## **DISASTER MANAGEMENT SYSTEM**

## A PROJECT REPORT

## (21CSC205P – Database Management Systems)

*Submitted by*

# Rashmika S [Reg No:RA2211030010089]

# Shreya De [Reg No:RA2211030010091]

# Chaarvi Ayineni [Reg No:RA2211030010122]

*Under the Guidance of*

# Dr.Kayalvizhi R

Assistant Professor, Department of Networking and Communications

*in partial fulfillment of the requirements for the degree of*

**BACHELOR OF TECHNOLOGY**

# in

**COMPUTER SCIENCE AND ENGINEERING specialization in Cyber Security**



**DEPARTMENT OF NETWORKING AND COMMUNICATIONS**

# COLLEGE OF ENGINEERING AND TECHNOLOGY

# SRM INSTITUTE OF SCIENCE AND TECHNOLOGY KATTANKULATHUR– 603 203

# MAY 2024



# SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

# KATTANKULATHUR–603 203

# BONAFIDE CERTIFICATE

Certified to be the bonafide work done by **Rashmika S (Reg. No. RA2211030010089)** of II year/IV sem B.Tech Degree Course in the Project Course – **21CSC205P Database Management Systems** in **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**, Kattankulathur for the academic year 2023-2024.

**Date:**

### Faculty in Charge

Dr.Kayalvizhi R

Assistant Professor

Department of Networking and Communications

SRM Institute of Science and Technology

Kattankulathur,

Chennai – 603 203

### HEAD OF THE DEPARTMENT

Dr. Annapurani K

Professor & Head

Department of Networking and Communications

SRM Institute of Science and Technology

Kattankulathur,

Chennai – 603 203

### 

**ABSTRACT**

The "Community Disaster Management System" is a multifaceted web-based platform designed to revolutionize local communities' responses to emergencies. By integrating various key components, such as a Resource Inventory, Incident Reporting and Alerts, Communication Hub, Mapping and Visualization, and Volunteer Management tools, the system enhances coordination, efficiency, and communication among stakeholders involved in disaster response efforts. Through advanced technologies, including web development tools, backend frameworks, databases, mapping libraries, and messaging APIs, the platform facilitates swift and effective resource allocation, timely incident reporting, real-time communication, and efficient volunteer management. Community members can input and update crucial information regarding emergency supplies, medical equipment, shelters, and volunteer availability, allowing for seamless resource coordination and allocation. Automated alerts ensure that relevant stakeholders are promptly informed of incidents based on severity and location, enabling swift and targeted responses. The Communication Hub serves as a centralized platform for real-time communication among emergency responders, volunteers, local authorities, and residents, fostering collaboration and coordination during crisis situations. Additionally, the Mapping and Visualization component offers an interactive map with GIS features, providing a clear overview of incident locations, resource distribution points, evacuation routes, and other critical information to aid decision-making and enhance situational awareness. Overall, the Community Disaster Management System holds the potential to significantly improve community resilience by empowering stakeholders to respond effectively to emergencies, fostering collaboration, and optimizing resource allocation and logistics management, thereby addressing essential challenges in local disaster management.

**PROBLEM STATEMENT**

The purpose of the "Disaster Management System" project is to empower local communities with the tools and resources necessary to orchestrate swift and effective responses during emergencies. In the face of escalating climate-related events and unforeseen disasters, the imperative for efficient and coordinated disaster management at the community level has become more pronounced than ever. This project signifies a proactive step towards bolstering community resilience and preparedness through a centralized hub that facilitates communication, resource allocation, and incident reporting. By incorporating key components such as a robust Resource Inventory, Incident Reporting and Alerts system, Communication Hub, Mapping and Visualization capabilities, and Volunteer Management tools, the platform equips communities with the means to respond promptly and efficiently to crises. Through seamless interaction among emergency responders, volunteers, local authorities, and residents, coupled with advanced mapping and visualization features, the system enhances coordination, information exchange, and situational awareness. Ultimately, the goal of the project is to foster collaboration, resilience, and preparedness within communities, enabling them to effectively mitigate the impacts of disasters and safeguard lives and property.

The imperative for the "Disaster Management System" arises from a pressing need to address the inherent challenges faced by local communities in effectively managing and responding to emergencies.

The increasing frequency and intensity of natural disasters, coupled with the unpredictable nature of emergencies, underscore the necessity for a comprehensive and coordinated disaster management solution tailored to the grassroots level.

**Limited Coordination**: In the absence of a centralized platform, community-level coordination during disasters is often fragmented and inefficient. Emergency responders, volunteers, and community members operate in silos, leading to suboptimal resource allocation and response efforts.

**Resource Scarcity:** Communities frequently grapple with insufficient resources during emergencies. A lack of real-time information on available supplies, medical equipment, shelters, and volunteer manpower hampers the ability to mobilize resources effectively, exacerbating the impact of disasters.

**Delayed Reporting:** Timely reporting of incidents is crucial for initiating swift responses. The absence of a streamlined incident reporting system results in delays, hindering the deployment of resources and assistance to affected areas.

**Communication Challenges**: Effective communication is paramount during emergencies. Existing communication channels may prove inadequate, causing delays in conveying critical information to relevant stakeholders, impeding the overall response coordination.

**Community Engagement:** A centralized system is needed to actively engage community members in disaster response efforts.

Volunteer management tools are essential for mobilizing local resources and manpower, fostering a sense of community resilience, and encouraging proactive participation. The "Disaster Management System" is conceived as a response to these challenges, aiming to bridge gaps in coordination, resource management, incident reporting, and community engagement.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Chapter No** | **Chapter Name** | **Page No** |
|  | Problem understanding, Identification of Entity and Relationships, Construction of DB using ER Model for the project | **8** |
|  | Design of Relational Schemas, Creation of Database Tables for the project. | **17** |
|  | Complex queries based on the concepts of constraints, sets, joins, views, Triggers and Cursors. | **27** |
|  | Analyzing the pitfalls, identifying the dependencies, and applying normalizations | **31** |
|  | Implementation of concurrency control and recovery mechanisms | **38** |
|  | Code for the project | **42** |
|  | Result and Discussion (Screen shots of the implementation with front end) | **46** |
|  | Attach the Real Time project certificate / Online course certificate | **56** |

**CHAPTER 1**

**INTRODUCTION**

**1.1 PROBLEM UNDERSTANDING**

As the "Community Disaster Management System" continues to evolve, its impact extends beyond theoretical frameworks into the realm of practical, real-time disaster response. Through ongoing development and refinement, this system stands poised to revolutionize the way communities, governments, and social organizations mitigate, respond to, and recover from emergencies.

