Relief Tracker

Mapping Help to Reach Those in Need After Natural Disasters

Architecture Design Document

CSE350

—

Supervisor: Mohammad Abdullah Al Mumin, PhD, Professor

Team members details

Registration No	Section	Name	Role
2019331027	A	Alomgir	Lead Software developer, Architect
2019331062	B	Ariful Islam Farhad	Project manager, Scribe
2019331063	A	Rubayet Sadman Sami	Lead Architect, software developer
2019331091	A	Md.Saidur Rahman	Quality assurance, Software developer
2019331097	A	Oli Hossain	Lead Scribe, Architect

Github Link: **https://github.com/Sami-63/Relief-Tracker**

Table of Contents

1. Introduction

1.1 Motivation

Every year, our country faces several natural disasters. People in the affected areas lose their belongings, homes and even closed ones. During or after the disaster period, different governmental or non-governmental organizations and individuals across the country come forward with donations and relief materials for the affected people to help them recover and rebuild.

But due to proper management, or let's say, lack of information, some areas in the affected region get donations and relief aids multiple times, but some localities may not get any relief materials even for a single time. This is a very common but serious issue in our country.

1.2 Aims and objectives of the project (Che, 2014).

We are making an android app which organizations and individuals can use to track the areas which have been affected due to any disaster, and then can see whether any area or localities have got any relief or donations.

- If one area has recently received any donations or relief materials, donation providers can skip this locality or area, and go to the areas which haven't got any.
- People in the affected areas can ask for donations through our app which will be shown on our map after verification. The more severe the condition, the higher priority they get.
- Donations providers can see if any area has received a sufficient amount of donations, if not, they can frequently visit the areas.
- There will be different events and volunteers can attend those events to work for disaster recovery and smoothening relief distribution processes.

1.3 Limitations of existing system

Before the development of Relief Tracker, existing systems faced several limitations that hindered effective coordination of relief efforts. Manual processes, limited visibility into real-time relief operations, and fragmented relief data posed challenges in prioritizing aid delivery and decision-making. Additionally, inadequate verification mechanisms, poor communication channels, and accessibility issues in remote areas further compounded these challenges. Security concerns, such as data breaches, also raised risks to the integrity of relief data. Relief Tracker was conceived to address these limitations by providing a comprehensive, efficient, and transparent platform for coordinating relief activities, enhancing communication among stakeholders, and ensuring the secure and effective distribution of aid in disaster-affected areas.

1.4 Purpose of the document

The purpose of this document is to document the architectural design phase for the Relief Tracker system. It outlines the logical and physical structure of the system, including strategic decisions made during development. The document serves as a guide for stakeholders and development teams, providing a detailed understanding of the system's architecture.

The Software Architecture Document (SAD) provides a comprehensive architectural overview of design and architecture of Relief_Tracker. It presents several different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system (Forward et al., 2002).

In order to depict the software as accurately as possible, the structure of this document is based on the "4+1" model view of architecture (Bachmann et al., 2000).

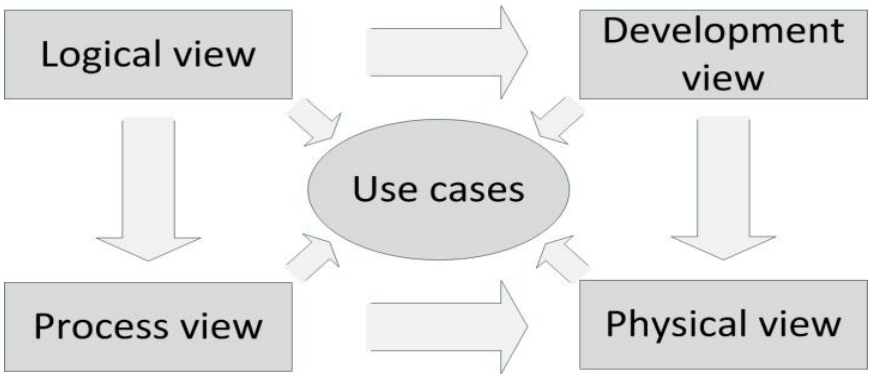


Figure: "4+1" model view of architecture (Barcelos et al., 2006).

The "4+1" View Model allows various stakeholders to find what they need in the software architecture.

1.5 **Scope of the system (Ernst et al., 2023).**

The objective of this project is to construct a versatile framework tailored for the Relief Tracker system. Our aim is to encompass a broad spectrum of functionalities crucial for efficient relief coordination. This includes features such as relief request submission, verification processes, real-time visualization of relief distribution, and user communication. By developing a comprehensive framework, we strive to address various aspects of relief operations, ranging from data submission to distribution mapping. This framework should accommodate different user environments and operational stages, whether it's for administrators, local moderators, or general users. The framework's flexibility is paramount, enabling easy integration with existing systems and adaptability to diverse scenarios. Thus, it's imperative to design an architecture with ample customizable elements, ensuring seamless alignment with specific relief challenges and interoperability with other frameworks.

1.6 Definitions, acronyms, and abbreviations

1.6.1. Definitions

Architecture: The software architecture of a program or computing system is the structure or structures of the system, which comprises software components, the externally visible properties of those components, and the relationships among them (Alexeeva et al., 2016).

Framework: A framework represents a collection of classes that provide a set of services for a particular domain; a framework exports several individual classes and mechanisms that clients can use or adapt. A framework realizes architecture (Che, 2014).

Component: A software component is a re-usable piece of software that has a well specified public interface, and it implements a limited functionality. Software components achieve reuse by following standard conventions (Alexeeva et al., 2016).

1.6.2. Acronyms (Zhu, 2005).

URD: User Requirements Document

ADD: Architecture Design Document

UML: Unified Modeling Language

ODBMS: Object-oriented Database Management System

APIs: Application Programming Interfaces

GIS: Geographic Information Systems

MySQL: relational database management system (RDBMS)

SAD: Software Architecture Document

RUP: Rational Unified Process

2. Design and Architecture

2.1 Overview

First step of building a software is to design it on the basis of a design pattern. Here we use the MVC design pattern to design our software. Model View Controller is the most commonly used design pattern. Developers find it easy to implement this design pattern.

Following is a basic architecture of the Model View Controller –

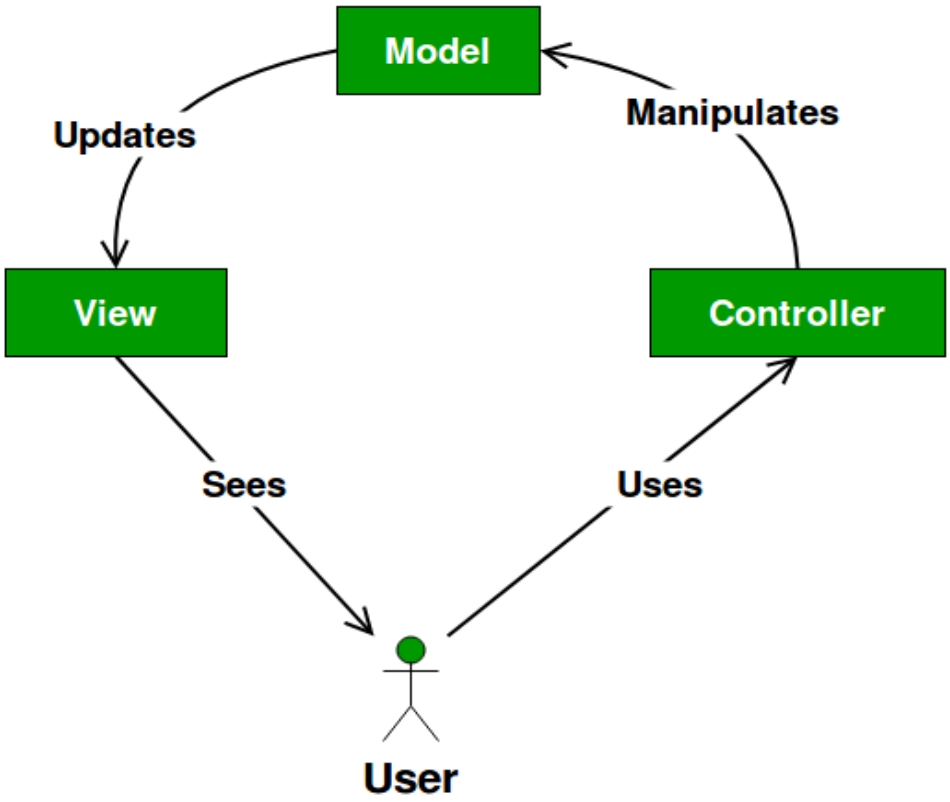


Figure: MVC Pattern

Let us now see how the structure works.

Model: It consists of pure application logic, which interacts with the database. It includes all the information to represent data to the end user.

View: View represents the HTML files, which interact with the end user. It represents the model's data to user.

Controller: It acts as an intermediary between view and model. It listens to the events triggered by view and queries model for the same.

2.2 Module Description

We have separated the software into different modules like admin, moderator, general user, these all modules have different tasks in the system.

“Admin” module will handle all the process of user management, validation, creating disaster information, verifying donation records etc. This module will interact with the database.

“General user” module is used to add individual donation records by donors, viewing recent disasters, viewing donation list from an area by searching the location or selecting from map.

“Moderator” module verifies the donation records made by general users, and requesting disaster information to the admin. This module will interact with the database.

2.3 System Architecture

The Relief Tracker application is built according to the pillars of the 3 tire architecture. Users can interact with the front-end system via any android device and it is called a separate API server connect to the database. Backend database is used to store images. Front end server can upload data and media directly to database storage. To request information and data, front end server has to invoke the API function and the API will deliver the requested resources.

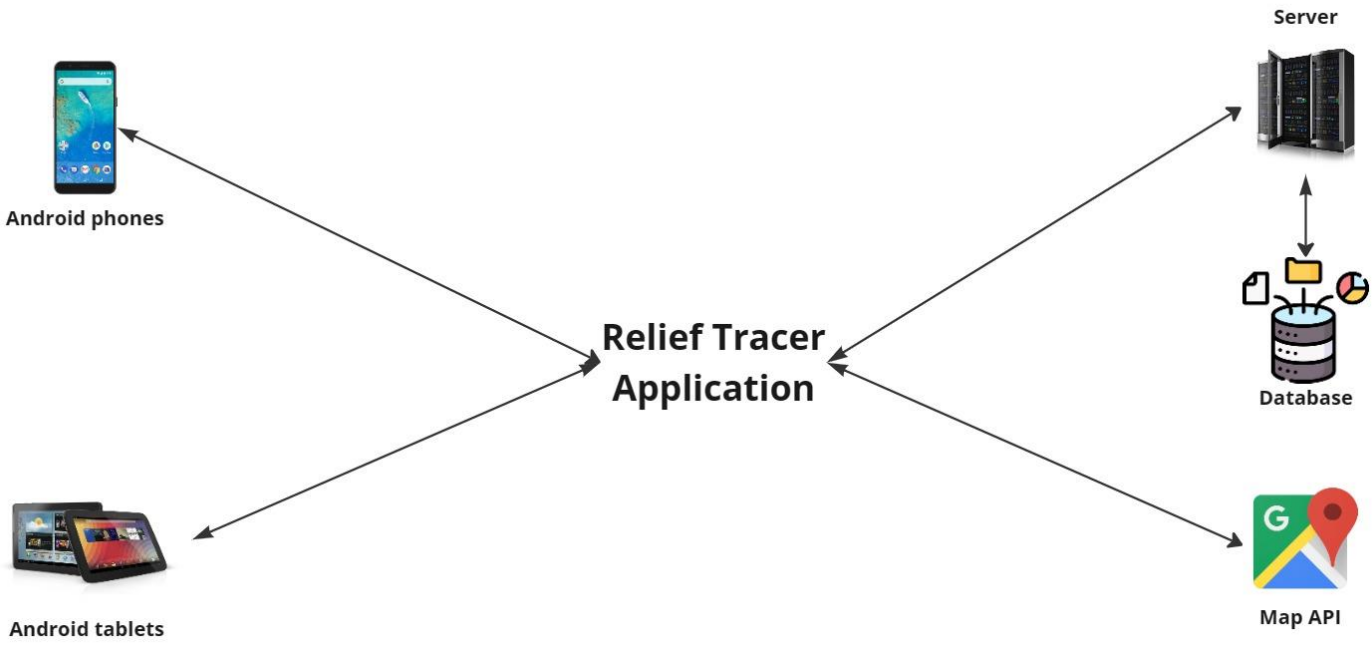


Figure: High-Level Architecture diagram

2.4 Architectural pattern

In the Relief Tracker project, we adopt a four-layered architectural pattern to provide a structured and modular design. Here's a breakdown of each layer:

Presentation Layer: This layer focuses on user interface components, map integration and managing how users interact with the system mobile interfaces.

Configuration Layer: Responsible for managing user interfaces, management of functionalities, authentication and authorization. It centralizes parameters like database connections, API endpoints, and security settings, facilitating easy deployment and maintenance.

Application Layer: At the heart of the system, the application layer houses the core application logic and rules. It orchestrates data flow and processes, handling tasks such as relief request processing, donation verification, record donations etc. This layer encapsulates the system's unique functionalities and defines how they interact with each other.

System Support Layer: Serving as the foundation for the entire system, the system support layer provides essential services and utilities. It includes components for logging, error handling, caching, API handling, and performance monitoring, addressing cross-cutting concerns that span multiple layers.

