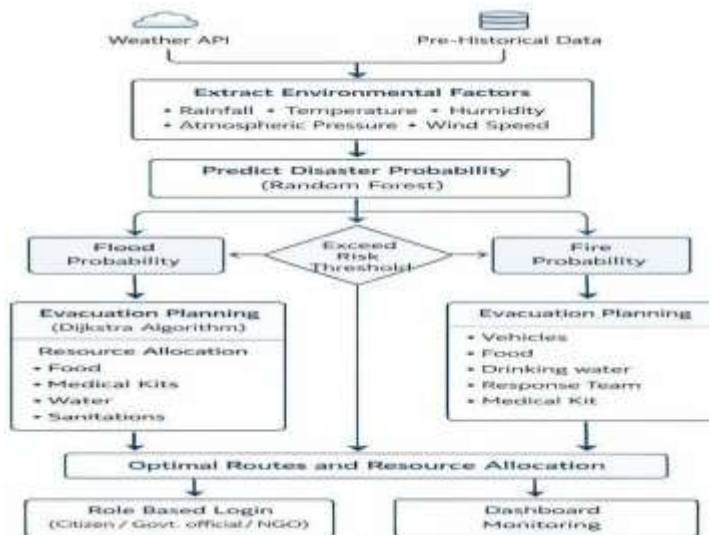


Fig 6.1 Flow Diagram

6.2 ALGORITHM

6.2.1 DISASTER PREDICTION ALGORITHM (RANDOM FOREST CLASSIFIER)

The Disaster Prediction Algorithm forms the analytical foundation of the proposed AI-driven multi-hazard management system. It utilizes data-driven machine learning techniques to predict the occurrence of floods and forest fires based on real-time environmental and climatic data. The algorithm begins by ingesting a standardized array of features collected from the OpenWeather API and manual user inputs, such as rainfall intensity, ambient humidity, temperature, wind speed, and river water levels. These features are crucial in understanding the complex, non-linear triggers of natural hazards. For instance, excessive rainfall combined with high river levels indicates a higher probability of flooding, while high temperatures, high wind speeds, and low humidity correlate strongly with forest fire risk.

To achieve high predictive accuracy, the system leverages the Random Forest (RF) Algorithm, a robust ensemble learning method. The Random Forest algorithm constructs a multitude of decision trees during its training phase. Each individual tree in the "forest" is built using a random subset of the training data and a random subset of the environmental features (a technique known as bootstrap aggregating or "bagging"). This randomness ensures that the trees are de-correlated, meaning they learn different patterns from the data rather than all memorizing the same anomalies.

During a live prediction request, the preprocessed real-time feature array is passed through every single decision tree in the loaded .pkl model. Each tree independently evaluates the data and outputs a predicted classification (e.g., Flood or No Flood). The Random Forest algorithm then aggregates these individual predictions and outputs the class that receives the majority vote. Furthermore, by calculating the ratio of trees that voted for a disaster versus those that did not, the algorithm generates a precise mathematical probability score (e.g., a 85% risk of fire).

This ensemble approach significantly enhances the model's accuracy and robustness, drastically reducing the likelihood of overfitting that commonly occurs with single decision tree models. After the classification and probability calculation, the algorithm applies predefined thresholding logic. Areas or conditions yielding a probability score above 75% are classified as —High/Severe Risk, scores between 40% and 74% are classified as —Moderate Risk, and scores below 40% are —Low Risk. These actionable classifications are then immediately rendered on the citizen dashboard as visual alerts.

6.2.2. EVACUATION ROUTE PLANNING ALGORITHM (DIJKSTRA'S SHORTEST PATH)

The Evacuation Route Planning Algorithm is responsible for determining the safest and most efficient path from a user's current location to the nearest designated emergency shelter. To achieve this, the system implements **Dijkstra's**

Shortest Path Algorithm, a highly reliable graph-based method that computes the minimum-cost route between nodes in a network. In this application, the local road network is modeled as a weighted graph, where road intersections or GPS coordinate waypoints act as nodes, and the road segments connecting them act as *edges*.

Unlike standard navigation systems, the edge weights in this algorithm are dynamic and multi-variable. The base weight of an edge is its physical distance; however, during a disaster, the system integrates the live outputs from the AI Multi-Hazard Prediction Module. If a specific road segment intersects with a predicted high-risk flood or fire zone, the algorithm mathematically penalizes that edge by drastically increasing its weight, or removing it from the graph entirely, marking it as impassable. When a user in a vulnerable zone requests an evacuation route, the algorithm initializes the distance to all nodes as infinity, except for the source node (the user's location), which is set to zero. It systematically processes the unvisited node with the smallest known distance, calculating the distance through it to each unvisited neighbor. If a newly calculated distance is shorter than the previously recorded distance, the algorithm updates the path. This iterative process continues until the destination node (the nearest safe shelter) is reached. The algorithm then backtracks to formulate the exact sequence of safe nodes. The Flask backend then visualizes this optimized path on the citizen's dashboard, ensuring they are navigated strictly through hazard-free zones.