One of the key benefits of further development lies in the system's ability to harness real-time data and analytics, enabling stakeholders to make informed decisions swiftly. By integrating live data feeds from various sources such as weather stations, social media platforms, and emergency services, the system can provide up-to-the-minute insights into evolving disaster scenarios. This real-time situational awareness is invaluable in orchestrating agile response strategies, optimizing resource allocation, and minimizing the impact of disasters on affected communities.

Moreover, as the system matures, its interoperability with existing government frameworks and social organizations becomes increasingly seamless. By establishing robust data-sharing protocols and interoperable interfaces, the system can integrate seamlessly with government agencies responsible for disaster management, emergency services, and infrastructure maintenance. This synergy facilitates a coordinated, multi-agency response to disasters, enhancing the overall effectiveness of response efforts.

Furthermore, the scalability and flexibility of the system make it an invaluable asset for social organizations involved in disaster relief and humanitarian aid. Whether responding to localized incidents or large-scale disasters, these organizations can leverage the system's capabilities to mobilize resources, coordinate volunteers, and deliver aid to affected communities more efficiently. This not only accelerates the pace of recovery but also ensures that assistance reaches those in need in a timely and targeted manner.

For governments, the benefits are manifold. By leveraging the Community Disaster Management System, authorities can enhance their disaster preparedness and response capabilities, leading to more resilient communities and reduced human and economic losses. The system's ability to facilitate data-driven decision-making, streamline resource allocation, and improve communication fosters greater efficiency and effectiveness in disaster management operations. Additionally, by empowering local communities to play an active role in disaster response, the system promotes civic engagement and strengthens social cohesion, laying the foundation for a more resilient society.

In summary, the further development of the Disaster Management System holds immense potential to transform disaster management practices, offering real-time insights, enhancing collaboration, and bolstering resilience in the face of adversity. By bridging the gap between technology and emergency management, this project emerges as a powerful tool in safeguarding communities and building a more resilient future for all.

**1.2 IDENTIFICATION OF ENTITY AND RELATIONSHIPS**

**1.2.1 ENTITIES:**

**User:**

UserID (Primary Key), Username, Email, Role

**Resource:**

ResourceID (Primary Key), Name, Type, Location, Availability, Quantity

**Incident**:

IncidentID(Primary Key), Title, Description, Location, Severity, Status, ReportTime

**Alert**:

AlertID (Primary Key), Type, Description, Severity, IssuedTime

**Communication**:

MessageID (Primary Key), Message, TimeSent, SenderID (Partial Key), ReceiverID (Partial Key)

**Volunteer**:

VolunteerID (Primary Key), Skills, Availability

**Task**:

TaskID (Primary Key), Title, Description, Location, Status, StartTime, EndTime, VolunteerID (Partial Key)

**Map**:

MapID (Primary Key), IncidentID (Partial Key), DisasterLevel

**GISFeature**:

GISFeatureID (Primary Key), MapID (Partial Key)

**Organization**:

OrganizationID (Primary Key), Name, Type, Location, ContactInfo

**Location**:

LocationID (Primary Key), Name, Latitude, Longitude, Description

**EmergencyPlan**:

PlanID (Primary Key), Title, Description, Steps, LastUpdate

**Funding**:

FundingID (Primary Key), Source, Amount, DateReceived, Purpose

**IncidentHistory**:

HistoryID (Primary Key), IncidentID (Partial Key), Description, Resolution, TimeResolved

**IncidentCategory**:

CategoryID (Primary Key), Name, Description

**Contribution**:

ContributionID (Primary Key), ResourceID (Partial Key), ContributorID (Partial Key), QuantityProvided

**Allocation**:

AllocationID (Primary Key), ResourceID (Partial Key), VolunteerID (Partial Key), AdministratorID (Partial Key), QuantityAllocated

**1.2.2 RELATIONSHIPS:**

**Reports**:

One User can report multiple Incidents.

**Triggers**:

Each Incident can trigger multiple Alerts.

**Sends**:

Users can send multiple Communications.

**RegistersAs**:

Users can register as Volunteers.

**AssignedTo**:

Each Volunteer can be assigned to multiple Tasks.

**Utilizes**:

Incidents can utilize multiple Resources.

**Provides**:

Users can provide multiple Resources.

**Maps**:

Each Map is associated with one Incident.

**Features**:

Multiple GIS Features can be associated with one Map.

**Manages**:

Organizations can manage multiple Incidents.

**LocatedAt**:

Resources are located at one Location.

**Funds**:

Organizations can provide funding for various purposes.

**Supervises**:

Users can supervise multiple Tasks.

**Allocates**:

Organizations can allocate multiple Resources.

**DevelopedBy**:

Organizations can develop multiple Emergency Plans.

**RecordedIn**:

Multiple Incident Histories can be recorded for one Incident.

**CategorizedAs**:

Incidents can be categorized into multiple Incident Categories.