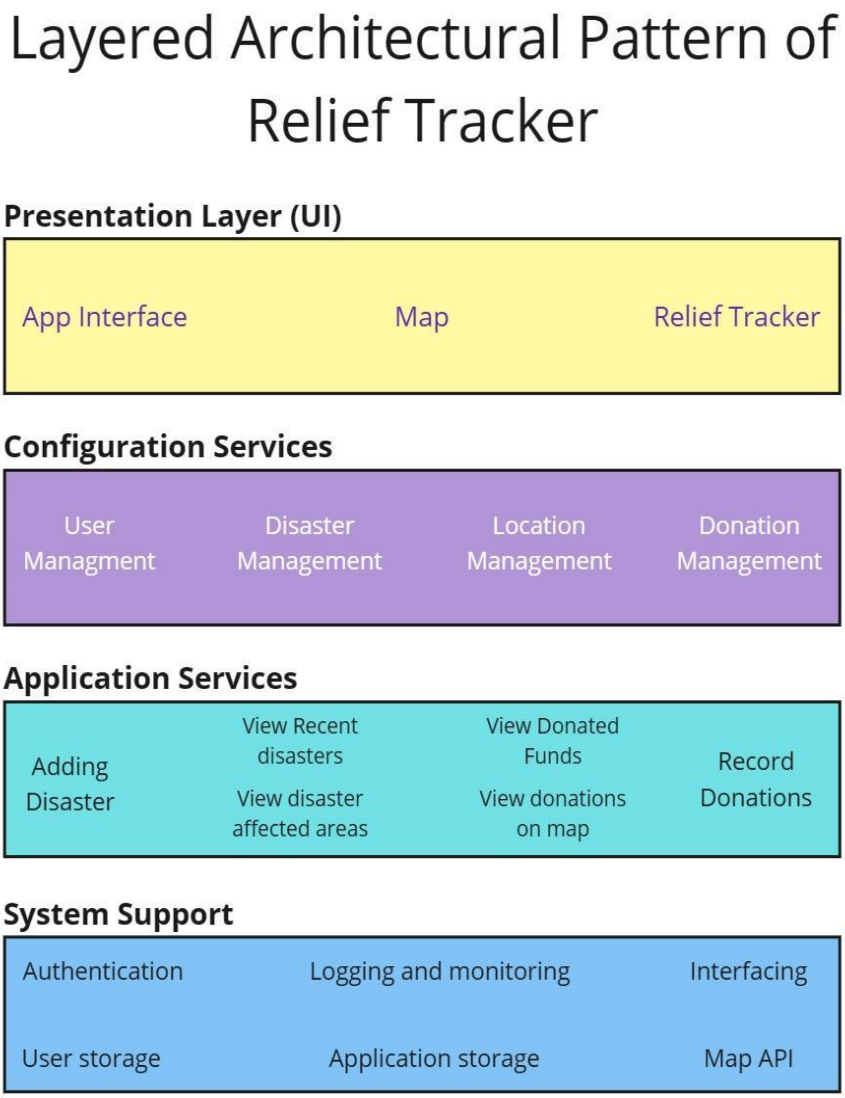


Figure: Architectural pattern of Relief Tracker

2.5 Architectural Views

This document details the architecture using the views defined in the “4+1” model but using the RUP naming convention. The views used to document the Relief Tracker tool application are:

Use Case view

Audience: all the stakeholders of the system, including the end-users.

Area: describes the set of scenarios and/or use cases that represent some significant, central functionality of the system. Describes the actors and use cases for the system, this view presents the needs of the user and is elaborated further at the design level to describe discrete flows and constraints in more detail. This domain vocabulary is independent of any processing model or representational syntax (i.e. XML).

Related Artifacts: Use-Case Model, Use-Case documents

Logical view

Audience: Designers.

Area: Functional Requirements: describes the design's object model. Also describes the most important use-case realizations and business requirements of the system.

Related Artifacts: Design model

Process view

Audience: Integrators.

Area: Non-functional requirements: describes the design's concurrency and synchronization aspects.

Related Artifacts: (no specific artifact).

Data view

Audience: Data specialists, Database administrators

Area: Persistence: describes the architecturally significant persistent elements in the data model

Related Artifacts: Data model.

Deployment view

Audience: Deployment managers.

Area: Topology: describes the mapping of the software onto the hardware and shows the system's distributed aspects. Describes potential deployment structures, by including known and anticipated deployment scenarios in the architecture we allow the implementers to make certain assumptions on network performance, system interaction and so forth.

Related Artifacts: Deployment model.

2.5.1. Logical View Diagram

Relief Tracker is divided into layers based on the N-tier architecture (Rost et al., 2013).

Client Layer: This is where users interact with the system through interfaces like web pages and mobile apps.

Business Layer: It contains the core logic and rules of the system, handling tasks like processing relief requests and managing users.

Data Layer: This layer manages data storage and retrieval, ensuring the integrity and security of the system's data assets.

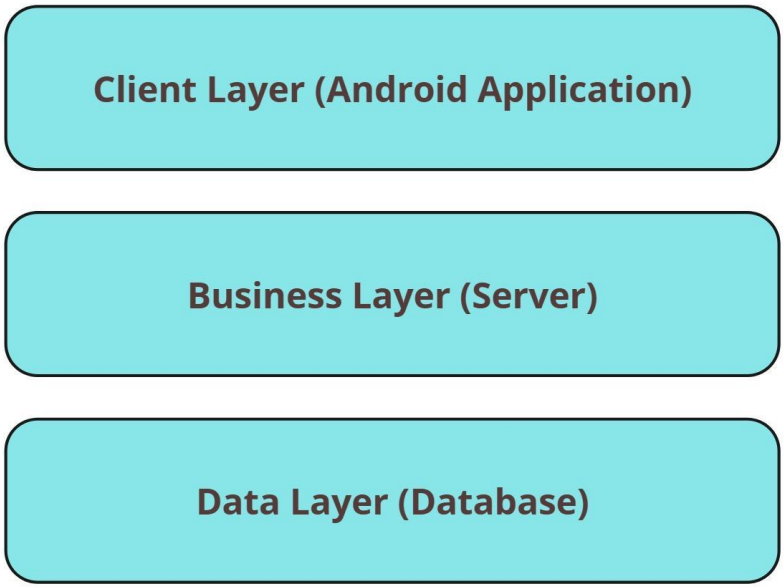


Figure: Logical view of Relief Tracker

The layering model of the Relief Tracker application is based on a responsibility layering strategy that associates each layer with a particular responsibility. This strategy has been chosen because it isolates various system responsibilities from one another, so that it improves both system development and maintenance.

2.5.2. Process View Diagram

The process view of the Relief Tracker system outlines how various processes interact and collaborate to achieve system functionality. It illustrates the flow of tasks and activities within the system, including user interactions, data processing, and communication between different components. This view provides insights into how the system operates at runtime, facilitating efficient coordination and execution of tasks to fulfill user requirements and system objectives.

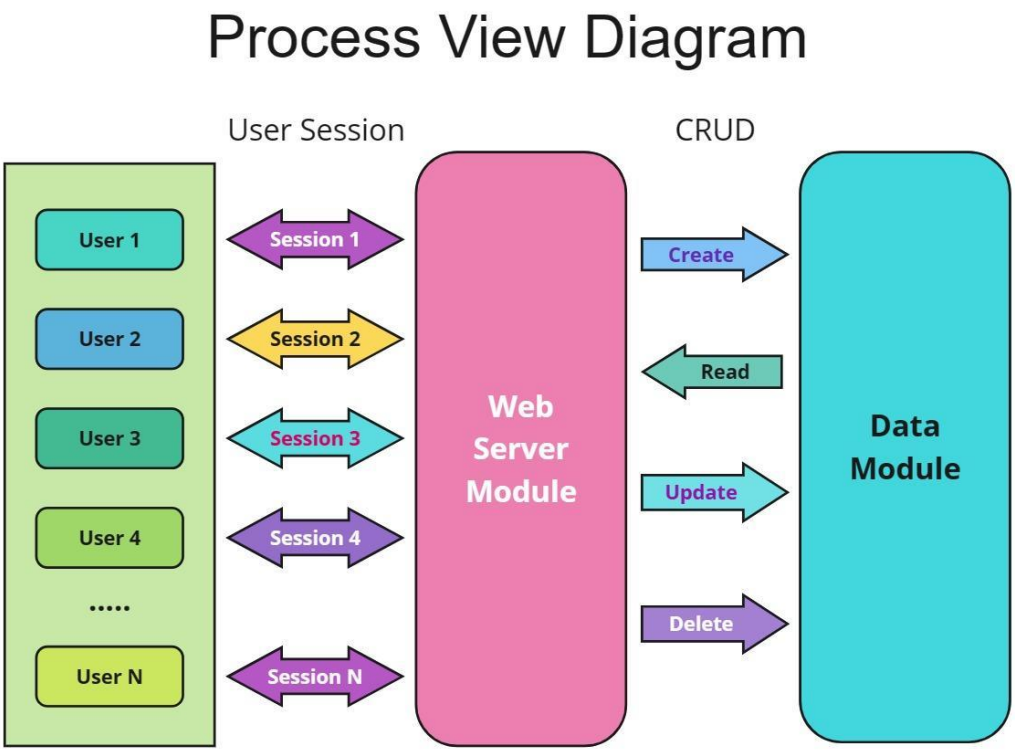


Figure: Process View diagram

2.5.3. Data view (Database Schema)

The data view of the Relief Tracker architecture delineates the structure and organization of data within the system. It encompasses the various data entities, their attributes, relationships, and the overall data flow. This view provides a comprehensive understanding of how data is stored, accessed, and manipulated across different layers and modules of the system. Additionally, it highlights data integrity measures, data storage mechanisms, and data retrieval methods, ensuring efficient data management and utilization throughout the system.

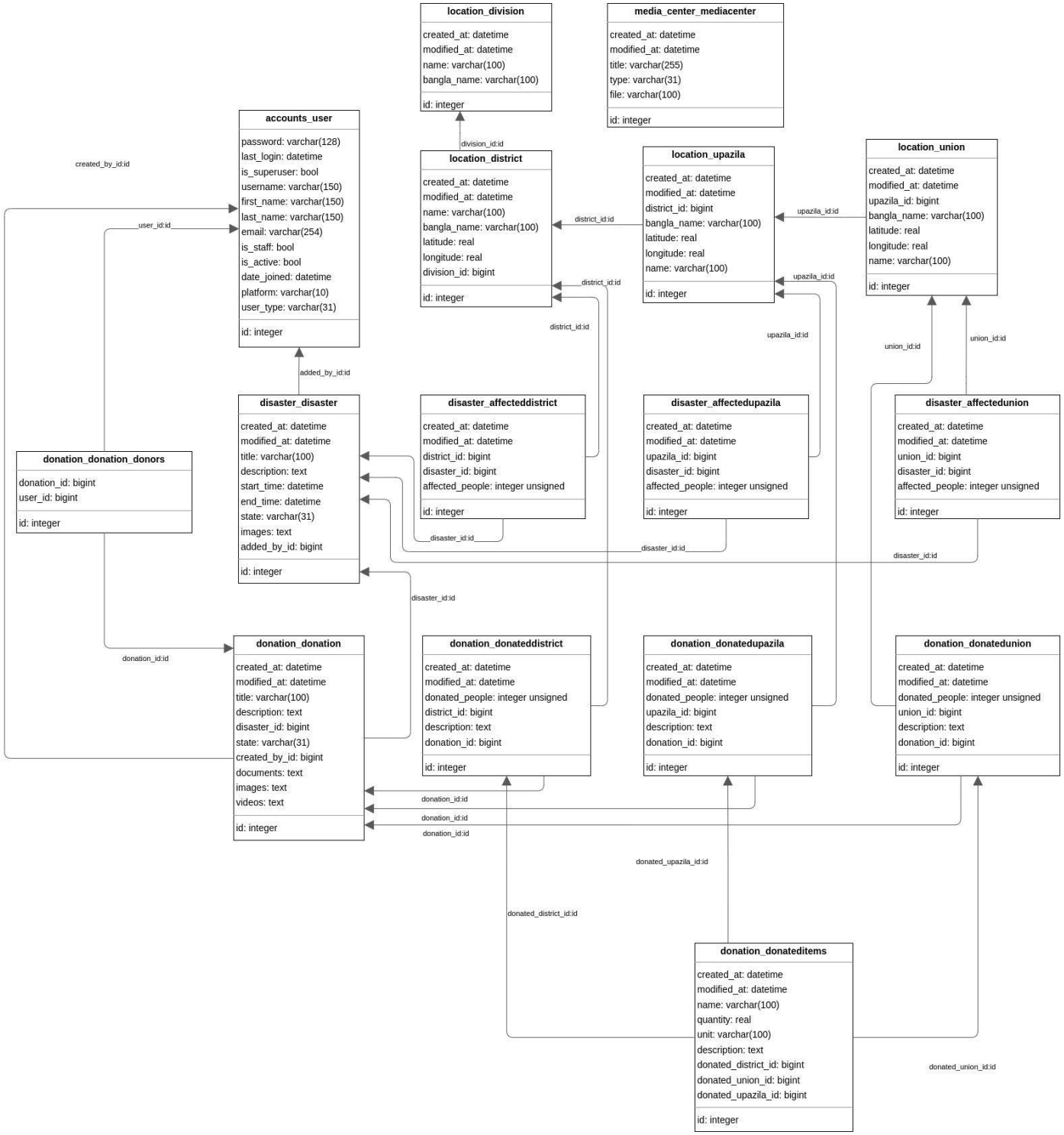


Figure: Data View of Relief Tracker

2.5.4. Use case diagram

Use case diagram is a methodology that used as system development. It is used to clarify, identify and organize requirements of a system. Which user performs which role, is the main concern of this diagram. Actor is used to define who is using the system and the use case that means what task they can do. Actor and use case are linked with direct association.

