Capital University of Science and Technology

Department of Software Engineering

COURSE TITLE: DATABASE

Semester: FALL 2024

Name: Syeda Anna Zainab

Zia Ur Rehman

Mansoor Ur Rehman

Instructor: Hina Rashid

Date: 19/01/2025

PROJECT: CHARITY DONATION MANAGEMENT SYSTEM

Group members:

1. **Syeda Anna Zainab (BSE233080)**
2. **Zia Ur Rehman (BSE233096)**
3. **Muhammad Mansoor Ur Rehman (BSE233094)**

GitHub Link:

1. **https://github.com/Anna-Zainab**
2. **https://github.com/DARKROARK**
3. **https://github.com/imnxr**

PROJECT SCOPE:

The project scope involves creating a database management system to manage donor information, donations, fundraising campaigns, and events. It will provide features for tracking donations, managing campaigns, and generating reports, while ensuring data security and user-friendly access for donors, admins, and event coordinators. The system will be built using MySQL and will include user authentication and data backup capabilities.

Database Development Life Cycle Phases

1. Requirements Gathering

Stakeholders:

- **Donors**: Provide donations and manage their preferences.
- **Admins**: Oversee campaigns, donations, donors, and events.
- **Campaign Managers**: Plan campaigns, set goals, and monitor progress.
- **Event Coordinators**: Organize and track events linked to campaigns.

Functional Requirements:

- **Donor Management:**
    - Add, view, update, and delete donor records.
    - Maintain donation history and preferences.
- **Donation Management:**
    - Log donations, including payment method, date, and amount.
    - Generate receipts linked to donations.
- **Campaign Management:**
    - Manage campaign goals, funds raised, start and end dates.
    - View campaign progress reports.
- **Event Management:**
    - Record event details (name, date, location) and link them to campaigns.
- **Reporting:**
    - Generate summaries for donations, campaigns, and donors.

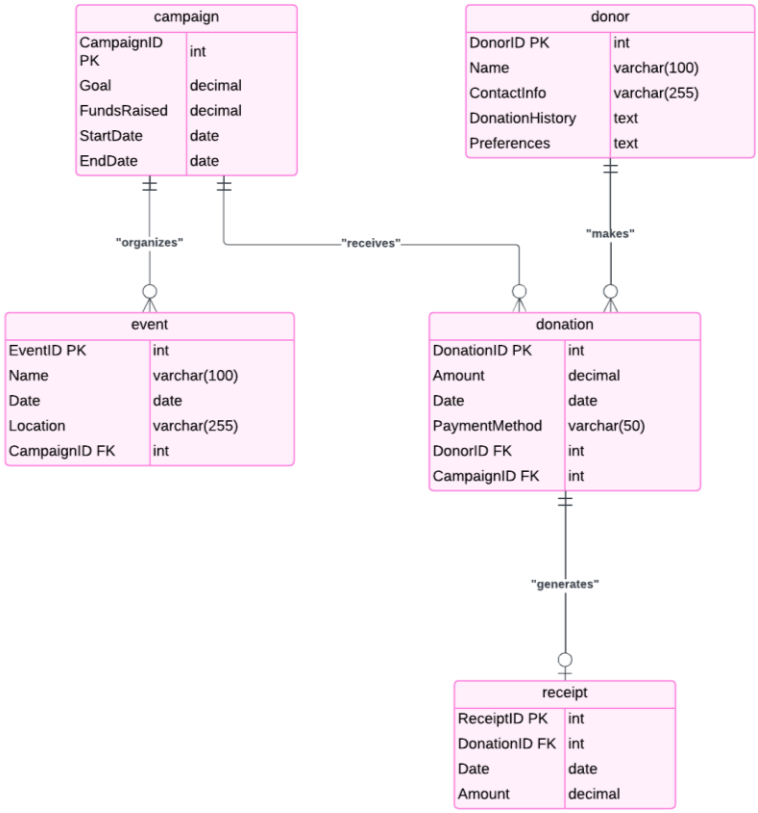
Non-Functional Requirements:

- **Performance**: Handle queries efficiently for large datasets.
- **Security**: Implement user authentication and authorization.
- **Usability**: Provide a user-friendly interface with minimal steps for key operations.

2. Design Phase

The Entity-Relationship Diagram (ERD) visually represents the relationships between key entities in the database, including Donors, Donations, Campaigns, Events, and Receipts. It illustrates how these entities interact, ensuring data integrity and facilitating efficient data management within the system.

Entity-Relationship Diagram (ERD):



RELATIONAL SCHEMA:

1. **Admin**

- **Primary Key**: `adminID`
- **Attributes**: `name`, `username`, `password`, `role`, `permissions`

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
adminID	int(11)		No		auto_increment			
name	varchar(100)		No					
username	varchar(50)		No					
password	varchar(255)		No					
role	varchar(50)		No					
permissions	text		Yes	NULL				

2. Campaign

- Primary Key: `CampaignID`
- Attributes: `Goal`, `FundsRaised`, `StartDate`, `EndDate`

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
CampaignID	int(11)		No		auto_increment			
Goal	decimal(10, 2)		No					
FundsRaised	decimal(10, 2)		Yes	0.00				
StartDate	date		No					
EndDate	date		No					

3. Donation

- **Primary Key**: `DonationID`
- **Attributes**: `Amount`, `Date`, `PaymentMethod`, `DonorID` (FK), `CampaignID` (FK)

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
DonationID	int(11)		No		auto_increment			
Amount	decimal(10, 2)		No					
Date	date		No					
PaymentMethod	varchar(50)		Yes	NULL				
DonorID	int(11)		No			-> donor.DonorID ON UPDATE RESTRICT ON DELETE CASCADE		
CampaignID	int(11)		No			-> campaign.CampaignID ON UPDATE RESTRICT ON DELETE CASCADE		

4. Donor

- **Primary Key**: `DonorID`
- **Attributes**: `Name`, `ContactInfo`, `DonationHistory`, `Preferences`

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
DonorID	int(11)		No		auto_increment			
Name	varchar(100)		No					
ContactInfo	varchar(255)		Yes	NULL				
DonationHistory	text		Yes	NULL				
Preferences	text		Yes	NULL				

5. Event:

- **Primary Key**: `EventID`
- **Attributes**: `Name`, `Date`, `Location`, `CampaignID` (FK)

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
EventID	int(11)		No		auto_increment			
Name	varchar(100)		No					
Date	date		No					
Location	varchar(255)		Yes	NULL				
CampaignID	int(11)		No			-> campaign.CampaignID ON UPDATE RESTRICT ON DELETE CASCADE		

6. Receipt

- **Primary Key**: `ReceiptID`
- **Attributes**: `DonationID` (FK), `Date`, `Amount`