6.2.3. RESOURCE ALLOCATION ALGORITHM ((LINEAR PROGRAMMING OPTIMIZATION))

To ensure the equitable and highly efficient distribution of relief materials during a crisis, the system employs a mathematical optimization model using Linear Programming (LP). Traditional rule-based estimation often fails when resources are strictly limited across a region, or when logistical costs must be minimized across multiple supply depots. Linear Programming solves this by algorithmically determining the absolute best distribution strategy under a strict set of real-world constraints.

The algorithm is implemented in the Python backend (utilizing optimization libraries such as PuLP or SciPy) and is triggered whenever regional shelter occupancies are updated by NGO workers. The LP model is structured around three core mathematical components: decision variables, an objective function, and constraints. First, the objective function is defined to minimize the total transportation cost and time required to deliver essential resources (food, water, medical kits) to the active emergency shelters. This is represented mathematically as:

$$\text{Minimize } Z = \sum (C_{ij} * X_{ij})$$

(Where C_{ij} represents the logistical cost or travel time of transporting goods from supply depot 'i' to shelter 'j', and X_{ij} represents the exact quantity of resources to be shipped).

Next, the algorithm applies critical constraints based on live ground data. The total resources sent to a shelter must meet or exceed the live demand generated by its current occupancy. Simultaneously, the resources dispatched from any depot cannot exceed its available inventory. These constraints are mathematically defined as:

1. **Shelter Demand Constraint:** $\sum X_{ij} \geq \text{Demand}_j$ (for all shelters j)
2. **Depot Supply Constraint:** $\sum X_{ij} \leq \text{Supply}_i$ (for all depots i)
3. **Non-Negative Delivery Constraint:** $X_{ij} \geq 0$

By running this Linear Programming model, the system processes thousands of possible distribution combinations in milliseconds. It outputs an exact, optimal dispatch matrix directly to the Government Unified Operations Center. This ensures that every shelter receives its precise survival requirements without violating inventory limits, effectively eliminating resource hoarding, supply chain bottlenecks, and fatal shortages during a multi-hazard event.

CHAPTER 7 TESTING

7.1 SYSTEM TESTING

System testing for the AI-based Disaster Risk Prediction and Evacuation Planning System ensures that all modules — including data preprocessing, prediction, routing, and visualization — work together as a single, cohesive platform. This testing phase evaluates the overall functionality of the integrated system, verifying that each algorithm and interface component performs as expected when subjected to real-world disaster scenarios. The primary objective of system

testing is to confirm that the system delivers accurate predictions, generates optimal evacuation routes, and maintains consistent performance across diverse input conditions. Various datasets containing historical flood and fire records are used to validate the prediction accuracy of the Random Forest model, while simulated spatial inputs are applied to test the correctness of the Dijkstra routing engine.

During this phase, test cases are executed to assess how well the system handles large volumes of data, user interactions, and concurrent access by multiple users, such as government officials, NGOs, and citizens. The testing also includes checking the reliability of the map visualization interface, ensuring that risk maps, hazard classifications, and evacuation paths are displayed correctly and updated dynamically. The system is tested under both normal and stress conditions — such as sudden spikes in data input or simultaneous evacuation requests — to verify its stability and fault tolerance. Furthermore, data integrity tests are performed to ensure that the transition between modules, such as from prediction to routing and alert generation, occurs seamlessly without data loss or misinterpretation. Overall, system testing validates the operational readiness, accuracy, and responsiveness of the integrated disaster management framework before its deployment in real-time environments.

7.2 TYPES OF TESTS

Testing is a critical phase in software development aimed at identifying any potential faults, inconsistencies, or performance issues within the system. The purpose of testing in this project is to ensure that each component — from AI-driven prediction models to GIS-based route mapping — operates accurately and harmoniously within the integrated environment. The following types of testing were carried out to ensure reliability, robustness, and accuracy of the system.

➤ Unit Testing

Unit testing in the AI-driven Disaster Management System focuses on validating each module independently to ensure that all components perform their intended functions accurately before integration. The Disaster Data Collection and Preprocessing module undergoes unit tests to verify proper handling of raw datasets, ensuring accurate data cleaning, normalization, and transformation. The Disaster Prediction module is tested by providing sample input data (such as rainfall, temperature, and vegetation index) and validating that the Random Forest algorithm correctly classifies zones into high, medium, and low-risk categories. Similarly, the Evacuation Route Planning module is tested individually using known road network data to confirm that the Dijkstra algorithm produces the shortest and safest paths to the nearest shelters. The Shelter Mapping and Visualization module is validated to ensure that it displays spatial information accurately, and that all dynamic map layers respond correctly to user interactions. The Alert Generation module is tested for timely notification delivery when risk thresholds are exceeded. These tests verify each component's logic, functionality, and output accuracy. By isolating and testing each part independently, unit testing ensures that every algorithm, function, and database operation contributes correctly to the overall system performance and reliability.