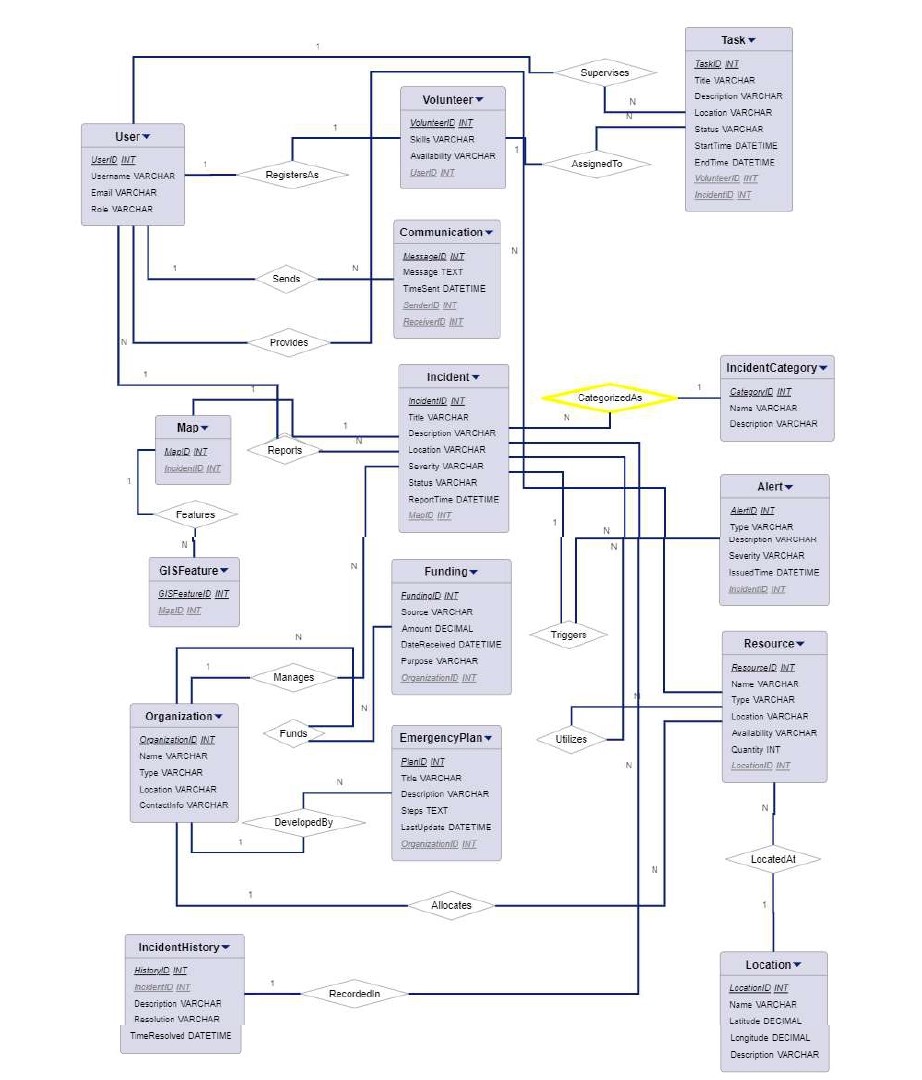
**Contributes**:

Users can contribute resources to the system, which are recorded as contributions.

**AllocatesToVolunteer**:

Administrators can allocate resources to volunteers for deployment during disaster response operations.

**1.3 CONSTRUCTION OF DB USING E-R MODEL**

****

**Fig.1.1** ER Diagram

The database schema for the Disaster Management System is designed to capture essential entities and their relationships using the Chen notation. The system includes entities such as Location, User, Volunteer, DisasterList, Disaster, VolunteerAssignment, and Task, each serving a distinct role in facilitating effective disaster management.The "Location" entity represents geographical locations with an assigned ID and a name. The "User" entity, linked to the "Volunteer" entity through a one-to-many relationship, represents individuals involved in disaster response efforts. The "Volunteer" entity includes details such as volunteer ID, name, date of birth, and an indicator of whether the volunteer is currently assigned to a task.The "DisasterList" entity defines types of disasters with an assigned ID and a name. The "Disaster" entity is linked to both the "Location" and "DisasterList" entities, capturing details such as the disaster ID, name, location, report time, and an indicator of whether the disaster is currently active.

The "VolunteerAssignment" entity establishes a many-to-one relationship between volunteers and disasters, allowing for the assignment of volunteers to specific disaster response operations. The "Task" entity, linked to the "Disaster" entity, represents tasks associated with a particular disaster, including task ID, name, and a status indicator.The relationships outlined in the schema contribute to a comprehensive Disaster Management System. For instance, volunteers are linked to both users and locations, enabling efficient management and deployment during emergencies. Disasters are associated with both specific locations and disaster types, providing a detailed context for response efforts. The volunteer assignments and tasks further enhance the system's functionality by facilitating the coordination and tracking of volunteers and tasks during disaster response operations. This structured database schema lays the foundation for a robust and organized system to support effective disaster management.

**CHAPTER 2**

**RELATIONAL MODEL**

**2.1 DESIGN OF RELATIONAL SCHEMA**

1. User (User\_id, Username, Email, Role)

2. Volunteer (Volunteer\_id, User\_id, Skills, Availability)

3. Resource (Resource\_ID, Name, Type, Location, Availability)

4. Incident (Incident\_ID, Title, Description, Location, Severity, Status, Report\_Time)

5. Alert (Alert\_ID, Type, Description, Severity, Issued\_time)

6. Communication (Message\_ID, Message, Time\_Spent, Sender\_ID, Receiver\_ID)

7.Task (Task\_ID, Title, Description, Location, Status, Start\_Time, Volunteer\_ID)

8. Map (Map\_ID, Incident\_ID)

9. GIS features (GIS\_Feature\_ID, Map\_ID)

10. Incident Category (Category\_ID, Name, Description)

11. Incident history (History\_ID, Incident\_Id, Description,Resolution, Time\_Solved)

12. Emergency plan (Plan\_ID, Title, Description, Steps, Last\_Update)

13. Organization (Organization\_ID, Name, Type, Location, Contact\_Info)

14. Location (Location\_Id, Name, Latitude, Longitude, Description)

15. Funding (Funding\_ID, Source, Amount, Date\_Received, Purpose)

**2.2 CREATION OF DATABASE TABLES**

**1. USER**

User ( User\_id ,Username , Email , role)

|  |  |  |
| --- | --- | --- |
| Attribute | Data type | Constraints |
| User\_ID | Int | Primary Key |
| User-Name | Varchar(50) | NOT NULL |
| Email | Varchar(25) | NOT NULL |
| Mobile | Int | NOT NULL |