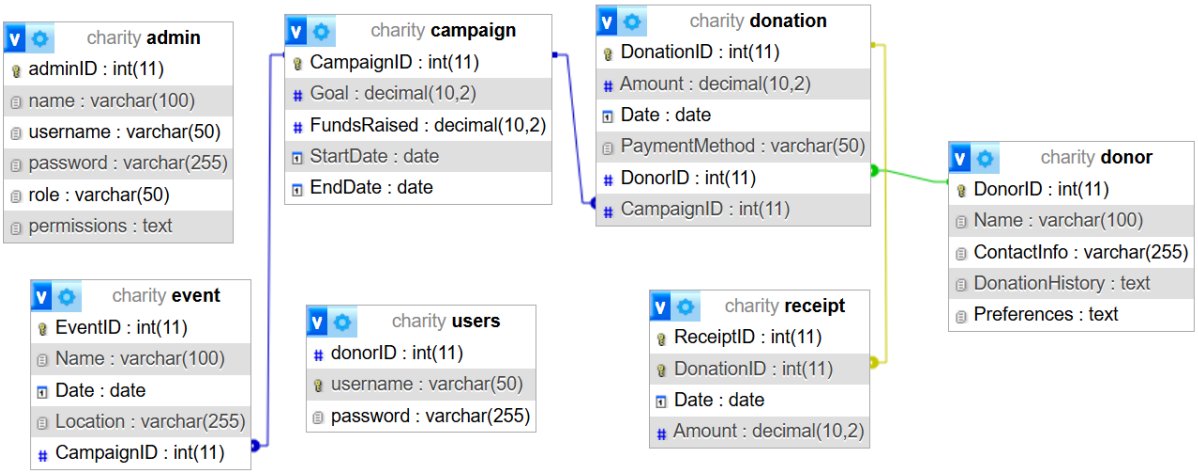
Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
ReceiptID	int(11)		No		auto_increment			
DonationID	int(11)		No			-> donation.DonationID ON UPDATE RESTRICT ON DELETE CASCADE		
Date	date		No					
Amount	decimal(10,2)		No					

7. Users

- **Primary Key**: `donorID`
- **Attributes**: `username`, `password`

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
donorID	int(11)		No					
username	varchar(50)		No					
password	varchar(255)		No					

SCHEMA:



NORMALIZATION:

The database schema is normalized to the third normal form (3NF) to eliminate redundancy and maintain data consistency. For instance, the Donors table contains general donor information such as name and contact details, while the Donations table records specific donation attributes like amount and date, linked by a unique DonorID. Similarly, the Campaigns table holds overarching campaign details, while the Events table captures specific event information, ensuring that each piece of data is stored only once and related through foreign keys. This structure enhances data integrity and simplifies updates across the charity management system.

3. Implementation Phase

- **DBMS:**
    - MYSQL
    - **Development Environment:** XAMPP with phpMyAdmin for easy local database management.
- **Database Table Implementations**

1. Campaign Table

- **Purpose**: Stores information about fundraising campaigns, including goals, progress, and timelines.
- **Attributes and Data Types**:
    - `CampaignID`: `int(11)` - Primary Key
    - `Goal`: `decimal(10,2)` - The target amount for the campaign
    - `FundsRaised`: `decimal(10,2)` - The total funds raised (default: 0.00)
    - `StartDate`: `date` - The start date of the campaign
    - `EndDate`: `date` - The end date of the campaign
- **Keys**:
    - **Primary Key (PK)**: `CampaignID`
    - **Foreign Key (FK)**: None
- **Relationships**:
    - Referenced by `donation` (via `CampaignID`)
    - Referenced by `event` (via `CampaignID`)

2. Donation Table

- **Purpose**: Tracks individual donations made to specific campaigns.
- **Attributes and Data Types**:
    - `DonationID`: `int(11)` - Primary Key
    - `Amount`: `decimal(10,2)` - The donation amount
    - `Date`: `date` - The date of the donation
    - `PaymentMethod`: `varchar(50)` - The payment method (e.g., Credit Card)
    - `DonorID`: `int(11)` - Foreign Key referencing `DonorID` in `donor`
    - `CampaignID`: `int(11)` - Foreign Key referencing `CampaignID` in `campaign`
- **Keys**:
    - **Primary Key (PK)**: `DonationID`
    - **Foreign Keys (FK)**:
        - `DonorID → donor.DonorID`
        - `CampaignID → campaign.CampaignID`
- **Relationships**:
    - References `donor` and `campaign`

- Referenced by `receipt` (via `DonationID`)

3. Donor Table

- **Purpose**: Holds details about donors, including their preferences and contact information.
- **Attributes and Data Types**:
  - `DonorID`: `int(11)` - Primary Key
  - `Name`: `varchar(100)` - The name of the donor
  - `ContactInfo`: `varchar(255)` - Contact information of the donor
  - `DonationHistory`: `text` - Historical data of the donor's donations
  - `Preferences`: `text` - The donor's preferences for contributions
- **Keys**:
  - **Primary Key (PK)**: `DonorID`
  - **Foreign Key (FK)**: None
- **Relationships**:
  - Referenced by `donation` (via `DonorID`)

4. Event Table

- **Purpose**: Manages details about events organized under specific campaigns.
- **Attributes and Data Types**:
  - `EventID`: `int(11)` - Primary Key
  - `Name`: `varchar(100)` - Name of the event
  - `Date`: `date` - Date of the event
  - `Location`: `varchar(255)` - Location of the event
  - `CampaignID`: `int(11)` - Foreign Key referencing `CampaignID` in `campaign`
- **Keys**:
  - **Primary Key (PK)**: `EventID`
  - **Foreign Key (FK)**:
    - `CampaignID → campaign.CampaignID`
- **Relationships**:
  - References `campaign`

5. Receipt Table

- **Purpose**: Maintains receipt records for donations made.
- **Attributes and Data Types**:
  - `ReceiptID`: `int(11)` - Primary Key
  - `DonationID`: `int(11)` - Foreign Key referencing `DonationID` in `donation`
  - `Date`: `date` - The date of the receipt issuance
  - `Amount`: `decimal(10,2)` - The amount on the receipt
- **Keys**:
  - **Primary Key (PK)**: `ReceiptID`

    - **Foreign Key (FK)**:
        - `DonationID → donation.DonationID`
- **Relationships**:
    - References `donation`

Relationships and Constraints

1. **Foreign Key Constraints**:
    - **Donation** references **Donor** to ensure all donations are associated with valid donors.
    - **Donation** references **Campaign** to ensure all donations are tied to valid campaigns.
    - **Event** references **Campaign** to ensure all events are linked to valid campaigns.
    - **Receipt** references **Donation** to ensure all receipts are generated for valid donations.
2. **Primary Key Constraints**:
    - Each table has a primary key to uniquely identify records.

3. Data Integrity

- **Foreign Key Constraints**:
    - Enforce referential integrity to prevent orphan records, ensuring that:
        - Donations cannot exist without a valid donor or campaign.
        - Events cannot exist without a valid campaign.
        - Receipts cannot exist without a valid donation.
- **Default Values**:
    - For attributes like `FundsRaised` in the **Campaign Table**, default values (e.g., `0.00`) maintain data consistency.
- **Data Types**:
    - Appropriate data types (e.g., `decimal` for monetary values, `date` for dates) are used to ensure accurate and consistent data representation.

4. Indexes

- **Primary Keys**:
    - Automatically indexed to ensure efficient lookups.
- **Foreign Keys**:
    - Indexed for faster join operations and to optimize database performance when retrieving related records.