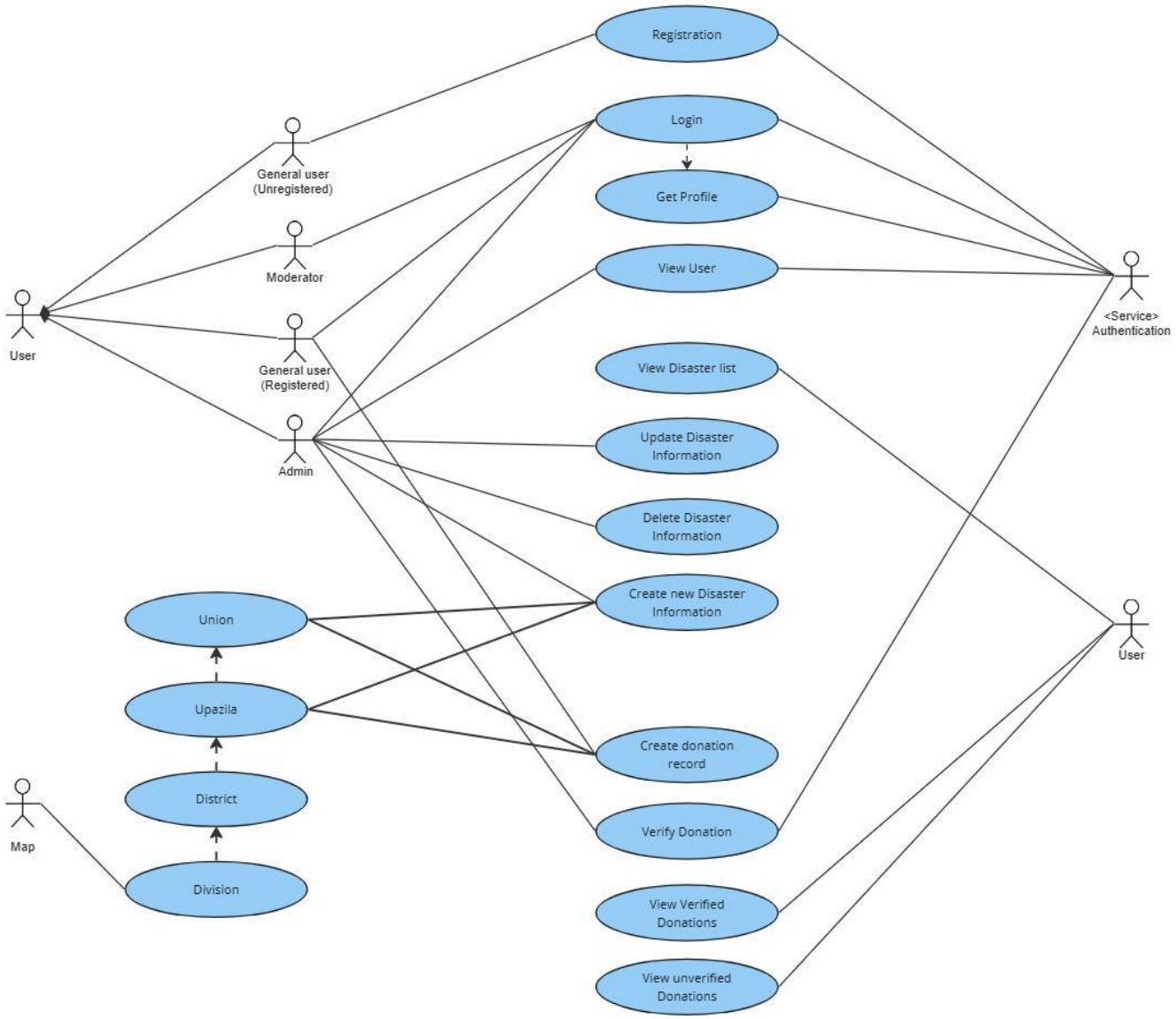
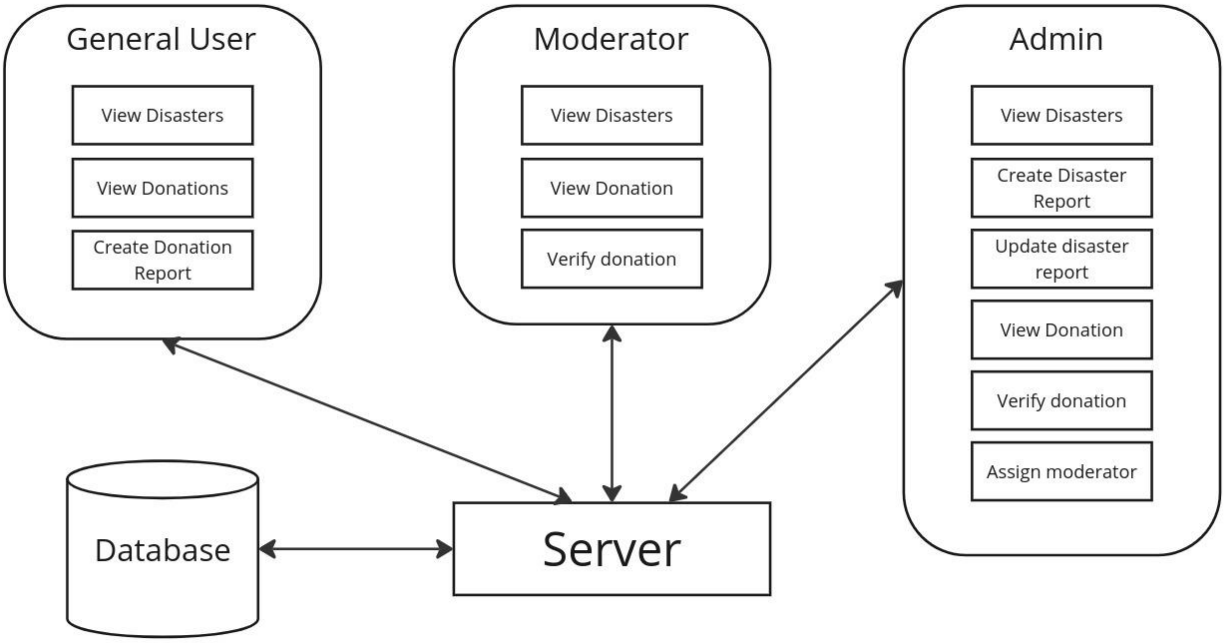


Figure: Use Case Diagram of Relief Tracker

2.5.5. Use Case Realization

The Relief Tracker system's use case realization encompasses several key functionalities crucial for effective relief management and coordination. Users can securely authenticate and register accounts, manage profiles, and create detailed relief requests specifying location, needs, and urgency. Admins validate and approve/reject relief requests based on authenticity and urgency criteria. The system visualizes relief distribution centers and affected areas on a map, highlighting regions requiring assistance for donor awareness. Users can communicate regarding relief efforts, and local moderators assess relief needs, track distributions, and validate requests within their jurisdiction. This comprehensive approach ensures efficient relief distribution and coordination, facilitating timely assistance to disaster-affected communities.



2.6 Workflow diagram

A workflow diagram in the Relief Tracker project offers a visual representation of the relief management process, depicting each step from initiation to completion. Utilizing standardized symbols and shapes, the workflow illustrates the sequence of actions involved in managing relief efforts, including donation submission, verification, distribution mapping, and user communication. By clearly outlining the responsibilities of each stakeholder at various stages of the process, the workflow helps ensure efficient coordination and understanding of roles, fostering unity and cohesion among team members.

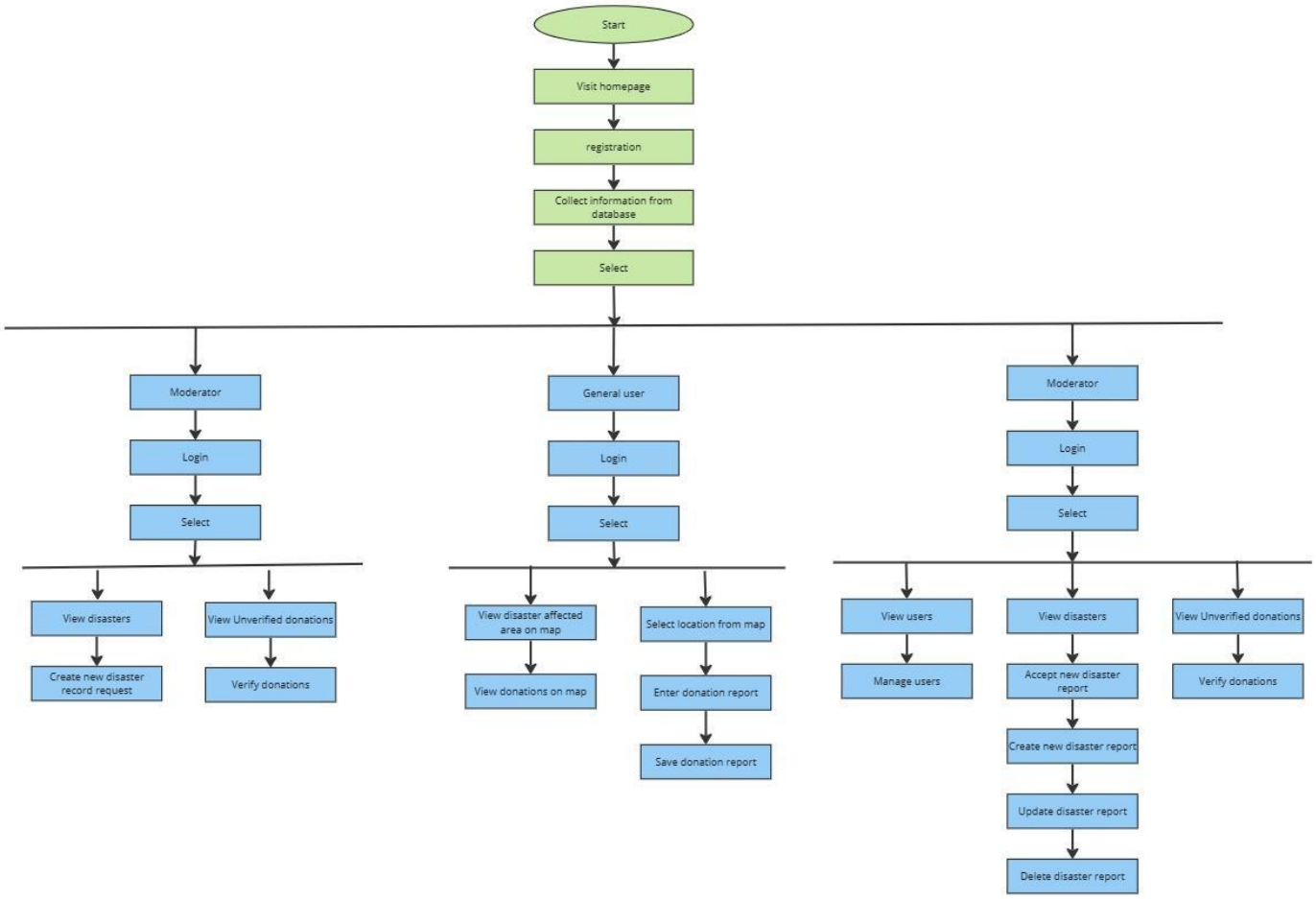


Figure: Workflow diagram of Relief Tracker

2.7 Flowcharts

In the context of the Relief Tracker project, a flowchart serves as a visual depiction of the relief management process, outlining the sequential steps involved in various tasks such as donation submission, verification, distribution mapping, and user communication. Each step is represented by a box, with different types of boxes indicating different actions or decisions. Arrows connect these boxes to show the flow of activities, guiding users through the step-by-step approach to managing relief efforts effectively.