**2. VOLUNTEER**

Volunteer ( volunteer\_id , user\_id ,skills , availability )

|  |  |  |
| --- | --- | --- |
| Attributes | Data Type | Constraints |
| Volunteer\_ID | Int | Primary Key |
| Skills | Varchar(50) |  |
| Availability | Varchar(50) |  |

**3. RESOURCE**

Resource ( Resource\_ID , Name ,Type , Location , Availability )

|  |  |  |
| --- | --- | --- |
| Attributes | Data Type | Constraints |
| Resource\_ID | Int | Primary Key |
| Name | Varchar(25) | NOT NULL |
| Type | Varchar(50) | NOT NULL |
| Location | Varchar(50) |  |
| Availability | Varchar(50) |  |
| Quantity | Int |  |

**4. INCIDENT**

Incident ( Incident\_ID , Title , Description , Location , Severity , Status , Report\_Time)

|  |  |  |
| --- | --- | --- |
| Attributes | Data Type | Constraints |
| Incident\_ID | Int | Primary Key |
| Title | Varchar(50) | NOT NULL |
| Description | Varchar(100) |  |
| Location | Varchar(50) | NOT NULL |
| Severity | Varchar(25) | NOT NULL |
| Status | Boolean | NOT NULL |
| Report\_ | time |  |

**5. ALERT**

Alert ( Alert\_ID , Type , Description , Severity , Issued\_time )

|  |  |  |
| --- | --- | --- |
| Attributes | Data Type | Constraints |
| Alert\_ID | int | Primary\_Key |
| Type | Varchar(25) | NOT NULL |
| Description | Varchar(100) |  |
| Severity | Varchar(50) | NOT NULL |
| Issued\_Time | TIME | Unique |

**6. COMMUNICATION**

Communication ( Message\_ID , Message , Time\_Spent , Sender\_ID , Receiver\_ID)

|  |  |  |
| --- | --- | --- |
| Attributes | Data Type | Constraints |
| Message\_ID | Int | Primary Key |
| Message | Varchar(100) | NOT NULL |
| Time\_Sent | TIME | Unique |
| Sender\_ID | Int | Foreign Key |
| Receiver\_ID | int | Foreign Key |

**7. TASK**

Task ( Task\_ID , Title , Description , Location , Status , Start\_Time ,Volunteer\_ID)

|  |  |  |
| --- | --- | --- |
| Attributes | Data Type | Constraints |
| Task\_ID | Int | Primary Key |
| Title | Varchar(25) | NOT NULL |
| Description | Varchar(100) |  |
| Location | Varchar(25) | NOT NULL |
| Status | Boolean | NOT NULL |
| Start\_Time | TIME | Unique |
| Volunteer\_ID | TIME | Foreign Key |

**8. MAP**

Map ( Map\_ID , Incident\_ID )

|  |  |  |
| --- | --- | --- |
| Attributes | Data Type | Constraints |
| Map\_ID | Int | Primary Key |
| Incident\_ID | int | Foreign Key |

**9. GIS FEATURES**

GIS features ( GIS\_Feature\_ID , Map\_ID )

|  |  |  |
| --- | --- | --- |
| Attributes | Data Types | Constraints |
| GIS\_Feature\_ID | Int | Primary Key |
| Map\_ID | int | Foreign Key |

**10. INCIDENT CATEGORY**

Incident Category ( Category\_ID , Name , Description )

|  |  |  |
| --- | --- | --- |
| Attributes | Data Types | Constraints |
| Category\_ID | int | Primary Key |
| Name | Varchar(25) | NOT NULL |
| Description | Vaarchar(100) |  |

**11. INCIDENT HISTORY**

Incident history ( History\_ID , Incident\_Id , Description , Resolution , Time\_Solved )

|  |  |  |
| --- | --- | --- |
| Attributes | Data Types | Constraints |
| History\_ID | Int | Primary Key |
| Incident\_ID | Int | Foreign Key |
| Description | Varchar(100) |  |
| Resolution | Varchar(100) |  |
| Time\_Solved | TIME | NOT NULL |

**12. EMERGENCY PLAN**

Emergency plan (Plan\_ID , Title ,Description ,Steps ,Last\_Update)

|  |  |  |
| --- | --- | --- |
| Attributes | Data Type | Constraints |
| Plan\_ID | int | Primary Key |
| Title | varchar(50) | NOT NULL |
| Description | Varchar(100) |  |
| Steps | Varchar(500) |  |
| Last\_Update | TIME | Unique |

**13. ORGANIZATION**

Organization ( Organization\_ID , Name , Type , Location , Contact\_Info )

|  |  |  |
| --- | --- | --- |
| Attributes | Data type | Constraints |
| Organization\_ID | int | Primary Key |
| Name | Varchar(25) | NOT NULL |
| Type | Varchar(50) |  |
| Location | Varchar(50) |  |
| Contact\_Info | Int | Unique |

