



Disaster Relief and Donation Management System

Project Report

Group 12

AS20240395- K I R Abeysekara

AS20240427- B P R Sachintha

AS20240404- A H M D Silva

AS20240577- N G Sachindu Sathsara

AS20240546- T H Sandaruwan

AS20240538- S M Wijesekara

AS20240559-E.K.S Govinda

CSC 1042 Database Management Systems

Lecturer in Charge: Dr. Surani Tissera

Department: Department of Computer Science

Submission Date: 01.02.2026

Table of Contents

1. Problem Statement	3
2. System Scope and Limitations	3
3. Conceptual Design	4
3.1 Assumptions	4
3.2 Entity-Relationship (ER) Diagram	6
4. Logical Design (Relational Schema)	7
5. Implemented Database	8
6. Populated Database Tables	8
Table:Donor	8
Table: Donation	9
Table: DonationItem	9
Table: Item	10
Table: Inventory	10
Table: Beneficiary	11
Table: Distribution	11
Table: DistributionItem	12
Table: User	12
7. Sample CRUD Operations	13
Table Creation	13
Table: Donor	13
Table: Donation	14
INSERT Operation (Create)	14
Table: Donor	14
Table: Donation	15
SELECT Operation (Read)	15
Table: Donor	15
Table: Donation	16
DELETE Operation (Delete)	16
Table: Donor	16
Table: Donation	17
8. User Interface Implementation	17
8.1 Login Interface	17
8.2 Dashboard	18
8.3 Donation Entry Form	18
8.4 Inventory View	18
9. Conclusion	19
10. Individual Contributions	20

1. Problem Statement

In the aftermath of natural disasters, the immediate management of relief efforts is often chaotic and inefficient. Manual methods of tracking donations, inventory, and distributions lead to significant data inconsistencies, loss of resources, and uneven aid distribution. There is often a lack of transparency regarding which donors have contributed, what items are currently in stock, and which beneficiaries have already received aid.

The Solution: We have designed and implemented the **Disaster Relief and Donation Management System**. This relational database system streamlines the workflow by digitalizing the tracking process. It ensures:

- **Donors** and their specific **Donations** are accurately recorded and retrievable.
- **Inventory** is automatically updated based on incoming donations and outgoing distributions.
- **Beneficiaries** are tracked to prevent duplicate distributions and ensure fair aid allocation.
- **Users** (Staff/Admins) have secure access to manage records.

2. System Scope and Limitations

This project focuses on designing and implementing a Disaster Relief and Donation Management System with primary emphasis on relational database design and SQL operations.

System Scope

The scope of the system includes:

- Designing a relational database schema using ER modeling and logical design principles.
- Defining primary keys and foreign key relationships between entities such as Donor, Donation, Inventory, Beneficiary, Distribution, and Item.
- Executing SQL DML operations (INSERT, SELECT, UPDATE, DELETE) to manage data in the database.
- Demonstrating database functionality using SQL queries executed directly in phpMyAdmin.
- Providing a basic PHP-based user interface to trigger selected SQL operations for ease of interaction and demonstration.
- Managing inventory and distribution data primarily through insert and select operations, reflecting real-world transactional behavior.

3. Conceptual Design

3.1 Assumptions

The following assumptions were made during the creation of the Entity-Relationship (ER) Diagram:

1) Each donor is uniquely identifiable

Assumption:

Each donor is uniquely identified by a system-generated DonorID.

Reason:

Donor names and contact numbers may not be unique. Therefore, a surrogate primary key is required to uniquely identify each donor.

Impact on ER Diagram:

DonorID is used as the primary key of the **Donor** entity.

2) A donor can make multiple donations

Assumption:

A single donor can make multiple donations over time, while each donation is associated with exactly one donor.

Reason:

This reflects real-world donation behavior where donors may contribute more than once.

Impact on ER Diagram:

A one-to-many relationship exists between **Donor** and **Donation**, with DonorID included as a foreign key in the **Donation** entity.

3) Inventory items are centrally managed

Assumption:

Inventory represents relief items stored and managed by the organization, and each item is uniquely identifiable.

Reason:

Centralized inventory management is required to accurately track available relief items.

Impact on ER Diagram:

The **Inventory** entity contains a unique primary key (InventoryID) and maintains item quantity information.

4) Inventory quantities change through system operations

Assumption:

Inventory quantities may change due to donations received or distributions made.

Reason:

Inventory levels must reflect real-time stock availability.

Impact on ER Diagram:

The **Inventory** entity allows controlled quantity updates through system operations.

5) Distribution records represent relief allocation events

Assumption:

Distribution records capture details of relief items distributed to beneficiaries and are mainly used for record-keeping.

Reason:

Distribution data should not be frequently modified once recorded to ensure data integrity.

Impact on ER Diagram:

The **Distribution** entity is modeled as a separate transactional entity with limited update and delete operations.