2.7.1. Flowchart for admin

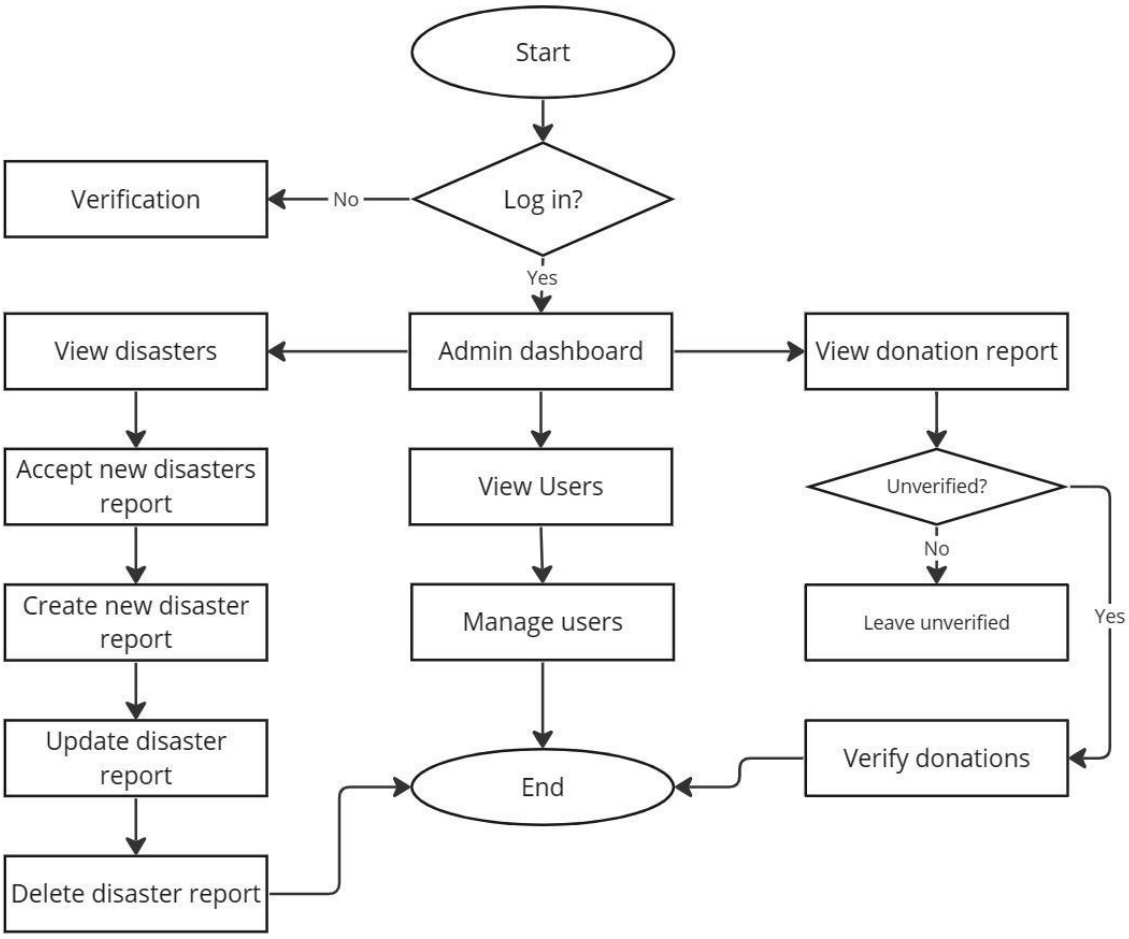


Figure: Flowchart for admin

2.7.2. Flowchart for users

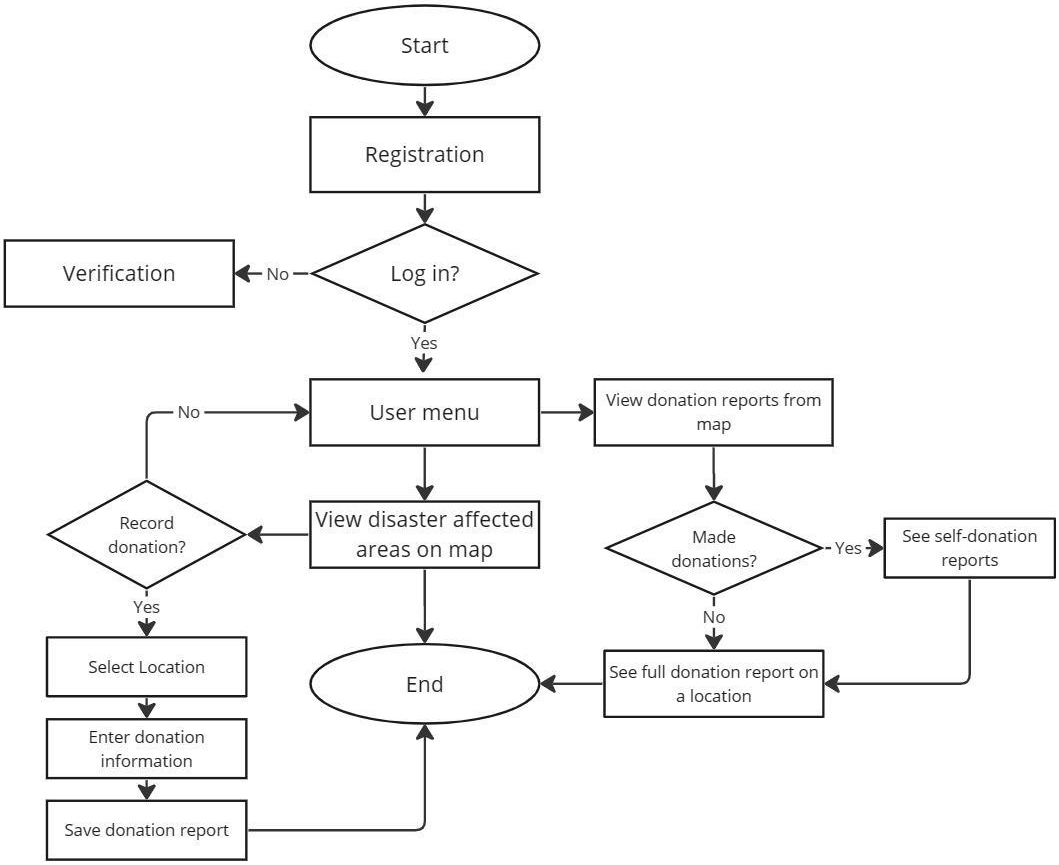


Figure: Flowchart for users

2.7.3. Flowchart for donation creation

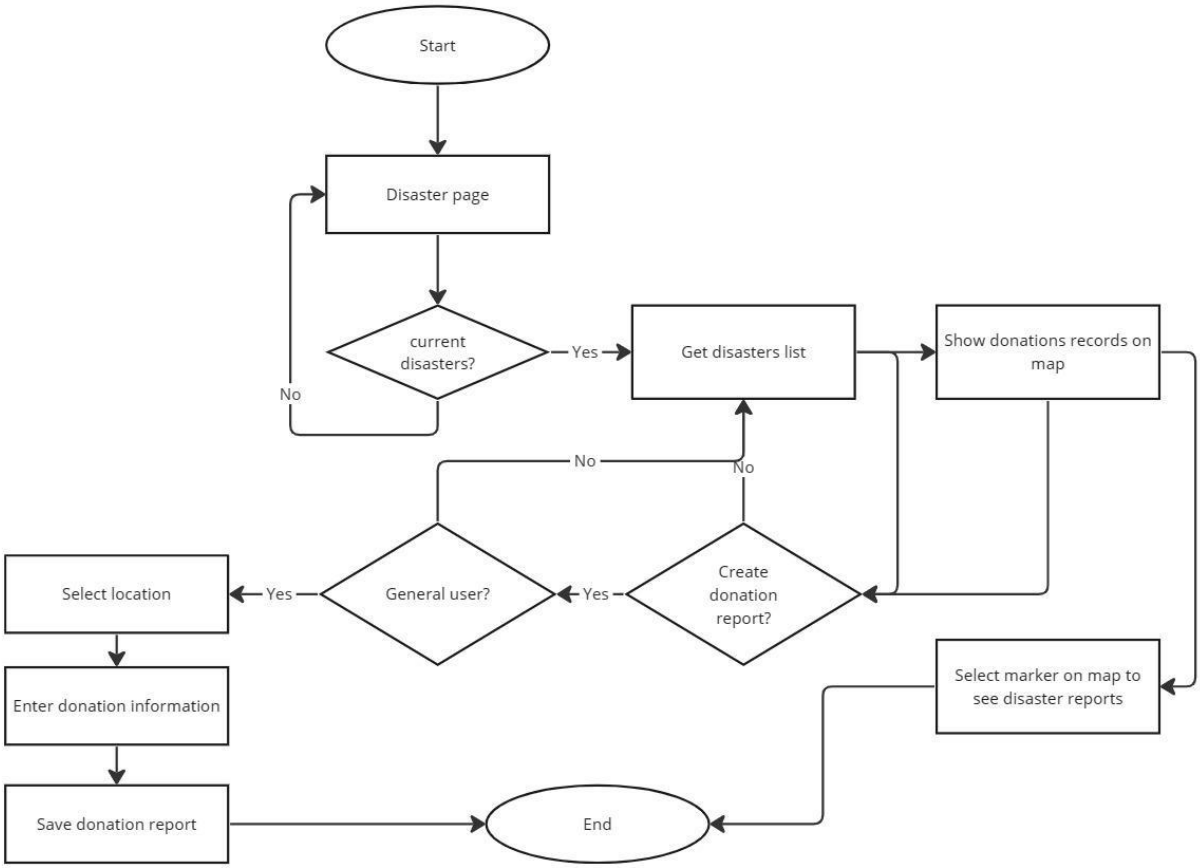


Figure: Flowchart for donation creation

2.7.4. Adding new disaster information flowchart

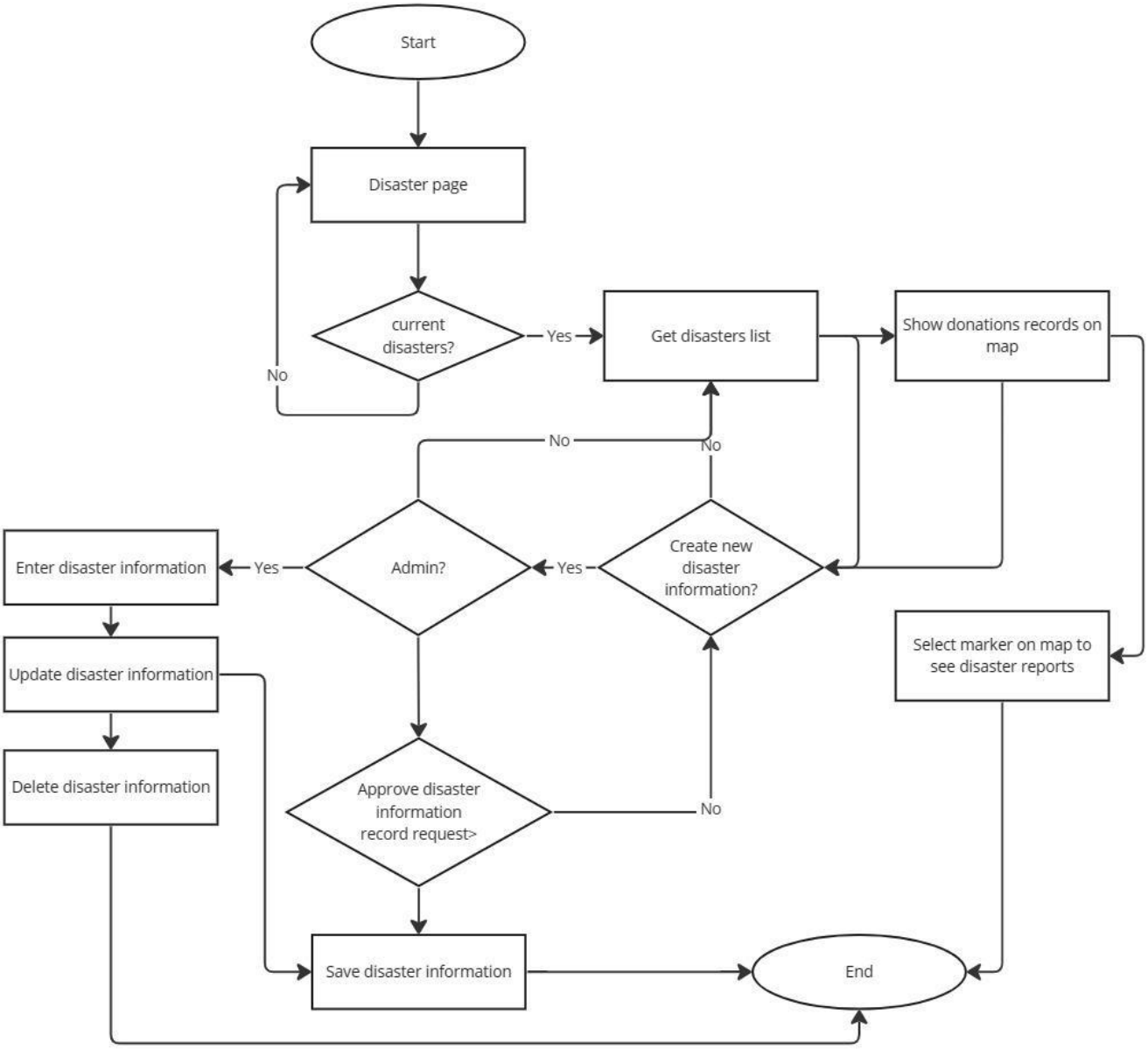
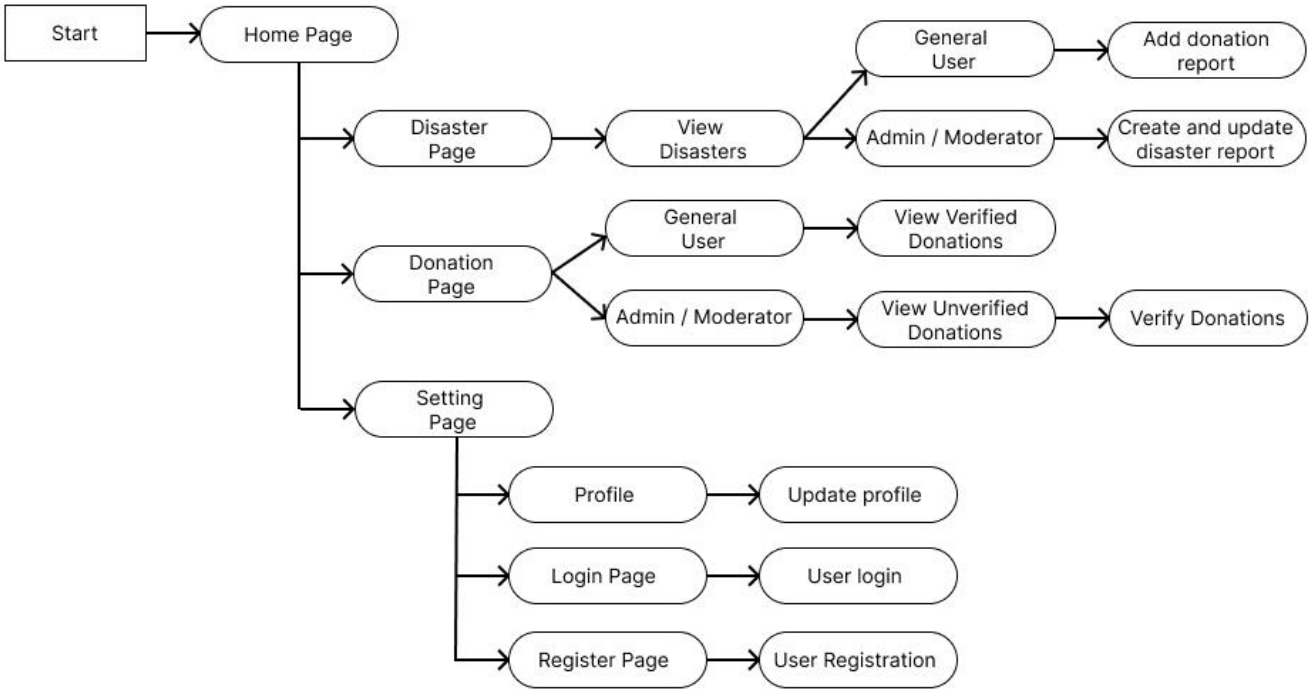