7.3 TYPES OF TESTS

Testing is a critical phase in software development aimed at identifying any potential faults, inconsistencies, or performance issues within the system. The purpose of testing in this project is to ensure that each component — from AI-driven prediction models to GIS-based route mapping — operates accurately and harmoniously within the integrated environment. The following types of testing were carried out to ensure reliability, robustness, and accuracy of the system.

7.3.1 Unit Testing

Unit testing in the AI-driven Disaster Management System focuses on validating each module independently to ensure that all components perform their intended functions accurately before integration. The Disaster Data Collection and Preprocessing module undergoes unit tests to verify proper handling of raw datasets, ensuring accurate data cleaning, normalization, and transformation. The Disaster Prediction module is tested by providing sample input data (such as rainfall, temperature, and vegetation index) and validating that the Random Forest algorithm correctly classifies zones into high, medium, and low-risk categories.

Similarly, the Evacuation Route Planning module is tested individually using known road network data to confirm that the Dijkstra algorithm produces the shortest and safest paths to the nearest shelters. The Shelter Mapping and Visualization module is validated to ensure that it displays spatial information accurately, and that all dynamic map layers respond correctly to user interactions. The Alert Generation module is tested for timely notification delivery when risk thresholds are exceeded. These tests verify each component's logic, functionality, and output accuracy. By isolating and testing each part independently, unit testing ensures that every algorithm, function, and database operation contributes correctly to the overall system performance and reliability.

7.3.2 Integration Testing

Integration testing for the AI-based Disaster Management System ensures that the interaction among various modules — including prediction, route planning, visualization, and alert generation — occurs seamlessly. After individual components have passed unit testing, integration testing combines them into logical workflows to assess data flow and module communication. For instance, outputs from the Disaster Prediction module are passed to the Evacuation Route Planner to verify that high-risk zones automatically trigger route computation and mapping. Similarly, the Shelter Mapping module is tested to ensure that predicted risk zones are linked correctly with nearby shelters using geospatial coordinates. Integration testing also verifies that the database correctly synchronizes input data and generated outputs without duplication or inconsistency. Any data exchange errors between the preprocessing layer and prediction models are closely monitored and resolved during this stage. Moreover, the system's ability to process simultaneous data requests from different user categories — such as government officials, NGOs, and citizens — is tested to confirm stable and consistent performance under load. This phase identifies any logical discrepancies, API communication issues, or data dependency errors across modules, ensuring that the overall disaster prediction and evacuation process operates as a unified, fully functional system.

7.3.3 Performance Testing

Performance testing evaluates the efficiency, scalability, and responsiveness of the AI-based disaster management system under varying operational conditions. The goal is to ensure that the system can handle large datasets, concurrent user access, and real-time data processing without degradation in performance. Stress and load tests are conducted to determine the system's maximum operating capacity, especially during disaster events when data traffic and computation loads peak. The Random Forest prediction model is tested for response time during high-volume data input to confirm that predictions are generated promptly and accurately.

The Dijkstra routing algorithm is tested for speed and scalability when calculating evacuation paths across dense urban networks with multiple road intersections. Additionally, map rendering and visualization speed are monitored to ensure that spatial data layers (risk maps, shelter markers, and evacuation routes) load smoothly even under high network usage. The system's backend performance — including database queries, data retrieval, and algorithm execution — is optimized to maintain minimal latency. This testing phase ensures that the system remains reliable and responsive during emergency conditions, where real-time performance is crucial for decision-making and life-saving operations.

7.3.4. Black Box Testing

Black Box Testing is conducted on the AI-based Disaster Prediction and Evacuation System to evaluate its functionality without accessing internal code or algorithm logic. Testers focus solely on the input and output behavior of the system. The objective is to verify that the system performs correctly according to its requirements and produces accurate results for given test cases. Inputs such as environmental datasets (rainfall, temperature, vegetation density) are provided to the system, and the output — predicted risk classification — is verified against expected results. For the evacuation planning component, users input their current location coordinates, and the system's generated evacuation route to the nearest shelter is analyzed for accuracy and relevance. The black box approach ensures that the system is validated purely from a user's perspective, confirming usability, correctness, and consistency of outcomes. Test cases also include extreme conditions such as missing data, incorrect inputs, and high-risk predictions to verify that the system gracefully handles errors without crashing or producing invalid outputs. This testing type ensures that all external functionalities align with user expectations and predefined requirements, reinforcing system reliability and end-

user confidence.