Sample Data Population:

CampaignID	Goal	FundsRaised	StartDate	EndDate
1	5000.00	2000.00	2025-01-01	2025-06-01
2	10000.00	4000.00	2025-02-01	2025-07-01
3	7000.00	3500.00	2025-03-01	2025-08-01

DonorID	Name	ContactInfo	DonationHistory	Preferences
1	John Doe	johndoe@example.com	History1	Preference1
2	Jane Smith	janesmith@example.com	History2	Preference2
3	Robert Brown	robertbrown@example.com	History3	Preference3

DonationID	Amount	Date	PaymentMethod	DonorID	CampaignID
1	500.00	2025-01-05	Credit Card	1	1
2	1500.00	2025-01-10	Bank Transfer	2	1
3	2000.00	2025-02-15	Credit Card	3	2
4	1000.00	2025-02-20	PayPal	1	2

EventID	Name	Date	Location	CampaignID
1	Charity Gala	2025-03-15	Grand Ballroom	1
2	5K Run	2025-04-10	Central Park	2
3	Auction Night	2025-05-20	City Hall	3

ReceiptID	DonationID	Date	Amount
1	1	2025-01-06	500.00
2	2	2025-01-11	1500.00
3	3	2025-02-16	2000.00
4	4	2025-02-21	1000.00

DATABASE creation:

Create database:

```
MariaDB [(none)]> create database Charity;
Query OK, 1 row affected (0.001 sec)
```

Use database:

```
MariaDB [(none)]> use Charity;
Database changed
```

Create table admin

```
MariaDB [(none)]> use Charity;
Database changed
MariaDB [Charity]> CREATE TABLE `admin` (
    ->   `adminID` int(11) NOT NULL,
    ->   `name` varchar(100) NOT NULL,
    ->   `username` varchar(50) NOT NULL,
    ->   `password` varchar(255) NOT NULL,
    ->   `role` varchar(50) NOT NULL,
    ->   `permissions` text DEFAULT NULL
    -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
Query OK, 0 rows affected (0.020 sec)
```

Describe table

```
MariaDB [Charity]> describe admin;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| adminID     | int(11)      | NO   |     | NULL    |       |
| name        | varchar(100) | NO   |     | NULL    |       |
| username    | varchar(50)  | NO   |     | NULL    |       |
| password    | varchar(255) | NO   |     | NULL    |       |
| role        | varchar(50)  | NO   |     | NULL    |       |
| permissions | text         | YES  |     | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
```

Create table campaign:

```
MariaDB [Charity]> CREATE TABLE `campaign` (
    ->   `CampaignID` int(11) NOT NULL,
    ->   `Goal` decimal(10,2) NOT NULL,
    ->   `FundsRaised` decimal(10,2) DEFAULT 0.00,
    ->   `StartDate` date NOT NULL,
    ->   `EndDate` date NOT NULL
    -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
Query OK, 0 rows affected (0.040 sec)
```

Describe campaign:

```
MariaDB [Charity]> Describe campaign;
+-------------+---------------+------+-----+---------+-------+
| Field       | Type          | Null | Key | Default | Extra |
+-------------+---------------+------+-----+---------+-------+
| CampaignID  | int(11)       | NO   |     | NULL    |       |
| Goal        | decimal(10,2) | NO   |     | NULL    |       |
| FundsRaised | decimal(10,2) | YES  |     | 0.00    |       |
| StartDate   | date          | NO   |     | NULL    |       |
| EndDate     | date          | NO   |     | NULL    |       |
+-------------+---------------+------+-----+---------+-------+
5 rows in set (0.018 sec)
```

Create table donation:

```
MariaDB [Charity]> CREATE TABLE `donation` (
    ->   `DonationID` int(11) NOT NULL,
    ->   `Amount` decimal(10,2) NOT NULL,
    ->   `Date` date NOT NULL,
    ->   `PaymentMethod` varchar(50) DEFAULT NULL,
    ->   `DonorID` int(11) NOT NULL,
    ->   `CampaignID` int(11) NOT NULL
    -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
Query OK, 0 rows affected (0.020 sec)
```

Describe donation:

```
MariaDB [Charity]> Describe donation;
+---------------+---------------+------+-----+---------+-------+
| Field         | Type          | Null | Key | Default | Extra |
+---------------+---------------+------+-----+---------+-------+
| DonationID    | int(11)       | NO   |     | NULL    |       |
| Amount        | decimal(10,2) | NO   |     | NULL    |       |
| Date          | date          | NO   |     | NULL    |       |
| PaymentMethod | varchar(50)   | YES  |     | NULL    |       |
| DonorID       | int(11)       | NO   |     | NULL    |       |
| CampaignID    | int(11)       | NO   |     | NULL    |       |
+---------------+---------------+------+-----+---------+-------+
6 rows in set (0.030 sec)
```

Create table donor:

```
MariaDB [Charity]> CREATE TABLE `donor` (
    ->   `DonorID` int(11) NOT NULL,
    ->   `Name` varchar(100) NOT NULL,
    ->   `ContactInfo` varchar(255) DEFAULT NULL,
    ->   `DonationHistory` text DEFAULT NULL,
    ->   `Preferences` text DEFAULT NULL
    -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
Query OK, 0 rows affected (0.017 sec)
```

Describe Donor:

```
MariaDB [Charity]> Describe donor;
+-----------------+-------------+------+-----+---------+-------+
| Field           | Type        | Null | Key | Default | Extra |
+-----------------+-------------+------+-----+---------+-------+
| DonorID         | int(11)     | NO   |     | NULL    |       |
| Name            | varchar(100) | NO   |     | NULL    |       |
| ContactInfo     | varchar(255) | YES  |     | NULL    |       |
| DonationHistory | text        | YES  |     | NULL    |       |
| Preferences     | text        | YES  |     | NULL    |       |
+-----------------+-------------+------+-----+---------+-------+
5 rows in set (0.025 sec)
```

Create Table Event:

```
MariaDB [Charity]> CREATE TABLE `event` (
    ->   `EventID` int(11) NOT NULL,
    ->   `Name` varchar(100) NOT NULL,
    ->   `Date` date NOT NULL,
    ->   `Location` varchar(255) DEFAULT NULL,
    ->   `CampaignID` int(11) NOT NULL
    -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
Query OK, 0 rows affected (0.017 sec)
```

Describe event:

```
MariaDB [Charity]> Describe event;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| EventID    | int(11)     | NO   |     | NULL    |       |
| Name       | varchar(100) | NO   |     | NULL    |       |
| Date       | date        | NO   |     | NULL    |       |
| Location   | varchar(255) | YES  |     | NULL    |       |
| CampaignID | int(11)     | NO   |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
5 rows in set (0.025 sec)
```

Create receipt:

```
MariaDB [Charity]> CREATE TABLE `receipt` (
    ->   `ReceiptID` int(11) NOT NULL,
    ->   `DonationID` int(11) NOT NULL,
    ->   `Date` date NOT NULL,
    ->   `Amount` decimal(10,2) NOT NULL
    -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
Query OK, 0 rows affected (0.017 sec)
```

Describe receipt:

```
MariaDB [Charity]> describe receipt;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| ReceiptID  | int(11)       | NO   |     | NULL    |       |
| DonationID | int(11)       | NO   |     | NULL    |       |
| Date       | date          | NO   |     | NULL    |       |
| Amount     | decimal(10,2) | NO   |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
4 rows in set (0.026 sec)
```

Create table users:

```
MariaDB [Charity]> CREATE TABLE `users` (
    ->   `donorID` int(11) NOT NULL,
    ->   `username` varchar(50) NOT NULL,
    ->   `password` varchar(255) NOT NULL
    -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
Query OK, 0 rows affected (0.017 sec)
```

Describe users:

```
MariaDB [Charity]> Describe Users;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| donorID  | int(11)      | NO   |     | NULL    |       |
| username | varchar(50)  | NO   |     | NULL    |       |
| password | varchar(255) | NO   |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
3 rows in set (0.028 sec)
```

4. Testing the Database:

CRUD Operations

1. Create: Add a new donor.

INSERT INTO Donor (Name, ContactInfo, Preferences)

VALUES ('David Green', 'davidg@example.com', 'Interested in environment campaigns');

Run SQL Query

SQL Query:

```
INSERT INTO Donor (Name, ContactInfo, Preferences)
VALUES ('David Green', 'davidg@example.com', 'Interested in environment campaigns');
```