Figure: Flowchart for disaster information entry

2.8 Activity diagram

An activity diagram is a behavioral diagram i.e., it depicts the behavior of a system. Activity Diagrams describe how activities are coordinated to provide a service which can be at different levels of abstraction. Typically, an event needs to be achieved by some operations, particularly where the operation is intended to achieve a number of different things that require coordination, or how the events in a single use case relate to one another, in particular, use cases where activities may overlap and require coordination. It is also suitable for modeling how a collection of use cases coordinates to represent business workflows.



2.9 Entity-Relationship Diagram:

Entity relationship diagram is a data model technique that is constructed to define entities used in a system. ER Diagram or ER model is a type of structural diagram for use in database design. An ERD contains different symbols and connectors that visualize two important pieces of information: The major entities within the system scope, and the inter-relationship among these entities.

Here's a breakdown of the entities and their attributes:

accounts_user: This entity stores information about the users who can access the system. It has attributes including user ID, username, email, password, user type, platform they use to access the system (mobile app), creation time, modification time, and whether the user is active or not.

Location: This entity stores information about geographical locations. It has attributes including division ID, district_ID, upazila_ID, union_ID, division_name, district_name, upazila_name, union_name, latitude, and longitude. It also has creation time, modification time for both the location data and the administrative unit data (division, district, etc.).

Disaster: This entity stores information about disasters. It has attributes including disaster ID, title, description, start time, end time, state (ongoing, resolved, etc.), a field for images, creation time, and modification time.

Donation: This entity stores information about donations made to relief efforts. It has attributes including donation ID, title, description, images, videos, documents, creation time, and modification time. There are also fields for the state of the donation (received, distributed, etc.)

Donation_donor: This entity stores information about the donors who make the donations. This is entitled in accounts_user as donor.

Affected_people: This entity stores the approximate number of people affected by the disaster.

Disaster_affected: This entity establishes a relationship between disasters and the areas affected by them. It likely has foreign keys for disaster ID, district ID, upazila ID, and union ID.

Donation_donated: This entity establishes a relationship between donations and the areas where they are distributed. It likely has foreign keys for donation ID, district ID, upazila ID, and union ID.

Donation_donated_item: This entity stores information about the items included in the donations. It has attributes including donation ID, item name, quantity, unit (e.g., box, bag, etc.), description, creation time, modification time.

Media_center_mediacenter: This entity likely stores information about multimedia files uploaded to the system. It has attributes including media ID, title, description, file type, creation time, and modification time.

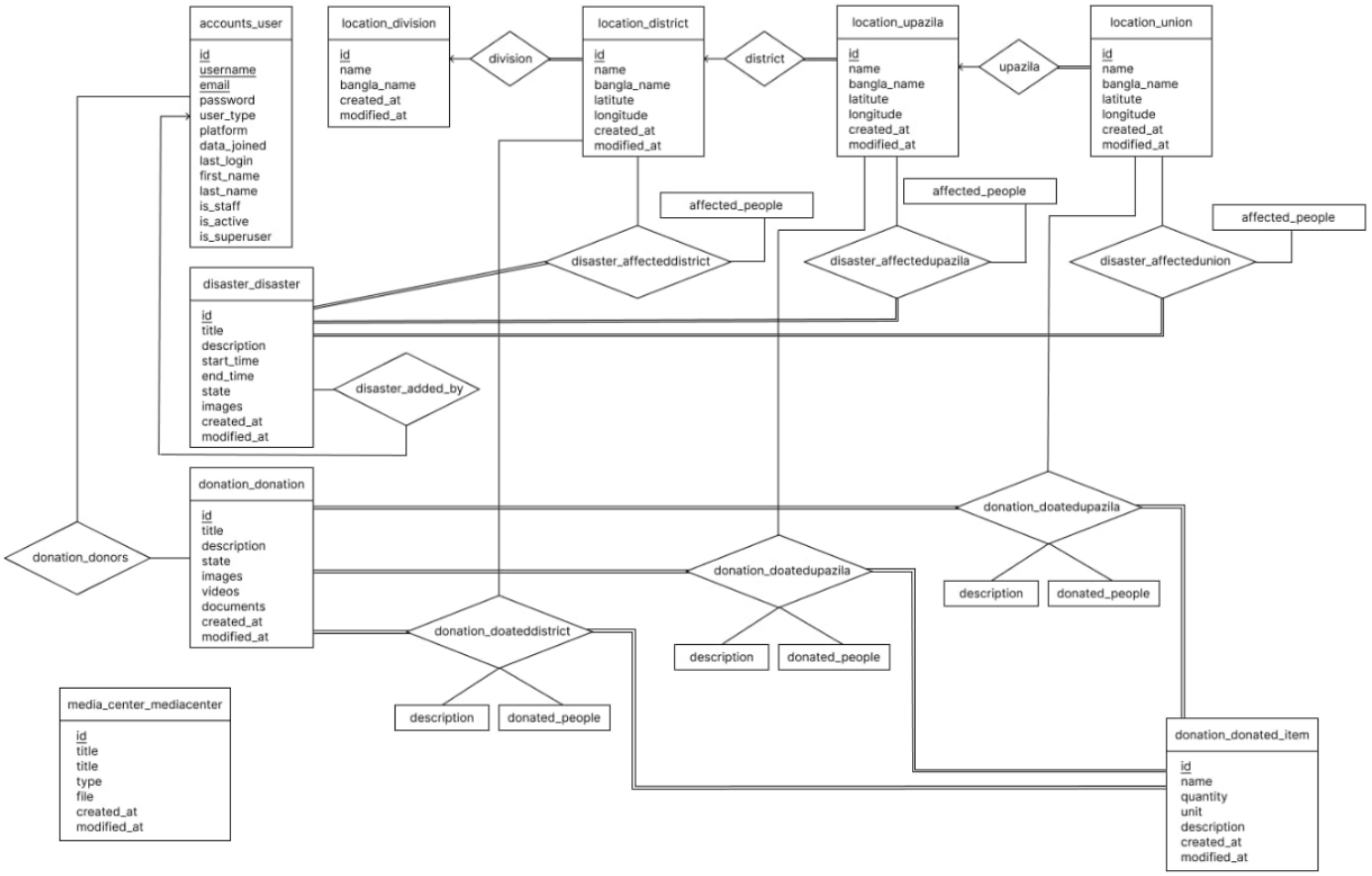
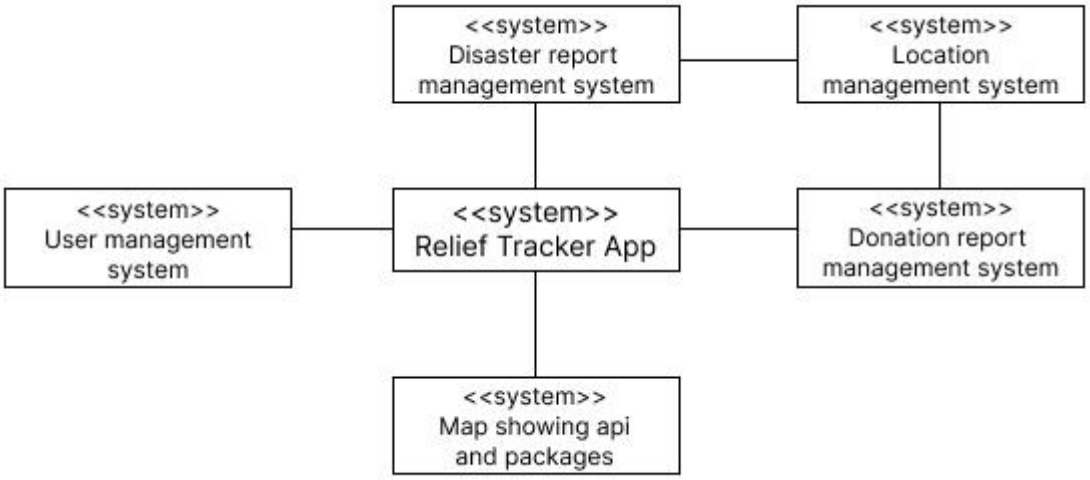


Figure: Entity-relationship Diagram of Relief_Tracker

2.10 Context diagram

In the context diagram for the Relief Tracker project, the main system, Relief Tracker, is depicted at the center. Surrounding this central system are several external systems or entities, including the Disaster Report Management System, Location Management System, User Management System, Donation Report Management System, and the Map API and Packages. Arrows extending from these external systems towards Relief Tracker illustrate the flow of data and interactions between them. This diagram provides a high-level overview of how Relief Tracker interacts with other systems or components within its environment.



2.11 Sequence Diagram

A sequence diagram is a type of interaction diagram because it describes how—and in what order - a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process (Jansen et al., 2009).

This report includes the sequence of diagram of the followings -

Login: The user initiates the login process by providing their credentials. The Relief Tracker System validates the credentials by communicating with the User Database. Once validated, the system displays the user dashboard.

Registration: In the registration process, the user submits their registration details. The Relief Tracker System communicates with the User Database to register the new user and confirms the registration with the user.

Enter New Disaster Information: The user inputs details about a new disaster. The Relief Tracker System prompts for disaster details, receives the input from the user, and stores the information in the User Database.

Record Donation Report: For recording a donation report, the system provides a reporting form to the user. The user fills out the form, and the Relief Tracker System stores the report in the User Database.

Utilize Map Functionality: To utilize the map functionality, the system requests the map interface from the Map API. The API responds with the map interface, which the Relief Tracker System displays to the user.

