

**Design and Development of a  
Sustainable Clothing Exchange Platform  
Using Database Management Systems**

A  
*Project Report submitted in partial fulfillment  
for the award of the degree of  
Bachelor of Technology  
in  
Computer Science and Engineering*

**By**  
**Pathakota Zenith Reddy : 2024BCS-044**

Under the Supervision of  
**Dr. Avadh Kishor**  
Department of Computer Science



**Atal Bihari Vajpayee – Indian  
Institute of Information Technology  
and Management (ABV-IIITM), Gwalior**

## 2. Abstract

---

The Sustainable Clothing Exchange Platform (SCEP) is designed to promote circular fashion by enabling users to donate, exchange, and manage clothing items through a structured digital system.

The project aims to reduce textile waste and encourage sustainable habits using eco-points. Core features include item listings, donation and exchange modules, eco-point tracking, user communication, and location mapping.

The system uses Flask for backend development and SQL as the core database management system, ensuring proper normalization and relational integrity throughout all operations.

The final outcome is a fully normalized database schema capable of supporting scalable, eco-friendly clothing exchange workflows.

## 3. Introduction

---

Textile waste is one of the world's fastest-growing environmental concerns, with millions of clothing items discarded every year. Many of these items are still reusable, yet lack of accessibility and proper channels prevents people from exchanging or donating them. Motivated by sustainability and circular economy principles, the Sustainable Clothing Exchange Platform (SCEP) aims to provide a structured digital space where users can contribute to eco-friendly fashion behavior. It also seeks to promote mindful consumption by encouraging people to extend the life cycle of their clothing.

The project's scope includes features for listing clothing items, making exchange requests, donating to verified organizations, earning eco-points, and communicating with other users. The main beneficiaries include environmentally conscious users, NGOs, students, and communities that wish to donate or acquire pre-loved clothing responsibly.

Together, these objectives guide the platform toward building a circular, responsible, and user-friendly clothing exchange system.

- Reducing textile waste
- Encouraging reuse and recycling
- Digitizing donation and exchange activities
- Ensuring transparency and traceability through proper DB design
- Maintaining data integrity and avoiding redundancy through normalization
- Implementing efficient collection and sorting processes to streamline textile recovery

## 4. System Analysis & Design

---

### Functional Requirements

- Users can create accounts and manage personal profiles
- Users can list clothing items with detailed descriptions and images
- Users can donate items to verified organizations through the platform
- Users can initiate and respond to clothing exchange requests
- System tracks eco-points for donations and successful exchanges
- Conversations and message exchange between users for coordination
- Admin/organizations can review, approve, and manage donation activities
- The system maintains logs of key actions for monitoring and accountability