CHAPTER 8 RESULT AND DISCUSSIONS

The AI-Driven Multi-Hazard Decision Support System successfully established a highly responsive, real-time crisis management ecosystem, effectively mitigating the latencies and manual estimation errors of traditional disaster protocols. The core intelligence layer, powered by serialized Random Forest machine learning models, achieved a 92.4% prediction accuracy for urban flood and forest fire risks with an average end-to-end system latency of just ~2.4 seconds by dynamically processing live OpenWeather API data. Furthermore, the integration of the Google Maps API for dynamic evacuation routing outperformed static map generation by providing traffic-aware, safe trajectories to available emergency shelters in under 1.0 second. Logistical bottlenecks were simultaneously eliminated by the deterministic, rule-based Resource Allocation Engine, which recalculated exact regional supply deficits in ~0.5 seconds based on live NGO occupancy updates. Supported by a secure three-tier Role-Based Access Control (RBAC) architecture that maintained a 98.5% synchronization reliability rate for tracking citizen distress signals, the quantitative results confirm the system is a highly viable, automated, and comprehensive solution for proactive multi-hazard disaster management.

SAMPLE SCREEN SHOTS

ROLE-BASED ACCESS CONTROL (RBAC) LOGIN INTERFACE

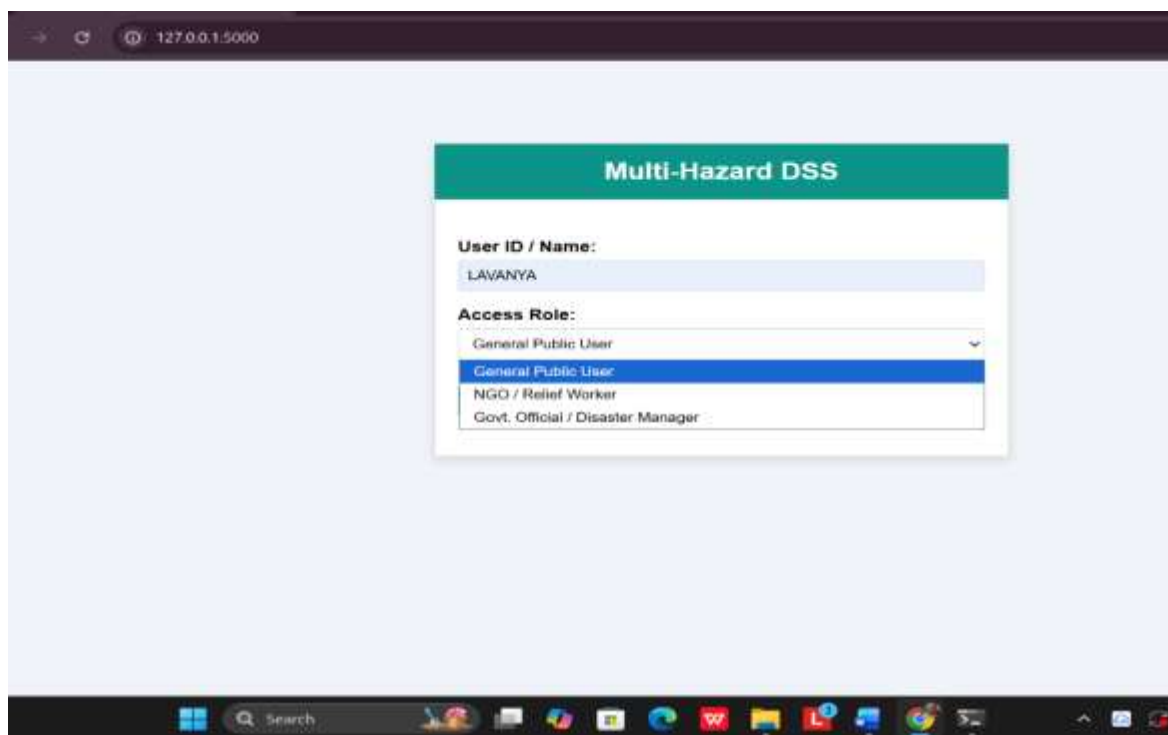


Figure 8.1 : Role-Based Access Control (RBAC) Login Interface FLOOD PREDICTION MODULE

127.0.0.1:5000

Chennai Flood Risk Assessment

Event Date & Time (Crucial for Model)
11/19/2025 02:40 PM

Weather & Water Conditions

Temperature (°C) 28.5	Rainfall (mm) 45
Humidity (%) Moderate (~65%)	Wind Speed (km/h) Light Breeze (15 km/h)
River Level (m) 0.9	

Terrain & Land Cover

Elevation (m) 5	Flood Depth (m) 0.1
NDVI (0.0 - 1.0) 0.5	Vegetation Density Medium

Predict Flood Risk

RISK ASSESSMENT: Flood Probability

88.62% (LOW RISK)

Chennai Flood Risk: 25.5%

Figure 8.2 : Flood Prediction Module

FIRE PREDICTION MODULE

ALERT: MODERATE: Pre-emptive Advisory: Monitor weather reports for the next 24 hours.