2.11.1. Login

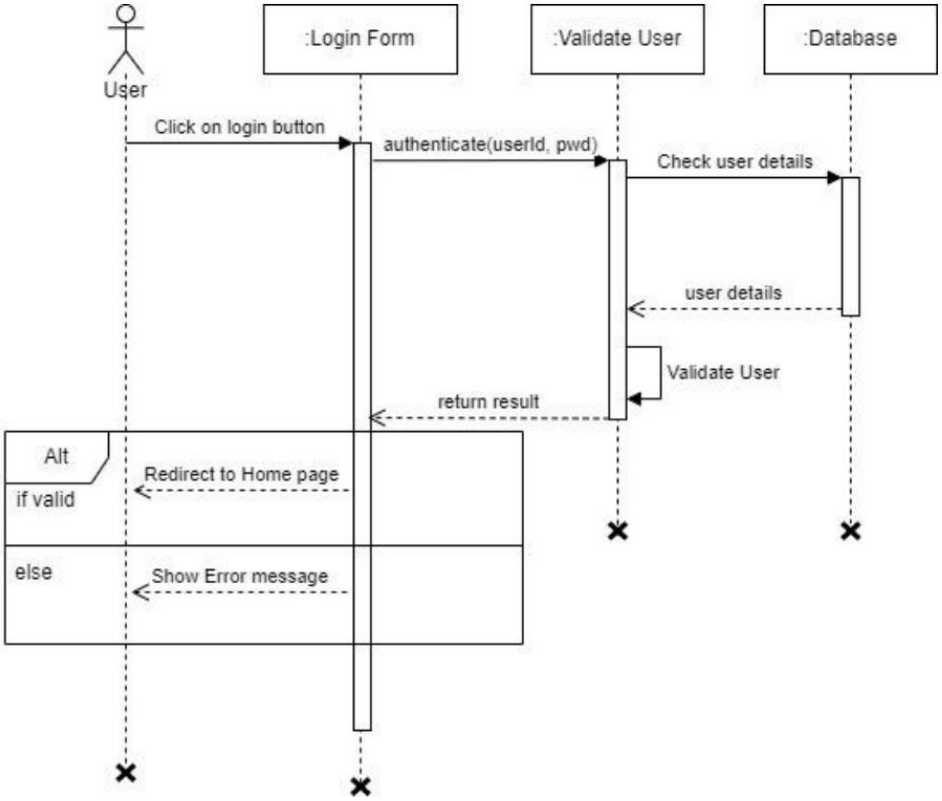


Figure: Sequence diagram for login

2.11.2. Registration

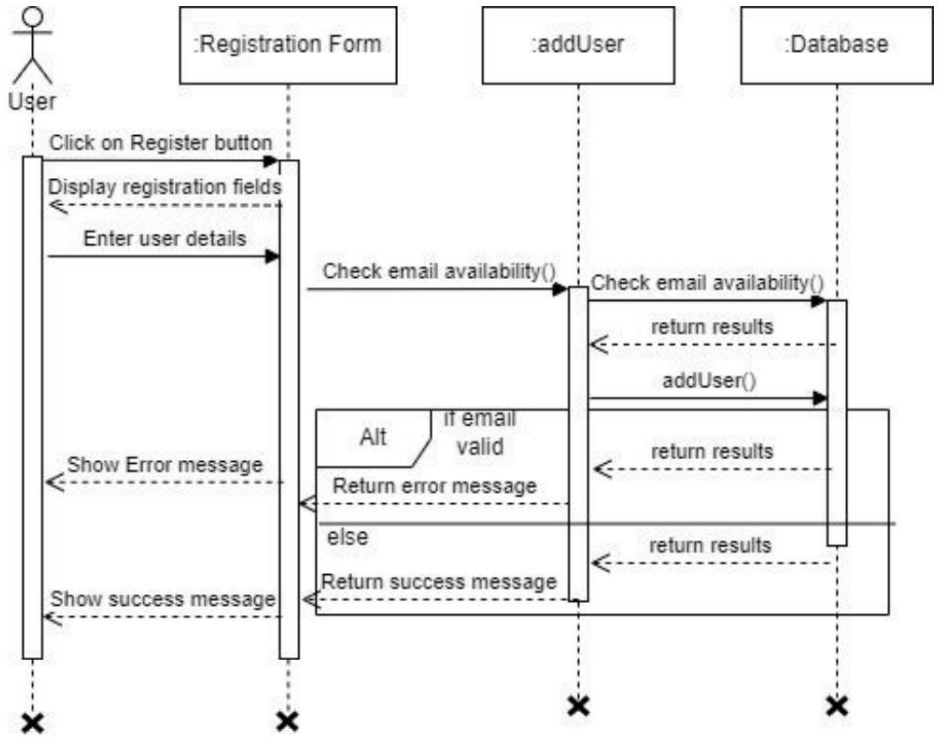


Figure: Sequence diagram for Registration

2.11.3. Enter new disaster information

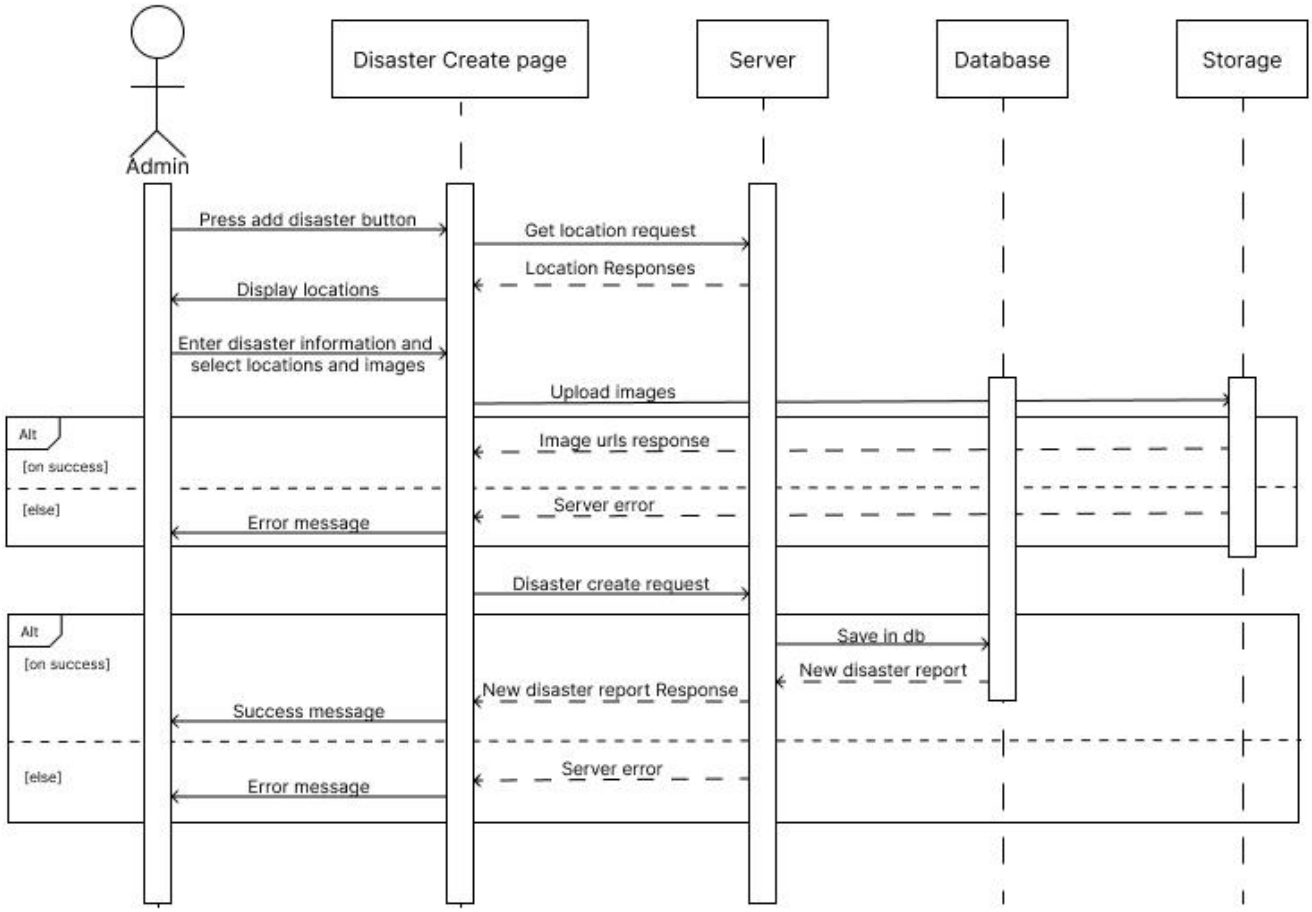


Figure: Sequence diagram for entering new disaster information

2.11.4. Submit Donation report

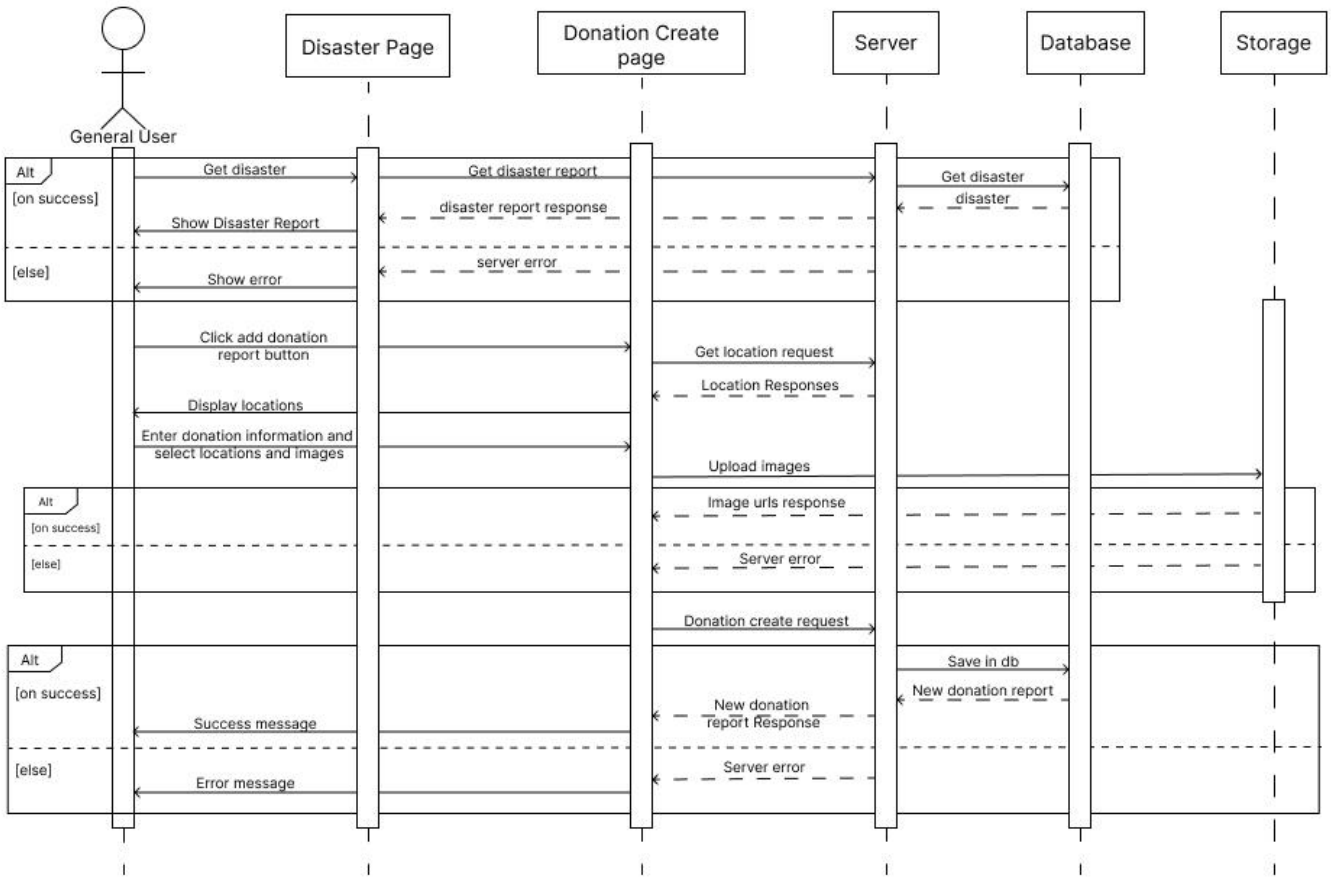


Figure: Sequence diagram for recording donation report

2.11.5. Map API Utilization

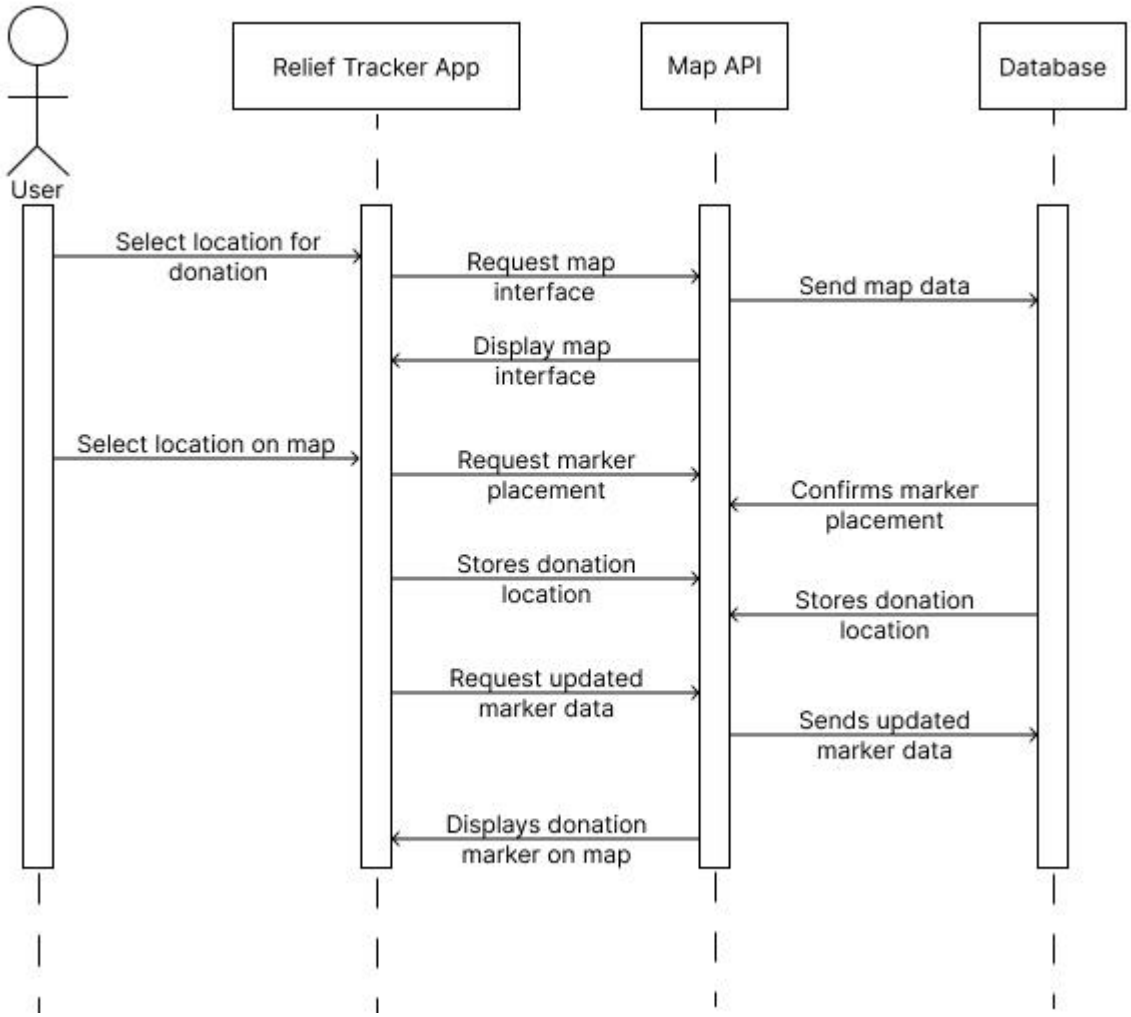


Figure: Sequence diagram for utilizing map API

2.12 Data Flow Diagram (DFD)

In the Relief Tracker project, a Data Flow Diagram (DFD) serves as a visual representation of the flow of data within the system and the processes that manipulate this data. The DFD helps to analyze the current system, define input, and output specifications, and outline the implementation plan for the relief tracking system (Ernst et al., 2023).