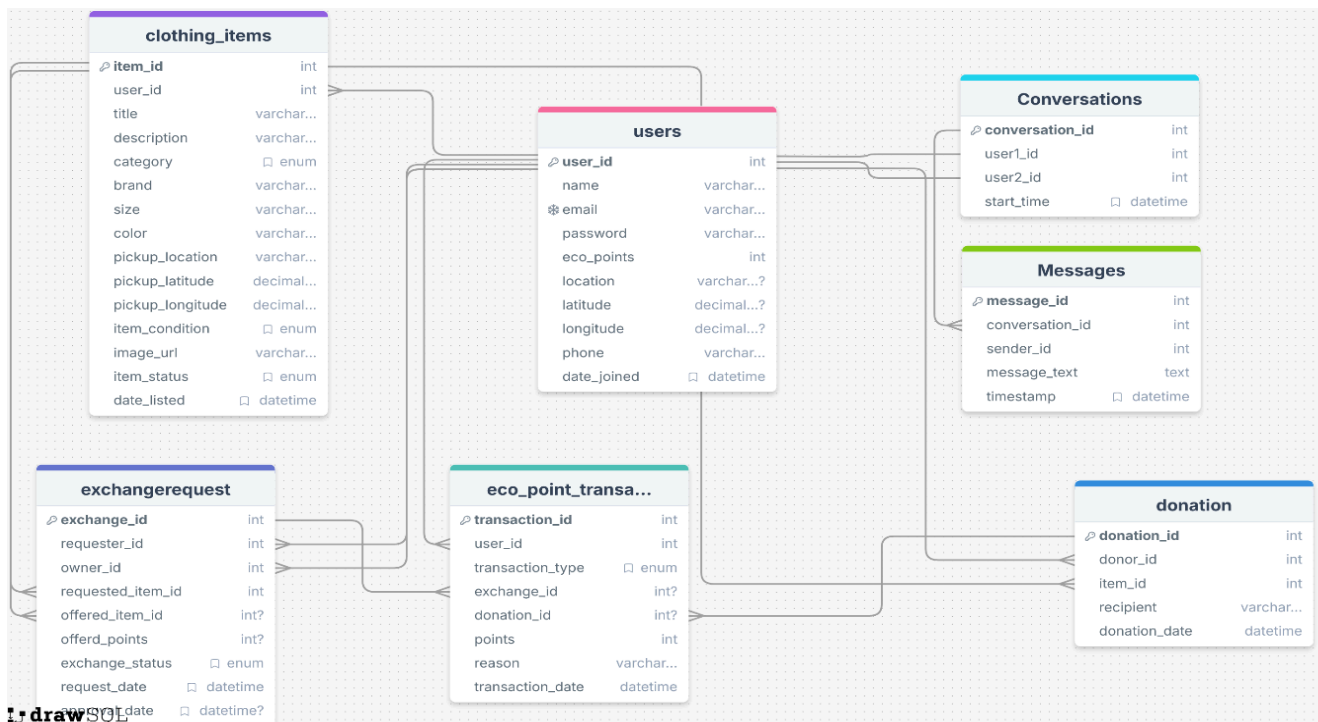
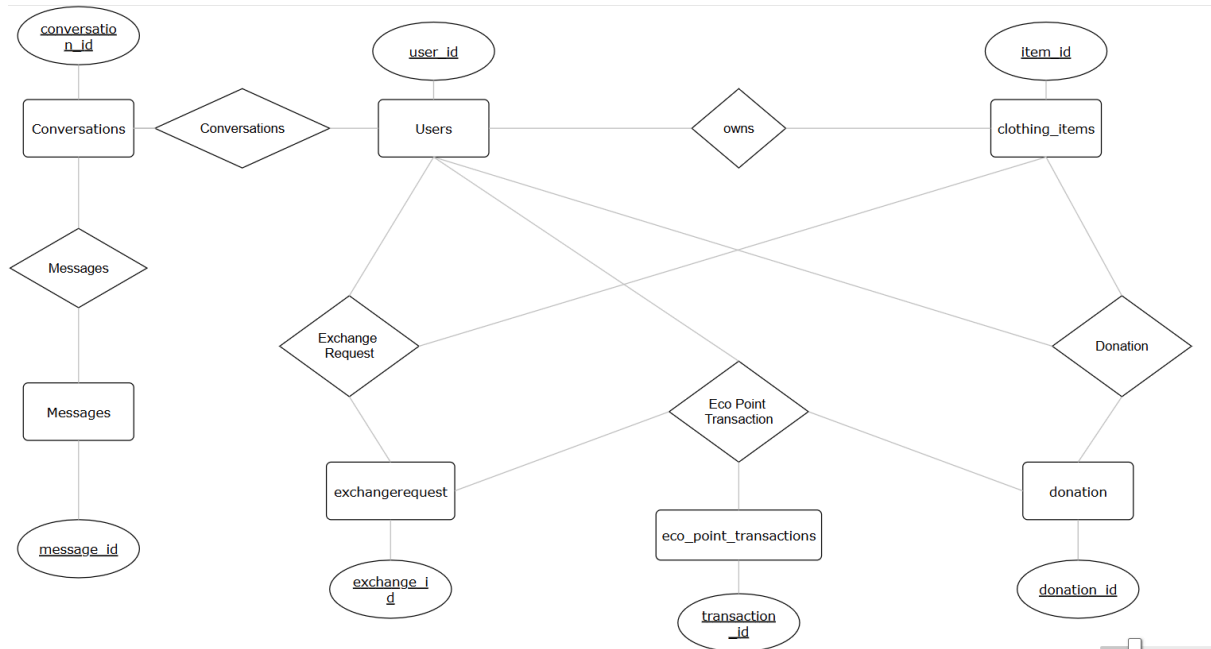
### Non-Functional Requirements