Run Query

Query Result:

DonorID	Name	ContactInfo	DonationHistory	Preferences
1	John Doe	johndoe@example.com		Prefers anonymity
2	Jane Smith	janesmith@example.com		Interested in medical campaigns
3	Alice Johnson	alicej@example.com		Focus on education
4	Bob Brown	bobbrown@example.com		Environment-focused campaigns
5	Charlie Wilson	charliew@example.com		No preference
6	Emily Davis	emilyd@example.com		Animal welfare
7	David Green	davidg@example.com		Interested in environment campaigns

2. Create: Record a new donation.

INSERT INTO Donation (Amount, Date, PaymentMethod, DonorID)

VALUES (150.00, '2025-01-20', 'Credit Card', 7);

Run SQL Query

SQL Query:

```
INSERT INTO Donation (Amount, Date, PaymentMethod, DonorID)
VALUES (150.00, '2025-01-20', 'Credit Card', 7);
```

Run Query

Query Result:

DonationID	Amount	Date	PaymentMethod	DonorID
1	100.00	2025-01-01	Credit Card	1
2	250.00	2025-01-03	Bank Transfer	2
3	75.00	2025-01-05	PayPal	3
4	500.00	2025-01-10	Credit Card	4
5	300.00	2025-01-12	Bank Transfer	5
6	200.00	2025-01-15	PayPal	6
8	150.00	2025-01-20	Credit Card	7

3. Read: Retrieve all admins with their roles.

SELECT Name, Role FROM Admin;

Run SQL Query

SQL Query:

```
SELECT Name, Role FROM Admin;
```

Run Query

Query Result:

Name	Role
Admin One	Manager
Admin Two	Assistant
Admin Three	Coordinator

4. Read: Show the total funds raised by all campaigns.

SELECT SUM(FundsRaised) AS TotalFunds FROM Campaign;

Run SQL Query

SQL Query:

```
SELECT SUM(FundsRaised) AS TotalFunds FROM Campaign;
```

Run Query

Query Result:

TotalFunds
0.00

5. Update: Update a campaign's funds raised after receiving a large donation.

UPDATE Campaign

SET FundsRaised = FundsRaised + 500

WHERE CampaignID = 1;

Run SQL Query

SQL Query:

```
UPDATE Campaign SET FundsRaised = FundsRaised + 500 WHERE CampaignID = 1;
```

Run Query

Query Result:

CampaignID	Goal	FundsRaised	StartDate	EndDate
1	5000.00	500.00	2025-01-01	2025-06-01
2	10000.00	0.00	2025-01-15	2025-07-15
3	15000.00	0.00	2025-02-01	2025-08-01
4	20000.00	0.00	2025-03-01	2025-09-01

6. Update: Update the contact information of a donor.

UPDATE Donor

SET ContactInfo = 'newemail@example.com'

WHERE Name = 'Jane Smith';

Run SQL Query

SQL Query:

```
UPDATE Donor SET ContactInfo = 'newemail@example.com' WHERE Name = 'Jane Smith';
```

Run Query

Query Result:

DonorID	Name	ContactInfo	DonationHistory	Preferences
1	John Doe	johndoe@example.com		Prefers anonymity
2	Jane Smith	newemail@example.com		Interested in medical campaigns
3	Alice Johnson	alicej@example.com		Focus on education
4	Bob Brown	bobbrown@example.com		Environment-focused campaigns
5	Charlie Wilson	charliew@example.com		No preference
6	Emily Davis	emilyd@example.com		Animal welfare
7	David Green	davidg@example.com		Interested in environment campaigns

7. Delete: Delete a donor who has opted out of the program.

DELETE FROM Donor WHERE Name = 'Bob Brown';

Run SQL Query

SQL Query:

```
DELETE FROM Donor WHERE Name = 'Bob Brown';
```

Run Query

Query Result:

DonorID	Name	ContactInfo	DonationHistory	Preferences
1	John Doe	johndoe@example.com		Prefers anonymity
2	Jane Smith	newemail@example.com		Interested in medical campaigns
3	Alice Johnson	alicej@example.com		Focus on education
5	Charlie Wilson	charliew@example.com		No preference
6	Emily Davis	emilyd@example.com		Animal welfare
7	David Green	davidg@example.com		Interested in environment campaigns

8. Delete: Remove donations made before January 2025.

DELETE FROM Donation WHERE Date < '2025-01-01';

Run SQL Query

SQL Query:

```
DELETE FROM Donation WHERE Date < '2025-01-01';
```

Run Query

Query Result:

DonationID	Amount	Date	PaymentMethod	DonorID
1	100.00	2025-01-01	Credit Card	1
2	250.00	2025-01-03	Bank Transfer	2
3	75.00	2025-01-05	PayPal	3
5	300.00	2025-01-12	Bank Transfer	5
6	200.00	2025-01-15	PayPal	6
8	150.00	2025-01-20	Credit Card	7

Joining Queries

1. **Get all donations along with the names of the donors.**

SELECT DonationID, Amount, Date, Name

FROM Donation

JOIN Donor ON Donation.DonorID = Donor.DonorID;

Run SQL Query

SQL Query:

```
SELECT DonationID, Amount, Date, Name FROM Donation JOIN Donor ON Donation.DonorID =
Donor.DonorID;
```

Run Query

Query Result:

DonationID	Amount	Date	Name
1	100.00	2025-01-01	John Doe
2	250.00	2025-01-03	Jane Smith
3	75.00	2025-01-05	Alice Johnson
4	500.00	2025-01-10	Bob Brown
5	300.00	2025-01-12	Charlie Wilson
6	200.00	2025-01-15	Emily Davis

2. Fetch all receipts with donation amounts and the corresponding donors' names.

```
SELECT ReceiptID, Receipt.Amount, Donor.Name
FROM Receipt
JOIN Donation ON Receipt.DonationID = Donation.DonationID
JOIN Donor ON Donation.DonorID = Donor.DonorID;
```

Run SQL Query

SQL Query:

```
SELECT ReceiptID, Receipt.Amount, Donor.Name FROM Receipt JOIN Donation ON Receipt.DonationID = Donation.DonationID JOIN Donor ON Donation.DonorID = Donor.DonorID;
```

Run Query

Query Result:

ReceiptID	Amount	Name
1	100.00	John Doe
2	250.00	Jane Smith
3	75.00	Alice Johnson
4	500.00	Bob Brown
5	300.00	Charlie Wilson
6	200.00	Emily Davis

3. List all events along with the campaigns they belong to.

```
SELECT Event.Name AS EventName, Campaign.Goal
FROM Event
JOIN Campaign ON Event.CampaignID = Campaign.CampaignID;
```

Run SQL Query

SQL Query:

```
SELECT Event.Name AS EventName, Campaign.Goal FROM Event JOIN Campaign ON
Event.CampaignID = Campaign.CampaignID;
```