The DFD consists of four main components:

External Entities: These represent sources or destinations of data that interact with the Relief Tracker system. External entities can include donors, local moderators, administrators, and other stakeholders involved in relief efforts. They communicate with the system by providing input or receiving output. External entities are typically depicted on the edges of the diagram.

Processes: Processes in the DFD represent actions or operations that occur within the Relief Tracker system. These processes manipulate the data flowing through the system. Examples of processes in the Relief Tracker system include submitting relief requests, verifying relief submissions, mapping relief distribution, and updating relief information. Each process is labeled with a short description of its function.

Data Stores: Data stores represent repositories where information is stored within the Relief Tracker system. These can include databases, files, or other storage mechanisms. Examples of data stores in the Relief Tracker system include databases storing user profiles, relief request details, verification status, and distribution information. Each data store is labeled with a descriptive name, such as "donation_donated_item" or "disaster_disaster"

Data Flows: Data flows depict the movement of data between external entities, processes, and data stores within the Relief Tracker system. These arrows indicate the flow of information and are labeled with brief descriptions of the data being transferred. Examples of data flows in the Relief Tracker system include the submission of donation records from donors, and the display of relief distribution on a map.

2.12.1. DFD Level 0

A context diagram is a top level (also known as "Level 0") data flow diagram It only contains one process node ("Process 0") that generalizes the function of the entire system in relation to external entities. Draw data flow diagrams can be made in several nested layers.

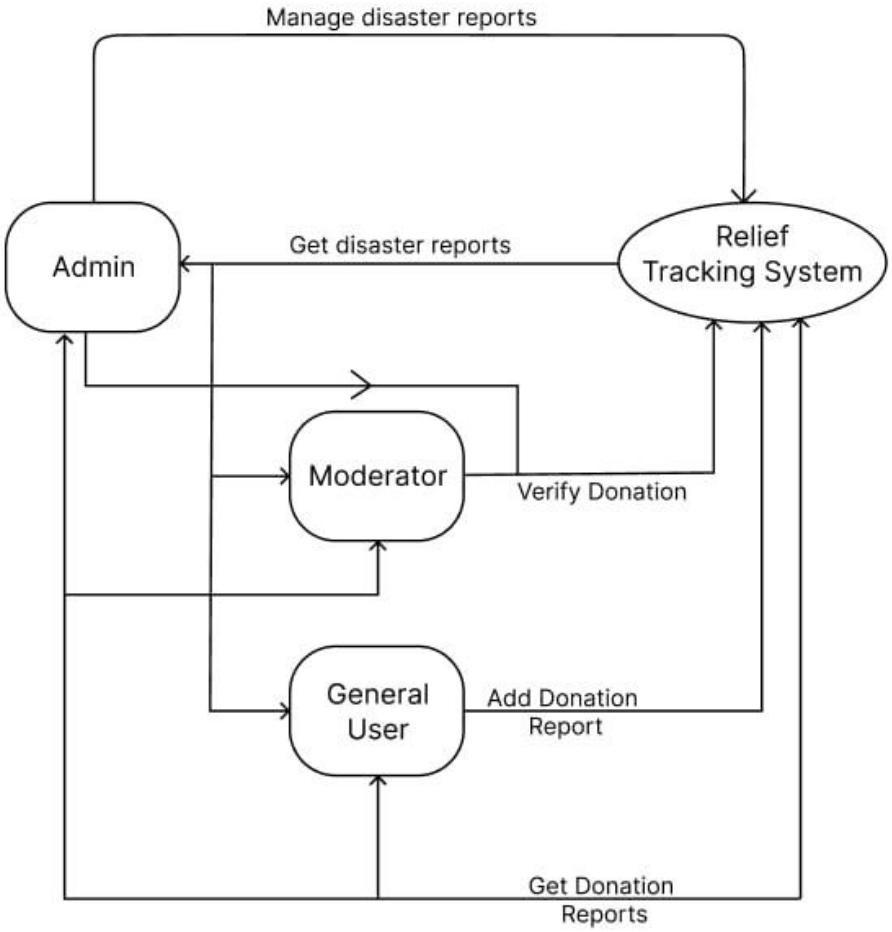


Figure: Data Flow Diagram (Level 0)

2.12.2. DFD level 1

Level 1 DFD's are still a general overview, but they go into more detail than a context diagram. In a level 1 data flow diagram, the single process node from the context diagram is broken down into sub processes. As these processes are added, the diagram will need additional data flows and data stores to link them together.

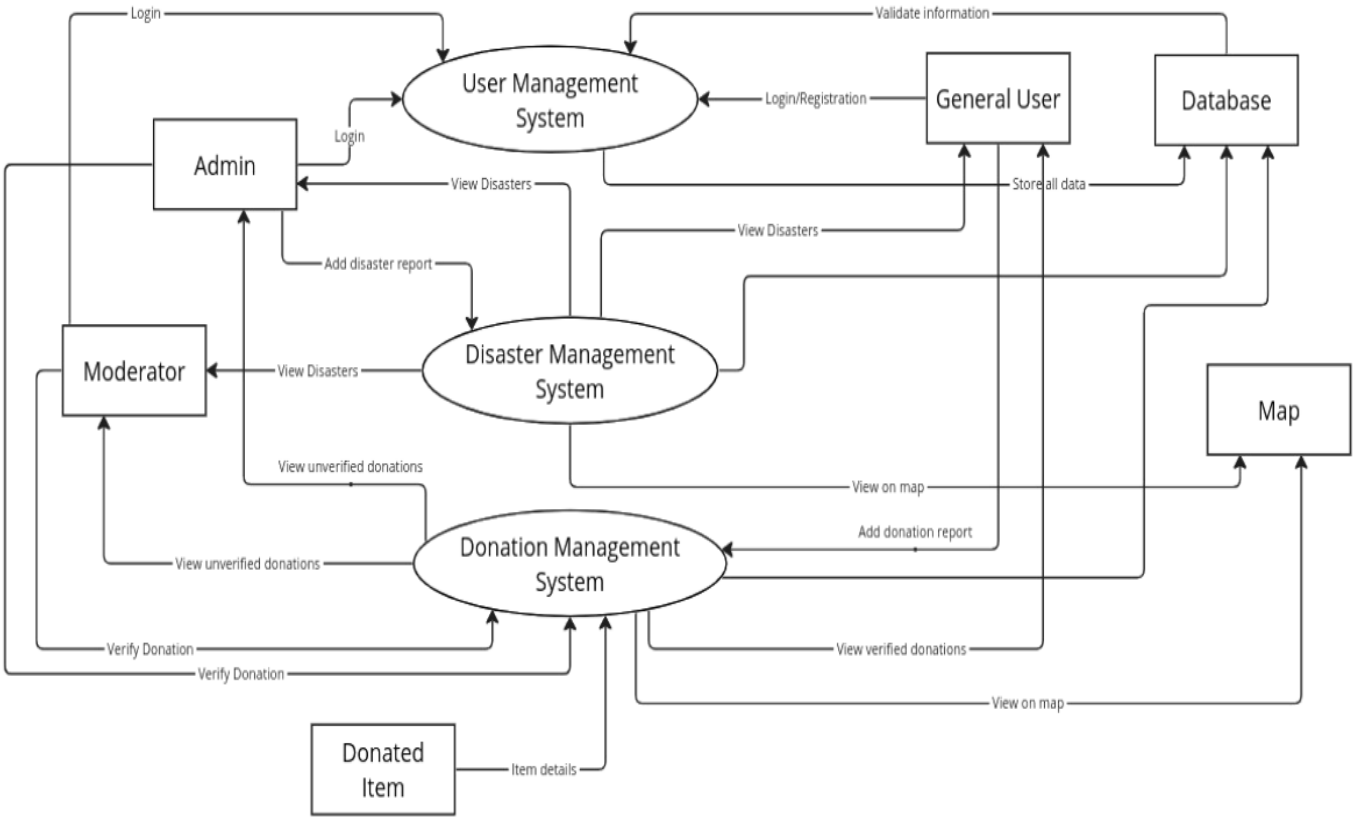


Figure: Data Flow Diagram (level 1)

2.12.3. DFD level 2

In a Level 2 Data Flow Diagram (DFD) for the Relief Tracker project, we'll provide a more detailed view of the system by expanding on the processes and data flows identified in the Level 1 DFD. Level 2 DFDs decompose each process from the Level 1 DFD into sub-processes, providing a more granular understanding of the system's functionality. Here's an example of how we can represent the Level 2 DFD for the Relief Tracker system.

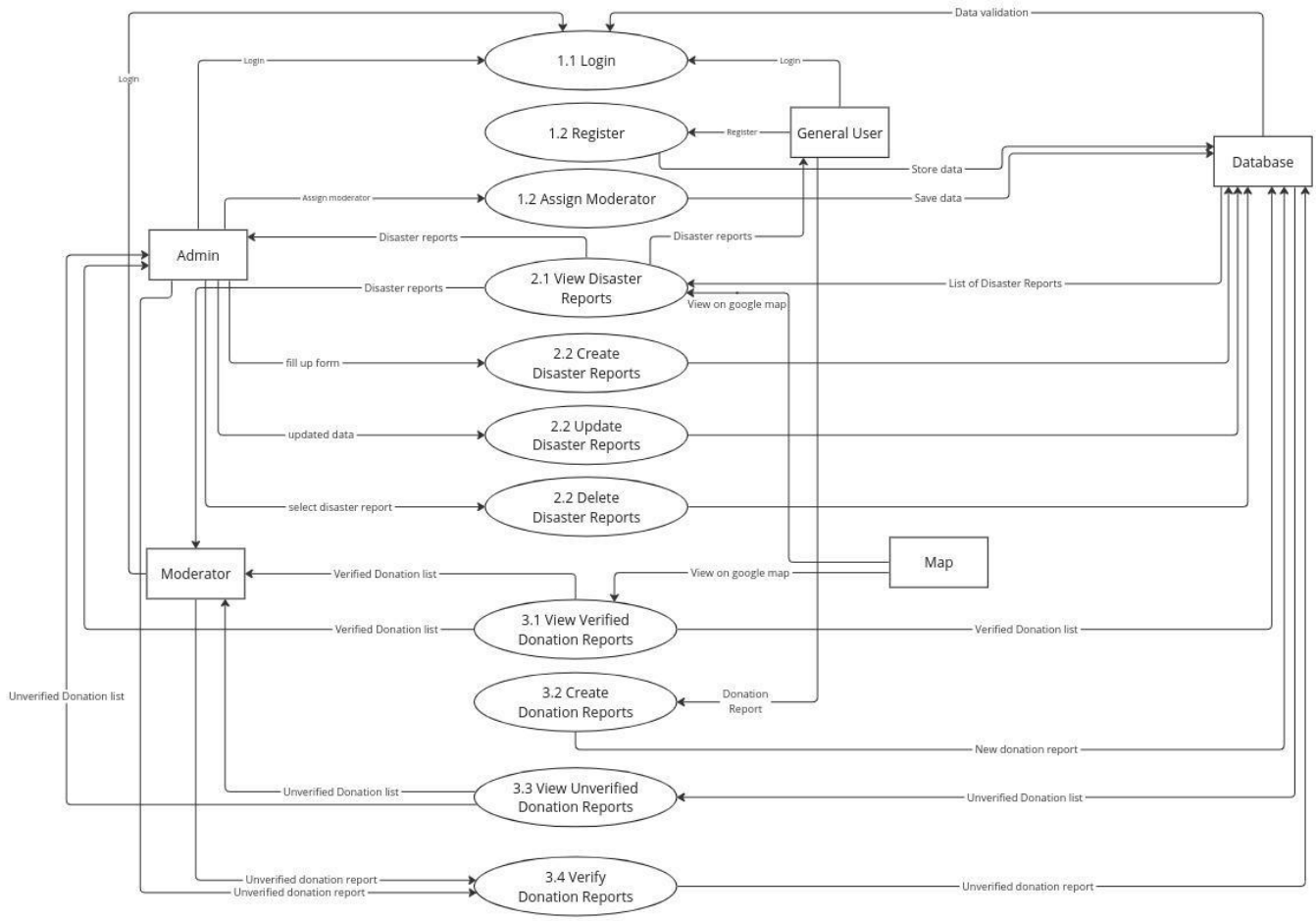


Figure: Data Flow Diagram (level 2)

3. Appendix: Design Documentation

Use case scenario in detail

Following is the detailed elaboration of use case scenarios mentioned in design and architecture section.