6) Each entity uses a single-attribute primary key

Assumption:

All entities use a single-column primary key.

Reason:

Single-attribute primary keys simplify database design and improve clarity.

Impact on ER Diagram:

No composite primary keys were used in the database design.

7) User authentication is outside the project scope

Assumption:

The system supports authenticated users with predefined roles (admin and staff), where administrative users have elevated privileges compared to staff users.

Reason:

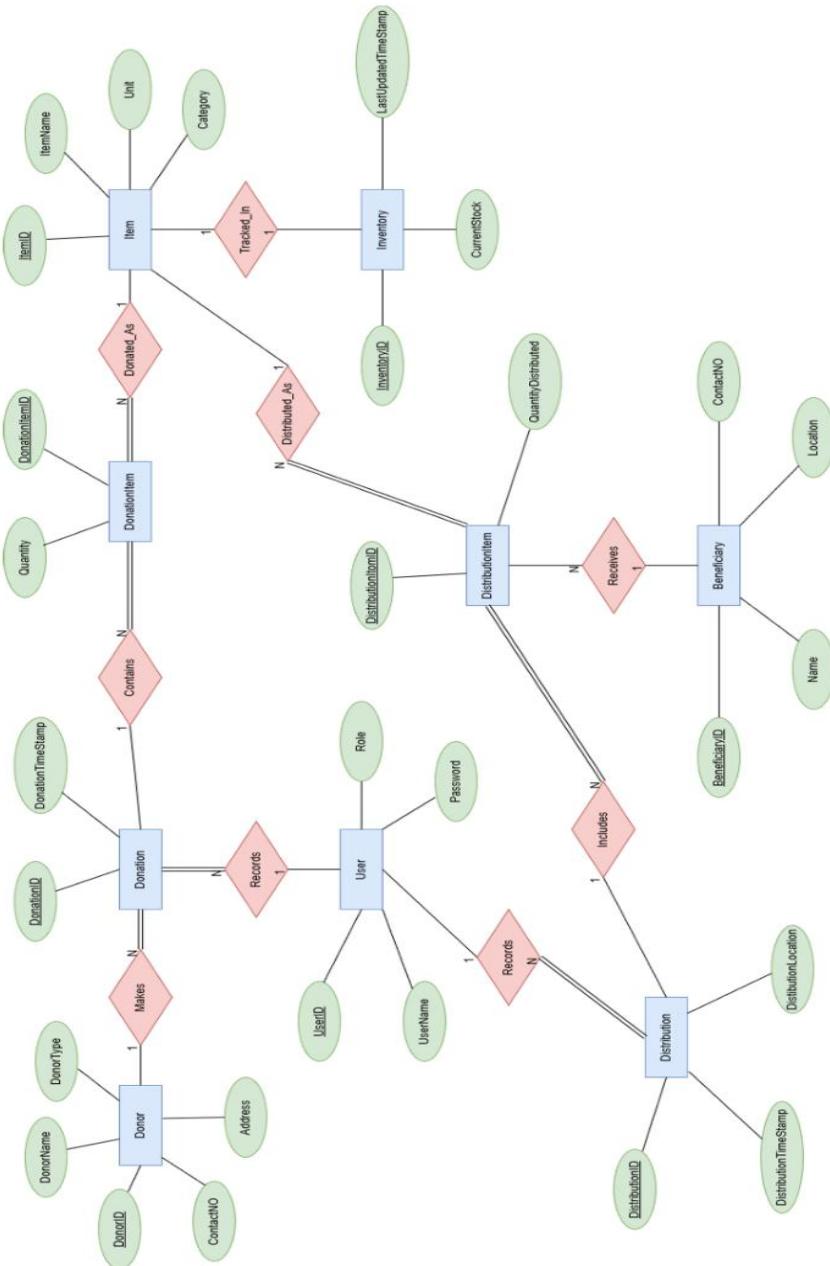
To ensure data security and integrity, role-based access control is enforced at the application level. Editing and deletion operations are restricted to administrative users, while staff users are limited to data entry and viewing functions.

Impact on ER Diagram:

The **User** entity includes a role attribute to represent different user types, while permission enforcement is implemented at the application level.