Run Query

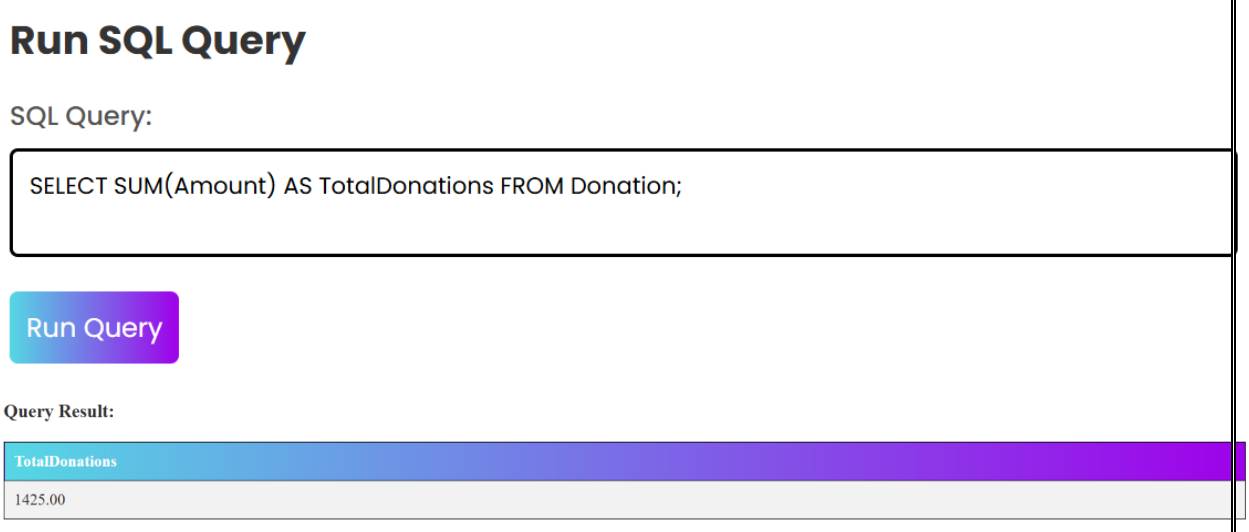
Query Result:

EventName	Goal
Charity Run	5000.00
Fundraising Gala	10000.00
Education Awareness Walk	15000.00
Green Earth Summit	20000.00

Aggregation Queries

1. **Find the total amount raised from all donations.**

   SELECT SUM(Amount) AS TotalDonations FROM Donation;

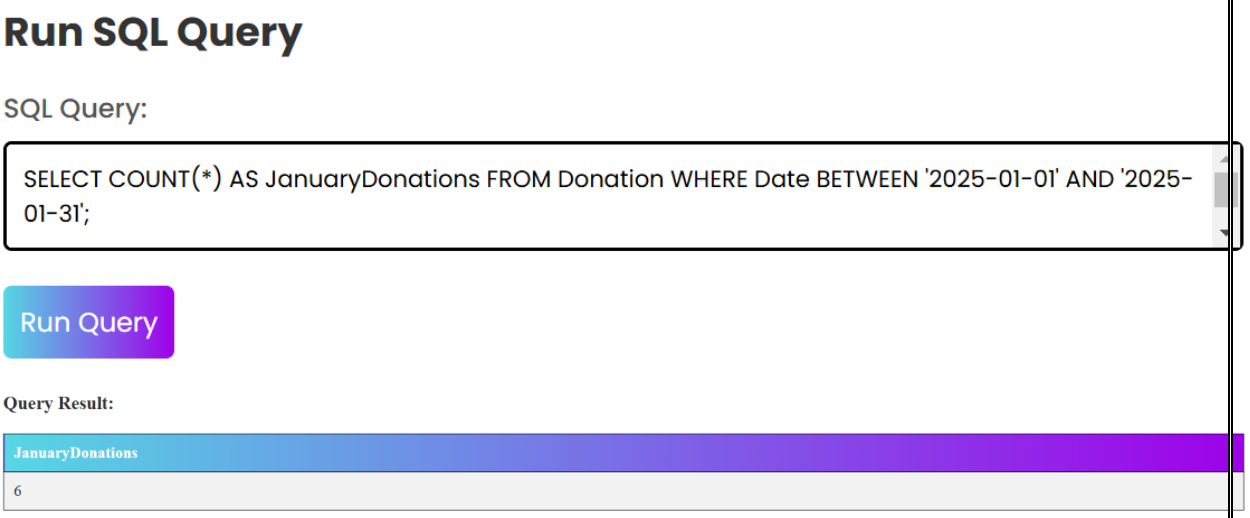


2. **Count the number of donations made in January 2025.**

   SELECT COUNT(*) AS JanuaryDonations

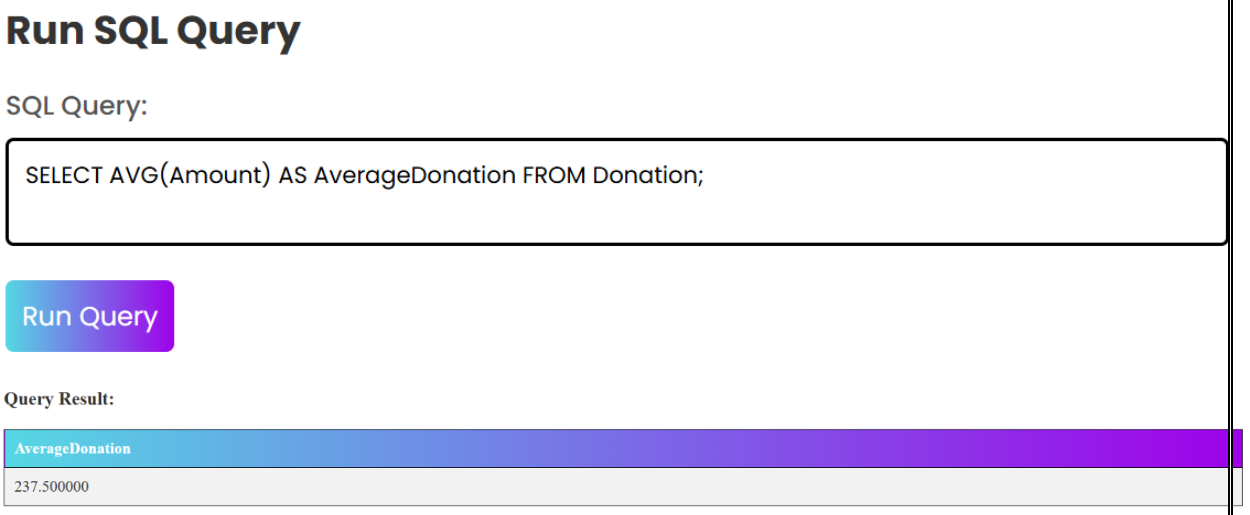
   FROM Donation

   WHERE Date BETWEEN '2025-01-01' AND '2025-01-31';



3. **Calculate the average donation amount.**

```
SELECT AVG(Amount) AS AverageDonation FROM Donation;
```



Subqueries

1. **Get the details of the highest donation.**

```
SELECT *

FROM Donation

WHERE Amount = (SELECT MAX(Amount) FROM Donation);
```

Run SQL Query

SQL Query:

```
SELECT * FROM Donation WHERE Amount = (SELECT MAX(Amount) FROM Donation);
```

Run Query

Query Result:

DonationID	Amount	Date	PaymentMethod	DonorID
4	500.00	2025-01-10	Credit Card	4

2. List donors who have donated more than $200.

```
SELECT Name, ContactInfo
FROM Donor
WHERE DonorID IN (SELECT DonorID FROM Donation WHERE Amount > 200);
```

Run SQL Query

SQL Query:

```
SELECT Name, ContactInfo FROM Donor WHERE DonorID IN (SELECT DonorID FROM Donation WHERE Amount > 200);
```

Run Query

Query Result:

Name	ContactInfo
Jane Smith	janesmith@example.com
Bob Brown	bobbrown@example.com
Charlie Wilson	charliew@example.com

3. Find campaigns that have no associated events.

```
SELECT *
FROM Campaign
WHERE CampaignID NOT IN (SELECT CampaignID FROM Event);
```

Run SQL Query

SQL Query:

```
SELECT * FROM Campaign WHERE CampaignID NOT IN (SELECT CampaignID FROM Event);
```