3.1 Use Case 1: Log in to the system

Primary Actor -: Donor/Campaign Organizer, Administration Staff

Pre-Condition -: Active Internet connection should be available.

Main Scenario -:

1) Enter into the Relief Tracker application using android phone or tablet.

2) Click Sign in button

3) Enter Username (Email address) and Password and click login

4) System checks for the authenticity of the username and password.

5) According to user roles the system will load the relevant landing page. For general users, the system will load the general dashboard. For admin staff system will load the admin dashboard.

Alternative Scenario -:

4(a) Authorization failed.

4(a) 1. An error message will pop up to the user informing that his/her username or Password is incorrect.

4(a) 2. The system asks the user to reenter the username or password.

3.2 Use Case 2: Register User

Primary Actor -: General users seeking for recent donation reports or adding new donation reports

Pre-Conditions -:

1) An active Internet connection should be available.

2) The user should have a valid email address.

Main Scenario -:

1) Enter into the Relief Tracker application using android phone or tablet.

2) Click the Signup button

3) Select the general user radio button.

4) This will initiate the registration process.

5) A basic questionnaire is given to the user which is relevant to his/her personal details and contact details.

6) The user answers the basic questionnaires and clicks the register button.

7) An invitation email will be sent to the user's given email address to confirm the user’s email address validity.

Alternative Scenario -:

5(a) Email already exists.

5(a) 1. An error message will pop up to the user informing that his/her email address is Already taken.

5(a) 2. The system asks the user to reenter the email address.

5(b) Donor doesn’t answer the mandatory questions.

5(b) 1. An error message will pop up to the user informing that he/she didn’t provide answers to all the mandatory questions.

5(b) 2. The system will highlight the question which isn’t answered by the user.

3.3 Use Case 3: Create Disaster Report

Primary Actor -: Admin

Pre-Conditions -:

1) The admin has access to the Relief Tracking application.
2) The admin is logged into the system.

Main Scenario -:

1) The admin logs in to the Relief Tracking application using android phone or tablet.
2) The admin clicks on the “Create Disaster Report” button.
3) The system displays a form for entering disaster information.
4) The admin enters the disaster information, such as the disaster title, description, location, start time, and end time.
5) The admin selects images or uploads image URLs for the disaster report.
6) The admin clicks the “Submit” button.
7) The system validates disaster information.

   7(a) 1. If the information is valid, the system saves the disaster information to the database and displays a success message.

   7(a) 2. If the information is invalid, the system displays an error message and highlights the fields that need correction.

   7(a) 3. The admin can then close the form or create another disaster report.

Alternative Scenarios -:

8(a)Network Error:

8(a) 1. If there is a network error while communicating with the server, the system will display an error message to the admin.

8(b)Server Error:

8(b) 1. If there is an error on the server while saving the disaster information, the system will display an error message to the admin.

3.4 Use Case 4: Submit Donation Report

Primary Actor -: General User

Pre-Conditions -:

1) The user has an internet connection.
2) The user has a Relief Tracker account.
3) The user has already made a donation to a disaster.

Main Scenario -:

1) The user logs in to the Relief Tracker application using android phone or tablet.
2) The user navigates to the donation reporting section of the application.
3) The system displays a list of disasters that the user has donated to.
4) The user selects the disaster that they want to submit a donation report for.
5) The system asks the user to select the location where they made the donation.
6) The system displays a form for entering donation information.
7) The user enters the donation information, such as the type of donation (e.g., money, goods, services), the amount of the donation, and the date of the donation.
8) The user uploads images or enters URLs for images of their donation.
9) The user submits the donation report.
10) The system validates the donation information.

    10(a) 1. If the information is valid, the system saves the donation information to the database and displays a success message.

    10(a) 2. If the information is invalid, the system displays an error message and highlights the fields that need correction.

Alternative Scenarios -:

11(a) Network Error -:

11(a) 1. If there is a network error while communicating with the server, the system will display an error message to the user.

11(b) Server Error -:

11(b) 1. If there is an error on the server while saving the donation information, the system will display an error message to the user.

11(c) User Doesn’t Select Disaster -:

11(c) 1. If the user does not select a disaster from the list, the system will prompt the user to select a disaster.

3.5 Use Case 5: View Location Reports

Primary Actor: General User

Pre-Conditions -:

1) The user has an internet connection.
2) The user has a Relief Tracker account.

Main Scenario -:

1) The user logs in to the Relief Tracker application.
2) The user navigates to the location reports section of the application.
3) The system utilizes a Map API to display a map.
4) The map might display various overlays or markers based on the type of location reports the user wants to view. For instance:
    a. Disaster locations
    b. Donation drop-off centers
    c. Areas requiring assistance
5) The user interacts with the map provided by the Map API to view details about these locations. This interaction might involve:
    a. Clicking on markers to see information about the location.
    b. Zooming in and out on the map to focus on specific areas.
    c. Using search functionality to find specific locations.

Alternative Scenarios -:

6(a) Network Error -:

If there is a network error while communicating with the server or the Map API, the system will display an error message to the user.

6(b) Server Error -:

If there is an error on the server while retrieving location report data, the system will display an error message to the user.

6(c) Map API Unavailable -:

If the Map API is unavailable, the system might:

- Display a limited view of the location reports without the map functionality.
- Inform the user that the map functionality is currently unavailable.

3.6 Use Case 6: Assign Moderator

Primary Actor -: Admin

Pre-Conditions -:

1) The admin has access to Relief Tracking application with appropriate permissions.
2) The admin is logged into the system.
3) There is a list of users available in the system.
4) There is a area requiring a moderator.

Main Scenario -:

1. The admin navigates to the disaster management or user management section of the system.
2. The system displays a list of disasters or areas requiring a moderator.
3. The admin selects the disaster or area where a moderator needs to be assigned.
4. The system displays a list of users who might be suitable moderators. This list will be filtered based on various criteria such as:
   4.1. Location of the user
   4.2. Expertise or experience relevant to the disaster
   4.3. Past moderation history
5. The admin selects a user from the list to be assigned as a moderator.
6. The system confirms the assignment and sends a notification to the selected user informing them of their new role as a moderator for the specific disaster or area.

Alternative Scenarios -:

7(a) No Suitable Users:

If there are no suitable users available in the system, the system might:

- Inform the admin that there are no users who meet the criteria for moderator.
- Allow the admin to search for users outside of the pre-defined list.

7(b) User Declines Moderation:

If the selected user declines the moderator role, the system might:

- Inform the admin of the user's declination.
- Allow the admin to select a different user from the list.

7(c) Network Error:

If there is a network error while communicating with the server, the system will display an error message to the admin.

7(d) Server Error:

If there is an error on the server while assigning the moderator role, the system will display an error message to the admin.

3.7 Use Case 7: Update Disaster Information

Primary Actor: Admin

Pre-Conditions:

1) The admin has access to the Relief Tracking application with appropriate permissions.
2) The admin is logged into the system.
3) There is an existing disaster record in the system.

Main Scenario:

1. The admin navigates to the disaster management section of the system.
2. The system displays a list of existing disaster records.
3. The admin selects the specific disaster record that needs to be updated.
4. The system displays the details of the selected disaster record, such as title, description, location, start time, end time, and current status.
5. The admin edits the disaster information as needed. This might involve updating details such as:
   5.1. Disaster title or description to reflect new information.
   5.2. Location of the disaster if the affected area has expanded.
   5.3. Start or end time of the disaster if the situation has changed.
   5.4. Status of the disaster (e.g., ongoing, transitioning to recovery, etc.)
   5.5. Adding images or videos to provide visual updates.
6. The admin submits the changes.
7. The system validates the updated information.
   7.1. If the information is valid, the system saves the updates to the disaster record and displays a confirmation message.
   7.2. If the information is invalid (e.g., missing required fields), the system displays an error message and highlights the fields that need correction.

Alternative Scenarios:

1. Network Error:
   1.1. If there is a network error while communicating with the server, the system will display an error message to the admin.
2. Server Error:
   2.1. If there is an error on the server while saving the updated information, the system will display an error message to the admin.
3. Permission Error:
   3.1. If the admin does not have the necessary permissions to update disaster information, the system will display an error message.

3.8 Use Case 8: Verify Donation Records

Primary Actor: Admin/Moderator

Pre-Conditions:

1) The admin/moderator has access to the Relief Tracking application with appropriate permissions.
2) The admin/moderator is logged into the system.
3) There are donation records submitted by users in the system.

Main Scenario:

1. The admin/moderator navigates to the donation management section of the system.
2. The system displays a list of donation records submitted by users. This list might be filtered by various criteria such as:
   2.1. Disaster the donation is associated with
   2.2. Date range of the donation
   2.3. Donation amount
   2.4. Verification status (verified, pending verification, unverified)
3. The admin/moderator selects a donation record for verification.
4. The system displays the details of the donation record, including:
   4.1. Donor information (name, contact details)
   4.2. Donation details (amount, type of donation, date)
   4.3. Images or descriptions uploaded by the donor as proof of donation
5. The admin/moderator reviews the donation record and compares it against any available verification methods:
   5.1. Payment gateway confirmation (if online donation)
   5.2. Third-party verification service (if applicable)
   5.3. Manual verification of proof of donation (e.g., receipts, photos)
6. Based on the review, the admin/moderator takes the following actions:
   6.1. Verify Donation: If the donation is confirmed as legitimate, the admin/moderator marks the record as verified and the donation amount is reflected in the disaster's total.
   6.2. Mark as Unverified: If the donation cannot be verified after due diligence, the admin/moderator marks the record as unverified and might contact the donor to inform them.

Alternative Scenarios:

1. Network Error:
   1.1. If there is a network error while communicating with the server, the system will display an error message to the admin/moderator.
2. Server Error:
   2.1. If there is an error on the server while processing the verification, the system will display an error message to the admin/moderator.
3. Missing Information:

4. If the donation record is missing crucial information for verification, the system might:
    4.1. Prompt the admin/moderator to request clarification from the donor.
    4.2. Not allow the admin/moderator to verify the record until the missing information is provided.
5. Donor Not Responsive:
    5.1. If the donor does not respond to requests for clarification within a reasonable timeframe, the admin/moderator might:
        5.1.1.1. Mark the donation as unverified.
        5.1.1.2. Flag the donor account for further investigation.

4. Conclusion

4.1 Business Prospect:

Relief Tracker connects donors, relief organizations, and affected communities on a single platform, fostering better communication and resource allocation during disasters. Real-time information sharing, streamlined donation management, and efficient reporting create a more transparent and efficient disaster relief ecosystem. Relief Tracker caters to individual donors, NGOs, and government agencies, with potential monetization through transaction fees, premium services, and corporate sponsorships.

4.2 Summary:

Relief Tracker has the potential to be a game-changer in disaster relief. By connecting various stakeholders on a single platform, it promotes better communication, transparency, and resource allocation. This translates to informed decision-making by relief organizations, ultimately leading to a more efficient and impactful disaster response. The app's diverse user base and potential for monetization solidify its position as a sustainable solution for the disaster relief landscape.

5. Github Link: https://github.com/Sami-63/Relief-Tracker

6. References

Alexeeva, Z., Perez-Palacin, D., & Mirandola, R. (2016). Design decision documentation: A literature overview. In *Software Architecture: 10th European Conference, ECSA 2016, Copenhagen, Denmark, November 28--December 2, 2016, Proceedings 10* (pp. 84-101). Springer International Publishing.

Bachmann, F., Bass, L., Carriere, J., Clements, P. C., Garlan, D., Ivers, J., ... & Little, R. (2000). Software architecture documentation in practice: Documenting architectural layers.

Barcelos, R. F., & Travassos, G. H. (2006). Evaluation Approaches for Software Architectural Documents: a Systematic Review. *CIbSE*, 433-446.

Che, M. (2014). Managing architectural design decision documentation and evolution (Doctoral dissertation).

Ernst, N. A., & Robillard, M. P. (2023). A study of documentation for software architecture. *Empirical Software Engineering*, *28*(5), 122.

Forward, A., & Lethbridge, T. C. (2002, November). The relevance of software documentation, tools and technologies: a survey. In *Proceedings of the 2002 ACM symposium on Document engineering* (pp. 26-33).

Jansen, A., Avgeriou, P., & van der Ven, J. S. (2009). Enriching software architecture documentation. *Journal of Systems and Software*, *82*(8), 1232-1248.

Rost, D., Naab, M., Lima, C., & von Flach Garcia Chavez, C. (2013). Software architecture documentation for developers: A survey. In *Software Architecture: 7th European Conference, ECSA 2013, Montpellier, France, July 1-5, 2013. Proceedings 7* (pp. 72-88). Springer Berlin Heidelberg.

Zhu, H. (2005). Software design methodology: From principles to architectural styles. Elsevier.