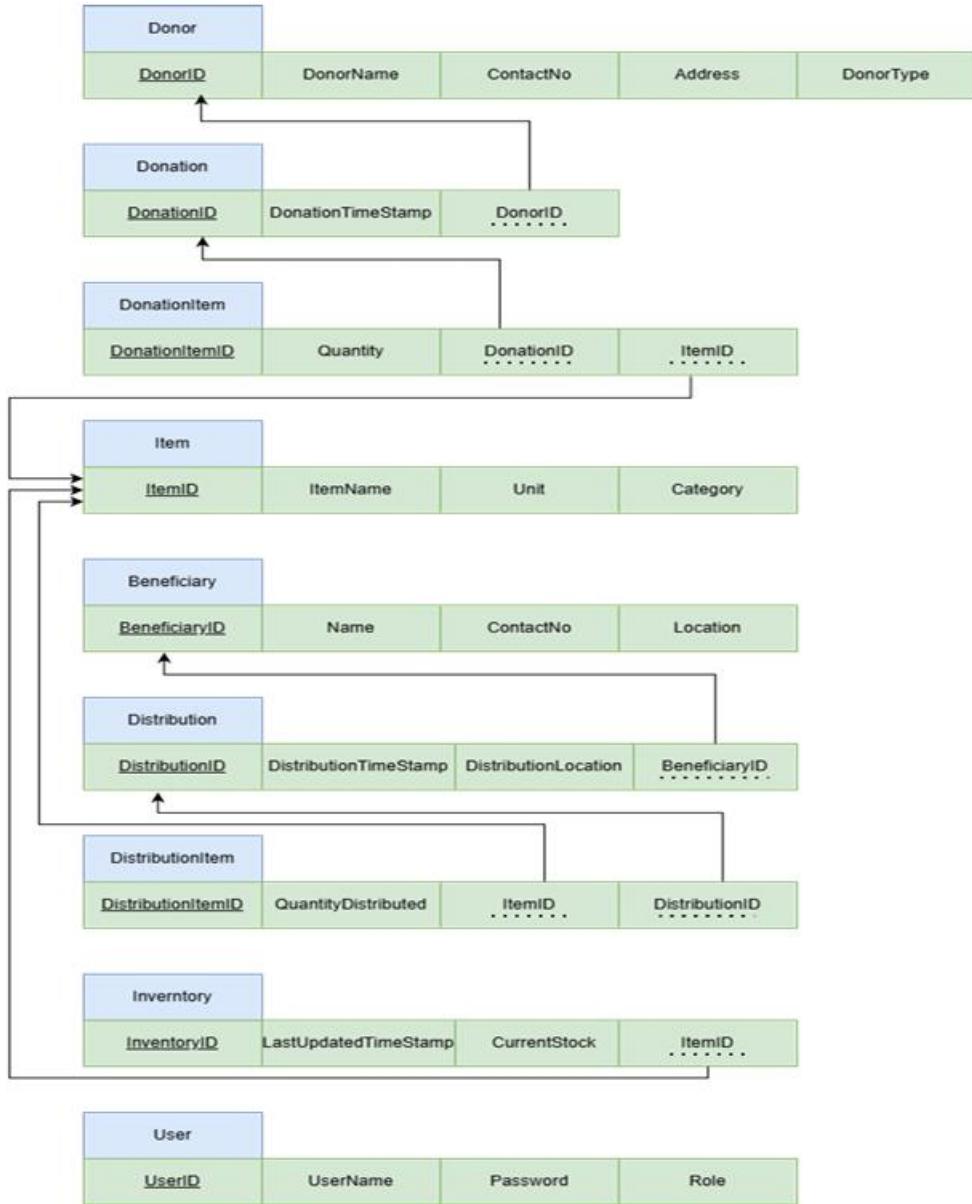
3.2 Entity-Relationship (ER) Diagram

The ER Diagram below illustrates the conceptual model of the system, defining the entities and their relationships.



4. Logical Design (Relational Schema)

The conceptual design was converted into the following Relational Schema. Primary Keys (PK) and Foreign Keys (FK) are identified to maintain referential integrity.



5. Implemented Database

The screenshot shows the phpMyAdmin interface for the 'disaster_mgt_db' database. The left sidebar lists databases and tables. The main area displays a table of 9 tables with columns for Action, Table, Rows, Type, Collation, Size, and Overhead. Below the table, there's a 'Create new table' section with fields for Table name (empty) and Number of columns (4), and a 'Create' button.

Action	Table	Rows	Type	Collation	Size	Overhead
Browse Structure Search Insert Empty Drop	beneficiary	4	InnoDB	utf8mb4_general_ci	16.0 Kib	-
Browse Structure Search Insert Empty Drop	distribution	4	InnoDB	utf8mb4_general_ci	32.0 Kib	-
Browse Structure Search Insert Empty Drop	distributionitem	3	InnoDB	utf8mb4_general_ci	48.0 Kib	-
Browse Structure Search Insert Empty Drop	donation	3	InnoDB	utf8mb4_general_ci	32.0 Kib	-
Browse Structure Search Insert Empty Drop	donationitem	0	InnoDB	utf8mb4_general_ci	48.0 Kib	-
Browse Structure Search Insert Empty Drop	donor	3	InnoDB	utf8mb4_general_ci	16.0 Kib	-
Browse Structure Search Insert Empty Drop	inventory	3	InnoDB	utf8mb4_general_ci	32.0 Kib	-
Browse Structure Search Insert Empty Drop	item	4	InnoDB	utf8mb4_general_ci	16.0 Kib	-
Browse Structure Search Insert Empty Drop	user	3	InnoDB	utf8mb4_general_ci	16.0 Kib	-
Sum	9 tables	27	InnoDB	utf8mb4_general_ci	256.0 Kib	0 B

6. Populated Database Tables

The following screenshots demonstrate that the database has been successfully created and populated with data.