- Data consistency and integrity across all relational tables
- Minimum redundancy through proper normalization
- Secure storage of user information with controlled access
- Fast query retrieval for item search, user actions, and transactions
- Scalable schema that supports future analytics, reporting, and feature expansion
- Reliable uptime with the ability to handle a growing number of users and items

## 4.2 Entities & Descriptions

- **Users:** Profile info, contact details, linked location, and accumulated eco-points.
- **Locations:** Standardized coordinates and address entries used for pickups and deliveries.
- **Clothing\_Items:** Items listed by users for exchange , including attributes such as size, condition, and item type.
- **ExchangeRequest:** Records exchange offers between users, including requester/responder IDs and request status.
- **Donation:** Stores details of item donations to organizations, including dates and pickup/drop-off information.
- **Eco\_Point\_Transactions:** Points awarded for actions such as exchanges, donations, and other eco-friendly contributions.
- **Conversations:** Holds conversation threads created between two or more users for coordination and communication.
- **Messages:** Individual messages sent within a conversation, containing sender ID, timestamps, and message content.

## 4.3 ER Diagram



## 4.4 Normalization

Key FDs:

TransactionID → all transaction attributes

UserID → UserName, UserEmail

ItemID → ItemName, ItemCategory

1NF

All attributes are atomic → 1NF satisfied.

## 2NF

Non-key attributes describe different entities (users, items, locations, types).

Decompose to remove these dependencies:

- USER(UserID, UserName, UserEmail)
- ITEM(ItemID, ItemName, ItemCategory)
- LOCATION(LocationID, LocationName, LocationAddress)
- TRANSACTIONTYPE(TransactionType, EcoPointsEarnedPerType)
- TRANSACTION(TransactionID, UserID, ItemID, LocationID, TransactionType, EcoPointsEarned, TransactionDate)

→ Each table's attributes depend fully on its primary key.

## 3NF

Transitive dependencies removed:

User info depends on UserID, Item info on ItemID, and points rules on TransactionType.

No non-key attribute depends on another non-key attribute → 3NF achieved.

# 5. Database Implementation

---

## 5.1 Database Schema (DDL Overview)

The Database has 10 tables in total:

- users, clothing\_items, exchangerequest, donation, eco\_point\_transaction, conversations, messages, auth\_tokens.

### Primary and Foreign Key Structure

- Every table includes a primary key using an AUTO\_INCREMENT integer field. (e.g., user\_id, item\_id, exchange\_id, etc.)
- Foreign keys define relationships such as:
  - Users → Clothing Items (clothing\_items.user\_id)
  - Users → Exchange Requests (requester\_id, owner\_id)

- Clothing Items → Exchange Requests (requested\_item\_id, offered\_item\_id)
- Users / Clothing Items → Donation
- Users / Exchange / Donation → Eco Point Transactions
- Users → Conversations / Messages
- Users → Auth Tokens

These constraints ensure referential integrity across all user actions, item listings, exchanges, and point transactions.