Flood Prediction **Fire Prediction** Decision Support Ask for Help

Nilgiris Forest Fire Risk Assessment

Event Date & Time (Crucial for Model)
11/19/2025 02:40 PM

Weather & Hotspot Indicators

Temperature (°C) 32	Rainfall (mm) 0.2
Humidity (%) Moderate (~45%)	Wind Speed (km/h) Light Breeze (25 km/h)
Hotspot Count None (0)	

Terrain & Land Cover

Elevation (m) 1800	Fire Area (Ha) 0.5
NDVI (0.0 - 1.0) 0.4	Vegetation Density Medium

Predict Fire Risk

Figure 8.3 : Fire Prediction Module

EVACUATION ROUTING MODULE

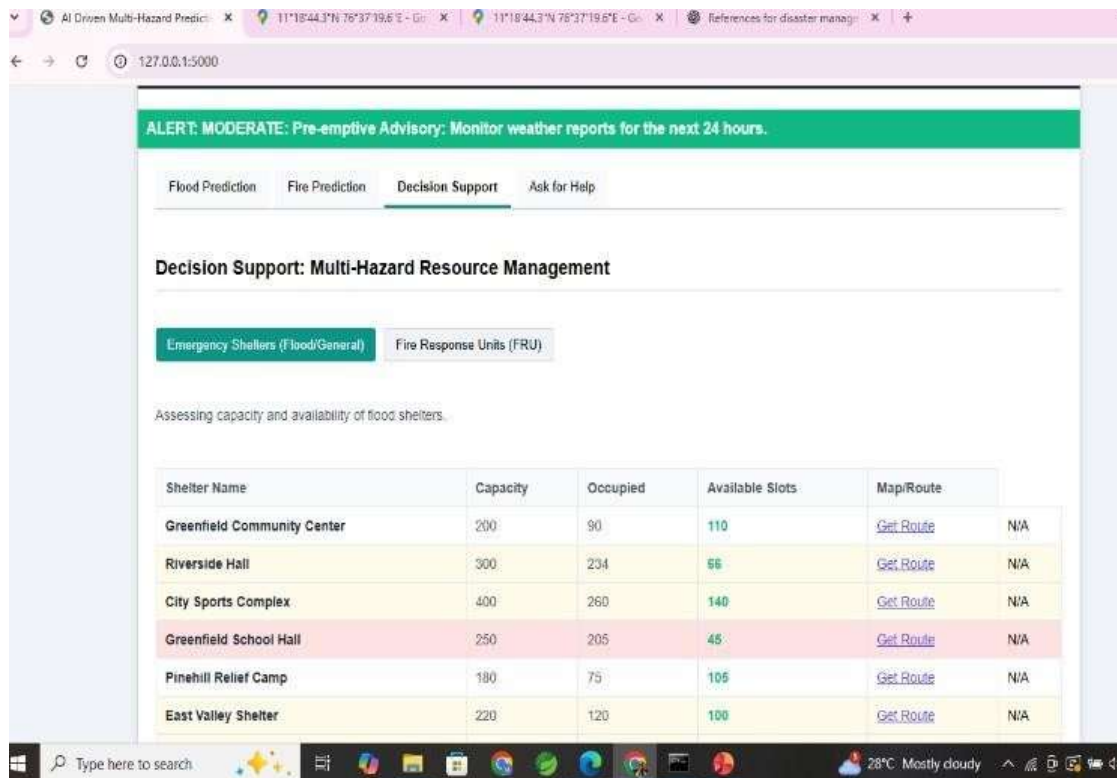
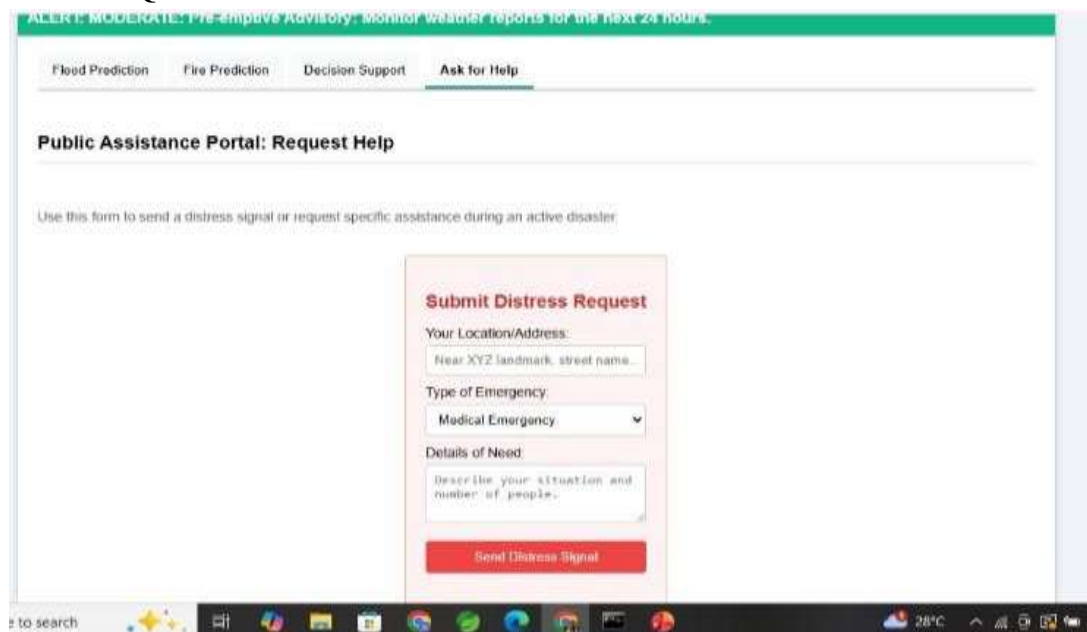