Table: Donor

The screenshot shows the phpMyAdmin interface for the 'donor' table. The top navigation bar includes tabs like Browse, Structure, SQL, and Insert. The main area contains a query editor with the SQL command 'SELECT * FROM donor;', a results grid showing three rows of donor data, and various navigation and search controls.

DonorID	DonorName	ContactNo	Address	DonorType
17	Saman Lakmal	0712345678	Colombo	Individual
18	Bright Org Ltd	0119876543	Main Street, Kandy	Organization
19	Kamala Perera	0771234567	Galle	Individual

Table: Donation

The screenshot shows the MySQL Workbench interface with the 'donation' table selected. The table has three columns: DonationID, DonationTimeStamp, and DonorID. The data shows three rows with values 10, 11, and 12 respectively.

	DonationID	DonationTimeStamp	DonorID
<input type="checkbox"/>	10	2026-01-27 10:00:00	17
<input type="checkbox"/>	11	2026-01-27 11:00:00	18
<input type="checkbox"/>	12	2026-01-27 12:00:00	19

Table: DonationItem

The screenshot shows the MySQL Workbench interface with the 'donationitem' table selected. The table has four columns: DonationItemID, Quantity, DonationID, and ItemID. The data shows four rows with values (1, 1, 10, 2), (2, 5, 10, 2), (3, 20, 12, 4), and (4, 15, 11, 3).

	DonationItemID	Quantity	DonationID	ItemID
<input type="checkbox"/>	1	1	10	2
<input type="checkbox"/>	2	5	10	2
<input type="checkbox"/>	3	20	12	4
<input type="checkbox"/>	4	15	11	3

Table: Item

Run SQL query/queries on table disaster_mgt_db.item:

```
1 SELECT * FROM item;
```

ItemID ItemName Unit Category

	ItemID	ItemName	Unit	Category
<input type="checkbox"/>	2	cereal	5 packs	cereal
<input type="checkbox"/>	3	Rice	kg	Food
<input type="checkbox"/>	4	Water Bottle	liters	Drink
<input type="checkbox"/>	5	Blanket	piece	Clothing

Table: Inventory

Run SQL query/queries on table disaster_mgt_db.inventory:

```
1 SELECT * FROM inventory;
```

InventoryID LastUpdatedTimeStamp CurrentStock ItemID

	InventoryID	LastUpdatedTimeStamp	CurrentStock	ItemID
<input type="checkbox"/>	2	2026-01-27 14:00:00	100	4
<input type="checkbox"/>	3	2026-01-27 14:30:00	50	3
<input type="checkbox"/>	4	2026-01-27 15:00:00	30	5

Table: Beneficiary

The screenshot shows the MySQL Workbench interface for the 'beneficiary' table in the 'disaster_mgt_db' database. The table has four columns: BeneficiaryID, Name, ContactNo, and Location. The data shows four rows of beneficiary information.

	BeneficiaryID	Name	ContactNo	Location
<input type="checkbox"/>	1	Saman Silva	0712345678	Kandy Relief Camp 1
<input type="checkbox"/>	2	Sithumini Gamage	0711111111	Colombo relief camp
<input type="checkbox"/>	3	A D Kelum	0722222222	Kandy relief camp 2
<input type="checkbox"/>	4	Nayani Silva	0733333333	Galle Town Hall

Table: Distribution

The screenshot shows the MySQL Workbench interface for the 'distribution' table in the 'disaster_mgt_db' database. The table has four columns: DistributionID, DistributionTimeStamp, DistributionLocation, and BeneficiaryID. The data shows four rows of distribution information.

	DistributionID	DistributionTimeStamp	DistributionLocation	BeneficiaryID
<input type="checkbox"/>	3	2026-01-25 15:00:00	Kandy Relief Camp 1	1
<input type="checkbox"/>	4	2026-01-27 16:00:00	Community Center Bandarawela	3
<input type="checkbox"/>	5	2026-01-27 17:00:00	Galle Town Hall	4
<input type="checkbox"/>	6	2026-01-27 23:52:00	Kandy Relief Camp 1	3

Table: DistributionItem

The screenshot shows the phpMyAdmin interface for the 'distributionitem' table in the 'disaster_mgt_db' database. The table structure is displayed on the right, and the data grid below it contains three rows:

	DistributionItemID	QuantityDistributed	ItemID	DistributionID
<input type="checkbox"/>	1	5	3	3
<input type="checkbox"/>	2	10	4	5
<input type="checkbox"/>	3	2	3	3

Below the table, there is a message indicating that the current selection does not contain a unique column, so grid edit, checkbox, Edit, Copy and Delete features are not available.