**14. LOCATION**

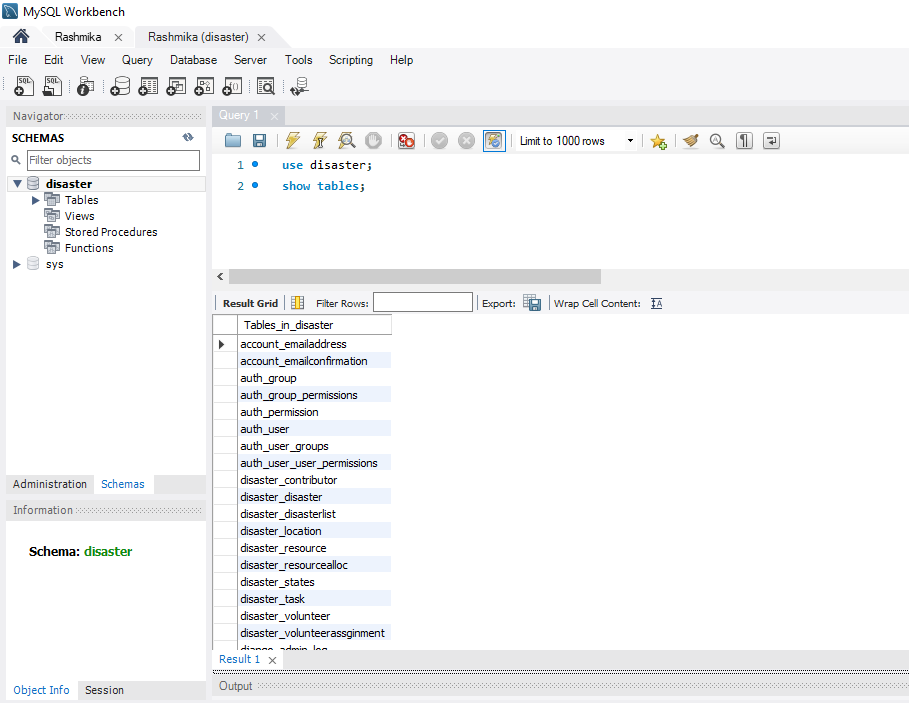
Location ( Location\_Id , Name , Latitude , Longitude , Description )

|  |  |  |
| --- | --- | --- |
| Attributes | Data Type | Constraints |
| Location\_ID | int | Primary Key |
| Name | Varchar(25) | NOT NULL |
| Latitude | Varchar(25) | Unique |
| Longitude | Varchar(25) | Unique |
| Description | Varchar(100) |  |

**15. FUNDING**

Funding ( Funding\_ID , Source , Amount ,Date\_Received ,Purpose )

|  |  |  |
| --- | --- | --- |
| Attributes | Data Type | Constraints |
| Funding\_ID | int | Primary Key |
| Source | Varchar(50) | NOT NULL |
| Amount | Numeric | NOT NULL |
| Date\_Received | DATE | NOT NULL |
| Purpose | Varchar(50) |  |

****

**Fig.2.1** List of Tables

The relational database serves as the backbone of an emergency response system, meticulously designed to manage and organize critical information vital for effective disaster management. At the core of this structure lies the "USER" table, housing essential user details such as User\_ID, Username, Email, and Role. Each user's involvement in volunteering efforts is seamlessly integrated, as evidenced by the "VOLUNTEER" table, which captures volunteer-specific information including Volunteer\_ID, associated User\_ID, Skills, and Availability.

Resources indispensable for emergency response operations are meticulously cataloged within the "RESOURCE" table. Here, Resource\_ID, Name, Type, Location, Availability, and Quantity attributes are meticulously maintained, ensuring accurate tracking and efficient utilization of vital supplies during crises.

The "INCIDENT" table stands as a cornerstone in tracking and managing incidents, recording crucial details such as Incident\_ID, Title, Description, Location, Severity, Status, and Report\_Time. This comprehensive repository enables swift and informed decision-making, facilitating effective response strategies tailored to the nature and severity of each incident.

To swiftly disseminate critical information and alerts, the "ALERT" table serves as a conduit, storing Alert\_ID, Type, Description, Severity, and Issued\_Time. This enables timely communication and proactive measures in response to emerging threats and evolving situations.

Effective communication among stakeholders is facilitated through the "COMMUNICATION" table, where Message\_ID, Message, Time\_Sent, Sender\_ID, and Receiver\_ID are meticulously recorded. This ensures seamless

information exchange and coordination among team members, enhancing situational awareness and response efficiency.

Tasks assigned to volunteers are systematically tracked within the "TASK" table, featuring Task\_ID, Title, Description, Location, Status, Start\_Time, and Volunteer\_ID attributes. This enables efficient task allocation, monitoring, and progress tracking, ensuring optimal utilization of volunteer resources.

The geographical aspect of emergency response is comprehensively addressed by the "LOCATION" table, housing Location\_ID, Name, Latitude, Longitude, and Description attributes. This spatial data facilitates accurate mapping, resource allocation, and navigation during response operations.

Additionally, the database encompasses features crucial for incident categorization and historical tracking. Incident categories and descriptions are organized within the "INCIDENT CATEGORY" table, while incident histories are meticulously logged in the "INCIDENT HISTORY" table. These components provide invaluable insights into past incidents, facilitating trend analysis, and informed decision-making.

Emergency plans, essential for guiding response efforts, are detailed in the "EMERGENCY PLAN" table, featuring Plan\_ID, Title, Description, Steps, and Last\_Update attributes.

This ensures that response efforts are guided by well-defined protocols and procedures, enhancing overall effectiveness and coordination.

Moreover, organizations involved in emergency response efforts are systematically cataloged within the "ORGANIZATION" table, which captures Organization\_ID, Name, Type, Location, and Contact\_Info. This fosters collaboration and partnership among stakeholders, facilitating coordinated response efforts.

Finally, funding sources and allocations are meticulously recorded within the "FUNDING" table, documenting Funding\_ID, Source, Amount, Date\_Received, and Purpose attributes. This transparent tracking of financial resources ensures accountability and effective resource management, bolstering the sustainability of emergency response operations.