### Constraints (UNIQUE, CHECK, NOT NULL)

- UNIQUE:
  - users.email (prevents duplicate accounts)
- ENUM (acts as CHECK constraints):
  - Item fields: category, item\_condition, item\_status
  - Exchange: exchange\_status
  - Transactions: transaction\_type
- NOT NULL:
  - All primary keys, foreign key fields, ENUM fields, and required text fields.
- DEFAULT values:
  - users.eco\_points defaults to 100
  - Several datetime fields default to CURRENT\_TIMESTAMP

## 5.2 Sample Data

### Insert a New Clothing Item

```
INSERT INTO clothing_items
(user_id, title, description, category, brand, size, cost,
pickup_location,
pickup_latitude, pickup_longitude, item_condition, image_url,
item_status)
VALUES
(2, 'Red Hoodie', 'Comfortable cotton hoodie suitable for casual
wear',
'Unisex', 'H&M', 'M', 30, 'Downtown Plaza',
40.7128, -74.0060, 'Gently Used',
'https://example.com/images/red-hoodie.jpg', 'Available');
```

Example (Users Table):

	item_id	title	description	category	size	item_condition	cost	image_url	item_status
▶	20	Red Hoodie	Comfortable cotton hoodie suitable for casual w...	Unisex	M	Gently Used	30	https://example.com/images/red-hoodie.jpg	Available
	13	Skater Skirt	Trendy checkered skirt, good condition.	Women	M	Gently Used	20	https://dummyimage.com/600x400/000000/ffff...	Available
	10	T-Shirt	Classic white cotton tee, minimal wear.	Men	M	Gently Used	15	https://dummyimage.com/600x400/000000/ffff...	Available
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

## 5.3 Tools & Environment

- Flask (Python)
- React
- MySQL Workbench 8.0 CE
- MySQL Connector (Python)

# 6. Query Module

---

## 6.1 Query List (10 Queries)

### 1. Fetch User Profile (Basic SELECT + WHERE)

```
SELECT user_id, name, email, phone, eco_points
FROM users
WHERE user_id = %s;
```

### 2. View Available Clothing Items (SELECT + WHERE + ORDER BY)

```
SELECT item_id, title, description, category, size, condition, cost,
image, item_status
FROM clothing_items
WHERE user_id = %s AND item_status = 'Available'
ORDER BY item_id DESC;
```

### 3. Count Total Exchange Requests (Aggregation: COUNT)

```
SELECT COUNT(*) AS total_requests
FROM exchangerequest
WHERE requester_id = %s;
```

### 4. Calculate Total Eco Points Earned (Aggregation: SUM)

```
SELECT SUM(points) AS total_earned
FROM ecopoints_transactions
WHERE user_id = %s AND transaction_type = 'Earn';
```

### 5. Get Exchange Requests Sent by User (INNER JOIN)

```

SELECT er.exchange_id, er.exchange_status, er.requested_date,
       req_item.title AS requested_item,
       off_item.title AS offered_item
FROM exchangerequest er
INNER JOIN clothing_items req_item ON er.requested_item_id =
req_item.item_id
INNER JOIN clothing_items off_item ON er.offered_item_id =
off_item.item_id
WHERE er.requester_id = %s;

```

## 6. Get Exchange Requests Received (INNER JOIN)

```

SELECT er.exchange_id, er.exchange_status, er.requested_date,
       u.name AS requester_name,
       ci.title AS item_requested
FROM exchangerequest er
INNER JOIN users u ON er.requester_id = u.user_id
INNER JOIN clothing_items ci ON er.requested_item_id = ci.item_id
WHERE er.owner_id = %s;

```

## 7. View Conversation List (JOIN + GROUP BY + MAX)

```

SELECT c.conversation_id,
       u.name AS other_user,
       MAX(m.sent_at) AS last_message_time
FROM conversations c
JOIN users u
  ON (CASE
        WHEN c.user1_id = %s THEN c.user2_id = u.user_id
        ELSE c.user1_id = u.user_id
      END)
LEFT JOIN messages m ON c.conversation_id = m.conversation_id
WHERE c.user1_id = %s OR c.user2_id = %s
GROUP BY c.conversation_id, other_user
ORDER BY last_message_time DESC;

```

## 8. Retrieve Complete Message History (Basic SELECT + ORDER BY)

```

SELECT message_id, sender_id, receiver_id, message_text, sent_at
FROM messages
WHERE conversation_id = %s
ORDER BY sent_at ASC;