Table: User

The screenshot shows the phpMyAdmin interface for the 'user' table in the 'disaster_mgt_db' database. The table structure is displayed on the right, and the data grid below it contains three rows:

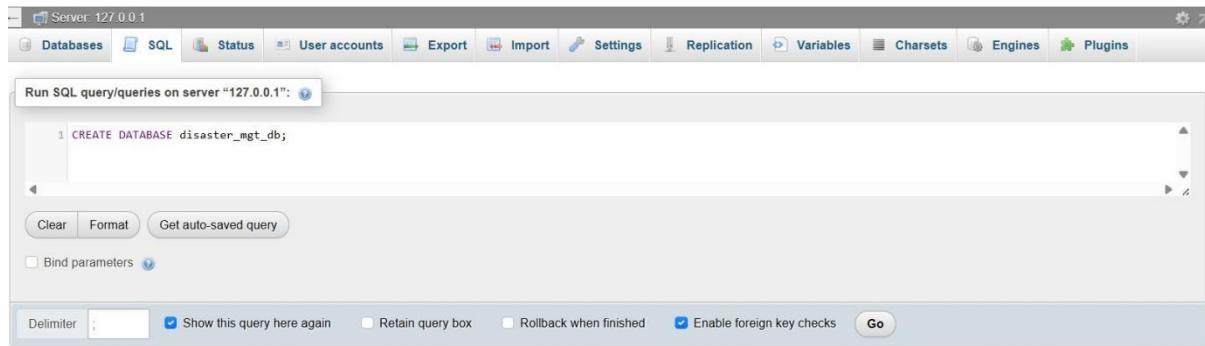
UserID	UserName	Password	Role
6	admin	admin123	Admin
7	staff1	staff123	Staff
8	staff2	staff456	Staff

Below the table, there is a message indicating that the current selection does not contain a unique column, so grid edit, checkbox, Edit, Copy and Delete features are not available.

7. Sample CRUD Operations

The following screenshots demonstrate the functional implementation of Create, Read, and Delete operations using SQL.

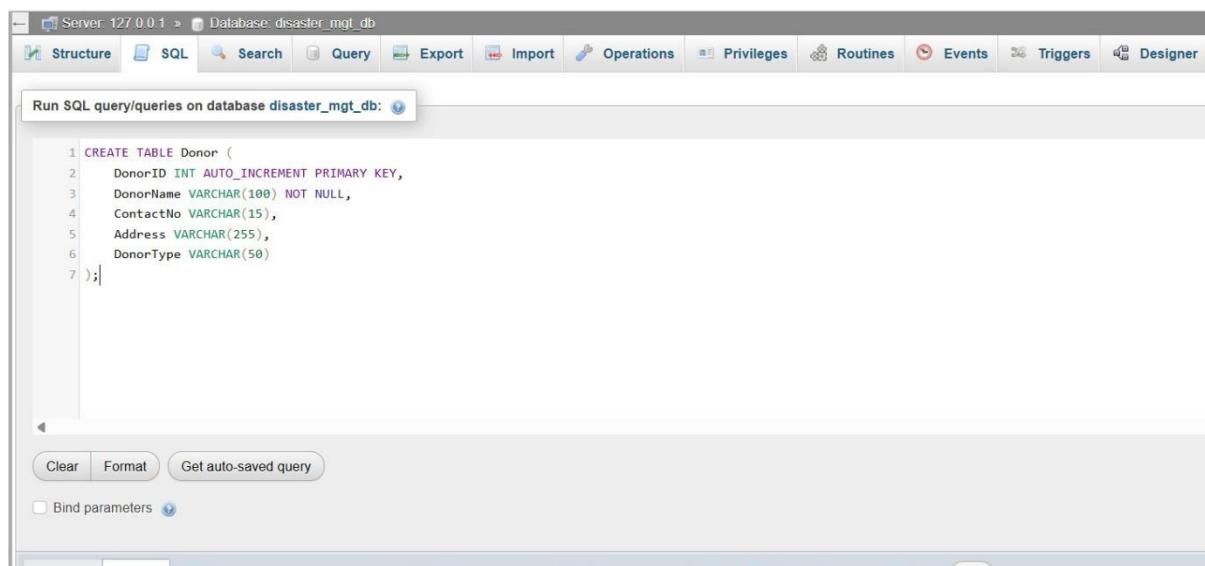
Database Creation



A screenshot of the MySQL Workbench interface. The title bar says "Server: 127.0.0.1". The main window has a toolbar with "Databases", "SQL", "Status", "User accounts", "Export", "Import", "Settings", "Replication", "Variables", "Charsets", "Engines", and "Plugins". Below the toolbar is a query editor titled "Run SQL query/queries on server '127.0.0.1':". It contains the SQL command: "CREATE DATABASE disaster_mgt_db;". Below the query editor are several buttons: "Clear", "Format", "Get auto-saved query", "Bind parameters", "Delimiter", "Show this query here again", "Retain query box", "Rollback when finished", "Enable foreign key checks", and a "Go" button.