**CHAPTER 3**

**COMPLEX QUERIES**

1. **Query to Save User Data:**

INSERT INTO auth\_user (username, email, password)

VALUES ('username', 'email@example.com', 'password');

*“This query inserts a new user into the auth\_user table with a specified username,*

*Email, and password.”*

1. **Query to Retrieve Location Data:**

SELECT \* FROM location WHERE id = 1;

*“Retrieves all columns of a location from the location table based on the location ID.”*

1. **Query to Update Task Data:**

UPDATE task SET status = True WHERE id = 1;

*“Updates the status of a task in the task table based on the task ID.”*

1. **Query to Delete Volunteer Data:**

DELETE FROM volunteer WHERE id = 1;

*“Deletes a volunteer from the volunteer table based on the volunteer ID.”*

1. **Query to Save Volunteer Assignment Data:**

INSERT INTO volunteerassginment (volunteer\_id, disaster\_id) VALUES (1, 1);

“*Inserts a new volunteer assignment into the volunteerassignment table linking*

*a volunteer to a disaster.”*

1. **Query to Retrieve User Data:**

SELECT \* FROM auth\_user WHERE id = 1;

*“Retrieves all columns of a user from the auth\_user table based on the user ID.”*

1. **Query using Constraints:**

ALTER TABLE Volunteer

ADD CONSTRAINT fk\_user\_id FOREIGN KEY (user\_id)

REFERENCES User(id);

*"To ensure data integrity and enforce relationships between tables, a foreign key constraint is added to the Volunteer table, referencing the User table.”*

1. **Query using Sets:**

SELECT v.\*

FROM Volunteer v

WHERE NOT EXISTS (

SELECT \*

FROM Disaster d

WHERE d.isactive = TRUE

AND NOT EXISTS (

SELECT \*

FROM VolunteerAssignment va

WHERE va.volunteer\_id = v.id

AND va.disaster\_id = d.id

)

);

*“This query identifies volunteers assigned to all active disasters, employing set operations for precise filtering.”*

1. **Query using Joins:**

SELECT v.name AS Volunteer\_Name, d.name AS Disaster\_Name

FROM Volunteer v

INNER JOIN VolunteerAssignment va ON v.id = va.volunteer\_id

INNER JOIN Disaster d ON va.disaster\_id = d.id;

*“Volunteer names along with their assigned disaster names are retrieved through a join operation, enhancing data visibility.”*

1. **Query using Views:**

REATE VIEW Active\_Disasters AS

SELECT \*

FROM Disaster

WHERE isactive = TRUE;

*“A view is created to display all active disasters, providing a convenient and pre-defined dataset for analysis.”*

1. **Query using Triggers:**

CREATE TRIGGER update\_task\_status

AFTER UPDATE ON Task

FOR EACH ROW

BEGIN

IF NEW.status = TRUE THEN

UPDATE Disaster d

SET d.status = TRUE

WHERE d.id = NEW.disaster\_id

AND NOT EXISTS (

SELECT \*

FROM Task t

WHERE t.disaster\_id = d.id

AND t.status = FALSE

);

END IF;

END;

*“A trigger updates the status of a task to 'Completed' when all associated tasks with a disaster are completed, automating task management.”*

1. **Query using Cursors:**

DECLARE disaster\_id INT;

DECLARE volunteer\_count INT;

DECLARE done INT DEFAULT FALSE;

DECLARE cur CURSOR FOR

SELECT id FROM Disaster;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO disaster\_id;

IF done THEN

LEAVE read\_loop;

END IF;

SELECT COUNT(\*) INTO volunteer\_count

FROM VolunteerAssignment

WHERE disaster\_id = disaster\_id;

-- Output or store the results

END LOOP;

CLOSE cur;

*“This cursor calculates the total number of volunteers assigned to each disaster, facilitating in-depth analysis of volunteer distribution.”*

**CHAPTER 4**

**PITFALLS, DEPENDENCIES AND NORMALIZATION**

**4.1 ANALYZING THE PITFALLS**

**Dependency on Technology:**

While technology facilitates efficient coordination, reliance on digital platforms may pose challenges during disasters, especially in areas with poor connectivity or infrastructure damage. Lack of access to the system during critical moments can impede response efforts.

**Data Integrity and Accuracy:**

User-generated incident reports and volunteer registrations may vary in quality and reliability, leading to inaccuracies in resource allocation and response planning. Without robust verification mechanisms, erroneous data could compromise decision-making processes.

**Security Risks:**

Storing sensitive information about volunteers and incident details requires stringent security measures. Inadequate safeguards could expose the system to data breaches, jeopardizing user privacy and potentially hindering trust in the platform.

**Resource Allocation Challenges:**

Despite efforts to streamline resource management, allocating resources effectively during large-scale disasters remains a complex task. Inaccurate or incomplete data, coupled with unpredictable demands, may result in inefficient resource distribution and underutilization.

**User Adoption and Training:**

The success of the system hinges on widespread user adoption and proficiency. Inadequate training or resistance to change among stakeholders could limit the system's effectiveness, particularly in communities with diverse demographics or limited digital literacy.

**Scalability and Resilience:** As the system expands to accommodate growing user bases and evolving needs, scalability becomes critical. Failure to scale infrastructure and operations appropriately could lead to performance bottlenecks and system failures during peak usage or crisis situations.

**Regulatory Compliance:** Compliance with data protection regulations and legal requirements is paramount, especially concerning sensitive information. Failure to adhere to regulatory standards may result in legal liabilities and damage the system's reputation.

**4.2 IDENTIFYING THE DEPENDENCIES**

**User Entity:**

Dependency: Functional Dependency

Dependent Attributes: Name, Email, Role

The User\_ID uniquely determines the values of Name, Email, and Role attributes in the User table. Knowing the User\_ID allows retrieving the corresponding Name, Email, and Role values.

**Volunteer Entity:**

Dependency: Functional Dependency

Dependent Attributes: Name, Date of Birth, Contact Information

The Volunteer\_ID uniquely determines the values of Name, Date of Birth, and Contact Information attributes in the Volunteer table. Knowing the Volunteer\_ID allows retrieving the corresponding details of a volunteer.

**Disaster Entity:**

Dependency: Functional Dependency

Dependent Attributes: Name, Location, Report Time

The Disaster\_ID uniquely determines the values of Name, Location, and Report Time attributes in the Disaster table. Knowing the Disaster\_ID allows retrieving the corresponding details of a disaster.

**DisasterList** **Entity:**

Dependency: Functional Dependency

Dependent Attribute: Disaster Type

The DisasterList\_ID uniquely determines the values of the Disaster Type attribute in the DisasterList table. Knowing the DisasterList\_ID allows retrieving the corresponding type of disaster.