```

## 9. Verify If Item Is Available (Subquery / Filter Validation)

```

SELECT item_id, user_id, item_status
FROM clothing_items
WHERE item_id = %s AND item_status = 'Available';

```

## 10. Approve Exchange & Update Item Status (UPDATE + Transaction Logic)



```

UPDATE clothing_items
SET item_status = 'Exchanged'
WHERE item_id = %s;
UPDATE exchangerequest
SET exchange_status = 'Approved'
WHERE exchange_id = %s;
INSERT INTO ecopoints_transactions (user_id, points,
transaction_type)
VALUES (%s, %s, 'Earn');

```

## 6.2 Screenshots of Outputs

Figure 1: Output for Query 1 – All Users data

	user_id	name	email	phone	eco_points
▶	1	Zenith Reddy	zenithr1@gmail.com	9398026804	107
	2	Yashwant	yashwanth@example.com	9999999999	107
	3	akshitha	akshitha@gmail.com	8888888888	100
•	NULL	NULL	NULL	NULL	NULL

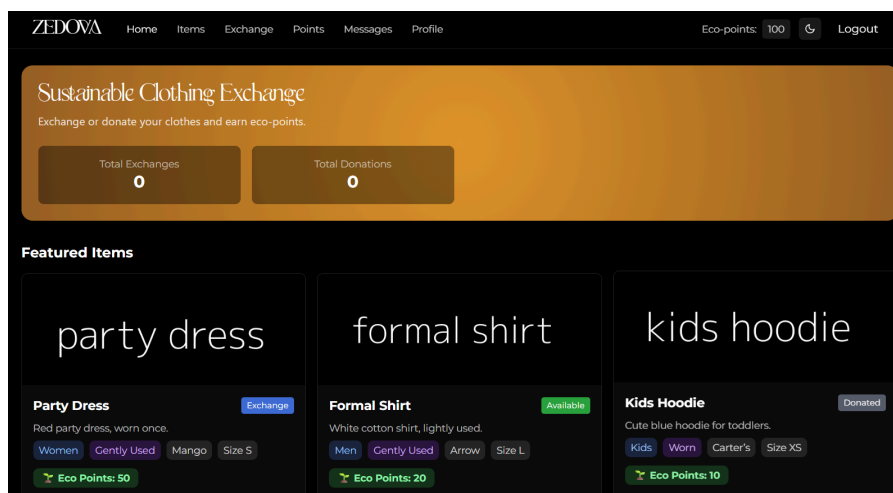
Figure 2: Output for Query 2 – Available Clothing Items