Table Creation

Table: Donor



A screenshot of the MySQL Workbench interface, specifically the "Query" tab for the "disaster_mgt_db" database. The title bar says "Server: 127.0.0.1 > Database: disaster_mgt_db". The toolbar includes "Structure", "SQL", "Search", "Query", "Export", "Import", "Operations", "Privileges", "Routines", "Events", "Triggers", and "Designer". The main area shows the SQL query editor with the following code:

```
1 CREATE TABLE Donor (
2     DonorID INT AUTO_INCREMENT PRIMARY KEY,
3     DonorName VARCHAR(100) NOT NULL,
4     ContactNo VARCHAR(15),
5     Address VARCHAR(255),
6     DonorType VARCHAR(50)
7 );
```

Below the query editor are buttons for "Clear", "Format", "Get auto-saved query", "Bind parameters", and other execution options.

Table: Donation

The screenshot shows the MySQL Workbench interface with the following details:

- Server:** 127.0.0.1
- Database:** disaster_mgt_db
- Tab:** SQL
- Query:** CREATE TABLE Donation (
1 DonationID INT AUTO_INCREMENT PRIMARY KEY,
2 DonationTimeStamp DATETIME NOT NULL,
3 DonorID INT NOT NULL
5);
- Buttons:** Clear, Format, Get auto-saved query, Bind parameters, Delimiter, Show this query here again, Retain query box, Rollback when finished, Enable foreign key checks, Go

INSERT Operation (Create)

SQL Query used to insert a new donation record into the system.

Table: Donor

The screenshot shows the MySQL Workbench interface with the following details:

- Server:** 127.0.0.1
- Database:** disaster_mgt_db
- Table:** donor
- Tab:** SQL
- Query:** INSERT INTO Donor (DonorName, ContactNo, Address, DonorType)
VALUES ('Nimal Perera', '0712345678', 'Kandy', 'Individual');
- Result pane:** Shows the inserted row: ✓ 1 row inserted. Inserted row id: 16 (Query took 0.0032 seconds.)
- Buttons:** SELECT *, SELECT, INSERT, UPDATE, DELETE, Clear, Format, Get auto-saved query, Bind parameters, Delimiter, Show this query here again, Retain query box, Rollback when finished, Enable foreign key checks, Go, Hide query box
- Table structure pane:** Shows columns: DonorID, DonorName, ContactNo, Address, DonorType

Table: Donation

The screenshot shows the MySQL Workbench interface for the 'disaster_mgt_db' database. The 'Table: donation' tab is selected. In the SQL pane, the following SQL code is entered:

```
1. INSERT INTO Donation (DonationTimeStamp, DonorID)
2. VALUES ('2026-01-27 10:00:00', 17),
3.          ('2026-01-27 11:00:00', 18),
4.          ('2026-01-27 12:00:00', 19);
5.
```

The results pane shows a successful insertion of 3 rows. The message reads: "3 rows inserted. Inserted row id: 12 (Query took 0.0022 seconds.)". The SQL code at the bottom of the results pane is identical to the one in the SQL pane.

SELECT Operation (Read)

A complex query used to view specific data

Table: Donor

The screenshot shows the MySQL Workbench interface for the 'disaster_mgt_db' database. The 'Table: Donor' tab is selected. In the SQL pane, the following SQL code is entered:

```
1. SELECT * FROM Donor;
```

The results pane shows the output of the query: "Showing rows 0 - 0 (1 total, Query took 0.0004 seconds.)". The SQL code at the bottom of the results pane is identical to the one in the SQL pane.

Below the results pane, the data grid displays the following row:

	DonorID	DonorName	ContactNo	Address	DonorType
<input type="checkbox"/>	16	Nimal Perera	0712345678	Kandy	Individual

Table: Donation

The screenshot shows the MySQL Workbench interface for the 'donation' table. The table structure is displayed on the right with columns: DonationID, DonationTimeStamp, and DonorID. The main pane shows the results of the query: 'SELECT * FROM donation;'. The results are:

	DonationID	DonationTimeStamp	DonorID
1	10	2026-01-27 10:00:00	17
2	11	2026-01-27 11:00:00	18
3	12	2026-01-27 12:00:00	19

DELETE Operation (Delete)

SQL Query demonstrating the removal of a record.