Run Query

Query Result:

CampaignID	Goal	FundsRaised	StartDate	EndDate

RELATIONAL ALGEBRA QUERIES:

Relational Algebra Queries

1. Retrieve all donors who prefer anonymity:

σ Preferences LIKE '%anonymity%' (Donor)

2. Fetch all donations made using 'Credit Card':

σ PaymentMethod = 'Credit Card' (Donation)

Query Result:

DonationID	Amount	Date	PaymentMethod	DonorID	CampaignID
5	100.00	2025-01-19	Credit Card	4	1

3. List all campaigns that end after June 2025:

σ EndDate > '2025-06-01' (Campaign)

Query Result:

CampaignID	Goal	FundsRaised	StartDate	EndDate
2	10000.00	4000.00	2025-02-01	2025-07-01
3	7000.00	3500.00	2025-03-01	2025-08-01

4. Show donor names and their contact info:

π Name, ContactInfo (Donor)

Query Result:

Name	ContactInfo
Zia Ur Rehman	zrehman@gmail.com
Anna Zainab	anna@gmail.com
Mansoor Ur Rehman	mnxr@gmail.com

5. List the names of all events with their corresponding dates:

π Name, Date (Event)

Query Result:

Name	Date
Charity Gala	2025-03-15
5K Run	2025-04-10
Auction Night	2025-05-20

6. Get all donations along with donor names:

Donation ⋈ Donation.DonorID = Donor.DonorID Donor

Query Result:

DonationID	Amount	Date	PaymentMethod	DonorID	CampaignID	DonorID	Name	ContactInfo	DonationHistory	Preferences
5	100.00	2025-01-19	Credit Card	4	1	Zia Ur Rehman	zrehman@gmail.com	NULL	Food Charity	
6	200.00	2025-01-19	PayPal	5	1	Anna Zainab	anna@gmail.com	NULL	Animal Shelter Charity	
7	300.00	2025-01-19	Bank Transfer	6	1	Mansoor Ur Rehman	mnxr@gmail.com	Food Donation	Rural Areas Donation	

7. Fetch all receipts with donation amounts and corresponding donors:

(Receipt ⋈ Receipt.DonationID = Donation.DonationID Donation)

⋈ Donation.DonorID = Donor.DonorID Donor

Query Result:

ReceiptID	DonationID	Date	Amount	DonationID	Amount	Date	PaymentMethod	DonorID	CampaignID	DonorID	Name	ContactInfo	Donatio
5	5	2025-01-19	100.00	Credit Card	4	1	Zia Ur Rehman	zrehman@gmail.com	NULL	Food Charity			
6	6	2025-01-19	200.00	PayPal	5	1	Anna Zainab	anna@gmail.com	NULL	Animal Shelter Charity			
7	7	2025-01-19	300.00	Bank Transfer	6	1	Mansoor Ur Rehman	mnxr@gmail.com	Food Donation	Rural Areas Donation			

8. List all events along with the campaigns they belong to:

Event ⋈ Event.CampaignID = Campaign.CampaignID Campaign

Query Result:

EventID	Name	Date	Location	CampaignID	CampaignID	Goal	FundsRaised	StartDate	EndDate
1	Charity Gala	2025-03-15	Grand Ballroom	1	5000.00	2000.00	2025-01-01	2025-06-01	
2	5K Run	2025-04-10	Central Park	2	10000.00	4000.00	2025-02-01	2025-07-01	
3	Auction Night	2025-05-20	City Hall	3	7000.00	3500.00	2025-03-01	2025-08-01	

9. Find the total amount raised from all donations:

γ SUM(Amount) → TotalDonations (Donation)

Query Result:

TotalDonations
600.00

10. Count the number of donations made in January 2025:

γ COUNT(*) → JanuaryDonations (σ Date ≥ '2025-01-01' ∧ Date ≤ '2025-01-31' (Donation))

Query Result:

JanuaryDonations
3

11. Calculate the average donation amount:

γ AVG(Amount) → AverageDonation (Donation)

Query Result:

AverageDonation
200.000000

12. Get the details of the highest donation:

Donation ⋈ Amount = MAX(Amount) (Donation)

13. List donors who have donated more than $200:

π Name, ContactInfo (σ Amount > 200 (Donation ⋈ Donation.DonorID = Donor.DonorID))

Query Result:

Name	ContactInfo
Mansoor Ur Rehman	mnxr@gmail.com

14. Find campaigns with no associated events:

Campaign − π CampaignID (Event)

Query Result:

CampaignID
1
2
3

15. List all donors who prefer anonymity or focus on education:

π Name (σ Preferences LIKE '%anonymity%' (Donor)) ∪

π Name (σ Preferences LIKE '%education%' (Donor))

Query Result:

DonorID	Name	ContactInfo	DonationHistory	Preferences

16. Find donors who made donations but are not interested in medical campaigns:

π DonorID (Donation) − π DonorID (σ Preferences LIKE '%medical%' (Donor))

Query Result:

DonorID	Name	ContactInfo	DonationHistory	Preferences

17. Identify donors who contributed to more than one campaign:

γ DonorID, COUNT(CampaignID) → CampaignCount

(Event ⋈ Event.CampaignID = Campaign.CampaignID ⋈ Campaign.CampaignID = Donation.DonationID)

Query Result:

DonorID	CampaignCount

18. Find the difference between total campaign goals and funds raised:

γ CampaignID, Goal - FundsRaised → RemainingGoal (Campaign)

Query Result:

CampaignID	RemainingGoal
1	3000.00
2	6000.00
3	3500.00

19. List campaigns along with the number of events associated with them:

γ CampaignID, COUNT(EventID) → EventCount (Event)

Query Result:

CampaignID	EventCount
1	1
2	1
3	1

20. Find the most generous donor (by total donation amount):

γ DonorID, SUM(Amount) → TotalDonated (Donation) ⋈ Donor.DonorID

σ TotalDonated = MAX(TotalDonated) (γ DonorID, SUM(Amount) → TotalDonated (Donation))

Query Result:

DonorID
6

Performance Testing:

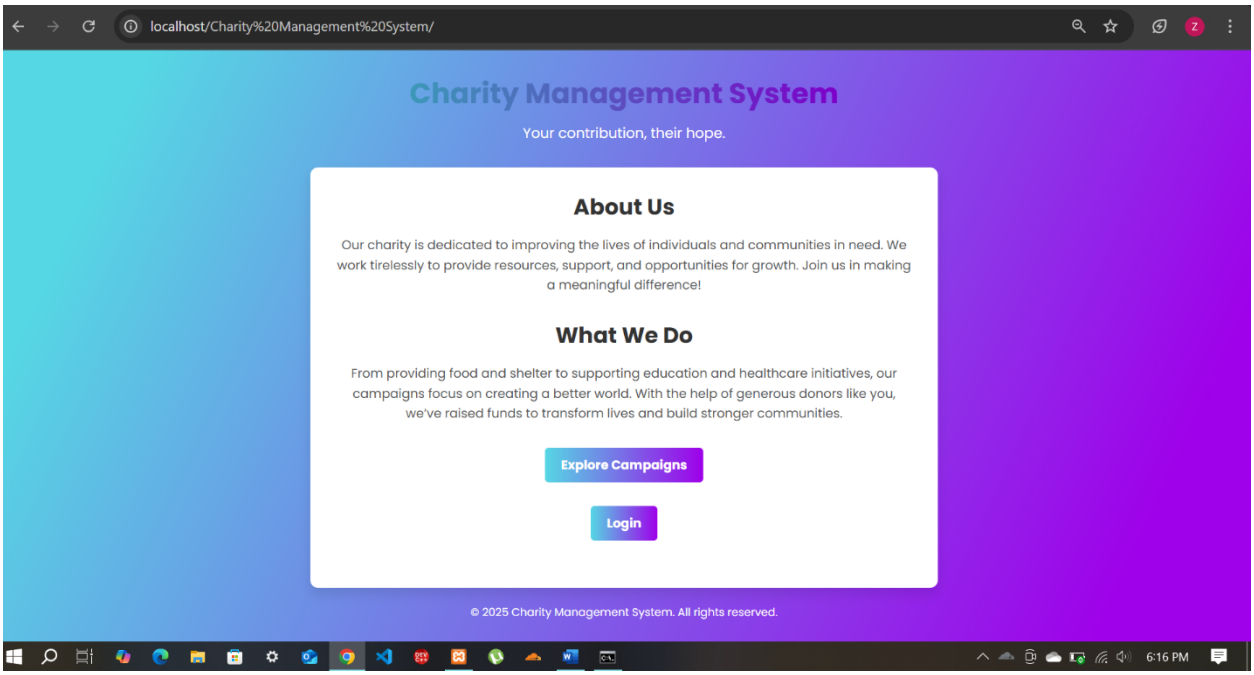
Measured query response times to ensure they met performance requirements. The performance time of each query is very short which make this DBMS efficient.