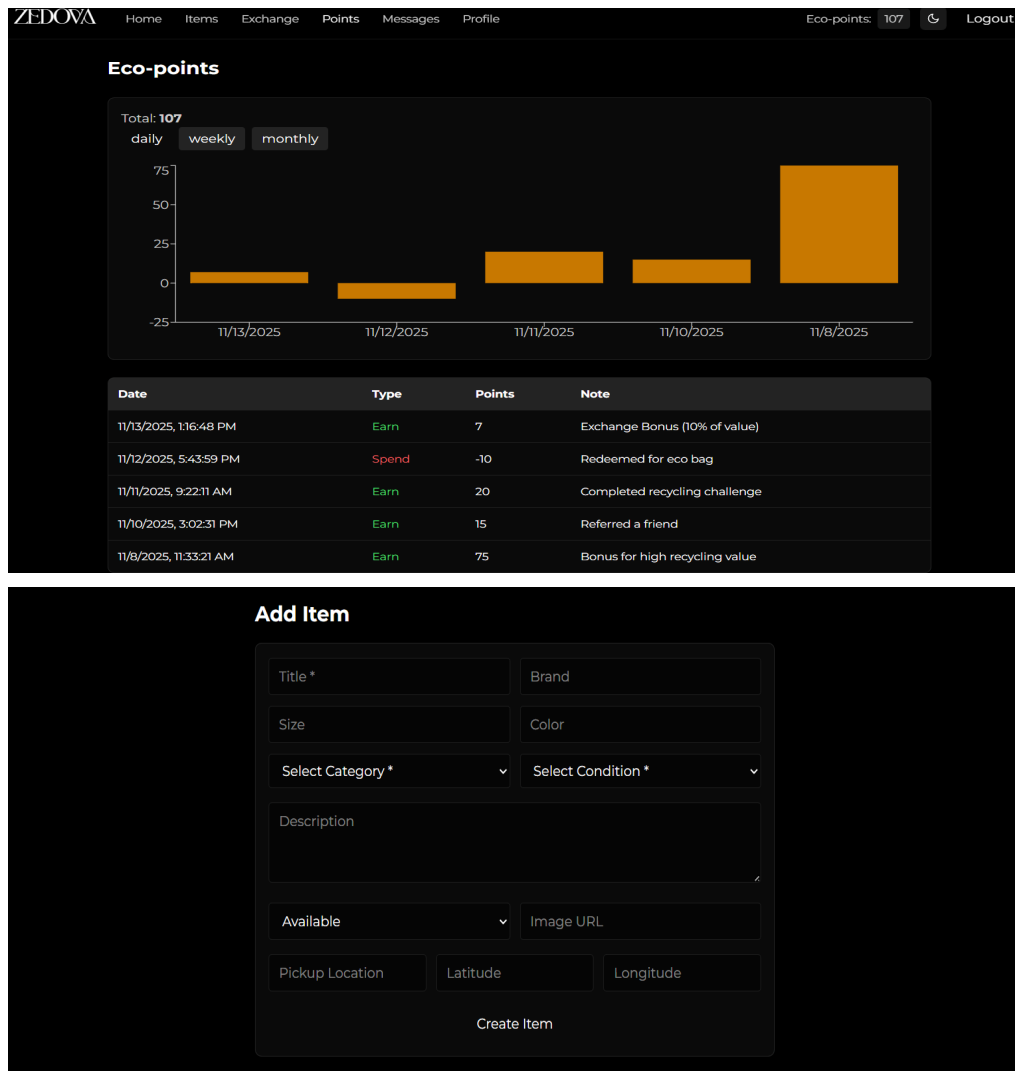
	item_id	title	description	category	size	item_condition	cost	image_url	item_status
▶	18	Formal Shirt	White cotton shirt, lightly used.	Men	L	Gently Used	20	https://dummyimage.com/600x400/000000/ffff...	Available
	16	Yoga Pants	Comfortable yoga pants, stretchable material.	Women	M	Gently Used	25	https://dummyimage.com/600x400/000000/ffff...	Available
	15	Puffer Jacket	Warm puffer jacket for winter.	Men	XL	New	60	https://dummyimage.com/600x400/000000/ffff...	Available
•	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 3: Output for Query 8 –Complete Message History

	message_id	sender_id	message_text	timestamp
▶	1	1	heyyy	2025-11-13 11:25:28
	2	2	hello	2025-11-13 11:25:45
	3	1	how are u	2025-11-13 11:26:01
	5	2	fine	2025-11-13 15:16:19
	6	1	i want the shirt from ur listing	2025-11-13 15:17:27
•	NULL	NULL	NULL	NULL

## 7. User Interface





## 8. Conclusion & Future Scope

The Sustainable Clothing Exchange Platform successfully models an efficient and eco-friendly fashion circulation system. The database design ensures minimal redundancy through proper normalization and provides strong relational support for key functionalities like donations, exchanges, communication, and eco-point tracking.

Limitations include lack of automated recommendation systems and limited platform scalability without sharding or indexing strategies.

Future enhancements may include:

- AI-based clothing recommendation
- Mobile app integration
- Geo-based pickup optimization
- Analytics dashboard for sustainability metrics

## 9. References

---

- DBS ppts
- W3Schools SQL Reference
- Youtube tutorial for connection
- Python(Flask) / React Documentation
- ERD & Normalization report