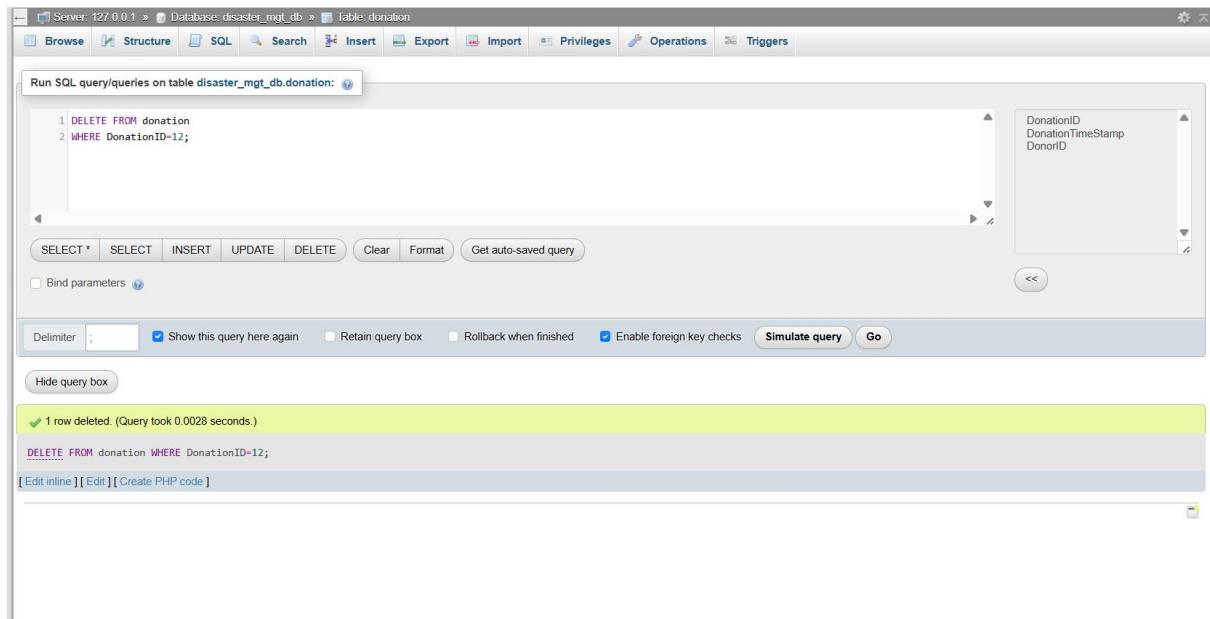
Table: Donor

The screenshot shows the MySQL Workbench interface for the 'donor' table. The table structure is displayed on the right with columns: DonorID, DonorName, ContactNo, Address, and DonorType. The main pane shows the results of the query: 'DELETE FROM Donor WHERE DonorName = 'Nimal Perera';'. The results indicate 1 row deleted.

1 row deleted. (Query took 0.0035 seconds.)

DELETE FROM Donor WHERE DonorName = 'Nimal Perera';

Table: Donation



The screenshot shows the MySQL Workbench interface. The top navigation bar includes 'Server: 127.0.0.1', 'Database: disaster_mgt_db', and 'Table: donation'. Below the navigation are tabs for 'Browse', 'Structure', 'SQL', 'Search', 'Insert', 'Export', 'Import', 'Privileges', 'Operations', and 'Triggers'. A sub-menu under 'Run SQL query/queries on table disaster_mgt_db.donation:' contains the following SQL code:

```
1 DELETE FROM donation
2 WHERE DonationID=12;
```

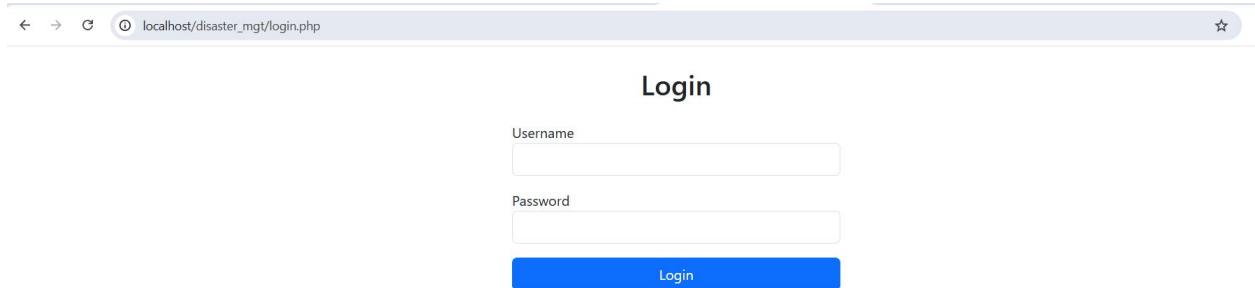
On the right side, there is a table structure view for the 'donation' table with columns: DonationID, DonationTimeStamp, DonorID.