For example:

```
MariaDB [Charity]> SELECT AVG(Amount) AS AverageDonation FROM Donation;
+-----------------+
| AverageDonation |
+-----------------+
|      200.000000 |
+-----------------+
1 row in set (0.001 sec)
```

Here the run time of query of this is 0.001 which show efficiency of this DBMS

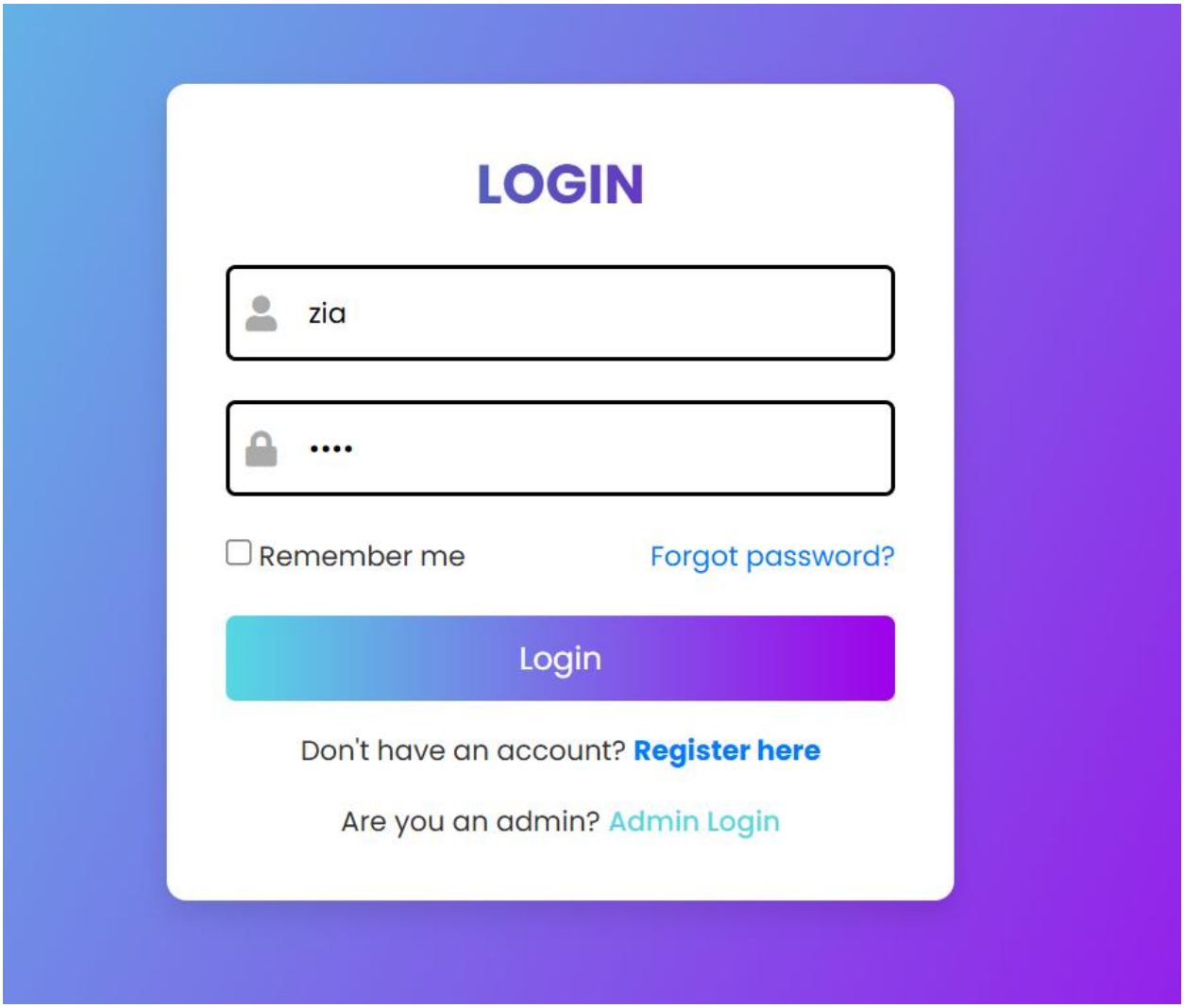
USER INTERFACE MAIN PAGE:



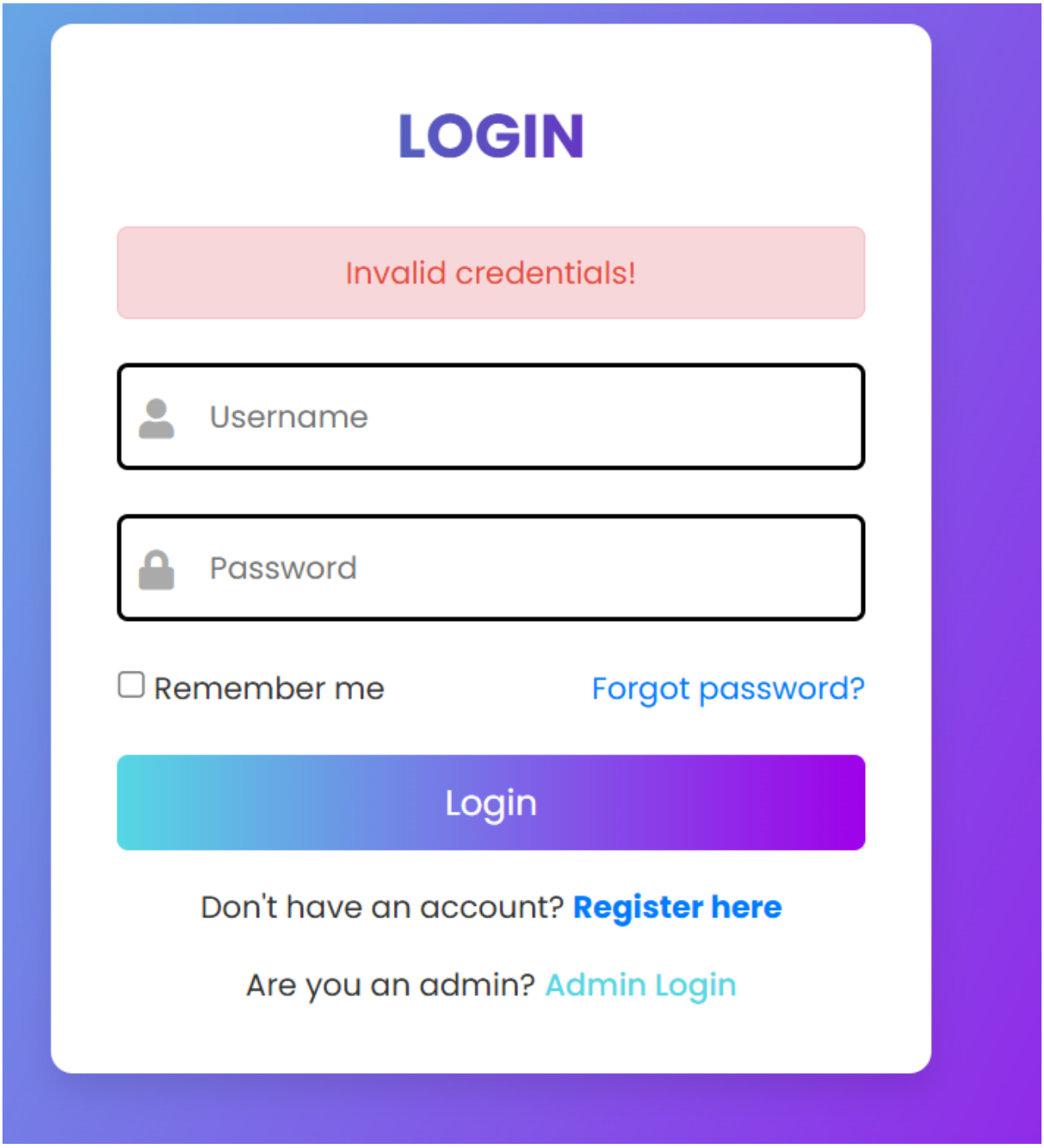
Secure user authentication with hashed passwords:

username	password
Xia	$2y$10$ABwQ08QfFmQq69mHZGsYge05YGpRS1pBPXdkfGZq3bc...
yahyasami	$2y$10$IEeVPHLMAOU1lCxjdzl8UuIGi9A2HR7zVV86avv9YVw...
zia	$2y$10$RUdffvfaEGMFweGl3LTzTuOg12sOrj0x.eDhdOa8f3h...

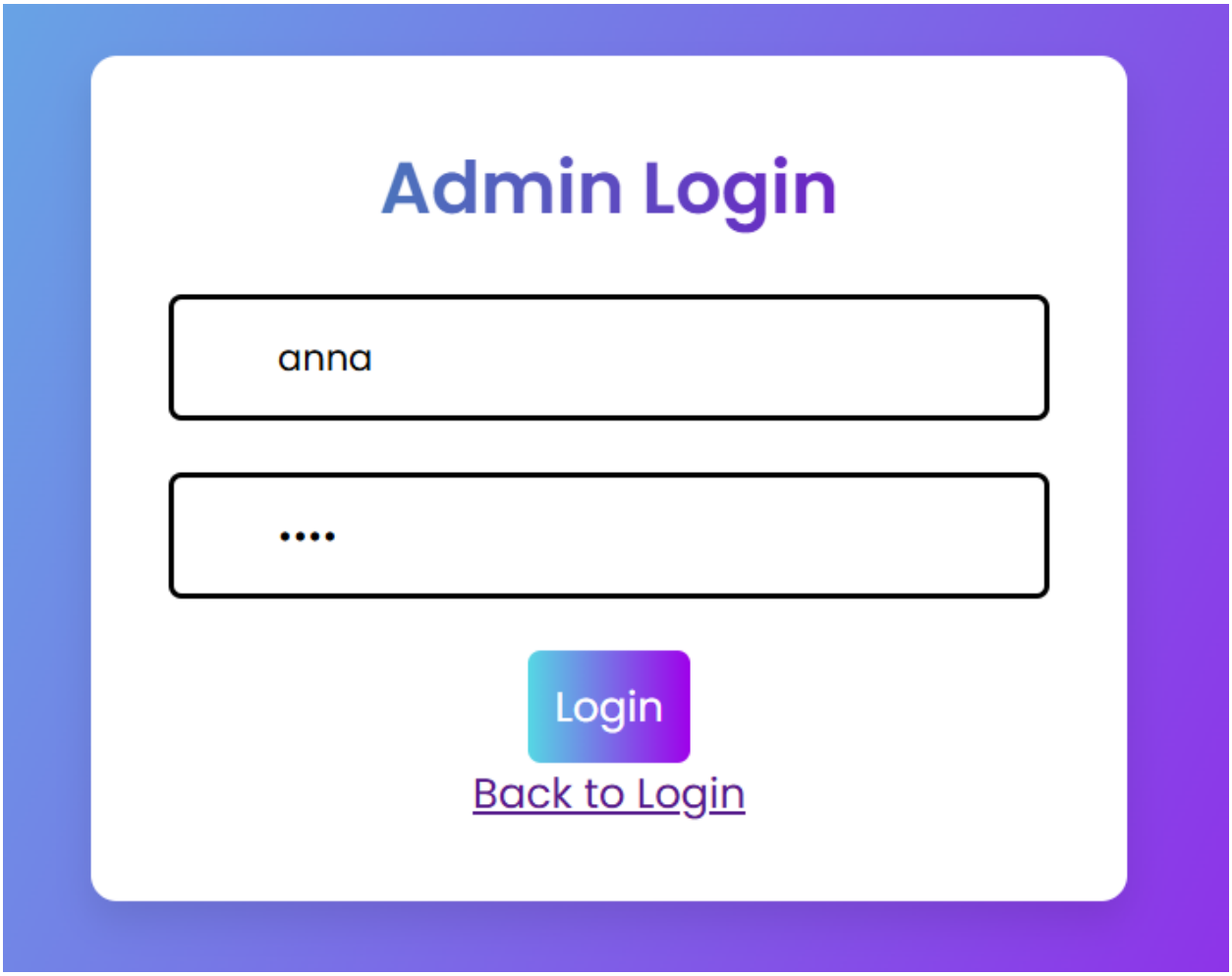
LOGIN FOR USERS:



WRONG LOGIN:


LOGIN
Invalid credentials!
Username
Password
Remember me
Forgot password?
Login
Don't have an account? Register here
Are you an admin? Admin Login

ADMIN LOGIN:



Database Backup and Recovery Testing

- Backup Process: Successfully executed a backup of the database using MySQL export functionality. This process involved exporting all necessary database tables and data into a structured format, ensuring that the backup was both comprehensive and reliable.
- Restoration Verification: Conducted a successful restoration of the database from the created backup file. This involved importing the data back into MySQL, where checks were performed to verify that all data, tables, and structures were restored accurately without any corruption or loss.
- Outcome: The successful completion of both the backup and recovery tests demonstrates the effectiveness of the current backup strategy in place. This provides

confidence in the system’s ability to recover from potential data loss events, ensuring business continuity.

CONCLUSION:

In conclusion, the charity management database system effectively meets the needs of donors, admins, campaign managers, and event coordinators. By normalizing the database to the third normal form (3NF), we have minimized redundancy and ensured data integrity. The system offers essential features for managing donors, donations, campaigns, and events, enhancing fundraising efforts. With secure user authentication and reliable data backup, the system protects sensitive information. Performance testing confirms its efficiency with quick query response times. Overall, this database will significantly improve the charity's operations and support its mission to maximize donations.