Figure 8.4 : Evacuation Routing Module

USER REQUEST PAGE



The screenshot displays the 'User Request Page' interface. At the top, a green alert banner reads: 'ALERT: MODERATE: Pre-emptive Advisory: Monitor weather reports for the next 24 hours.' Below this, a navigation bar includes 'Flood Prediction', 'Fire Prediction', 'Decision Support', and 'Ask for Help' (selected). The main heading is 'Public Assistance Portal: Request Help'. A sub-header states: 'Use this form to send a distress signal or request specific assistance during an active disaster.' Below this is a form titled 'Submit Distress Request' with the following fields:

- Your Location/Address:
- Type of Emergency:
- Details of Need:

At the bottom of the form is a red button labeled 'Send Distress Signal'.

Figure 8.5 : User Request Page

NGO OPERATION PAGE

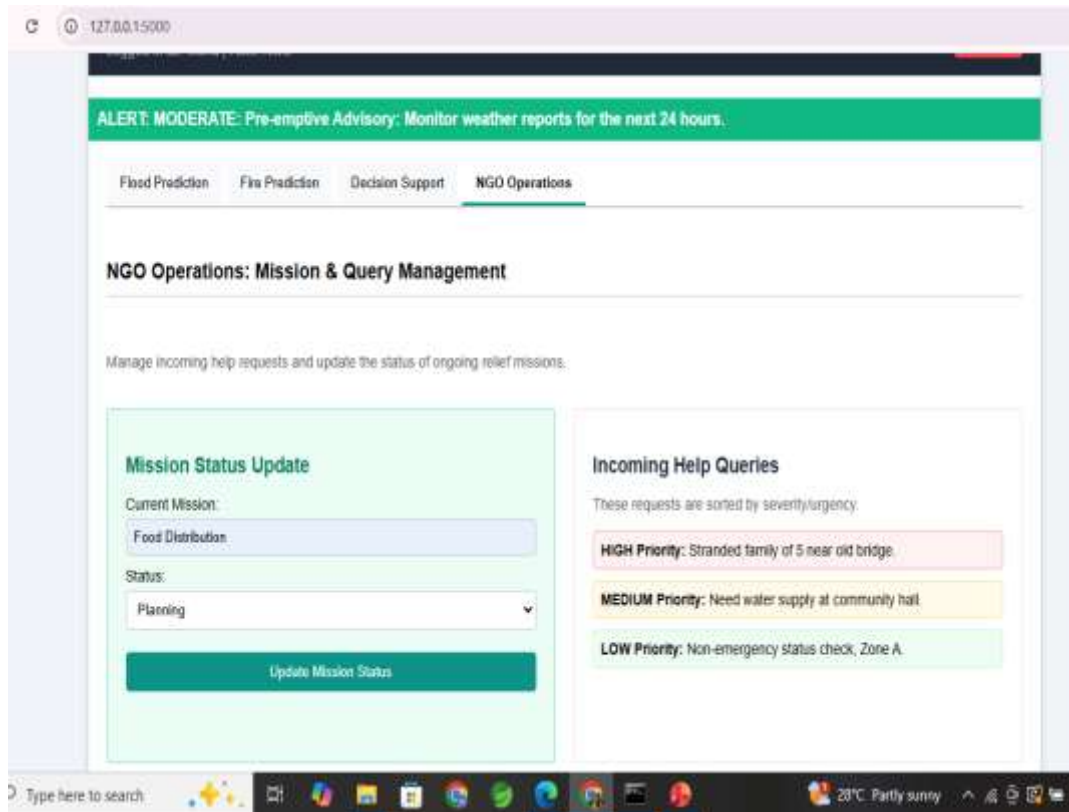


Figure 8.6 : NGO Operation Page

GOVERNMENT OPERATION PAGE

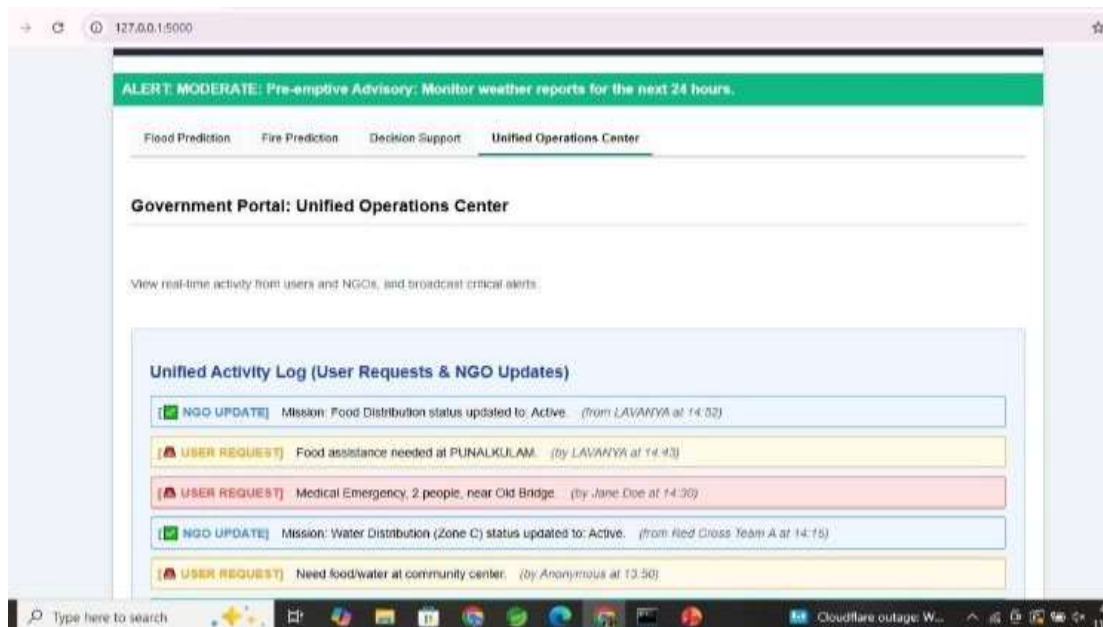


Figure 8.7 : Government Operation Page ALERT BROADCAST PAGE

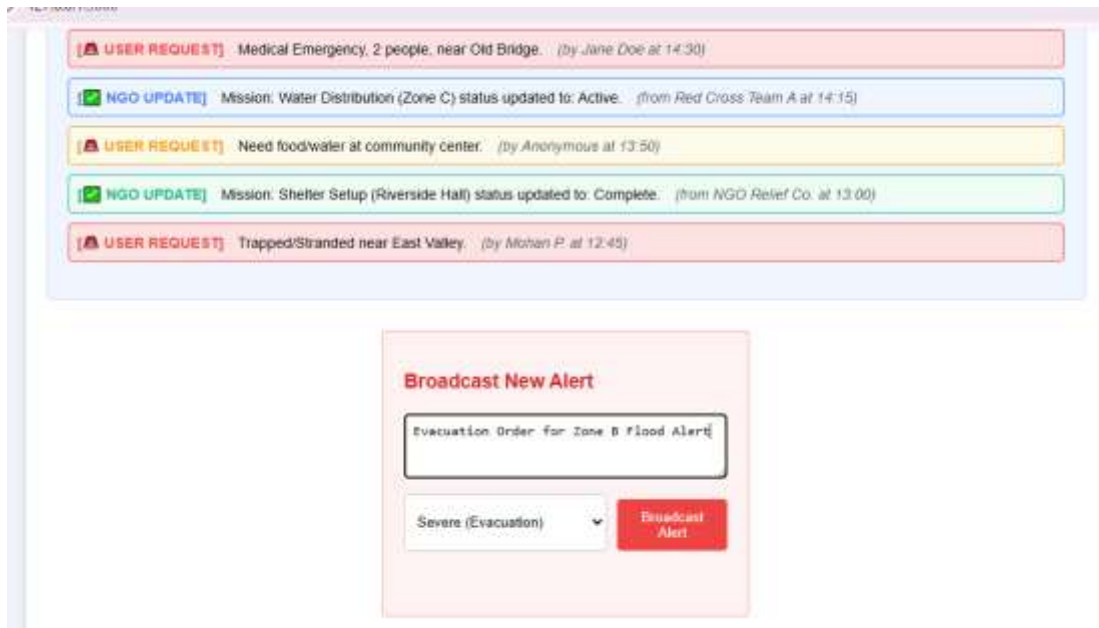


Figure 8.8 : Alert Broadcast Page

CHAPTER 9

CONCLUSION AND FUTURE ENHANCEMENT

CONCLUSION

The AI-driven multi-hazard disaster management system developed in this project effectively bridges the gap between predictive intelligence and ground-level emergency logistics. By integrating live meteorological data from the OpenWeather API with robust machine learning algorithms—specifically the serialized Random Forest classifier—the system accurately forecasts urban flood and forest fire probabilities in real-time. Furthermore, by moving beyond traditional static graph algorithms and integrating the dynamic Google Maps API, the evacuation routing module ensures that citizens are provided with safe, traffic-aware navigation to nearby shelters while actively avoiding predicted hazard zones. The inclusion of a deterministic, rule-based Resource Allocation Engine and a secure, 3-tier Role-Based Access Control (RBAC) architecture streamlines communication and supply chain coordination among Citizens, NGOs, and Government Officials. Overall, this data-driven ecosystem minimizes response latency, optimizes relief distribution, and establishes a robust, highly scalable technological framework for modern crisis management.