Below the code editor, there are buttons for 'SELECT *', 'SELECT', 'INSERT', 'UPDATE', 'DELETE', 'Clear', 'Format', 'Get auto-saved query', and checkboxes for 'Bind parameters', 'Delimiter', 'Show this query here again', 'Retain query box', 'Rollback when finished', and 'Enable foreign key checks'. There are also 'Simulate query' and 'Go' buttons. At the bottom, a message box indicates: '1 row deleted. (Query took 0.0028 seconds.)' followed by the executed query: 'DELETE FROM donation WHERE DonationID=12;'. Below the message are links for '[Edit inline]', '[Edit]', and '[Create PHP code]'.

8. User Interface Implementation

The following screenshots illustrate the developed frontend application, which provides a user-friendly interface for the database.

8.1 Login Interface



The screenshot shows a web browser window with the URL 'localhost/disaster_mgt/login.php'. The page title is 'Login'. It features a 'Username' input field and a 'Password' input field, both currently empty. Below these fields is a large blue 'Login' button.

8.2 Dashboard

The screenshot shows the main dashboard of the system. At the top right, it says "Welcome, admin" and has a "Logout" button. The dashboard is organized into a grid of nine boxes:

- Donor Management**: Contains "Add Donor" and "View Donors" buttons.
- Donation Management**: Contains "Add Donation" and "View Donations" buttons.
- Donation Items**: Contains "Add Donation Item" and "View Donation Items" buttons.
- Inventory Management**: Contains "Add Inventory" and "View Inventory" buttons.
- Distribution Management**: Contains "Add Distribution" and "View Distributions" buttons.
- Distribution Items**: Contains "Add Distribution Item" and "View Distribution Items" buttons.
- User Management**: Contains "Add User" and "View Users" buttons.
- Beneficiary Management**: Contains "Add Beneficiary" and "View Beneficiary" buttons.
- Item Management**: Contains "Add Item" and "View Item" buttons.

8.3 Donation Entry Form

The screenshot shows the "Add Donation" form. It includes fields for "Donation Timestamp" (a date input field), "Donor" (a dropdown menu with placeholder "-- Select Donor --"), and two buttons at the bottom: "Add Donation" and "Back to Dashboard".

8.4 Inventory View

The screenshot shows the "Inventory" page. It features a table with columns "Item", "Stock", and "Last Updated". The data is as follows:

Item	Stock	Last Updated
Water Bottle	100	2026-01-27 14:00:00
Rice	50	2026-01-27 14:30:00
Blanket	30	2026-01-27 15:00:00

At the bottom left is a "Back to Dashboard" button.

9. Conclusion

The **Disaster Relief and Donation Management System** has been successfully implemented, meeting the requirements of Phase 1, 2 and 3. By converting the conceptual ER model into a relational schema and implementing it within a DBMS, we have created a robust solution for tracking relief efforts. The system effectively handles data integrity through Foreign Key constraints, preventing orphaned records in critical areas like Donations and Distributions. The User Interface further enhances usability, allowing non-technical staff to perform CRUD operations without interacting directly with SQL code. This project demonstrates a complete lifecycle of database development, from problem definition to physical deployment.

10. Individual Contributions

The following table details the specific responsibilities and contributions of each group member towards the completion of the project.

Student ID	Name	Contribution	Signature
AS20240395	K I R Abeysekara	<ul style="list-style-type: none">Proposed the project topic and prepared the project proposalConverted ER diagram into relational schemaDesigned and implemented the complete database using phpMyAdminImplemented PHP-based UIProvided necessary screenshots for the report.Compiled and finalized the complete project report.	
AS20240427	B P R Sachintha	<ul style="list-style-type: none">Finalized the Project Proposal.Designed the complete Entity-Relationship (ER) Diagram.Identified entities, attributes, relationships, and cardinalities.Conducted user interface research and provided feedback.Assisted in preparing and reviewing SQL queriesPerformed System Testing and Data Validation.	
AS20240404	A H M D Silva	<ul style="list-style-type: none">Assisted in analyzing system requirements and project scopeAssisted in converting the ER diagram into relational tablesReviewed primary key and foreign key mappingsPerformed System Testing and Data Validation.Assisted in reviewing the final report.	
AS20240577	N G S Sathsara	<ul style="list-style-type: none">Reviewed screenshots and formatting consistencyAssisted in database testing	

AS20240546	T H Sandaruwan	<ul style="list-style-type: none"> • Assisted in structuring the project report • Assisted in database testing 	
AS20240559	E K S Govinda	<ul style="list-style-type: none"> • Performed System Testing and Data Validation. • Assisted in final documentation review 	
AS20240538	S M Wijesekara	<ul style="list-style-type: none"> • Adding table of contents and formatting for the final report. 	

Group Members

AS20240395-K I R Abeysekara

AS20240427- B P R Sachintha

AS20240404- A H M D Silva

AS20240577- N G S Sathsara

AS20240546- T H Sandaruwan

AS20240559- E K S Govinda

AS20240538- S M Wijesekara