**VolunteerAssignment Entity:**

Dependency: Composite Dependency

Dependent Attribute: VolunteerAssignment\_ID

The combination of Volunteer\_ID and Disaster\_ID together uniquely determines the VolunteerAssignment\_ID attribute in the VolunteerAssignment table. This means that a volunteer's assignment to a particular disaster is identified by both the Volunteer\_ID and Disaster\_ID together.

**4.3 APPLYING NORMALIZATION**

**Tables in All Normal Forms:**

USER: This table adheres to all normalization forms (1NF, 2NF, 3NF, BCNF, 4NF, 5NF). It has a well-defined primary key (User\_ID), and all attributes directly depend on it, eliminating redundancy and ensuring data integrity.

RESOURCE: Similar to the USER table, RESOURCE complies with all normal forms. It has a primary key (Resource\_ID), and all attributes directly depend on it, minimizing data duplication and maintaining data consistency.

INCIDENT: This table satisfies all normalization forms. It has a primary key (Incident\_ID), and all attributes directly depend on it, ensuring efficient data retrieval and management.

ALERT: The ALERT table adheres to all normal forms. It has a primary key (Alert\_ID) with a unique constraint on Issued\_Time, and all attributes directly depend on it, promoting data accuracy and reducing redundancy.

COMMUNICATION: This table fulfills all normalization forms. It has a primary key (Message\_ID), foreign keys for Sender\_ID and Receiver\_ID, and all attributes directly depend on the primary key, guaranteeing data integrity and efficient communication tracking.

TASK: The TASK table adheres to all normalization forms. It has a primary key (Task\_ID), a foreign key for Volunteer\_ID, and all attributes directly depend on the primary key, ensuring clear task assignment and data integrity.

MAP: This table satisfies all normal forms. It has a primary key (Map\_ID), a foreign key for Incident\_ID, and all attributes directly depend on the primary key, promoting efficient incident mapping and data management.

GIS FEATURES: Similar to the MAP table, GIS FEATURES adheres to all normalization forms. It has a primary key (GIS\_Feature\_ID), a foreign key for Map\_ID, and all attributes directly depend on the primary key, ensuring accurate geographic representation and data consistency.

INCIDENT CATEGORY: This table fulfills all normal forms. It has a primary key (Category\_ID), and all attributes directly depend on it, simplifying incident categorization and data retrieval.

INCIDENT HISTORY: The INCIDENT HISTORY table adheres to all normal forms. It has a primary key (History\_ID), a foreign key for Incident\_ID, and all attributes directly depend on the primary key, enabling efficient incident tracking and historical data management.

EMERGENCY PLAN: This table satisfies all normal forms. It has a primary key (Plan\_ID) with a unique constraint on Last\_Update, and all attributes directly depend on the primary key, ensuring clear and up-to-date emergency plans.

**Tables Requiring Further Analysis:**

VOLUNTEER: This table might violate 3NF (and higher forms) if user\_id references detailed skills or availability information in the USER table. This creates a transitive dependency. To address this, consider creating a separate VOLUNTEER\_DETAILS table with volunteer\_id (foreign key referencing VOLUNTEER) and attributes like skills, availability.

ORGANIZATION: The ORGANIZATION table might violate 2NF and higher forms if Contact\_Info stores multiple contact methods (e.g., phone number, email). To address this redundancy, consider a separate table like ORGANIZATION\_CONTACTS with organization\_id (foreign key referencing ORGANIZATION) and contact\_type (e.g., phone, email), contact\_details.

**CHAPTER 5**

**CONCURRENCY CONTROL**

**5.1 IMPLEMENTATION**

The disaster management system prioritizes data consistency during concurrent access, particularly for contributor information and resource allocation. To address this challenge, optimistic locking is employed. Unlike pessimistic locking, which restricts access through constant locks, optimistic locking allows for quicker updates by multiple responders. However, it ensures data integrity through versioning.When a responder retrieves data for modification (e.g., updating resource availability), the system captures the current version. Later, during commit, the system checks the version against the one retrieved earlier. If versions differ, signifying another responder modified the data, the commit fails. This prevents overwrites and ensures data consistency.

Optimistic locking is well-suited for this system as it balances efficiency with data integrity. Responders can make quicker updates without impacting performance, crucial during time-sensitive situations. However, robust error handling is essential to inform responders about conflicts and guide them on how to proceed, ensuring a smooth workflow.

**Efficiency:**

This approach allows for quicker updates to contributor information and resource availability without significantly impacting performance, crucial during time-sensitive disaster response situations.

**Data Integrity:**

By preventing overwrites, optimistic locking ensures data consistency, safeguarding against accidental or malicious modifications.

**Error Handling:**

Robust error handling mechanisms are essential to inform users about conflicts and guide them on how to proceed (e.g., refreshing data, retrying updates). This ensures a smooth workflow for disaster responders.

**Data Sensitivity:**

While optimistic locking promotes efficiency, it's crucial to evaluate the sensitivity of the data being accessed. For highly critical data, additional control mechanisms might be considered.

**5.2 RECOVERY MECHANISMS**

The disaster management system prioritizes data availability and integrity, especially during critical response situations. To ensure data remains accessible and consistent even in the face of unforeseen system failures, recovery mechanisms are implemented. Here's an overview of how these mechanisms achieve this objective:

**Transaction Logging: The Backbone of Recovery**

The system maintains a transaction log, essentially a detailed record of all database modifications. This log acts as the backbone for recovery efforts. It captures crucial information like the transaction ID, the type of operation performed (insert, update, delete), and the specific data affected by the operation.

**Rollback: Undoing Uncommitted Changes**

In the event of a system crash or error, the transaction log empowers the system to perform a rollback. This process essentially undoes all uncommitted transactions. By rolling back these transactions, the system ensures that only successful and verified modifications are reflected in the database, preventing inconsistencies and data corruption. This rollback functionality ensures a consistent database state even after unexpected failures.