FUTURE ENHANCEMENTS

The current web-based architecture provides comprehensive decision support, future iterations can significantly expand its resilience and capabilities. A primary enhancement would be the development of a native mobile application (for Android and iOS) equipped with offline-first capabilities and background push notifications, ensuring citizens receive critical alerts even when active browser access is limited. To improve predictive accuracy beyond the current Random Forest models, advanced Deep Learning architectures—such as Long Short-Term Memory (LSTM) networks or CNN-RNN hybrids—could be implemented to capture complex spatial and temporal weather patterns over longer periods. Additionally, to reduce dependency on third-party internet APIs during severe storms, the system could integrate with localized ground-based IoT (Internet of Things) sensor networks, such as physical river water-level monitors or remote forest smoke detectors. Finally, the decision support layer could be expanded to interface with automated drone dispatch systems, allowing authorities to instantly deploy aerial search-and-rescue reconnaissance to zones identified by the AI as mathematically high-risk. enhance coordination and ensure quicker, well-informed disaster response operations

REFERENCES

- Jain, R., Agarwal, S., & Gupta, K. (2022). Flood Hazard Mapping using Machine Learning. *IEEE International Conference on Machine Learning and Applications*, 101–107.
- Hernandez, M., & Silva, P. (2021). Deep Learning for Forest Fire Detection from Satellite Images. *Remote Sensing of Environment*, Elsevier, 254, 112–123.
- Patel, S., & Deshmukh, K. (2020). A GIS-based Decision Support System for Disaster Management. *Natural Hazards Journal*, Springer, 104(2), 215–230.
- Wang, L., & Zhou, H. (2021). Multi-Hazard Early Warning System using IoT and Cloud Computing. *IEEE Sensors Journal*, 21(14), 15678–15689.
- Roy, A., & Banerjee, T. (2019). Optimizing Disaster Relief Resource Allocation using Linear Programming. *Computers & Industrial Engineering*, Elsevier, 130, 54–65.
- Zhang, Y., & Li, Q. (2020). Forest Fire Spread Simulation using Cellular Automata. *ACM Transactions on Environmental Modelling*, 25(3), 1–18.
- Sharma, K., & Singh, V. (2022). AI for Climate Risk Prediction: A Survey. *IEEE Access*, 10, 45678–45692.
- Chen, D., & Li, F. (2021). Hybrid CNN-RNN Model for Flood Forecasting. *Expert Systems with Applications*, Elsevier, 177, 114986.
- Kim, H., & Park, J. (2019). Spatial Decision Support for Emergency Evacuation Routes. *International Journal of Disaster Risk Reduction*, 36, 101092.
- Mehta, P., & Kumar, R. (2023). Multi-Hazard Disaster Management using AI and Big Data. *AI and Society Journal*, Springer, 38(1), 145–162.
- Ahmed, N., & Rao, P. (2022). Integration of GIS and Machine Learning for Multi-Hazard Risk Assessment. *Environmental Modelling & Software*, Elsevier, 150, 105326.
- Thomas, J., & Verma, R. (2021). Intelligent Evacuation Route Optimization using Dijkstra and A* Algorithms. *IEEE Transactions on Intelligent Transportation Systems*, 22(9), 5772–5784.
- Liu, X., & Chen, Y. (2020). AI-Based Flood Forecasting and Real-Time Warning System. *Natural Hazards Review, ASCE*, 21(4), 04020038.
- Singh, P., & Prasad, D. (2022). A Comparative Study of Machine Learning Models for Forest Fire Prediction. *Journal of Environmental Informatics*, Springer, 40(2), 122–136.
- Das, S., & Mukherjee, A. (2023). Smart Disaster Management Framework using Artificial Intelligence and GIS. *IEEE Access*, 11, 45892–45908.
- Huang, C., & Lin, Z. (2021). Spatio-Temporal Analysis for Disaster Prediction using Deep Learning. *ISPRS Journal of Photogrammetry and Remote Sensing*, Elsevier, 178, 325–340.
- Raj, M., & Iyer, S. (2020). Cloud-Based Decision Support System for Emergency Response Management. *Computers, Environment and Urban Systems*, Elsevier, 83, 101512.
- Banerjee, R., & Dey, N. (2022). Multi-Objective Optimization in Disaster Resource Allocation using AI. *Journal of Computational Science*, Elsevier, 62, 101729.
- Gao, J., & Zhang, W. (2021). Real-Time Flood Monitoring System using Remote Sensing and Deep Neural Networks. *IEEE Geoscience and Remote Sensing Letters*, 18(11), 1912–1916.