**Enhancing Recovery with Redo Logging (Optional)**

For an even more comprehensive recovery strategy, the system can optionally implement redo logging. This log stores information about committed transactions. In case of a system crash,

the redo log can be used to replay and re-apply these committed transactions after the system is restored. This minimizes data loss by ensuring all verified modifications are reflected in the database.

**Regular Backups: A Safety Net for Critical Data**

Regularly backing up the database to a separate location is an essential element of the recovery strategy. This backup serves as a safety net in case of major failures that might render the transaction logs unusable. If a critical failure occurs, the system can be restored from the latest backup, minimizing data loss and ensuring a quicker recovery time.

**Choosing the Right Recovery Approach:**

The specific recovery mechanisms chosen will depend on several factors, including:

1. Frequency of Anticipated Failures: The likelihood of system crashes or errors will influence the chosen approach. More frequent occurrences might necessitate more robust mechanisms.
2. Data Sensitivity: The importance of data integrity will guide the selection of mechanisms. Highly sensitive data might require additional safeguards.
3. Recovery Time Objective (RTO): The acceptable downtime after a failure should be considered when choosing mechanisms. A faster RTO might necessitate prioritizing faster recovery techniques.

**CHAPTER 6**

**CODE**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A screen shot of a computer code

Description automatically generated**

**A screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generated**

**A screenshot of a computer code

Description automatically generated**

**A computer screen shot of a computer code

Description automatically generated**

**CHAPTER 7**

**CONCLUSION**

**7.1 RESULT**

**A screenshot of a login form

Description automatically generated**

**Fig.7.1** Login page

The login page, where the administrator can login through the admin credentials and the volunteer can login through their credentials. If a contributor wants to contribute something for the disaster recovery, they have their own login. User authentication ensures the safety of the user’s details.

**A screenshot of a computer

Description automatically generated**

**Fig.7.2** Admin Dashboard

**A screenshot of a computer

Description automatically generated**

**Fig.7.3** Volunteers

**A screenshot of a computer

Description automatically generated**

**Fig.7.4 New Disaster**

The admin can view the list of disasters that are currently active and the location in which they are occurring. They can also view the list of volunteers who have signed up along with their personal information. The admin can add new disasters to the list along with the type of disaster and the location it’s affecting.

A screenshot of a computer

Description automatically generated

**Fig.7.5** Tasks

A screenshot of a computer

Description automatically generated

**Fig.7.6** New Task

The admin can view the list of tasks that have to be done for the recovery. And they can create new tasks if need be.

A screenshot of a computer

Description automatically generated

**Fig.7.7** Location

A screenshot of a computer

Description automatically generated

**Fig.7.8** New Location

A map of india with a blue spot

Description automatically generated

**Fig.7.9** Map

The admin can view the list of places where disasters could possibly happen. And they can create new locations if they predict or know a disaster could happen. A map is generated that shows the severity of each disaster particular to the location and can be differentiated with the difference in the color ‘blue’ here.

A screenshot of a computer

Description automatically generated

**Fig.7.10** Contributors

A screenshot of a computer

Description automatically generated

**Fig.7.11** Resources

A screenshot of a computer

Description automatically generated

**Fig.7.12** Assign Resources

A screenshot of a computer

Description automatically generated

**Fig.7.13** Assigned Resources

The admin can view the list of the contributors and the list of resources that they have contributed. They can view the assigned resources and assign the resources according to the disaster, the location, the need and the number of volunteers available.

A screenshot of a computer

Description automatically generated

**Fig.7.14** Dashboard from Volunteers’

The volunteer can view the disasters and the tasks required to be done for the recovery. They will be able to join whichever they can help with.

A white rectangular object with black lines

Description automatically generated

**Fig.7.15** Contributor Registration

A screenshot of a computer

Description automatically generated

**Fig.7.16** Unique id for Contributor

A screenshot of a computer

Description automatically generated

**Fig.7.17** Resources Contribution

The contributor can register through this portal. They can register using their personal details and ID proof. They can donate the resources they want to in the first place, using the unique id provided by the system itself when registering. When donating the resources they can mention the quantity.

**7.2 DISCUSSION**

In conclusion, the "Disaster Management System" project represents a significant step forward in enhancing local communities' resilience and preparedness in the face of escalating climate-related events and unforeseen disasters. By leveraging advanced technologies and innovative approaches, this web-based platform serves as a vital tool for orchestrating swift and effective responses during emergencies. Through its integrated components, including the Resource Inventory, Incident Reporting and Alerts system, Communication Hub, Mapping and Visualization capabilities, and Volunteer Management tools, the platform empowers communities with the means to mitigate the impacts of disasters and safeguard lives and property.

The project's emphasis on proactive communication, resource allocation, and incident reporting underscores its commitment to facilitating collaboration and coordination among stakeholders. By providing a centralized hub for real-time interaction and information exchange, the platform fosters a culture of resilience and community engagement, enabling individuals and organizations to work together towards a common goal of disaster preparedness and response.

Furthermore, the project's focus on inclusivity and accessibility ensures that community members of all backgrounds and abilities can actively participate in disaster response operations. Whether through volunteering, reporting incidents, or accessing critical information, the platform offers avenues for meaningful engagement and contribution, thereby strengthening social cohesion and solidarity within communities.

Looking ahead, the success of the "Disaster Management System" project hinges on ongoing collaboration, innovation, and adaptation to evolving challenges. Continued efforts to enhance the platform's functionalities, expand its reach, and incorporate feedback from stakeholders will be crucial in ensuring its effectiveness and relevance in an ever-changing landscape of disasters and emergencies.

Ultimately, by investing in robust disaster management systems like this, communities can build resilience, save lives, and minimize the devastating impacts of disasters, thereby creating safer and more sustainable environments for all.

**CHAPTER 8**

**ONLINE COURSE CERTIFICATION**

**A close-up of a certificate

Description automatically generated**

**GITHUB LINK**

https://github.com/RASHMIKA4506/Disaster-Management-Final